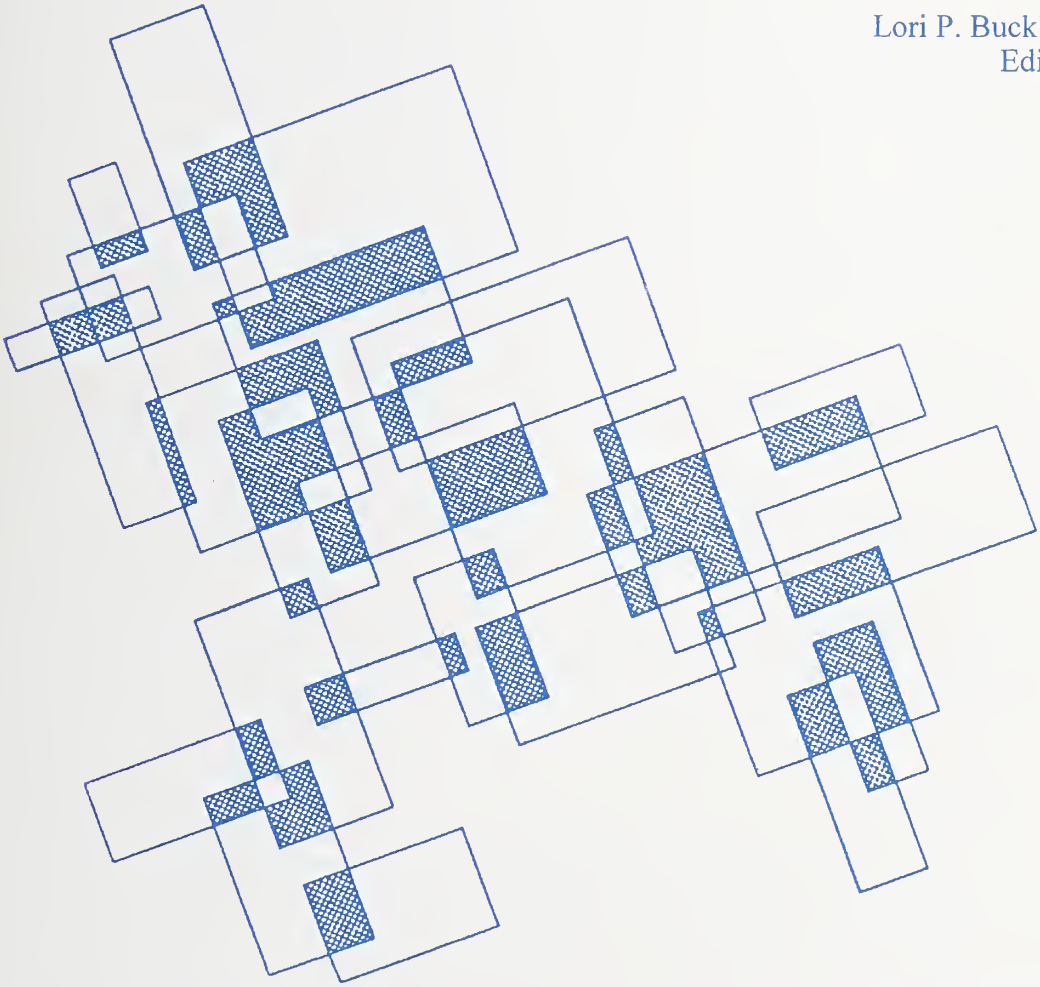
NIST Special Publication 500-255

Information Technology:
The Twelfth Text REtrieval Conference, TREC 2003

Ellen M. Voorhees
and
Lori P. Buckland
Editors



NIST
National Institute of Standards and Technology
Technology Administration, U.S. Department of Commerce

The National Institute of Standards and Technology was established in 1988 by Congress to "assist industry in the development of technology . . . needed to improve product quality, to modernize manufacturing processes, to ensure product reliability . . . and to facilitate rapid commercialization . . . of products based on new scientific discoveries."

NIST, originally founded as the National Bureau of Standards in 1901, works to strengthen U.S. industry's competitiveness; advance science and engineering; and improve public health, safety, and the environment. One of the agency's basic functions is to develop, maintain, and retain custody of the national standards of measurement, and provide the means and methods for comparing standards used in science, engineering, manufacturing, commerce, industry, and education with the standards adopted or recognized by the Federal Government.

As an agency of the U.S. Commerce Department's Technology Administration, NIST conducts basic and applied research in the physical sciences and engineering, and develops measurement techniques, test methods, standards, and related services. The Institute does generic and precompetitive work on new and advanced technologies. NIST's research facilities are located at Gaithersburg, MD 20899, and at Boulder, CO 80303. Major technical operating units and their principal activities are listed below. For more information visit the NIST Website at http://www.nist.gov, or contact the Publications and Program Inquiries Desk, 301-975-3058.

Office of the Director

- National Quality Program
- International and Academic Affairs

Technology Services

- Standards Services
- Technology Partnerships
- Measurement Services
- Information Services
- Weights and Measures

Advanced Technology Program

- Economic Assessment
- Information Technology and Applications
- Chemistry and Life Sciences
- Electronics and Photonics Technology

Manufacturing Extension Partnership Program

- Regional Programs
- National Programs
- Program Development

Electronics and Electrical Engineering Laboratory

- Microelectronics
- Law Enforcement Standards
- Electricity
- Semiconductor Electronics
- Radio-Frequency Technology[1]
- Electromagnetic Technology[1]
- Optoelectronics[1]
- Magnetic Technology[1]

Materials Science and Engineering Laboratory

- Intelligent Processing of Materials
- Ceramics
- Materials Reliability[1]
- Polymers
- Metallurgy
- NIST Center for Neutron Research

Chemical Science and Technology Laboratory

- Biotechnology
- Process Measurements
- Surface and Microanalysis Science
- Physical and Chemical Properties[2]
- Analytical Chemistry

Physics Laboratory

- Electron and Optical Physics
- Atomic Physics
- Optical Technology
- Ionizing Radiation
- Time and Frequency[1]
- Quantum Physics[1]

Manufacturing Engineering Laboratory

- Precision Engineering
- Manufacturing Metrology
- Intelligent Systems
- Fabrication Technology
- Manufacturing Systems Integration

Building and Fire Research Laboratory

- Applied Economics
- Materials and Construction Research
- Building Environment
- Fire Research

Information Technology Laboratory

- Mathematical and Computational Sciences[2]
- Advanced Network Technologies
- Computer Security
- Information Access
- Convergent Information Systems
- Information Services and Computing
- Software Diagnostics and Conformance Testing
- Statistical Engineering

[1] At Boulder, CO 80303.

[2] Some elements at Boulder, CO.

NIST Special Publication 500-255

Information Technology: The Twelfth Text Retrieval Conference, TREC 2003

Ellen M. Voorhees and
Lori P. Buckland
Editors

Information Technology Laboratory
Information Access Division
National Institute of Standards and Technology
Gaithersburg, MD 20899-8940

May 2004

U.S. Department of Commerce
Donald L. Evans, Secretary

Technology Administration
Phillip J. Bond, Under Secretary of Commerce for Technology

National Institute of Standards and Technology
Arden L. Bement, Jr., Director

Reports on Information Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) stimulates U.S. economic growth and industrial competitiveness through technical leadership and collaborative research in critical infrastructure technology, including tests, test methods, reference data, and forward-looking standards, to advance the development and productive use of information technology. To overcome barriers to usability, scalability, interoperability, and security in information systems and networks, ITL programs focus on a broad range of networking, security, and advanced information technologies, as well as the mathematical, statistical, and computational sciences. This Special Publication 500-series reports on ITL's research in tests and test methods for information technology, and its collaborative activities with industry, government, and academic organizations.

Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

National Institute of Standards and Technology
Special Publication 500-255
Natl. Inst. Stand. Technol.
Spec. Publ. 500-255
1284 pages (May 2004)
CODEN: NSPUE2

U.S. GOVERNMENT PRINTING OFFICE
WASHINGTON: 2004

For sale by the Superintendent of Documents, U.S. Government Printing Office
Internet: bookstore.gpo.gov—Phone: (202) 512-1800—Fax: (202) 512-2250
Mail: Stop SSOP, Washington, DC 20402-0001

Foreword

This report constitutes the proceedings of the 2003 edition of the Text REtrieval Conference, TREC 2003, held in Gaithersburg, Maryland, November 18–21, 2003. The conference was co-sponsored by the National Institute of Standards and Technology (NIST), the Advanced Research and Development Activity (ARDA), and the Defense Advanced Research Projects Agency (DARPA). Approximately 200 people attended the conference, including representatives from 22 different countries. The conference was the twelfth in an on-going series of workshops to evaluate new technologies for text retrieval and related information-seeking tasks. Ninety-three groups submitted retrieval results to one or more of the workshop's tracks.

The workshop included plenary sessions, discussion groups, a poster session, and demonstrations. Because the participants in the workshop drew on their personal experiences, they sometimes cite specific vendors and commercial products. The inclusion or omission of a particular company or product implies neither endorsement nor criticism by NIST. Any opinions, findings, and conclusions or recommendations expressed in the individual papers are the authors' own and do not necessarily reflect those of the sponsors.

The sponsorship of the U.S. Department of Defense is gratefully acknowledged, as is the tremendous work of the program committee and the track coordinators.

Ellen Voorhees
April 27, 2004

TREC 2003 Program Committee

Ellen Voorhees, NIST, chair
James Allan, University of Massachusetts at Amherst
Nick Belkin, Rutgers University
Chris Buckley, Sabir Research, Inc.
Jamie Callan, Carnegie Mellon University
Gordon Cormack, University of Waterloo
Susan Dumais, Microsoft
Fred Gey, University of California at Berkeley
Donna Harman, NIST
David Hawking, CSIRO
Bill Hersh, Oregon Health & Science University
James Mayfield, APL, Johns Hopkins University
John Prange, U.S. Department of Defense
Steve Robertson, Microsoft
Karen Sparck Jones, University of Cambridge, UK
Ross Wilkinson, CSIRO

TABLE OF CONTENTS

PAPERS

APPENDIX

INDEX OF TREC 2003 PAPERS BY TASK/TRACK

Genomics

HARD

Novelty

Question Answering

Indian Institute of Technology

ITC-Irst

Language Computer Corporation

LexiClone Inc.

Macquarie University

MIT Computer Science and Artificial Intelligence Laboratory

The MITRE Corporation

National Institute of Standards and Technology

National University of Singapore

National University of Singapore-MIT Alliance

Queens College, CUNY

SUNY Albany

University of Amsterdam

University of Wales, Bangor

University of Edinburgh

The University of Iowa

University of Limmerick

Robust

Web

Abstract

This report constitutes the proceedings of the 2003 edition of the Text REtrieval Conference, TREC 2003, held in Gaithersburg, Maryland, November 18–21, 2003. The conference was co-sponsored by the National Institute of Standards and Technology (NIST), the Advanced Research and Development Activity (ARDA), and the Defense Advanced Research Projects Agency (DARPA). Ninety-three groups including participants from 22 different countries were represented.

TREC 2003 is the latest in a series of workshops designed to foster research in text retrieval and related technologies. This year's conference consisted of six different tasks: web-based retrieval, novelty detection, question answering, retrieval in the genomics domain, improving the consistency of retrieval systems across queries, and improving retrieval effectiveness by focusing on user context.

The conference included paper sessions and discussion groups. This proceedings includes papers from the majority of participants (some groups did not submit papers), track reports that define the problem addressed by the track plus summarize the main track results, and tables of individual group results. The TREC 2003 proceedings web site also contains system descriptions that detail the timing and storage requirements of the different runs.

Overview of TREC 2003

Ellen M. Voorhees
National Institute of Standards and Technology
Gaithersburg, MD 20899

1 Introduction

The twelfth Text REtrieval Conference, TREC 2003, was held at the National Institute of Standards and Technology (NIST) November 18–21, 2003. The conference was co-sponsored by NIST, the US Department of Defense Advanced Research and Development Activity (ARDA), and the Defense Advanced Research Projects Agency (DARPA).

TREC 2003 is the latest in a series of workshops designed to foster research on technologies for information retrieval. The workshop series has four goals:

- to encourage retrieval research based on large test collections;
- to increase communication among industry, academia, and government by creating an open forum for the exchange of research ideas;
- to speed the transfer of technology from research labs into commercial products by demonstrating substantial improvements in retrieval methodologies on real-world problems; and
- to increase the availability of appropriate evaluation techniques for use by industry and academia, including development of new evaluation techniques more applicable to current systems.

TREC 2003 contained six areas of focus called "tracks". Three of the tracks, novelty, question answering, and web, were continuations of tracks that had run in earlier TRECs. The remaining three tracks, genomics, High-Accuracy-Retrieval-from-Documents (HARD), and robust retrieval, were new tracks in 2003. The retrieval tasks performed in each of the tracks are summarized in Section 3 below.

Table 1 lists the 93 groups that participated in TREC 2003. The participating groups come from 22 different countries and include academic, commercial, and government institutions.

This paper serves as an introduction to the research described in detail in the remainder of the volume. The next section provides a summary of the retrieval background knowledge that is assumed in the other papers. Section 3 presents a short description of each track—a more complete description of a track can be found in that track's overview paper in the proceedings. The final section looks forward to future TREC conferences.

2 Information Retrieval

Information retrieval is concerned with locating information that will satisfy a user's information need. Traditionally, the emphasis has been on text retrieval: providing access to natural language texts where the set of documents to be searched is large and topically diverse. There is increasing interest, however, in finding appropriate information regardless of the medium that happens to contain that information. Thus "document" can be interpreted as any unit of information such as a web page or a MEDLINE record.

The prototypical retrieval task is a researcher doing a literature search in a library. In this environment the retrieval system knows the set of documents to be searched (the library's holdings), but cannot anticipate the particular topic that will be investigated. We call this an *ad hoc* retrieval task, reflecting the arbitrary subject of the search and its short duration. Other examples of ad hoc searches are web surfers using Internet search engines, lawyers performing patent searches or looking for precedences in case law, and analysts searching archived news reports for particular events. A retrieval system's response to an ad hoc search is generally a list of documents ranked by decreasing similarity to the query.

Table 1: Organizations participating in TREC 2003

- Ajou University
- Axontologic, Inc.
- BBN
- California State University San Marcos
- Carnegie Mellon University (2 group)
- Center for Computing Science & U. Maryland
- Chinese Academy of Sciences (2 groups)
- Chinese Information Processing Center
- Clairvoyance Corporation
- CL Research
- Copernic Research
- CSIRO
- Dublin City University
- Erasmus MC
- Fondazione Ugo Bordoni
- Fraunhofer Institute (SCAI)
- Fudan University
- Hummingbird
- IBM Research, Haifa
- IBM TJ Watson Research Center (2 groups)
- Illinois Institute of Technology
- Indiana University, Bloomington
- Indian Institute of Technology Bombay
- IRIT/SIG
- ITC-irst
- Johns Hopkins University/APL
- Kasetsart University
- Korea University
- Language Computer Corporation
- Lehigh University
- LexiClone, Inc.
- Macquarie University
- Massachusetts Institute of Technology
- Meiji University
- Microsoft Research Asia
- Microsoft Research Ltd
- MITRE Corp.
- National Library of Medicine & U. Maryland
- National Research Council Canada
- National Taiwan University
- National University of Singapore (2 groups)
- New Mexico State University
- NTT Communication Science Laboratories
- OcE Technologies
- Oregon Health and Science University
- Queens College, CUNY
- RMIT University
- Rutgers University (3 groups)
- Saarland University (2 groups)
- Sabir Research, Inc.
- State University of New York at Buffalo
- StreamSage, Inc.
- Tarragon Consulting Corporation
- Tsinghua University (2 groups)
- Universitat Politecnica de Catalunya & Universitat de Girona
- Université de Neuchatel
- University Hospital of Geneva
- University of Alaska, Fairbanks
- University of Albany
- University of Amsterdam
- University of California, Berkeley
- University of Colorado & Columbia U.
- University of Edinburgh
- University of Edinburgh & Stanford U.
- University of Glasgow
- University of Helsinki
- University of Illinois at Chicago
- University of Illinois at Urbana-Champaign
- University of Iowa
- University of Limerick
- University of Maryland
- University of Maryland Baltimore County
- University of Massachusetts
- University of Melbourne
- University of Michigan
- University of Pisa
- University of Sheffield
- University of Southern California/ISI
- University of Sunderland
- University of Tampere
- University of Tokyo
- University of Wales, Bangor
- University of Waterloo (2 groups)
- Virginia Tech

A *known-item search* is similar to an ad hoc search but the target of the search is a particular document (or a small set of documents) that the searcher knows to exist in the collection and wants to find again. Once again, the retrieval system's response is usually a ranked list of documents, and the system is evaluated by the rank at which the target document is retrieved.

In a document routing or *filtering* task, the topic of interest is known and stable, but the document collection is constantly changing [1]. For example, an analyst who wishes to monitor a news feed for items on a particular subject requires a solution to a filtering task. The filtering task generally requires a retrieval system to make a binary decision whether to retrieve each document in the document stream as the system sees it. The retrieval system's response in the filtering task is therefore an unordered set of documents (accumulated over time) rather than a ranked list. TREC 2003 did not contain an explicit filtering task, though aspects of the filtering task were present in the novelty track tasks.

Information retrieval has traditionally focused on returning entire documents that contain answers to questions rather than returning the answers themselves. This emphasis is both a reflection of retrieval systems' heritage as library reference systems and an acknowledgement of the difficulty of question answering. However, for certain types of questions, users would much prefer the system to answer the question than be forced to wade through a list of documents looking for the specific answer. To encourage research on systems that return answers instead of document lists, TREC has had a question answering track since 1999. The information extraction task in the genomics track is similar to a question answering task in that the goal was to extract a short segment of a document as a description of a gene.

2.1 Test collections

Text retrieval has a long history of using retrieval experiments on test collections to advance the state of the art [4, 6, 10], and TREC continues this tradition. A test collection is an abstraction of an operational retrieval environment that provides a means for researchers to explore the relative benefits of different retrieval strategies in a laboratory setting. Test collections consist of three parts: a set of documents, a set of information needs (called *topics* in TREC), and *relevance judgments*, an indication of which documents should be retrieved in response to which topics.

2.1.1 Documents

The document set of a test collection should be a sample of the kinds of texts that will be encountered in the operational setting of interest. It is important that the document set reflect the diversity of subject matter, word choice, literary styles, document formats, etc. of the operational setting for the retrieval results to be representative of the performance in the real task. Frequently, this means the document set must be large. The primary TREC test collections contain about 2 gigabytes of text (between 500,000 and 1,000,000 documents). The document sets used in various tracks have been smaller and larger depending on the needs of the track and the availability of data.

The primary TREC document sets consist mostly of newspaper or newswire articles, though there are also some government documents (the *Federal Register*, patent applications) and computer science abstracts (*Computer Selects* by Ziff-Davis publishing) included. High-level structures within each document are tagged using SGML, and each document is assigned an unique identifier called the DOCNO. In keeping of the spirit of realism, the text was kept as close to the original as possible. No attempt was made to correct spelling errors, sentence fragments, strange formatting around tables, or similar faults.

2.1.2 Topics

TREC distinguishes between a statement of information need (the topic) and the data structure that is actually given to a retrieval system (the query). The TREC test collections provide topics to allow a wide range of query construction methods to be tested and also to include a clear statement of what criteria make a document relevant. The format of a topic statement has evolved since the earliest TRECs, but it has been stable since TREC-5 (1996). A topic statement generally consists of four sections: an identifier, a title, a description, and a narrative. An example topic taken from this year's robust retrieval track is shown in figure 1.

The different parts of the TREC topics allow researchers to investigate the effect of different query lengths on retrieval performance. For topics 301 and later, the "title" field was specially designed to allow experiments with very short queries; these title fields consist of up to three words that best describe the topic. The description field is a one sentence description of the topic area. The narrative gives a concise description of what makes a document relevant.

Participants are free to use any method they wish to create queries from the topic statements. TREC distinguishes among two major categories of query construction techniques, automatic methods and manual methods. An automatic method is a means of deriving a query from the topic statement with no manual intervention whatsoever; a manual method is anything else. The definition of manual query construction methods is very broad, ranging from simple

```
<num> Number:  602
<title> Czech, Slovak sovereignty

<desc> Description:
Retrieve information regarding the process by which the Czech and Slovak
republics of Czechoslovakia established separate sovereign countries.

<narr> Narrative:
A relevant document will provide specific dates and details regarding the
separation movement.  Documents relating to normal activities of the separate
nations, both internal and external are not relevant.
```

Figure 1: A sample TREC 2003 topic from the robust retrieval track.

tweaks to an automatically derived query, through manual construction of an initial query, to multiple query reformulations based on the document sets retrieved. Since these methods require radically different amounts of (human) effort, care must be taken when comparing manual results to ensure that the runs are truly comparable.

TREC topic statements are created by the same person who performs the relevance assessments for that topic (the *assessor*). Usually, each assessor comes to NIST with ideas for topics based on his or her own interests, and searches the document collection using NIST's PRISE system to estimate the likely number of relevant documents per candidate topic. The NIST TREC team selects the final set of topics from among these candidate topics based on the estimated number of relevant documents and balancing the load across assessors.

2.1.3 Relevance judgments

The relevance judgments are what turns a set of documents and topics into a test collection. Given a set of relevance judgments, the retrieval task is then to retrieve all of the relevant documents and none of the irrelevant documents. TREC almost always uses binary relevance judgments—either a document is relevant to the topic or it is not. To define relevance for the assessors, the assessors are told to assume that they are writing a report on the subject of the topic statement. If they would use any information contained in the document in the report, then the (entire) document should be marked relevant, otherwise it should be marked irrelevant. The assessors are instructed to judge a document as relevant regardless of the number of other documents that contain the same information.

Relevance is inherently subjective. Relevance judgments are known to differ across judges and for the same judge at different times [7]. Furthermore, a set of static, binary relevance judgments makes no provision for the fact that a real user's perception of relevance changes as he or she interacts with the retrieved documents. Despite the idiosyncratic nature of relevance, test collections are useful abstractions because the *comparative* effectiveness of different retrieval methods is stable in the face of changes to the relevance judgments [11].

The relevance judgments in early retrieval test collections were complete. That is, a relevance decision was made for every document in the collection for every topic. The size of the TREC document sets makes complete judgments utterly infeasible—with 800,000 documents, it would take over 6500 hours to judge the entire document set for one topic, assuming each document could be judged in just 30 seconds. Instead, TREC uses a technique called pooling [9] to create a subset of the documents (the "pool") to judge for a topic. Each document in the pool for a topic is judged for relevance by the topic author. Documents that are not in the pool are assumed to be irrelevant to that topic.

The judgment pools are created as follows. When participants submit their retrieval runs to NIST, they rank their runs in the order they prefer them to be judged. NIST chooses a number of runs to be merged into the pools, and selects that many runs from each participant respecting the preferred ordering. For each selected run, the top X documents (usually, $X = 100$) per topic are added to the topics' pools. Since the retrieval results are ranked by decreasing similarity to the query, the top documents are the documents most likely to be relevant to the topic. Many documents are retrieved in the top X for more than one run, so the pools are generally much smaller than the theoretical maximum of X × *the-number-of-selected-runs* documents (usually about 1/3 the maximum size).

The use of pooling to produce a test collection has been questioned because unjudged documents are assumed to be not relevant. Critics argue that evaluation scores for methods that did not contribute to the pools will be deflated relative to methods that did contribute because the non-contributors will have highly ranked unjudged documents.

Zobel demonstrated that the quality of the pools (the number and diversity of runs contributing to the pools and the depth to which those runs are judged) does affect the quality of the final collection [14]. He also found that the TREC collections were not biased against unjudged runs. In this test, he evaluated each run that contributed to the pools using both the official set of relevant documents published for that collection and the set of relevant documents produced by removing the relevant documents uniquely retrieved by the run being evaluated. For the TREC-5 ad hoc collection, he found that using the unique relevant documents increased a run's 11 point average precision score by an average of 0.5 %. The maximum increase for any run was 3.5 %. The average increase for the TREC-3 ad hoc collection was somewhat higher at 2.2 %.

A similar investigation of the TREC-8 ad hoc collection showed that every automatic run that had a mean average precision score of at least 0.1 had a percentage difference of less than 1 % between the scores with and without that group's uniquely retrieved relevant documents [13]. That investigation also showed that the quality of the pools is significantly enhanced by the presence of recall-oriented manual runs, an effect noted by the organizers of the NTCIR (NACSIS Test Collection for evaluation of Information Retrieval systems) workshop who performed their own manual runs to supplement their pools [5].

While the lack of any appreciable difference in the scores of submitted runs is not a guarantee that all relevant documents have been found, it is very strong evidence that the test collection is reliable for comparative evaluations of retrieval runs. The differences in scores resulting from incomplete pools observed here are smaller than the differences that result from using different relevance assessors [11].

2.2 Evaluation

Retrieval runs on a test collection can be evaluated in a number of ways. In TREC, all ad hoc tasks are evaluated using the `trec_eval` package written by Chris Buckley of Sabir Research [3]. This package reports about 85 different numbers for a run, including *recall* and *precision* at various cut-off levels plus single-valued summary measures that are derived from recall and precision. Precision is the proportion of retrieved documents that are relevant, while recall is the proportion of relevant documents that are retrieved. A cut-off level is a rank that defines the retrieved set; for example, a cut-off level of ten defines the retrieved set as the top ten documents in the ranked list. The `trec_eval` program reports the scores as averages over the set of topics where each topic is equally weighted. (The alternative is to weight each relevant document equally and thus give more weight to topics with more relevant documents. Evaluation of retrieval effectiveness historically weights topics equally since all users are assumed to be equally important.)

Precision reaches its maximal value of 1.0 when only relevant documents are retrieved, and recall reaches its maximal value (also 1.0) when all the relevant documents are retrieved. Note, however, that these theoretical maximum values are not obtainable as an average over a set of topics at a single cut-off level because different topics have different numbers of relevant documents. For example, a topic that has fewer than ten relevant documents will have a precision score less than one after ten documents are retrieved regardless of how the documents are ranked. Similarly, a topic with more than ten relevant documents must have a recall score less than one after ten documents are retrieved. At a single cut-off level, recall and precision reflect the same information, namely the number of relevant documents retrieved. At varying cut-off levels, recall and precision tend to be inversely related since retrieving more documents will usually increase recall while degrading precision and vice versa.

Of all the numbers reported by `trec_eval`, the recall-precision curve and mean (non-interpolated) average precision are the most commonly used measures to describe TREC retrieval results. A recall-precision curve plots precision as a function of recall. Since the actual recall values obtained for a topic depend on the number of relevant documents, the average recall-precision curve for a set of topics must be interpolated to a set of standard recall values. The particular interpolation method used is given in Appendix A, which also defines many of the other evaluation measures reported by `trec_eval`. Recall-precision graphs show the behavior of a retrieval run over the entire recall spectrum.

Mean average precision is the single-valued summary measure used when an entire graph is too cumbersome. The average precision for a single topic is the mean of the precision obtained after each relevant document is retrieved (using zero as the precision for relevant documents that are not retrieved). The mean average precision for a run consisting of multiple topics is the mean of the average precision scores of each of the individual topics in the run. The average precision measure has a recall component in that it reflects the performance of a retrieval run across all relevant documents, and a precision component in that it weights documents retrieved earlier more heavily than documents retrieved later. Geometrically, mean average precision is the area underneath a non-interpolated recall-precision curve.

Table 2: Number of participants per track and total number of distinct participants in each TREC

	TREC											
Track	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003
Ad Hoc	18	24	26	23	28	31	42	41	—	—	—	—
Routing	16	25	25	15	16	21	—	—	—	—	—	—
Interactive	—	—	3	11	2	9	8	7	6	6	6	—
Spanish	—	—	4	10	7	—	—	—	—	—	—	—
Confusion	—	—	—	4	5	—	—	—	—	—	—	—
DB Merging	—	—	—	3	3	—	—	—	—	—	—	—
Filtering	—	—	—	4	7	10	12	14	15	19	21	—
Chinese	—	—	—	—	9	12	—	—	—	—	—	—
NLP	—	—	—	—	4	2	—	—	—	—	—	—
Speech	—	—	—	—	—	13	10	10	3	—	—	—
Cross-Language	—	—	—	—	—	13	9	13	16	10	9	—
High Precision	—	—	—	—	—	5	4	—	—	—	—	—
VLC	—	—	—	—	—	—	7	6	—	—	—	—
Query	—	—	—	—	—	—	2	5	6	—	—	—
QA	—	—	—	—	—	—	—	20	28	36	34	33
Web	—	—	—	—	—	—	—	17	23	30	23	27
Video	—	—	—	—	—	—	—	—	—	12	19	[a]
Novelty	—	—	—	—	—	—	—	—	—	—	13	14
Genome	—	—	—	—	—	—	—	—	—	—	—	29
HARD	—	—	—	—	—	—	—	—	—	—	—	14
Robust	—	—	—	—	—	—	—	—	—	—	—	16
Total participants	22	31	33	36	38	51	56	66	69	87	93	93

[a] The video track was spun off as a separate evaluation effort in 2003.

As TREC has expanded into tasks other than the traditional ad hoc retrieval task, new evaluation measures have had to be devised. Indeed, developing an appropriate evaluation methodology for a new task is one of the primary goals of the TREC tracks. The details of the evaluation methodology used in a track are described in the track overview paper.

3 TREC 2003 Tracks

TREC's track structure was begun in TREC-3 (1994). The tracks serve several purposes. First, tracks act as incubators for new research areas: the first running of a track often defines what the problem *really* is, and a track creates the necessary infrastructure (test collections, evaluation methodology, etc.) to support research on its task. The tracks also demonstrate the robustness of core retrieval technology in that the same techniques are frequently appropriate for a variety of tasks. Finally, the tracks make TREC attractive to a broader community by providing tasks that match the research interests of more groups.

Table 2 lists the different tracks that were in each TREC, the number of groups that submitted runs to that track, and the total number of groups that participated in each TREC. The tasks within the tracks offered for a given TREC have diverged as TREC has progressed. This has helped fuel the growth in the number of participants, but has also created a smaller common base of experience among participants since each participant tends to submit runs to fewer tracks.

This section describes the tasks performed in the TREC 2003 tracks. See the track reports later in these proceedings for a more complete description of each track. Some of the descriptions given here are taken directly from the track overview papers.

3.1 The genomics track

The genomics track was a new track for TREC 2003. It is the first TREC track devoted to retrieval within a specific domain, and one of the goals of the track is to see how exploiting domain-specific information improves retrieval effectiveness. The track contained two tasks, the primary task that was an ad hoc retrieval task and the secondary task that was an information extraction task.

The scenario that motivated the primary task was that of a biological researcher or graduate student—that is, someone who already has considerable domain knowledge—confronted with the need to learn about a new gene very quickly. Since NIST assessors do not have the expertise to make judgments for the track, this first track made use of existing data that could serve as surrogate relevance judgments. The document collection consisted of approximately 526,000 MEDLINE records that were indcxed between April 1, 2002 and April 1, 2003, and were donated to the track by the U.S. National Library of Medicine. A topic consisted of a gene name and an organism, and was to be interpreted as a request for the basic biology of the gene and its protein products in the designated organism. This is the information given by the Gene Reference into Function (GeneRIF) data in the LocusLink database, a database of biological information created by the National Center for Biotechnology Information. The GeneRIF data were used as the relevance judgments for the track.

An analysis of the use of GeneRIF data showed that the vast majority of GeneRIF references pointed to relevant documents, but the GeneRIF references were incomplete (i.e., there were many more relevant documents than those included as GeneRIF references). Incompleteness is not necessarily a problem for retrieval system evaluation in that *unbiased* incomplete judgments allow for fair comparisons. Unfortunately, the GeneRIF data are not unbiased relevance judgments: the Berkeley group was able to build a classifier that could distinguish documents likely to be GeneRIFs [2]. The track will need to obtain relevance judgments in some other manner in future years.

Twenty five groups submitted 49 primary task runs to the genomics track. The best performing runs did use domain-specific knowledge as part of the retrieval. Exploiting the Medical Subject Headings (MeSH) and substance name fields of the MEDLINE records and filtering for species were particularly beneficial.

Part of the GeneRIF data is a text snippet that summarizes the main point of the referred to document with respect to the gene and organism. The secondary task was an information extraction task with the goal of creating this GeneRIF annotation automatically. The test set for the secondary task consisted of 139 GeneRIFs. Effectiveness was measured as a function of the overlap between the words nominated by the system and the actual GeneRIF text.

Fourteen groups submitted 24 secondary task runs. Since the actual GeneRIF text for many of the annotations is taken directly from the title of the target document, a baseline run consisting of the title of each target document was very hard to beat. The few runs that were able to beat the baseline used classifiers to rank sentences likely to contain GeneRIF text.

3.2 The HARD track

The HARD track was another new track in TREC 2003. HARD stands for High Accuracy Retrieval from Documents, and the goal of the track was to improve retrieval performance by targeting retrieval results to the specific user. Of course, to target retrieval results in such a manner the system needs to have some knowledge about the user. The HARD track provided this information in the form of biographical data about the user, information regarding the search context, and a statement of the expected type of a result.

The underlying task in the HARD track was an ad hoc retrieval task. However, for some topics the expected type of a result was passages rather than documents. Combining document and passage retrieval into a common evaluation methodology was one of the aspects explored in the track. Another aspect was the use of "clarifying forms" to gather information about the searcher. A clarification form was a single web page that solicited information about the query from the user. Any information from the user could be collected by the form subject to the constraints that the user would spend no more than 3 minutes filling out any one form and that the form had to be entirely self-contained HTML.

The document set used in the track was the set of documents from 1999 from the AQUAINT corpus plus a set of *Congressional Record* and *Federal Register* articles also from 1999. This collection consisted of approximately 372,000 documents and 1.7 GB of text. The topics were created by assessors from the Linguistics Data Consortium (LDC). The topics were patterned after standard TREC ad hoc topics, but included a set of metadata elements that described the searcher and/or the context of the search. For example, the PURPOSE metadata field explained why the user was searching for the information (its value could be one of background, details, answer, or any) and the FAMILIARITY field represented how familiar the searcher is with the general subject area of the topic (value between 1 and 5 with 1 meaning no prior knowledge and 5 meaning detailed knowledge of the subject; value could also be unknown). Biographical data such as the age, sex, and occupation of the searcher were also recorded.

Participants first ran their systems using just the standard TREC portions of the topic and no other information. They then repeated the search using any information from the metadata and/or their clarification forms. The goal was to see if the additional information helped systems to create a more effective retrieved set than the initial baseline result.

Relevance judgments were made at the LDC by the same assessor who created the topic. Two types of judgments were made, document-level judgments and passage-level judgments. Document-level judgments made without reference to the metadata are the same as standard TREC relevance judgments. Documents that are relevant in the standard TREC sense but do not meet the requirements specified by the metadata are called "SOFT-REL" documents, while relevant documents that also satisfy the metadata are called "HARD-REL". For document-level evaluation, HARD track runs were evaluated using the standard `trec_eval` measures, treating either both SOFT-REL and HARD-REL documents as relevant, or just HARD-REL documents as relevant.

Passage-level judgments were also made by the LDC assessor. If the metadata for a topic specified that the user wanted something smaller than a full document as a response, the assessor looked at each HARD-REL document in turn and marked the passages within the document that satisfied the topic. Passages were specified by an offset from the beginning of the document and a length. A single document could contain multiple relevant passages, but relevant passages never overlapped (overlapping passage were combined into a single passage if necessary). The relevance judgments were assumed to contain all the relevant passages for the topic.

The main measure used for passage-based evaluation was R-precision where R is the number of relevant passages for a topic. The passage-based evaluation treated all system responses as passages (i.e., a retrieved document was considered a single long passage). Precision was calculated on the basis of characters: the passage-based precision for a system response at rank R was the proportion of characters in the sum of the passages at ranks 1–R that were contained in a relevant passage.

Fourteen groups submitted 88 runs to the HARD track. For most groups, runs based on data obtained from clarification forms improved results as compared to the corresponding baseline run. Evaluation based on passages differs from that based on documents in that systems ranked differently when evaluated by passage-based R-precision than when evaluated by document-based R-precision.

3.3 The novelty track

The goal of the novelty track is to investigate systems' abilities to locate relevant and new (nonredundant) information within an ordered set of documents. This task models an application where the user is skimming a set of documents and the system highlights the new, on-topic information. The track was first introduced in TREC 2002, though this year's track had a number of significant changes from the initial track.

The basic task in the novelty track is as follows: given a topic and an ordered set of relevant documents segmented into sentences, return sentences that are both relevant to the topic and novel given what has already been seen. To accomplish this task, participants must first identify relevant sentences (a passage retrieval task) and then identify which sentences contain new information (a filtering task). To allow participants to focus on the filtering and passage retrieval aspects separately, four different tasks were included in the track where each task differed by the amount and kind of training data that was provided to the systems.

Fifty new topics were created for the novelty track by NIST assessors. Half of the topics focused on events and the other half focused on opinions about controversial subjects. For each topic, the assessor created a statement of information need and queried the document collection using the NIST PRISE search engine. The assessor selected 25 relevant documents and labeled the relevant and new sentences in each. The document collection used was the *AQUAINT Corpus of English News Text* assembled for the TREC 2002 question answering track. This corpus is comprised of documents from three different sources: the AP newswire from 1998–2000, the New York Times newswire from 1998–2000, and the (English portion of the) Xinhua News Agency from 1996–2000. There are approximately 1,033,000 documents and 3 gigabytes of text in the collection. The choice of the collection was motivated by a desire to increase the amount of redundancy in the relevant set as compared to last year's track. The 25 relevant documents for each topic were ordered chronologically for system processing, which is easily accomplished for a newswire collection.

The four tasks in the track allowed the participants to test their approaches to novelty detection using no, partial, or complete relevance information.

Task 1. Given the set of 25 relevant documents for a topic, identify all relevant and novel sentences.

Task 2. Given the relevant sentences in all 25 documents, identify all novel sentences.

Task 3. Given the relevant and novel sentences in the first 5 documents for the topic, find the relevant and novel sentences in the remaining 20 documents.

Task 4. Given the relevant sentences in all 25 documents, and the novel sentences in the first 5 documents, find the novel sentences in the remaining 20 documents.

Given the set of relevant and new sentences selected by the assessor who created the topic, the score for a novelty topic was computed as the F measure where sentence set recall and sentence set precision are equally weighted. Let M be the number of matched sentences, i.e., the number of sentences selected by both the assessor and the system, A be the number of sentences selected by the assessor, and S be the number of sentences selected by the system. Then sentence set recall is M/A and precision is M/S. The F score is then computed as $F = \frac{2*P*R}{(P+R)}$.

Fourteen groups submitted 179 runs to the novelty track. All but one group submitted a run for Task 1, and most groups tried all tasks. The results showed that for the basic task in which systems were given no sentence-level training data, the best systems were more effective than human performance. That is, a second assessor who selected relevant and novel sentences based on the topic statement generally scored lower when evaluated by the author's sentences than did the systems. More data is required to determine if systems are indeed performing at the level of a human at this task.

3.4 The question answering (QA) track

The question answering track addresses the problem of information overload by encouraging research into systems that return actual answers, as opposed to ranked lists of documents, in response to a question. The track has run since TREC-8 (1999), but has expanded in both scope and difficulty since the initial tracks. The TREC 2003 track contained two tasks, the main task and the passages task. Both tasks used the *AQUAINT Corpus of English News Text* used in the novelty track as the source of answers.

In TREC 2002, the QA task was defined such that systems were required to return exact answers, text strings consisting of a complete answer and nothing else. However, pinpointing the precise extent of an answer is a more difficult problem than finding a text segment that contains an answer, and there are applications of QA technology that do not require this extra step. The passages task provided a forum for research groups interested in these applications. A passages task run consisted of exactly one response for each of a set of 413 factoid questions. A response was either a document extract (not longer than 250 characters) believed to contain an answer to the question or the string "NIL" used to indicate the system's belief that there was no correct answer in the collection. Responses were judged as either correct, unsupported, or incorrect by human assessors. The final score for a passages task run was accuracy, the percentage of responses judged correct.

Twenty-one passages task runs from eleven different groups were submitted to the QA track. As determined by comparing mean accuracy scores, the passages task was not a noticeably easier task than the exact answer task. Accuracy scores for the passages task were in general no better than accuracy scores for the factoid component in the main task that required exact answers. Two of the three groups that submitted runs for both tasks had higher accuracy scores for the exact-answer case.

The main task was a combination task consisting of three different types of questions: factoids, lists, and definitions. The goal in combining the different question types into a single task was to increase the number of systems that attempted to answer the different question types. Each question was tagged as to its type in the test set. The three question types were evaluated separately, and the final score for a main task run was a combination of the scores for the three question types.

The factoid component of the main task was identical to the passages task except responses were required to be exact answers rather than document extracts that contained an answer. A fourth value for the judgments, inexact, was added to indicate when an otherwise correct response contained too much information. As in the passages task, the score for the factoid component of the main task was accuracy.

The list component of the main task required systems to assemble an answer from information located in multiple documents. In TREC, a list question asks for different instances of a particular kind of information to be retrieved, such as *List the names of chewing gums*. List questions can be thought of as a shorthand for asking the same factoid question multiple times; the set of answers that satisfy the factoid question is the appropriate response for the list question. Unlike the previous two times the list task was run in TREC, this year's list questions did not specify a target number of instances to return. Instead, systems were expected to return all of the correct, distinct answers contained in the document collection. There were 37 list questions in the main task test set.

Within the response returned for a single question by one system, assessors judged individual items as the factoid responses were judged. In addition, the assessor marked exactly one of a set of equivalent correct items as distinct. The final answer list for a question was created by the assessor based on the answers the assessor found during question development and the set of distinct, correct answers found by the systems. This final answer list was used to compute the instance recall and instance precision of a system's response. Instance recall is the fraction of answers on the final answer list that the system returned. The corresponding instance precision measure is the fraction of instances returned by the system that are on the final answer list. Instance recall and precision were combined using the F measure with recall and precision equally weighted ($F = \frac{2*IP*IR}{(IP+IR)}$) as the final score for a list question. The score for the entire list component of the main task was the average of the F scores over the 37 questions.

Definition questions are questions such as *Who is Colin Powell?* or *What is mold?*. This was the first time definition questions were evaluated in TREC. The evaluation was based on a small pilot evaluation of definition questions that was held as part of the ARDA AQUAINT program in the fall of 2002 [12]. Evaluating systems that answer definition questions is much more difficult than evaluating systems that answer factoid questions because it is no longer useful to judge a system response as simply right or wrong. Assigning partial credit to a system response requires some mechanism for matching the concepts in the desired response to the concepts present in the system's response. The issues are similar to those that arise in the evaluation of machine translation and automatic summarization.

The following scenario was assumed for definition questions:

> The questioner is an adult, a native speaker of English, and an "average" reader of US newspapers. In reading an article, the user has come across a term that they would like to find out more about. They may have some basic idea of what the term means either from the context of the article (for example, a bandicoot must be a type of animal) or basic background knowledge (Ulysses S. Grant was a US president). They are not experts in the domain of the target, and therefore are not seeking esoteric details (e.g., not a zoologist looking to distinguish the different species in genus Perameles).

The definition question test set contained 50 questions drawn from search engine logs; the set contained 30 questions for which the target was a (perhaps fictional) person, 10 questions for which the target was an organization, and 10 questions for which the target was some other thing.

A system response for a definition question was an unordered set of [*document-id*, *answer-string*] pairs. Each string was presumed to be a facet in the definition of the target. There were no limits placed on either the length of an individual answer string or on the number of pairs in the list, though systems were penalized for retrieving extraneous information.

Judging the quality of the systems' responses was done in two steps. In the first step, all of the answer-strings from all of the responses were presented to the assessor in a single (long) list. Using these responses and his own research done during question development, the assessor first created a list of "information nuggets" about the target. An information nugget was defined as a fact for which the assessor could make a binary decision as to whether a response contained the nugget. At the end of this step, the assessor decided which nuggets were vital—nuggets that must appear in a definition for that definition to be good. The assessor went on to the second step once the nugget list was created. In this step the assessor went through each of the system responses in turn and marked where each nugget appeared in the response. If a system returned a particular nugget more than once, it was marked only once. A single item in a system's response may match zero, one, or more than one nuggets.

Given the judgments as described above, it is straightforward to compute the nugget recall of a response: it is simply the ratio between the number of correctly retrieved nuggets to the number of nuggets on the assessor's list. But the corresponding measure of nugget precision, the ratio between the number of nuggets correctly retrieved to the total number of nuggets retrieved, is problematic since the correct value for the denominator is unknown. A trial evaluation prior to the pilot showed that assessors found enumerating *all* concepts represented in a response to be so difficult as to be unworkable. Instead, we used length as a crude approximation to precision. The length-based measure captures the intuition that users would prefer the shorter of two definitions that contain the same concepts. The final score for a definition question was the F measure where nugget recall was given five times as much emphasis as nugget precision. The score for the definition component of the main task was the average F over the 50 definition questions.

The final score for a main task run was computed as a weighted average of the three component scores:

$$\text{FinalScore} = 1/2 * \text{FactoidScore} + 1/4 * \text{ListScore} + 1/4 * \text{DefScore}.$$

Since each of the component scores ranges between 0 and 1, the final score is also in that range. The final score emphasizes the factoid component, which represented the largest number of questions and is the task people are most familiar with. The weight for the other components was made large enough to encourage participation in those subtasks.

Fifty-four main task runs from 25 different groups were submitted to the track. The results demonstrate that the list and definition tasks are challenging for systems, and that they present challenges for evaluation as well. For the definition task, the difference in evaluation scores required to have confidence in the conclusion that one run is better than another is large relative to the observed scores. This results in a fairly insensitive test since many comparisons are inconclusive. The list task scores are much more stable, but the stability is due in large part to the fact that the scores for the list task are very low.

3.5 The robust retrieval track

The robust retrieval track was another new track in TREC 2003. The goal of the track was to focus research on improving the consistency of retrieval technology by concentrating on poorly performing topics. In addition, the track brought back a classic ad hoc retrieval task to TREC.

The topic set used in the track was a set of 100 topics. Fifty of the topics were new, created by NIST assessors using the standard topic development procedure. The other 50 topics were old topics first used in the ad hoc tasks of TRECs 6–8. NIST selected these 50 topics based on the median mean average precision (MAP) score when the topic was first used: the 50 topics all had low median MAP scores with at least one run that did much better than the median to rule out flawed topics.

Since 50 of the topics were from previous TRECs, the track used the same document set as those years, namely the set of documents on TREC disks 4 and 5 minus the *Congressional Record* documents. No new relevance judgments were made for these topics. The 50 new topics were judged using pools created from all runs using a depth of 125 documents per topic per run. Evaluation was performed using `trec_eval` on each subset of the topics and on the combined set of 100 topics. Two new measures that focused on the poorly-performing topics were also introduced. The first of these measures was the percentage of topics that returned no relevant documents in the top ten documents retrieved. The second measure is a much more sensitive, but far less intuitive measure. If there are a total of Q topics in the test set, plot the MAP score computed over a system's worst X topics (as measured by average precision) against X for $X = 1 \ldots Q/4$. The measure is the area underneath this curve. Note that since the measure is computed over the individual system's worst X topics, different systems' scores are computed over a different set of topics in general.

The robust track received a total of 78 runs from 16 participants. All of the runs submitted to the track were automatic runs. The results of the track provide strong confirmation that average values of the traditional effectiveness measures do not reflect poorly performing topics. The new measures do emphasize systems' worst topics, but because they are defined over a subset of the topics, they are much less stable than traditional measures for a given test set size.

3.6 The web track

The goal in the web track is to investigate retrieval behavior when the collection to be searched is a large hyperlinked structure such as the World Wide Web. This year's track focused on finding homepages, the main entry pages to sites. There were two non-interactive tasks and one interactive task in the track.

All tasks used the .GOV collection created for the TREC 2002 web track and distributed by CSIRO (see `http://www.ted.cmis.csiro.au/TRECWeb/govinfo.html`). This collection is based on a January, 2002 crawl of .gov web sites. The documents in the collection contain both page content and the information returned by the http daemon; text extracted from the non-html pages is also included in the collection.

The two non-interactive tasks in the track were a topic distillation task and a navigational task known as the home/named page finding task. In the topic distillation task, the systems were given a broad information request and were to return a list of relevant home pages. A relevant home page was defined as the entry page to a credible site that is principally devoted to the topic. The emphasis was on returning home pages rather than pages themselves since a result list of homepages provides a better overview of the coverage of a topic in the collection. The primary effectiveness measure used was R-precision (precision after R relevant documents are retrieved) since many of the topics within the set of 50 test topics had fewer than 10 relevant home pages.

The navigational task was a known-item search task. The queries consisted of a very short description of a page such as "Tennessee Valley Authority", and the systems were to return the target page (in this case, `www.tva.gov`). The test set consisted of 300 queries, half of which had a home page as the target page. Effectiveness was measured by the mean over the 300 topics of the reciprocal of the rank at which the target page was returned.

Twenty-seven groups submitted a total of 166 runs to the non-interactive part of the web track. Ninety-three of the runs were topic distillation runs and 73 of the runs were navigational task runs. Results from both tasks showed that exploiting anchor text is an important element of effective homepage finding. Methods that exploited URL syntax and link structure had more mixed results, especially for the navigational task. Attempts to differentiate processing for named pages vs. homepages in the navigational task did not improve effectiveness.

The interactive task within the web track explored the role of the human searcher in the topic distillation task. Eight of the topics used in the non-interactive version of the task were expanded to include a search scenario to provide context for the searcher. The searchers produced a list of home pages for the topic which were then judged by the assessors along four dimensions: relevance, depth, coverage, and repetition. Each dimension was judged using a 5-point Likert scale.

Two groups participated in the task. Both groups explored whether a more structured presentation of the search results (rather than a simple ranked list) would better support a searcher in the topic distillation task. The searchers liked the structured result format better, and were somewhat more efficient with it, but there were no significant differences between the list and structured formats in the quality of the homepage lists the searchers assembled.

4 The Future

Since three of the six tracks offered in TREC 2003 were new tracks, the set of tracks to be offered in TREC 2004 will be little changed from this year. Each of the six tracks will continue in TREC 2004. In addition, a new track, currently known as the terabyte track, will be added. The main objective in the terabyte track will be to investigate ad hoc evaluation methodologies for terabyte scale collections [8]. Of course, the track will also offer participants the opportunity to see how well their retrieval methods scale to significantly larger collections.

Acknowledgements

Special thanks to the track coordinators who make the variety of different tasks addressed in TREC possible.

References

[1] Nicholas J. Belkin and W. Bruce Croft. Information filtering and information retrieval: Two sides of the same coin? *Communications of the ACM*, 35(12):29–38, December 1992.

[2] G. Bhalotia, P.I. Nakov, A.S. Schwartz, and M.A. Hearst. BioText team report for the TREC 2003 genomics track. In *Proceedings of the Twelfth Text REtrievalxi Conference (TREC 2003)*, 2004.

[3] Chris Buckley. trec_eval IR evaluation package. Available from `ftp://ftp.cs.cornell.edu/pub/smart`.

[4] C. W. Cleverdon, J. Mills, and E. M. Keen. Factors determining the performance of indexing systems. Two volumes, Cranfield, England, 1968.

[5] Noriko Kando, Kazuko Kuriyama, Toshihiko Nozue, Koji Eguchi, Hiroyuki Kato, and Souichiro Hidaka. Overview of IR tasks at the first NTCIR workshop. In *Proceedings of the First NTCIR Workshop on Research in Japanese Text Retrieval and Term Recognition*, pages 11–44, 1999.

[6] G. Salton, editor. *The SMART Retrieval System: Experiments in Automatic Document Processing.* Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1971.

[7] Linda Schamber. Relevance and information behavior. *Annual Review of Information Science and Technology*, 29:3–48, 1994.

[8] Ian Soboroff, Ellen Voorhees, and Nick Craswell. Summary of the SIGIR 2003 workshop on defining evaluation methodologies for terabyte-scale test collections. *SIGIR Forum*, 37(2):55–58, 2003.

[9] K. Sparck Jones and C. van Rijsbergen. Report on the need for and provision of an "ideal" information retrieval test collection. British Library Research and Development Report 5266, Computer Laboratory, University of Cambridge, 1975.

[10] Karen Sparck Jones. *Information Retrieval Experiment.* Butterworths, London, 1981.

[11] Ellen M. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. *Information Processing and Management*, 36:697–716, 2000.

[12] Ellen M. Voorhees. Evaluating answers to definition questions. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003), Volume 2*, pages 109–111, May 2003.

[13] Ellen M. Voorhees and Donna Harman. Overview of the eighth Text REtrieval Conference (TREC-8). In E.M. Voorhees and D.K. Harman, editors, *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, pages 1–24, 2000. NIST Special Publication 500-246. Electronic version available at `http://trec.nist.gov/pubs.html`.

[14] Justin Zobel. How reliable are the results of large-scale information retrieval experiments? In W. Bruce Croft, Alistair Moffat, C.J. van Rijsbergen, Ross Wilkinson, and Justin Zobel, editors, *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 307–314, Melbourne, Australia, August 1998. ACM Press, New York.

TREC Genomics Track Overview

William Hersh, Ravi Teja Bhupatiraju
Oregon Health & Science University
Portland, OR, USA
{hersh,bhupatir}@ohsu.edu

The first year of TREC Genomics Track featured two tasks: ad hoc retrieval and information extraction. Both tasks centered around the Gene Reference into Function (GeneRIF) resource of the National Library of Medicine, which was used as both pseudorelevance judgments for ad hoc document retrieval as well as target text for information extraction. The track attracted 29 groups who participated in one or both tasks.

The growing amount of scientific discovery in genomics and related biomedical disciplines has led to a corresponding growth in the amount of on-line data and information. A growing challenge for biomedical researchers is how to access and manage this ever-increasing quantity of information. This situation presents opportunities and challenges for the information retrieval (IR) field. IR has historically focused on document retrieval, but the field has expanded in recent years with the growth of new information needs (e.g., question-answering, cross-lingual), data types (e.g., video) and platforms (e.g., the Web). This paper describes the events leading up to the first year of TREC Genomics Track, the first year's results, and future directions for subsequent years.

Genomics and Information Resources

The field of *genomics* is concerned with the *genome*, which is usually defined as the genetic material of living organisms. Its research focuses on the *central dogma* of biology: deoxyribonucleic acid (DNA) is transcribed into ribonucleic acid (RNA), which serves to translate the nucleotide sequences of DNA into proteins. The latter are responsible for functions in living organisms and the collection of all proteins in is increasingly called the *proteome.* With the advent of new technologies for sequencing the genome and proteome, along with other tools for identifying the expression of genes, structures of proteins, and so forth, the face of biological research has become increasingly data-intensive, creating great challenges for scientists who formerly dealt with relatively modest amounts of data in their research.

The growth of biological data has resulted in a correspondingly large increase in scientific knowledge in what biologists sometimes call the *bibliome* or literature of biology. A great deal of biological information resources have become available in recent years (Baxevanis, 2003). Probably the most important of these are from the National Center for Biotechnology Information (NCBI, www.ncbi.nlm.nih.gov), a division of the National Library of Medicine (NLM, www.nlm.nih.gov) that maintains most of the NLM's genomics-related databases (Wheeler, Church et al., 2003).

Key features of NCBI resources include linkage and annotation. Linkage among resources allows the user to explore different types of knowledge across resources. For example, the original research documenting the discovery of a gene function appears in MEDLINE (the bibliographic database of medical literature, accessed by PubMed and other systems), with links to the nucleotide sequence in GenBank, the structure of the protein in the Molecular Modeling Database (MMDB), and an overview of the diseases it may cause in humans in the Online Mendelian Inheritance in Man (OMIM) textbook. LocusLink (http://www.ncbi.nlm.nih.gov/LocusLink/) serves as a switchboard to integrate these resources together as well as provide annotation of the gene's function using the widely accepted GeneOntology (GO, www.geneontology.org). Genes with known locations are also into maps which denote their locations of genes on chromosomes. PubMed also provides linkages to full-text journal articles on the Web sites of publishers.

Additional genomics resources exist beyond the NCBI. Of particular note are the model organism genome databases, such as:

Mouse Genome Informatics - www.informatics.jax.org
Saccharomyces Genome Database - http://genome-www.stanford.edu/Saccharomyces/
Flybase Database of the Drosophilia Genome - flybase.bio.indiana.edu

As with the NCBI resources, these resources provide rich linkage and annotation.

Preliminary Activities of the Genomics Track

The genesis of the Genomics Track came in 2001, when interest was expressed by the TREC Program Committee for moving into new types of data, including types which were more structured than the usual document collections from newswire. A number of discussions among interested people led to the first activity of the track, which was a Web survey soliciting ideas that took place in early 2002. Over 80 individuals responded, revealing diverse interests in IR and information extraction (IE) tasks, but clustered around three areas: extraction of knowledge from databases, automated or semi-automated annotation of genes and proteins, and retrieval across heterogeneous databases. Respondents from the IR community expressed the most enthusiasm for the latter task. All respondents were interested in using public databases, mainly those from the NCBI.

Activity also consisted of three workshops, held at the Joint Conference on Digital Libraries (JCDL) 2002, TREC 2002, and the Pacific Symposium on Biocomputing (PSB) 2003. These workshops led to the plan for the first year of the track, which was hoped would take place in 2003. A significant constraint on the track was lack of resources; i.e., NIST did not have the in-house resources to obtain documents or perform relevance judgments in this domain. As such, the choice of tasks, queries, documents, etc. would need to be guided by the availability of existing resources, including something that could be used to serve as proxies for relevance judgments. Fortunately, the track identified a valuable resource from NCBI: Gene Reference into Function (GeneRIF) data in the LocusLink database. Each GeneRIF entry consists of a statement about the function of a gene along with a pointer to the MEDLINE reference for the article that discovered that data (see Table 1).

A preliminary analysis in January, 2003 identified nearly 7,000 genes with one or more GeneRIFs. There were 246 genes with 10 or more GeneRIFs. As past IR work has shown that the "stability" of recall-precision numbers in batch retrieval experiments requires at least 25 and ideally 50 topics (Buckley and Voorhees, 2000), this would provide ample data for experiments.

The workshops also resulted in a use case guiding the first year's experiments, which was the biological researcher or graduate student (i.e., someone who already has considerable general domain knowledge) who is confronted with the need to learn about a new scientific area quickly. Perhaps he or she has performed a gene expression array experiment identifying genes not previously known to be involved in the biological process he or she has been investigating. Now he or she must get up to speed quickly with knowledge of these genes.

The GeneRIFs allowed the track to pursue two tasks satisfying the interests of a larger audience: an ad hoc retrieval task and an IE task. The ad hoc task was designated the primary task and was structured very similar to most previous TREC ad hoc tasks (e.g., ad hoc tasks of TREC 1-10, Web track, etc.). It was recognized that GeneRIFs could serve as pseudorelevance judgments, even though it was suspected (and later verified, see below) that they were incomplete from that standpoint.

GeneRIFs could also be used as targets for IE, and this was chosen to be the secondary task. The secondary task was more exploratory in nature: extracting the GeneRIF statement from the MEDLINE record or the article proper.

Table 1 - GeneRIFs for the gene *Interleukin 3 (colony-stimulating factor, multiple)* from LocusLink. The PubMed ID and citation are from the MEDLINE database.

LocusLink ID	PubMed ID	Citation	GeneRIF text
3562	11763346	Antisense Nucleic Acid Drug Dev 2001 Oct;11(5):289-300.	inhibition of signaling by antisense oligodeoxynucleotides targeting the common beta chain of receptors
3562	11861295	Blood 2002 Mar 1;99(5):1776-84.	ectopically expressed in myeloid leukemic cells with t(5;12)(q31;p13), suggesting that expression of IL3 was deregulated by the translocation, indicating a variant leukemogenic mechanism for translocations involving the 5' end of ETV6
3562	12002675	Folia Biol (Praha) 2002;48(2):51-7.	Antiapoptotic cytokine IL-3 + SCF + FLT3L influence on proliferation of gamma-irradiated AC133+/CD34+ progenitor cells.
3562	12055233	J Immunol 2002 Jun 15;168(12):6199-207.	Monocytes cultured in the presence of IL-3 (plus IL-4) differentiate into dendritic cells that produce less IL-12 and shift T helper (Th) cell responses toward a Th2 cytokine pattern.
3562	12093816	J Biol Chem 2002 Oct 11;277(41):38764-71.	Data suggest that increased activity of mutated interleukin 3 is due to a change from a rare ligand to a common one, allowing the increase in IL-3-dependent signaling.
3562	12135758	FEBS Lett 2002 Jul 31;524(1-3):149-53.	role in potentiating hematopoietic cell migration
3562	12165512	J Immunol 2002 Aug 15;169(4):1876-86.	The IL-3 gene is regulated by two enhancers that have distinct but overlapping tissue specificities.

Research groups were charged with maximizing the lexical overlap of the GeneRIF statement as measured by the Dice coefficient and some derivatives of it. Full-text articles were provided through Highwire Press (www.highwire.org), which publishes the full text of over 400 biomedical journals. Highwire does not own the copyrights to the journals, but has served as an intermediary to help various IR and other research groups obtain journal data for their work. Highwire facilitated interaction with publishers to obtain content for experiments.

Primary task

As noted above, the primary task for 2003 consisted of ad hoc document retrieval. This type of task requires a document collection, topics, and relevance judgments.

Documents

The document collection consisted of 525,938 MEDLINE records where indexing was completed between 4/1/2002 and 4/1/2003. The MEDLINE records were provided in the standard NLM MEDLINE format (although an XML version was available). The fields were indicated by their 2-3 letter abbreviation. The fields likely to be most important to track participants were: PubMed Unique Identifier (PMID), title (TI), abstract (AB), and MeSH headings (MH). A description of all the fields in a MEDLINE record can be found in the PubMed help file at: http://www.ncbi.nlm.nih.gov/entrez/query/static/help/pmhelp.html#MEDLINEDisplayFormat

Topics

The topics consisted of gene names, with the specific task being deriving from the definition of a GeneRIF (Mitchell, Aronson et al., 2003):

> For gene X, find all MEDLINE references that focus on the basic biology of the gene or its protein products from the designated organism. Basic biology includes isolation, structure, genetics and function of genes/proteins in normal and disease states.

We distributed training and test topic sets of 50 genes each. The training data were distributed first, allowing groups to get an idea of what the data in the track were like and tune their systems. The test data were the topics for the official runs in the track. For each set of 50 topics, we randomly chose gene names that were distributed across the spectrum of organisms, the number of GeneRIFs (many to few), name types (see below), and whether or not the gene names were Medical Subject Heading (MeSH) indexing terms. We also distributed the GeneRIFs for all of the training topics to allow groups to see the targets of their systems' retrieval efforts. GeneRIFs for the test topics were not distributed until after the deadline for the submission of official results.

Gene Names

LocusLink also contains a variety of names for each gene. Many researchers have lamented the pervasiveness of synonymy and polysemy in gene naming (O'Neill, 2003). Table 2 shows the multiple gene names for the *Interleukin 3 (colony-stimulating factor, multiple)* gene whose GeneRIFs were shown in Table 1.

Although many genes are present across multiple organisms (e.g., humans, mice, and rats produce and utilize insulin), LocusLink maintains a separate record for each gene in a given species. We chose to limit genes to four possible organisms:

Homo sapiens - human
Mus musculus - mouse
Rattus norvegicus - rat
Drosophila melanogaster - fruit fly

Relevance Judgments

For reasons described above, the relevance judgments for the 2003 track consisted of GeneRIFs. Track participants were not allowed to use GeneRIF data to augment their queries. While we recognized that GeneRIFs were, like the rest of LocusLink, publicly available, we worked on the *honor system* of research groups not using GeneRIF data.

Table 2 - Names for the gene *Interleukin 3 (colony-stimulating factor, multiple)* from LocusLink.

LocusLink ID	Organism	Gene name type	Gene name
3562	Homo sapiens	OFFICIAL_GENE_NAME	interleukin 3 (colony-stimulating factor, multiple)
3562	Homo sapiens	OFFICIAL_SYMBOL	IL3
3562	Homo sapiens	ALIAS_SYMBOL	IL-3
3562	Homo sapiens	ALIAS_SYMBOL	MCGF
3562	Homo sapiens	ALIAS_SYMBOL	MULTI-CSF
3562	Homo sapiens	PREFERRED_PRODUCT	interleukin 3 precursor
3562	Homo sapiens	PRODUCT	interleukin 3 precursor
3562	Homo sapiens	ALIAS_PROT	mast-cell growth factor
3562	Homo sapiens	ALIAS_PROT	P-cell stimulating factor
3562	Homo sapiens	ALIAS_PROT	hematopoietic growth factor
3562	Homo sapiens	ALIAS_PROT	multilineage-colony-stimulating factor

We calculated recall and precision in the classic IR way, using the preferred TREC statistic of *mean average precision* (average precision at each point a relevant document is retrieved, also called MAP). This was done in the standard TREC fashion of participants submitting their results in the format for input to the trec_eval program. (Groups were directed to the repository of code for trec_eval at ftp://ftp.cs.cornell.edu/pub/smart/. There are several versions of trec_eval, which differ mainly in the additional statistics they calculate in their output.) The trec_eval program requires two files for input. One file is the topic-document output, sorted by each topic and then subsorted by the order of the IR system output for a given topic. The second file required for trec_eval is the relevance judgments, which are called *qrels* in TREC jargon. (More information about qrels can be found at http://trec.nist.gov/data/qrels_eng/).

Training Data Runs

The training data runs not only allowed groups to become familiar with the data, but also allowed for discovery of some "quirks" with the training data qrels:

A number of qrels represented documents not present in the document collection.

Three topics had no qrels in the document collection: 21, 35, and 49.

This enabled us to make sure these problems did not exist before finalizing the test data. Due to the unstable nature of recall-precision for topics with very small numbers of qrels, we made the decision to use only gene names that had a minimum of three qrels in the collection for the test topics.

We also performed an analysis of relevance for 10 queries from the training data. This was done by manually judging relevance for all GeneRIFs as well as all other documents in the top 20 retrieved by the best OHSU training data run (or all documents if less than 20 retrieved). The relevance judgments were performed by an individual with a medical background enrolled in the OHSU medical informatics graduate program who had taken a course in IR. This analysis validated our *a priori* assumptions that all articles pointed to by GeneRIFs were relevant in the classic IR sense and that there were many "false negatives" (i.e., articles that were relevant but did not have a GeneRIF designation). We also discovered another phenomenon: documents that were relevant but for the gene in a species other than that designated by the LocusLink record. In the analysis of the top 20 ranking documents retrieved from the best OHSU training run, we found that:

35.0% of documents retrieved were not relevant

10.5% of documents retrieved were relevant and were GeneRIFs

42.5% of documents retrieved were relevant and not GeneRIFs

12.5% of documents retrieved were relevant and from a different species

Official Runs

A total of 25 groups submitted 49 official runs for scoring. Table 3 lists the results for each run, consisting of the run tag, whether the run was purely automatic or used some manual processing, and three results: MAP, number relevant at 10 documents retrieved, and number relevant at 20 documents retrieved. The final two rows of the table show the mean and median for each result. An analysis of variance with posthoc pairwise comparisons will be reported in a subsequent paper.

The run with the highest MAP was NLMUMDSE, with a mean MAP of 0.4165. This and the next-highest performing run came from an NLM-based research group (not affiliated with the operations of the library) (Kayaalp, Aronson et al., 2003). They used a search engine developed for the ClinicalTrials.gov database. They achieved good results from:

Identifying species through use of MeSH terms and other simple rules

Recognizing terms or their synonyms or lexical variants in non-text fields, in particular MeSH and substance name (RN)

Using additional general key words, such as genetics, sequence, etc.

A second run with a system that added MeSH terms and other controlled vocabulary along with collocation networks did not improve performance with this data.

Runs from UC Berkeley (Bhalotia, Nakov et al., 2003) and the National Research Council of Canada (deBruin and Martin, 2003) ranked next highest. Both of their approaches benefited from rules for recognizing gene name synonyms and filtering for organism name. The UC Berkeley approach included a machine learning algorithm to classify documents likely to have GeneRIFs assigned to them and document ranking based on gene name occurrence rules. The NRC approach added unsupervised relevance feedback to find additional relevant articles and ranking based on TF*IDF query term weighting.

The Waterloo group also did well, using what could be best described as "database-specific" (as opposed to "domain-specific") techniques that included (Yeung, Clarke et al., 2003):

Query formulation using fusion of Okapi weighting plus handling of punctuation plus pluralization as well as gene name bigrams

Recognition of gene name in substance name field

Query expansion on relevant substance names

It was apparent from the above groups that searching in the MeSH and substance name fields, along with filtering for species, accounted for the best performance. At least two other groups also found substantial benefit from organism name filtering, the National Research Council of Canada (deBruin and Martin, 2003) and Tarragon Consulting (Tong, Quackenbush et al., 2003). No groups attempted to model gene "function" in the sense of the GeneRIFs.

Approaches that used standard IR techniques shown to work best with traditional TREC data (i.e., newswire) performed less well. The Neuchatel group tried many permutations of advanced features from SMART (Savoy, Rasolofo et al., 2003). They obtained their best results with Okapi weighting, pivoted normalization, and query expansion, but they fell near the median of all groups. Likewise, the Illinois-UC group used a variant of language modeling and also performed near the median (Zhai, Tao et al., 2003).

As with many TREC experiments over the years, variation across topics and even within them across groups was substantial. Table 4 shows the variation of results for topic 35, the gene whose GeneRIFs and gene names were displayed in earlier tables.

We also carried out a relevance similar to that described for the training data with all 50 test topics. Again, all GeneRIFs as well as the top 20 documents retrieved (or all documents if less than 20 retrieved) in the best OHSU run (ohsuboost) were analyzed by the individual described above. Once again, we found that virtually all GeneRIFs were relevant (551/566, 97.3%), although a small number were relevant in other species (13/566, 2.3%) or indeterminate because the abstract was not accessible in MEDLINE to allow judgment (2/566, 0.4%). However, we also found again that substantial numbers of documents deemed relevant by our judge were not designated as GeneRIFs. Table 5 summarizes the analysis of retrieved documents.

Secondary Task

There is much interest in the bioinformatics community in IE. This comes in part from the desire to allow scientists to learn about new topics as quickly as they can, preferably without having to read and synthesize many papers. The specific task was to reproduce the GeneRIF annotation. As this task was more exploratory in nature and had an uncertain "gold standard," groups were instructed to attempt the task and compare their methods and results. Because of the exploratory nature of the secondary task, we did not provide any training data.

Table 3 - Official primary task runs, sorted by mean average precision.

Run Tag	Run Type	Mean Average Precision	Relevant @ 10 documents retrieved	Relevant @ 20 documents retrieved
NLMUMDSE	automatic	0.4165	3.16	4.84
NLMUMDSRB	manual	0.3994	3.20	4.56
nrc1	automatic	0.3941	2.94	4.38
biotext1	automatic	0.3912	3.06	4.46
nrc2	automatic	0.3771	2.76	4.36
biotext0	automatic	0.3753	2.92	4.30
uwmtg03btrf	automatic	0.3534	2.28	3.68
uwmtg03atrf	automatic	0.3479	2.48	4.00
axon2	automatic	0.3173	2.50	3.86
axon1	automatic	0.3118	2.40	3.78
CSUSM2	automatic	0.3079	2.68	3.76
edstanrecall	automatic	0.3015	2.60	3.74
edstanprec	automatic	0.2984	2.60	3.74
KUBIOIRNE	automatic	0.2980	2.32	3.42
KUBIOIRRAW	automatic	0.2937	2.24	3.38
CSUSM1	automatic	0.2859	2.56	3.52
tgnBaseline	manual	0.2837	2.18	3.52
IBMbt1	automatic	0.2823	2.26	3.32
tgnVariant1	manual	0.2791	2.22	3.56
aoyama	automatic	0.2277	1.90	2.92
aoyama2	automatic	0.2276	1.92	2.92
IBMbt2	automatic	0.2259	1.80	2.84
UIowaGN1	automatic	0.2064	2.02	3.40
UIUC03Gb	automatic	0.2001	1.50	2.44
SCAI	automatic	0.1960	1.42	2.60
utafil	manual	0.1931	1.48	2.40
utaband	manual	0.1927	1.54	2.62
UIUC03Ga	automatic	0.1925	1.58	2.32
UBgenomeBGNE	automatic	0.1867	1.44	2.14
UniNEg1	automatic	0.1852	1.28	2.12
humG03ns	automatic	0.1847	1.58	2.46
UniNEg2	automatic	0.1802	1.30	2.10
ErasmusMC3	automatic	0.1770	1.36	2.28
ErasmusMC2	automatic	0.1754	1.38	2.32
humG03ns5	automatic	0.1753	1.48	2.34
ohsuboost	automatic	0.1747	1.58	2.36
DcuMesh1	automatic	0.1669	1.36	2.08
DcuMesh2	automatic	0.1667	1.36	1.96
dayrutgers1	automatic	0.1652	1.34	2.40
dayrutgers2	automatic	0.1636	1.32	2.06
UniNEg5	automatic	0.1635	1.28	2.00
UniNEg4	automatic	0.1623	1.30	2.10
balsc3	automatic	0.1528	1.44	2.10
UBgenomRFB1	automatic	0.1511	1.16	1.84
UBgenomRFB2	automatic	0.1493	1.12	1.80
balsc2	automatic	0.1481	1.36	2.34
StreamSage3	automatic	0.0508	0.70	0.80
StreamSage4	automatic	0.0508	0.70	0.80
vvP05mil3	automatic	0.0271	0.22	0.60
Mean		0.2313	1.85	2.85
Median		0.1960	1.58	2.60

Table 4 - Best, median, and worst scores for the topic, *Interleukin 3 (colony-stimulating factor, multiple).*

Score	Best	Median	Worst
MAP	0.4136	0.0647	0
Relevant @ 10	4	1	0
Relevant @ 20	6	1	0

Table 5 - Classification of relevance of retrieved documents from best OHSU run organized by whether document, for a given query, is or is not a GeneRIF and is relevant, not relevant, relevant in another species, or unable to be judged due to no abstract in MEDLINE record.

GeneRIF	And	Number	Percentage
GeneRIF	Relevant	117	12.7%
GeneRIF	Not relevant	0	0.0%
GeneRIF	Relevant in another species	2	0.2%
GeneRIF	No abstract (unable to judge)	0	0.0%
Not a GeneRIF	Relevant	386	41.8%
Not a GeneRIF	Not relevant	85	9.2%
Not a GeneRIF	Relevant in another species	333	36.1%
Not a GeneRIF	No abstract (unable to judge)	0	0.0%
Total		923	100.0%

Consensus discussions yielded the notion that measuring success would be best calculated by some sort of overlap measure between words nominated for annotation and those actually selected in the GeneRIF. A problem, however, was that while some GeneRIF snippets were direct quotations from article abstracts, others were paraphrased. Furthermore, there were other legitimate references to basic gene biology beyond the official GeneRIF snippet. A preliminary analysis by Jim Mork and Lan Aronson of NLM found that 95% of GeneRIF snippets contained some text from the title or abstract of the article. About 42% of the matches were direct "cut and paste" from the title or abstract, and another 25% contained significant runs of words from pieces of the title or abstract.

Data

The data for the secondary task consisted of 139 GeneRIFs representing all of the articles appearing in five journals for which we could obtain full text from Highwire (*Journal of Biological Chemistry*, *Journal of Cell Biology*, *Nucleic Acids Research*, *Proceedings of the National Academy of Sciences*, and *Science*) that were published during the latter half of 2002.

Performance Measures

The original plan for assessing the secondary task was to use the Dice coefficient, which measures overlap of two strings. In this instance, the Dice coefficient would calculate the overlap between the candidate GeneRIF and actual GeneRIF. For two strings A and B, define X as the number of words in A, Y as the number of words in B, and Z as the number of words occurring in both A and B. The Dice coefficient is calculated as:

$$\text{Dice}(A, B) = (2 * Z)/(X + Y)$$

It quickly became apparent that this measure was quite limited. It did not, for example, perform any "normalization" of words, such as stop word removal or stemming. It also did not give any credit for words occurring more than once in both strings. Finally, it assumed the strings were simply bags of words and did account for word order or phrases.

Marti Hearst and Presley Nakov developed four derivatives (and Perl code, enhanced by Ravi Teja Bhupatiraju to calculate them) of the classic Dice measurement for the task:

Classic Dice The Dice formula from above applied to words, which were defined as successive alphanumeric characters delimited by white space.

Modified Unigram Dice - This measure gave added weight to terms that occurred multiple times in both strings. In particular, each set of words in a string was a multi-set, with the number of co-occurring words measured by the minimum number of co-occurences.

Bigram Dice - This measure gave additional weight to proper word order. Instead of measuring the unigram Dice coefficient on single words, it measured it on bigrams.

Bigram Phrases - Bigrams do not always represent legitimate phrases. Stop words such as articles and prepositions sometimes occur between content words such that straight bigrams of content words do not represent real phrases. A further measure therefore only included bigrams that did not have intervening stop words filtered.

Official Runs

A total of 14 groups submitted 24 runs. Table 6 lists the runs, sorted by Classic Dice score. The top-ranking run (emc4) came from Erasmus University. The mean and median results are shown at the bottom of the table, followed by the results of a run using simply the document titles.

Most participants found that the GeneRIF text most often came from sentences in the title or abstract of the MEDLINE record, with the title being used most commonly. As such, just using the text of the titles alone achieved a baseline performance that few groups were able to

Table 6 - Official secondary task runs, sorted by classic Dice score.

Run Tag	Classic	Unigram	Bigram	Phrases
emc4	57.83	59.63	46.75	49.11
biotextTask2	53.04	54.65	38.62	41.17
tgIlhugLASt	52.78	54.33	37.72	40.65
UniNEie1	52.28	54.78	37.43	40.35
UniNEie2	51.72	54.27	36.62	39.71
UIowaSecCan	50.68	52.72	35.32	37.87
IBMbtT2	50.47	52.60	34.82	37.91
IUB2003	50.40	52.56	34.83	37.97
NLMUMDLIN	50.36	52.65	35.03	38.34
UniNEie3	49.46	51.42	33.62	36.99
UBGenT2R2	49.40	51.30	33.59	36.99
CSUSMcand	49.31	51.30	34.99	37.80
UBGenT2BL1	49.28	51.25	33.59	36.99
UBGenT2R1	49.03	51.16	33.94	37.35
balscsec1	48.90	50.52	32.36	34.61
we	48.15	49.78	32.31	35.63
nwe	47.62	49.37	31.61	34.80
uwb3	46.48	48.25	29.53	32.82
uwb2	44.41	44.07	2.33	1.80
uwb4	36.28	35.21	22.73	24.52
EDISTFruns2	35.76	35.85	20.05	21.84
tg2hug	35.20	34.57	20.04	21.58
UniNEie4	25.88	25.29	12.03	13.61
UniNEie5	9.42	14.20	0.15	0.17
Mean	45.59	47.16	29.58	32.11
Median	49.30	51.28	33.61	36.99
Titles Only	50.47	52.60	34.82	37.91

outperform. The best approaches (Erasmus (Jelier, Schuemie et al., 2003) and Berkeley (Bhalotia, Nakov et al., 2003)) used classifiers to rank sentences likely to contain the GeneRIF text. No groups much improvement beyond using titles alone.

Future Directions

Despite the limited type of data, relevance judgments, and tasks, the track organizers were pleased with the results and enthusiasm of the participants. We are fortunate to have been awarded a National Science Foundation Information Technology Research grant to provide funding to the track for the next years. The first year's activities also consisted of laying out a roadmap for future iterations of the track. Described in more detail on the track Web site, this will include, over the years, real relevance judgments, use of additional documents beyond MEDLINE, user experiments, and use cases of different types of users.

References

Baxevanis, A. (2003). The Molecular Biology Database Collection: 2003 update. *Nucleic Acids Research,* 31: 1-12.

Bhalotia, G., Nakov, P., et al. (2003). BioText team report for TREC 2003 Genomics Track. *The Twelfth Text REtrieval Conference: TREC 2003*, Gaithersburg, MD. NIST.

Buckley, C. and Voorhees, E. (2000). Evaluating evaluation measure stability. *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Athens, Greece. ACM Press. 33-40.

deBruin, B. and Martin, J. (2003). Finding gene function using LitMiner. *The Twelfth Text REtrieval Conference: TREC 2003*, Gaithersburg, MD. NIST.

Jelier, R., Schuemie, M., et al. (2003). Searching for GeneRIFs: concept-based query expansion and Bayes classification. *The Twelfth Text REtrieval Conference: TREC 2003*, Gaithersburg, MD. NIST.

Kayaalp, M., Aronson, A., et al. (2003). Methods for accurate retrieval of MEDLINE citations in functional genomics. *The Twelfth Text REtrieval Conference: TREC 2003*, Gaithersburg, MD. NIST.

Mitchell, J., Aronson, A., et al. (2003). Gene indexing: characterization and analysis of NLM's GeneRIFs. *Proceedings of the AMIA 2003 Annual Symposium*, Washington, DC. Hanley & Belfus, 460-464.

O'Neill, G. (2003). Gruber winner Botstein calls for better gene-naming system. Bio-IT World. http://www.bio-itworld.com/news/070903_report2840.html.

Savoy, J., Rasolofo, Y., et al. (2003). Report on the TREC 2003 experiment: Genomics and Web searches. *The Twelfth Text REtrieval Conference: TREC 2003*, Gaithersburg, MD. NIST.

Tong, R., Quackenbush, J., et al. (2003). Knowledge-based access to the biomedical literature. *The Twelfth Text REtrieval Conference: TREC 2003*, Gaithersburg, MD. NIST.

Wheeler, D., Church, D., et al. (2003). Database resources of the National Center for Biotechnology. *Nucleic Acids Research,* 31: 28-33.

Yeung, D., Clarke, C., et al. (2003). Task-specific query expansion (MultiText experiments for TREC 2003). *The Twelfth Text REtrieval Conference: TREC 2003*, Gaithersburg, MD. NIST.

Zhai, C., Tao, T., et al. (2003). Improving the robustness of language models: UIUC TREC 2003 Genomics and Robust Track experiments. *The Twelfth Text REtrieval Conference: TREC 2003*, Gaithersburg, MD. NIST.

HARD Track Overview in TREC 2003
High Accuracy Retrieval from Documents

James Allan
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts Amherst

1 Introduction

The effectiveness of ad-hoc retrieval systems appears to have reached a plateau. After several years of 10% gains every year in TREC, improvements dwindled or even stopped. This lack of progress was undoubtedly one of the reasons behind abandoning suspending the ad-hoc TREC after TREC-9.

One plausible reason that document retrieval has been unable to improve is that the nature of the task requires that systems adopt "one size fits all" approaches. Given a query, a system will generally do best to return results that are good for an "average" user. Doing otherwise (i.e., targeting the results for a particular type of user) might result in substantial improvements on a query, but it is just as likely (in a TREC environment) to cause horrible degradation. By ignoring the user (or, more accurately, by treating all users identically), systems cannot possibly advance beyond a particular level of accuracy on average for a specific user.

The goal of this track is to bring the user out of hiding, making him or her an integral part of both the search process and the evaluation. Systems do not have just a query to chew on, but also have as much information as possible about the person making the request, ranging from biographical data, through information seeking context, to expected type of result.

The HARD track is a variant of the ad-hoc retrieval task from the past. It was a "pilot" track in 2003 because of the substantial extension on past evaluation—i.e., it is not clear how best to evaluate some of the aspects of the track, so at least for this year it was intended to be very open ended. HARD is also running as a track of TREC 2004.

The HARD 2003 track ran in three phases: baseline, clarifying, and final. In the first phase, sites received and ran topics that were essentially idential to a classic TREC topic: title, description, and narrative fields.

In the second phase, sites were able to acquire clarifying information about the topics. They had two means and could use either or both of them:

1. Biographical, contextual, preferred result format, and any information that disambiguates the query was captured when the topics were generated. This metadata about the query was made available for phase two.

2. Sites were permitted to generate a single "clarifying form" that the searcher would answer. For each topic, this form was a Web page that solicited useful information about the query or the searcher (e.g., disambiguating words in the query or finding out more information about what the searcher wants). The assumption was that the "clarification" would be generated automatically, but sites could have opted 0to generate manual clarification questions (can a librarian beat the best IR systems?). None did.

In the final phase of the track, sites used all user- and query-information that they acquired to construct a better and more accurate ranked list. That substantially improved (because it is more targeted) list was submitted to NIST for evaluation.

Because accurate retrieval could also just be pinpointed retrieval, an extension of the HARD track evaluated passage retrieval, a system's ability to select passages within documents that are relevant. Passage retrieval was an option available to sites, but could be ignored by returning full documents.

2 Participation

The following 14 sites participated in the HARD track. A summary of each group's activity is provided below. The summaries were written by the site (except for those that are in italics) and are listed in alphabetical order.

Clairvoyance [Shanahan et al., 2004]

The Clairvoyance team participated in the HARD Track, submitting fifteen runs. Our experiments focused primarily on exploiting user feedback through clarification forms for query expansion. We made limited use of the genre metadata. Within the clarification form feedback framework we explored the following hypothesis: could we organize the top retrieved documents for a query into intuitive groups (through clustering) that the user then selects as representative/relevant for the topic. Within this we explored two types of clarification forms: one was based upon representing each group using a list of terms, corresponding to typical terms for the group, and a list of documents, where each document was represented using its title and the information source (very similar to the forms used in Scatter-Gather [Hearst et al., 1995]); the second form represented each group using just a list of terms (forty), corresponding to typical terms for the group. The user was asked to judge the relevance of each group as being "On Topic", "Not on Topic", "Unsure", or "Unjudged". The group judgments were then used to expand (as feedback) the query. We explored various schemes for expansion. Overall, our results for the experiments suffered due to a bad baseline run that was used in the generation of both clarification forms. The mean average precision (MAP) for the top 1000 documents was 0.23 (about median for this track), which has since been improved to 0.31. Having said that, when we did incorporate feedback from the title-based form, the MAP was improved to 0.29 (from 0.23). The second term-based form did not yield any significant improvement. Follow-up experiments on our new baseline, where we choose a single group that yields the best performance for a topic (assume we have an oracle), has yielded an overall MAP of 0.37. We are currently regenerating the title-based forms using our new baseline run and will have a human re-evaluate them. We are also exploring how to automatically select these groups for feedback.

IIT Bombay [Ramakrishnan et al., 2004]

The description of the IIT Bombay system is minimal. They used the open source retrieval system Lucene to index the document collection and select passages for retrieval. The focus of their work in TREC 2003 was text summarization and they applied summarization techniques for several of the tracks.

Illinois Urbana-Champaign [Shen and Zhai, 2004]

For the clarification forms, we focused on studying the problem of "active feedback": Given that a user is willing to make relevance judgments on k documents, how do we choose k documents to present to the user so that we can learn *most* from the user's feedback on them?

The simplest baseline approach is to present the top k documents. However, this may not be the best strategy; for one thing, some of the top k documents may be redundant. Thus we proposed and tested three other methods: (1) "Gap-based methods": We sample k documents from the top ranked documents so that these k documents would form some "gaps" between them. E.g., we can pick k documents with ranks 1, 1+g, 1+2*g, 1+3*g, ..., 1+(k-1)*g. In this way, the gap between two adjacent documents in the ranked list is "g-1". If we assume that documents that are close in the ranked list are likely similar to each other, then this method would help reduce redundancy. (2) The Marximal marginal relevance (MMR) method: Here we select the k documents with a greedy algorithm. At each step, we try to pick a document that is both relevant and novel. (3) The clustering centroid method: We cluster the top N documents into k clusters, and pick the centroid document from each cluster.

To reduce the labor on the user side for judging documents, we presented the best (fixed window) passage, for each document in the clarification form, rather than the whole document. Our form contains essentially just 6 passages.

After obtaining the judgments, we explore two ways of using the judgments: (1) Using them as "passage judgments" and perform passage-based feedback (i.e., query expansion); (2) Using them to infer relevance status of the corresponding document – actually we simply take the judgment on a passage as if it were a judgment on the document that contains the passage.

Our basic retrieval approach is the KL-divergence retrieval formula with mixture model for feedback.

The results so far suggest that (1) All feedback methods perform better than the baseline no-feedback method. This isn't surprising at all, as it just shows that relevance feedback is effective. It does show that language model based feedback is effective. (2) Passage-based feedback performs *substantially* better than document-based feedback, which is also consistent with what others have seen (e.g., the local context analysis method?). But again, we did it with language models. (3) The gap-based method performs slightly better than the clustering method in terms of average precision, but the clustering method performs slightly better by pr@10 docs and R-precision. We are waiting for results of the MMR approach and of the baseline "top k" document approach. The comparison with the "top k" method would be most interesting, as it would indicate how effective our "active feedback methods" is as compared with the more standard way of presenting top k documents. (4) Compared with the group, our performances seem to be usually above medians.

Microsoft Research Cambridge [Robertson et al., 2004]

For the HARD task, we concentrated on the use of clarification forms (system-generated forms offered to the assessor who originated the topic for a one-pass user interaction). The primary intention was to obtain some data that could be used in a relevance feedback algorithm, The limitations of the clarification form (both screen space and assessor time) prevent the presentation of entire documents or even substantial passages, but it is possible to offer the user small, query-specific snippets. We used a form of passage retrieval, where the passages are pre-defined exclusive units at a little above the sentence level – each passage consisting of one or a few sentences, with no overlap between passages. The main focus was on an "active learning" approach to selecting the snippets to show the user: we wanted to choose items that would give us most information for relevance feedback purposes. Out of the top 30 documents, we attempted to choose a set of five which would maximise the expected change to the query after judgement (which might be positive or negative) by the user. The particular functions chosen to measure this effect will be described. The result of this selection process generally differed from the top 5 documents (our baseline run). The user was not asked specifically for a relevance judgement on each snippet, but rather for something that might correspond to click-through data. We also presented the user with some phrases selected from the top snippets, and invited them to select positive or negative phrases. Various possible ways to use this data will be discussed.

We incorporated minimal use of metadata. Since we are doing a form of relevance feedback with the clarification forms, we include the relt texts along with the rest of our relevant fragments (although the difference in length might be problematic). We did the obvious thing with the US Govt stuff.

For use of passages, see the description above. This is a step backwards from the kind of overlapping, any-size passages we used to do with Okapi, which we haven't yet reproduced in the new environment. We did some rather obvious matching of metadata granularity onto these passages.

Chinese Academy of Sciences [Wu et al., 2004]

They used natural language processing to restrict queries to just nouns and verbs and to classify them into positive and negative classes that could be treated differently. Their primarily emphasis was on how to train the relative weights of the positive and negative words in the query.

Queens College, CUNY [Grunfeld et al., 2004]

Basic retrieval was done using our PIRCS system with pseudo relevance feedback (expand using 20 docs and 60 terms). Three runs were submitted: pircHDBt1 and pircHDBt2 which are title runs but with slightly different parameters. pircHDBtd1 is another run using title and description.

Three clarification forms were submitted QCSU:1-3. The QCSU:2-3 forms were designed to display data produced by the BASIC retrieval for evaluation – essentially 20 feedback *terms* (out of 60) that have lowest document frequencies. In addition we displayed top 10 (out of 20) PRF documents with their title/first *sentence* for the user to judge. We also make available a window for the user to add whatever they want as *key* terms to improve the topic description. We believe the user can complete these in 3 minutes. The first form QCSU1 has synonym terms from WordNet based on the topic title only. Phrases were sent to WordNet first. If synonyms were found, their constituent single words were removed; else single words were used to find synonyms. This does not involve a BASIC retrieval, and is more efficient. We like to compare results to see if a BASIC retrieval is necessary.

For enhanced retrieval without metadata, *terms* and *keys* from the clarification form were added to enhance the raw query. A full retrieval was repeated including PRF. During PRF we make sure that the docs corresponding to the *sentences* marked relevant are included for feedback for QCSU:2-3. For QCSU1, no *sentence* information is available, just topic enhancement. These submissions are respectively; pircHDC1t1, pircHDC2t1 and pircHDC3td1. For QCSU:2-3, some clarification results lead to fewer than 3 relevant *sentences* (less than 3 relevant docs among the 10 displayed). This may signal that the BASIC retrieval is bad and the topic is hard. Even with enhanced *keys*, retrieval may only be mediocre. We regard this as suggestion that one should disallow PRF for these queries - 16 queries in QCSU2 and 20 in QCSU3. These submissions are pircHDC2t2, pircHDC3t2.

For enhanced retrieval using granularity metadata, starting with the results above, we further processed the 1000 retrieved docs of each topic by our QA system that returns 250 bytes as answer to a question. 250 bytes is like 40 words, close to a passage size; no effort was made to return sentences. Since such QA clue words as WHO, WHAT, HOW LONG, WHEN, etc are not available, the QA system Essentially defaults to finding text spans that contain most topic words and of higher weights. We believe answers could be words/phrases interspersed among these topic words in such a window. These runs are pircHDC1tp, pircHDC2tp and pircHDC3tdp.

Rutgers University [Belkin et al., 2004]

We were particularly concerned with such knowledge which could be gained through implicit sources of evidence, rather than explicit questioning of the information seeker. We therefore did not submit any clarification form, preferring to rely on the categories of supplied metadata concerning the user which we believed could, at least in principle, be inferred from user behavior, either in past or the current information seeking episode. We did not attempt to retrieve only passages. Below, we describe how we used the supplied metadata.

FAMILIARITY This we addressed by promoting the value of documents which score toward the unreadable end of readability scales for people highly familiar with the topic, and by promoting the value of documents which scored toward the easily readable end of the scales for people unfamiliar with the topic.

GENRE This we addressed in two ways. One was by constructing language models for all the retrieved documents for each training topic and for just the completely relevant documents for each topic. We then identified words which occurred with greater than expected probability, based on the entire topic language model, in the relevant documents, for all topics which had the same genre. These words were considered to be indicators of the genre. We added the words associated with a particular genre to queries for topics which requested that genre. The second way was to promote documents from certain sources to the top of the retrieved list for topics with some genres, by removing documents from some sources entirely from the retrieved list for topics with some genres, and by demoting the value of documents from some sources in the retrieved list for topics with some genres.

RELEVANT TEXTS We used relevant texts as the basis for automatic query expansion.

GRANULARITY If the desired granularity of the retrieval result was passage, we ranked documents on the basis of their best passage, rather than on the document as a whole.

Tsinghua University, CS IR Group [Ma et al., 2004]

Main idea: Though the HARD is new experimental track, our research work mainly focus on delivering a practical solution for applied search environment. Therefore, all the submitted results(including CF and runs) are constructed in a automatic way, for we think it is more feasible than manual mode.

1. Baseline run. We get the baseline run (only with document) using the initial query by a TF*IDF scoring schema (BM 25). For each topic, the initial query is constructed simply by the task description(The detail restriction for none-relevant document are ignored). For the search items, different weights are set according to their position and importance in the task description. No positive training documents are used to refine the query, because usually the training resource is unlikely to be provided for various immediate search requirements in Web IR.

2. Clarification form. In the form, all the potential search issues to be confirmed by user are listed with checkbox, together with a text field to fill if he/she find there are something we missed. The search issues are presented as keywords or phrases, which are automatically extracted by two methods: (1) the kernel words/phrase in topic description. (2) terms with high statistical weight in top-100 ranked documents in search result. To keep the search deviation under control, we limit the search items up to 10 issues. It is an efficient method for delivering clarification form to the user, while the accurate of the question seem not satisfied.

3. Final Run.

1. *refine the query term.* The resource to refine the query terms is from: (1) attached text field in the return CF form; (2) the searchitem field in metadata. The new terms are added by Rocchio-like style.

2. *focus probe and reconstruct the query.* From the CF result returned from LDC, we do the work in two ways: In first method, all the items in selected checkbox in return CF are thought as one search focus. Based on the kernel terms in initial query and the current search item, a sub-query is constructed for a specific search focus. Then the initial query is divided into several queries for different search focus. And the final result of the topic is the combine of the results from all the sub-query. In second method, all the search terms of the search focus are simply taken as new weighted terms to be added into the initial query. Then using the new refined query, we get the final run.

3. *return type detection.* There are three different types available. We return document if topic require so. For passage and sentence, we usually return the single paragraph(For sentence, it is nearly impossible to present an efficient result in such rough retrieval). If any type is welcomed, we analyze the topic description and decide the result should be passage or document.

UMass Amherst [AbdulJaleel et al., 2004]

The CIIR at UMass Amherst participated in all three aspects of the HARD task. First, we mapped query metadata values to document metadata values that we assigned. We then adjusted the ranking of documents depending on whether their metadata matched the query metadata.

We also generated clarification forms to tease more information out of the searcher. We tried several types of clarification forms, including providing a list of keywords that might appear in relevant documents, a list of top-ranking clusters that might contain relevant documents, and a list of passages that might appear in relevant documents.

Finally, we explored passage-level retrieval of documents to see if we could pinpoint the relevant portions of documents.

In the final analysis, all runs using metadata or clarification forms failed to outperform our best baseline run (which included query expansion). Further exploration is needed to understand why the adjustments did not help more. Passage retrieval provided a gain for a subset of queries, but there were just as any that did not improve or dropped in effectiveness.

University of Buffalo, CEDAR [Srikanth et al., 2004]

Metadata: We used the purpose, genre and granularity metadata items in our solution to the HARD problem. Documents were processed by InfoXtract - an information extraction engine from Cymfony Inc. InfoXtract parses the documents to tag named entities, semantic structures and discovers relations between entities. HARD queries are parsed by InfoXtract to identify question type. The occurrence of these features and query keywords in documents and text snippets (in the case of passage, sentence or phrase granularity) are used to model some of the metadata values of the query and rank the answers.

Passages: Document snippets are selected for all granularity values except 'Document'. For granularity of sentence/phrase, each sentence with at least one query keyword is shortlisted as candidate answer snippets. For passage granularity, contiguous sentences are selected based on keyword hits based on minimum (3 sentences) and maximum (6 sentences) window sizes.

Documents processed by InfoXtract are indexed (words and extracted features are used as index terms) by TAPIR toolkit. Document retrieval for HARD queries is based on Concept Language Models. Expected answers for a user query is modeled as a sequence of keyword and non-keyword features (e.g. passage position in document, answer-type match, occurrence and count of location/time/person/organization features). Document and/or text snippets are ranked based on the probability of their model *generating* the query features. adhoc weights were assigned to query features.

University of Helsinki

No information provided.

University of Maryland [He and Demner-Fushman, 2004]

The goal of University of Maryland team (UMD team) in this year's HARD experiment is to leverage existing theories, and models about information need negotiation in information science literature to design and implement an automated process of generating clarification questions and utilizing the answers to improve the ranked list of documents for a given query statement.

The clarification questions generated by UMD team came from four aspects of context information related to a given query, which was motivated by research work about information need negotiation. The four aspects are 1) characteristics of the subject area that the user is querying; 2) user's motivation/background, especially user's recent experience with searching on the subject area; 3) user's preference to sub-collections within the document collection; and 4) user's anticipation to result format. The questions were then further narrowed down to those whose answers can be utilized in an automated process. UMD team also preferred those questions that would probe complimentary information to the metadata provided. They applied three techniques in their automated process to utilize the extra information obtained from clarification forms and meta data. The three techniques are query expansion based on keyword extraction, document reranking within a ranked list, and ranked lists merging.

Not all metadata were used in UMD team's HARD experiment. The used metadata satisfied two conditions: 1) the data were able to apply in the automated process, and 2) the data were not covered by the answers from their clarification forms. Therefore, only genre and granularity information were used, where the first one was used in document reranking, and the latter was used to trigger passage retrieval.

UMD team designed their own simple passage retrieval module. Their passage retrieval module assumes that the relevance of a passage is related to how many query terms it contains, how important those query terms are, and how relevant the document containing the passage is. Among the three, they gave more emphasis to the document containing the passage. All passages are ranked according to their relevance, with the condition that only three passages were allowed from the same document. The final result is top 1000 passages for a given query.

University of Waterloo and Bilkent University [Vechtomova et al., 2004]

This year we decided to focus on developing techniques for eliciting additional search criteria from users, preparing the ground for the next year's participation, where we plan to focus on techniques that exploit genre, familiarity and purpose metadata. We developed two topic clarification techniques for this year's entry:

1) The first technique consists in selecting one representative sentence from each of the top-ranked documents retrieved by the terms taken from the topic titles. The selected sentences were presented in the clarification form, and the users were asked to choose those sentences that are likely to represent relevant documents. We selected sentences on the basis of the sum of idf of query term instances in the sentence. Sentences with equal scores were ranked by the sum of tf*idf of all terms in the sentence, normalised by the sentence length. The documents which contained sentences chosen by users were used for query expansion. We used word co-occurrence measure of Z-score to select the query expansion terms.

2) The second technique is to show to the users phrases, selected from the two top-scoring sentences in each document from the initially retrieved set, and asking them to choose those phrases that are likely to represent relevant documents. We used a POS tagger and a noun phrase chunker to identify noun phrases, which were ranked by the sum of idf weights of their constituents. Top-ranked phrases were included in the clarification form. Terms from user-selected phrases were then used for query expansion.

We used Okapi BM250 for document and passage retrieval. For the topics requiring retrieval of best sentences, we used the sentence selection method described above.

3 HARD Corpus

The evaluation corpus is a combination of newswire text from the 1999 portion of the AQUAINT corpus and of U.S. government documents. The following table provides details on the make-up of the corpus. All information is from only 1999 because only documents from that year are included:

NYT	APW	XIE	CR	FR	Totals
137,806	77,876	104,698	16,609	35,230	372,219
750Mb	245Mb	310Mb	147Mb	330Mb	1.7Gb
Jan-Dec	Jan-Nov	Jan-Dec	Jan-Dec	Jan-Dec	

The New York Times (NYT), Associated Press Worldstream (APW), and Xinghua English (XIE) articles are all available on the AQUAINT disks. Those disks were available free-of-charge to all TREC participants.

The Congressional Record (CR) and Federal Register (FR) data set was gathered by the LDC for this track. Particularly lengthy documents from either source were not included because they cause serious annotation problems. This set of data was provided free-of-charge to all participants in the HARD track.

4 Topics

Topics follow the basic TREC style, but are more richly annotated with metadata that describes the searcher and the context of the query. The format of a topic is:

<top>
<num> Number: HARD-*nnn*
<title> *Web-style description of topic*
<desc> Description: *Sentence-length description of topic*
<narr> Narrative: *Paragraph-length description of topic, indended primarily to help future relevance assessors*
<hard> item=*label*, value=*value*
<hard> item=*label*, value=*value*
<hard> item=*label*, value=*value*
<hard> item=*label*, value=*value*
...
</top>

The following metadata items and values are provided for each topic:

1. *item=PURPOSE* represents why the user is searching for the information.
    - value=BACKGROUND indicates that the searcher wants to know where the topic came from.
    - value=DETAILS means the searcher wants to know the details of the topic.
    - value=ANSWER indicates the user is looking for an answer to a specific question. (This value is implicitly linked to some of the GRANULARITY values.)
    - value=ANY means that the user has no specific purpose in mind or, at least, has not specified one.
2. *item=GENRE* represents the type of material the searcher is interested in.
    - value=OVERVIEW means the searcher is interested in general news related to the topic.
    - value=REACTION indicates the searcher is looking for news commentary on the topic.
    - value=I-REACTION is like REACTION but is specifically about non-U.S. news commentary.
    - value=ADMINISTRATIVE means the search is interested in official US government documents.
    - value=ANY indicates that any genre is acceptable or none was indicated.
3. *item=FAMILIARITY* represents how familiar the searcher is with the topic. Presumably a user who is fully aware of the details of a topic would not be interested in background material, for example.
    - value=1, no prior knowledge
    - ...
    - value=5, know details of topic
    - value=UNKNOWN means that the user does not know his or her familiarity or has not specified one.
4. *item=GRANULARITY* captures the amount of text that the searcher is anticipating in a value response.
    - value=DOCUMENT means the searcher is expecting complete documents (one or more).
    - value=PASSAGE will be selected when the search expects extracts from documents that are on the paragraph or multi-paragraph level.
    - value=SENTENCE means that the retrieved units should be roughly at the sentence level.
    - value=PHRASE means that user is expecting a small number of words (including just one) as a response.
    - value=ANY means the user has no specific granularity in mind or did not specify one.
5. *item=RELATED-TEXT*. This item includes sample relevant text. It may be repeated if there are multiple sample texts to be included.
    - value="..." identified text that is known to be related to the topic being specified. This provides a kind of pre-query relevance feedback. The intent is that this text not come from the evaluation corpus.

During topic creation, the LDC made an effort to have topics vary across each of the indicated metadata items.

When the LDC created the *evaluation* topics, they also gathered additional metadata beyond what was required by the HARD track. That information is provided along with the HARD metadata after the baseline runs and may be used in any way a site likes. The items collected were:

- OCCUPATION

- SPECIAL TRAINING
- SPECIAL INTERESTS, where the annotator can candidly explain why he or she chose this topic
- LANGUAGES SPOKEN
- AGE
- SEX

5 Relevance judgments

For each topic, documents that are annotated get one of the following judgments:

- NON-RELEVANT means that the document is known not to be relevant to the topic. (As is common in TREC, a document without any judgment is assumed to be non relevant.)
- SOFT-REL means that the document is relevant to the topic but that it does not satisfy the appropriate metadata. Given the metadata items listed above, that means it either does not satisfy the PURPOSE, GENRE, or the FAMILIARITY items (the others are not document-level items).
- HARD-REL means that the document is relevant *and* it satisfies the appropriate metadata.

In addition, if the GRANULARITY value is not DOCUMENT, then each judgment will come with information that specifies which portion of the documents is relevant.

To specify passages, HARD used the same approach used by the question answering track. A passage is specified by its byte offset and length. The offset is from the "<" in the "<DOC>" tag of the original document (an offset of zero would mean include the "<" character). The length indicates the number of bytes that are included. If a document contains multiple relevant passages, the document is listed multiple times.

The HARD track used the standard TREC pooling approach to find possible relevant documents. The top 75 documents from one baseline and one final run from each submitted system were pooled (i.e., 75 times 14 times 2 documents). The LDC considered each of those documents as possibly relevant to the topic.

Judging was done in three passes:

1. Decide if the document contains relevant material (soft rel)
2. Decide if the document matches metadata restrictions (hard rel)
3. Select relevant passages within the document

Across all topics, the LDC annotated 42,016 documents, finding 5,123 that were HARD-REL and another 2,533 that were HARD-REL. Topics ranged from three HARD-REL documents to 400, and from 6 to 714 if SOFT-REL is also included.

6 Training data

The LDC provided 10 training topics. The topics incorporated a selection of metadata values and came with relevance judgments (though the relevance judgments were delayed).

In addition, the LDC provided a mechanism to allow sites to validate their clarification forms. Sites could send a form to the LDC and get back confirmation that the form was viewable and some "random" completion of the form. The resulting information was sent back to the site in the same format that was used in the evaluation.

7 Results format

Results were returned for evaluation in standard TREC format extended, though, to support passage-level submissions since it possible that the searcher's preferred response is the best passage (or sentence or phrase) of relevant documents. Results included the top 1000 documents (or top 1000 passages) for each topic, one line per document/passage per topic. Each line will have the format:

topic-id Q0 docno rank score tag psg-offset psg-length

where:

- *topic-id* represents the topic number from the topic (e.g., HARD-001)
- *"Q0"* is a constant provided for historical reasons
- *docno* represents the document that is being retrieved (or from which the passage is taken)
- *rank* is the rank number of the document/passage in the list. Rank should start with 1 for the document/passage that the system believes is most likely to be relevant and continue to 1000.
- *score* is a system-internal score that was assigned to the document/passages. High values of score are assumed to be better, so score should generally drop in value as rank increases.
- *tag* is a unique identifier for this run by the site.
- *psg-offset* indicates the byte-offset in document docno where the passage starts. A value of zero represents the "<" in "<DOC>" at the start of the document. A value of negative one (-1) means that no passage has been selected and the entire document is being retrieved.
- *psg-length* represents how many bytes of the document are included in the passage. A value of negative one (-1) must be supplied when psg-offset is negative one.

8 Evaluation approach

Results were evaluated at the document level, both in light of and ignoring the query metadata. Ranked lists were also evaluated incorporating passage-level judgments. We discuss each evaluation in this section.

Two of the 50 HARD topics (155 and 231) had no "HARD rel" documents. That is, although there were documents that matched the topics, no document in the pool matched the topic *and* the query metadata. Accordingly, those two topics were dropped from both the HARD and SOFT evaluations. (They could have been kept for the SOFT evaluation, but then the scores would not have been comparable.)

8.1 Document-level evaluation

In the absence of passage information, evaluation was done using standard mean average precision. There were two variants, one for HARD-REL judgments and one for SOFT-REL.

Some of the runs evaluated in this portion were actually passage-level runs and could therefore include a document at multiple points in the ranked list—i.e., because more than one passage was considered likely to be relevant. For the document-level evaluation, only the first occurrence of a document in the ranked list was considered. Subsequent occurrences were "deleted" from the ranked list.

Figure 1 shows a tradeoff between the number of relevant documents found in the first ten retrieved and the system's average precision. Each submitted run generates a point on the scatter plot, the main purpose of which is to show the range of scores that came in. Not surprisingly, there is a clear relationship between the two values. Figure 2 shows the same relationship for the "hard relevance" condition.

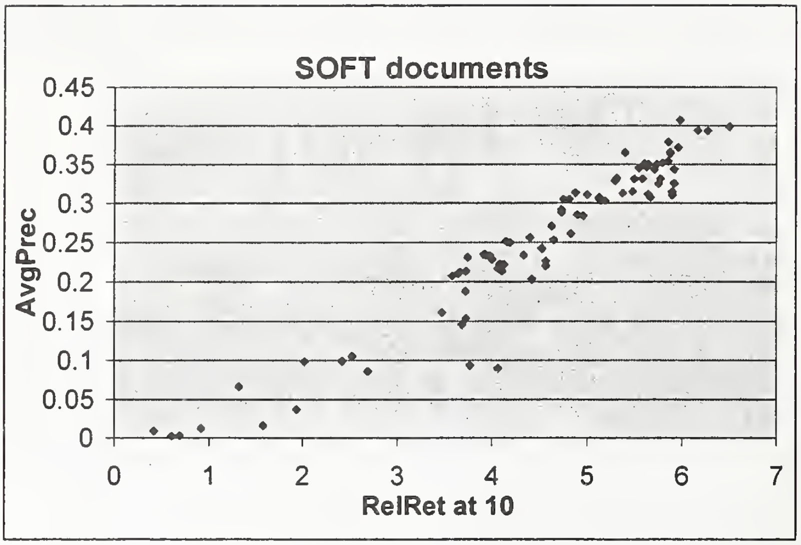


Figure 1: Scatter plot of number of relevant documents found in the first 10 compared to average precision, for "soft rel" documents.

8.2 Passage-level evaluation

The HARD track participants floated several passage evaluation measures. In the end, the track coordinator and NIST used one that was easy to implement and that attempted to match the goals of the community discussion.

The following operational description of passage recall and passage precision is provided by Ellen Voorhees of NIST to the HARD track participants.

> The passage level evaluation for a topic consists of values for passage recall, passage precision, and the F score at cutoff 5, 10, 15, 20, 30, 50, and 100, plus a R-precision score. As with standard document level evaluation, a cutoff is the rank within the result set such that passages at or above the cutoff are "retrieved" and all other passages are not retrieved. So, for example, if the cut-off is 5 the passage recall and precision are computed over the top 5 passages. R-precision is defined similarly to the document level counterpart: it is the passage precision after R passages have been retrieved where R is the number of relevant passages for that topic. We are using passage R-precision as the main evaluation measure reported for the track because it is a cutoff-based measure that tracks mean average precision extremely closely in document evaluations.
>
> For each relevant passage, allocate a string representing all of the character positions contained within the relevant passage (i.e., a relevant passage of length 100 has a string of length 100 allocated). Each passage in the retrieved set marks those character positions in the relevant passages that it overlaps with. A character position can be marked at most once, regardless of how many different retrieved passages contain it. (Retrieved passages may overlap, but relevant passages do not overlap.) The passage recall is then defined as the average over all relevant passages of the fraction of the passage that is marked. The passage precision is defined as the total number of marked character positions divided by the total number of characters in the retrieved set. The F score is defined in the same way as for documents, assigning equal weight

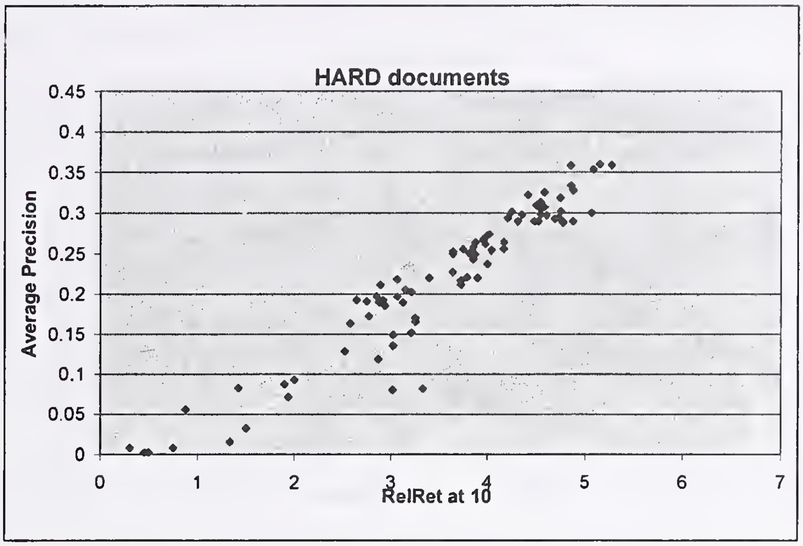


Figure 2: Scatter plot of number of relevant documents found in the first 10 compared to average precision, for "hard rel" documents.

to recall and precision:

$$F = (2 * \text{prec} * \text{recall})/(\text{prec} + \text{recall})$$

where F is defined to be 0 if prec+recall is 0. We included the F score because set-based recall and precision average extremely poorly but F averages well. R-precision also averages well.

In all of the above, a document is treated as a (potentially long) passage. That is, for topics where the granularity is "document" the relevant passage starts at the beginning of the document and is as long as the document. (These are represented in the judgment file as passages with -1 offset and -1 length, but are treated as described above.) For any topic, a retrieved document (i.e., where offset and length are negative one) is again just a passage with offset 0 and length the length of the document.

Using the above definition of passage recall, passage recall and standard document level recall are identical when both retrieved and relevant passages are whole documents. That is not true for this definition of passage precision. Passage precision will be greater when a shorter irrelevant document is retrieved as compared to when a longer irrelevant document is retrieved. This makes sense, but is different from standard document level precision.

Figure 3 shows the tradeoff between three measures for all submitted runs. Note that even runs that did not attempt any passage retrieval are included here; their "passages" are entire documents.

9 Conclusion

The HARD track is running again for TREC 2004. The experience of this track suggests the following changes, some of which have been adopted already:

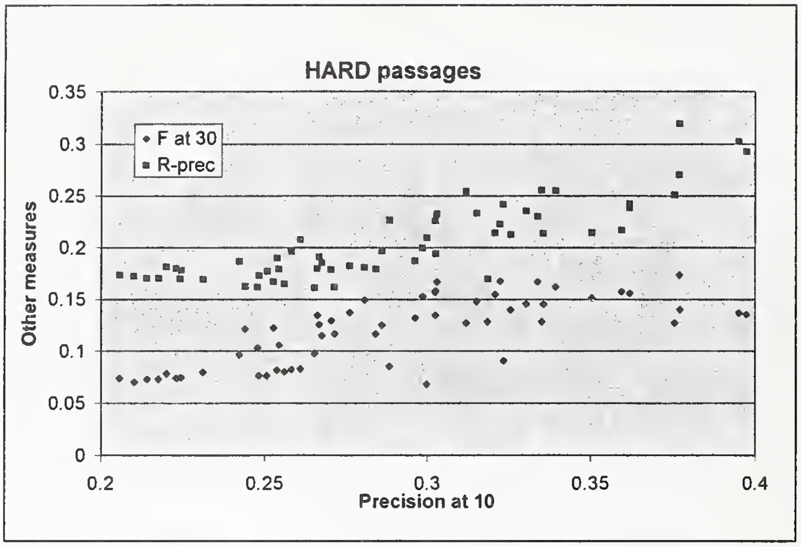


Figure 3: Scatter plot of precision at 10 documents retrieved compared to both the F measure at 30 documents retrieved and to R-precision, for "hard rel" passages.

- A corpus that permits a wider range of "interesting" metadata values would be useful. The current corpus was intended to provide a contrast between news and US government documents, but they were not different enough for metadata to be clearly useful.

  HARD 2004 will use a corpus of news from 2003. This does not provide the wide range we dream of, but it is a richer set of news than was used for HARD 2003. Also, because it is more recent news, it will be more pleasurable for the annotators to read.

- Passage-level judging was a terrifically difficult task for the LDC and needs to be revisited. The LDC has some thoughts on this but they have not been finalized at this time.

- As is typical with new tracks, many decisions were made quite late in the process. Next year they need to happen more quickly.

Acknowledgments

The HARD corpus creation and annotation was funded by DARPA under the TIDES program. NIST provided coordination and evaluation support for the work in the track as part of its support for TREC.

The coordination at the CIIR was supported in part by the Center for Intelligent Information Retrieval, in part by SPAWARSYSCEN-SD grant number N66001-02-1-8903, and in part by NSF grant number IIS-9907018. Any opinions, findings and conclusions or recommendations expressed in this material are the author's and do not necessarily reflect those of the sponsor.

References

[AbdulJaleel et al., 2004] AbdulJaleel, N., Corrada-Emmanuel, A., Li, Q., Liu, X., Wade, C., and Allan, J. (2004). UMass at TREC 2003: HARD and QA. In *Proceedings of the Twelfth Text Retrieval Conference (TREC-2003)*, Washington, DC. U.S. Government Printing Office. NIST Special Publication 500-255.

[Belkin et al., 2004] Belkin, N., Kelly, D., Lee, H.-J., Li, Y.-L., Muresan, G., Tang, M.-C., Yuan, X.-J., and Zhang, X.-M. (2004). Rutgers' HARD and web interactive track experiments at TREC 2003. In *Proceedings of the Twelfth Text Retrieval Conference (TREC-2003)*, Washington, DC. U.S. Government Printing Office. NIST Special Publication 500-255.

[Grunfeld et al., 2004] Grunfeld, L., Kwok, K., Dinstl, N., and Deng, P. (2004). TREC 2003 robust, HARD and QA track experiments using PIRCS. In *Proceedings of the Twelfth Text Retrieval Conference (TREC-2003)*, Washington, DC. U.S. Government Printing Office. NIST Special Publication 500-255.

[He and Demner-Fushman, 2004] He, D. and Demner-Fushman, D. (2004). HARD experiment at Maryland: From need egotiation to automated HARD process. In *Proceedings of the Twelfth Text Retrieval Conference (TREC-2003)*, Washington, DC. U.S. Government Printing Office. NIST Special Publication 500-255.

[Hearst et al., 1995] Hearst, M. A., Karger, D. R., and Pedersen, J. O. (1995). Scatter/gather as a tool for the navigation of retrieval results. In *The proceedings of the 1995 AAAI Fall Symposium on Knowledge Navigation.*

[Ma et al., 2004] Ma, L., Tan, W., Chen, Q., Ma, S., , Shi, S., Xiao, S., Wang, H., and Wang, H. (2004). THUIR at TREC 2003: HARD experiments. In *Proceedings of the Twelfth Text Retrieval Conference (TREC-2003)*, Washington, DC. U.S. Government Printing Office. NIST Special Publication 500-255.

[Ramakrishnan et al., 2004] Ramakrishnan, G., Bellare, K., Shah, C., and Paranjpe, D. (2004). Generic text summarization using wordnet for novelty and hard. In *Proceedings of the Twelfth Text Retrieval Conference (TREC-2003)*, Washington, DC. U.S. Government Printing Office. NIST Special Publication 500-255.

[Robertson et al., 2004] Robertson, S., Zaragoza, H., and Taylor, M. (2004). Microsoft cambridge at TREC-12: HARD track. In *Proceedings of the Twelfth Text Retrieval Conference (TREC-2003)*, Washington, DC. U.S. Government Printing Office. NIST Special Publication 500-255.

[Shanahan et al., 2004] Shanahan, J., Bennett, J., Evans, D. A., Hull, D. A., and Montgomery, J. (2004). Clairvoyance Corporation experiments in the TREC 2003 high accuracy retrieval from documents (HARD) track. In *Proceedings of the Twelfth Text Retrieval Conference (TREC-2003)*, Washington, DC. U.S. Government Printing Office. NIST Special Publication 500-255.

[Shen and Zhai, 2004] Shen, X. and Zhai, C. (2004). Active feedback - UIUC TREC-2003 HARD experiments. In *Proceedings of the Twelfth Text Retrieval Conference (TREC-2003)*, Washington, DC. U.S. Government Printing Office. NIST Special Publication 500-255.

[Srikanth et al., 2004] Srikanth, M., Ruiz, M., and Srihari, R. (2004). UB at TREC 12: HARD and genomics tracks. In *Proceedings of the Twelfth Text Retrieval Conference (TREC-2003)*, Washington, DC. U.S. Government Printing Office. NIST Special Publication 500-255.

[Vechtomova et al., 2004] Vechtomova, O., Lam, E., and Karamuftuoglu, M. (2004). Interactive search refinement techniques for HARD tasks. In *Proceedings of the Twelfth Text Retrieval Conference (TREC-2003)*, Washington, DC. U.S. Government Printing Office. NIST Special Publication 500-255.

[Wu et al., 2004] Wu, Z., Du, L., Sun, L., and Ye, S. (2004). TREC12 HARD track at ISCAS. In *Proceedings of the Twelfth Text Retrieval Conference (TREC-2003)*, Washington, DC. U.S. Government Printing Office. NIST Special Publication 500-255.

Overview of the TREC 2003 Novelty Track

Ian Soboroff and Donna Harman
National Institute of Standards and Technology
Gaithersburg, MD 20899

Abstract

The novelty track was first introduced in TREC 2002. Given a TREC topic and an ordered list of documents, systems must find the relevant and novel sentences that should be returned to the user from this set. This task integrates aspects of passage retrieval and information filtering. This year, rather than using old TREC topics and documents, we developed fifty new topics specifically for the novelty track. These topics were of two classes: "events" and "opinions". Additionally, the documents were ordered chronologically, rather than according to a retrieval status value. There were four tasks which provided systems with varying amounts of relevance or novelty information as training data. Fourteen groups participated in the track this year.

1 Introduction

The novelty track was introduced as a new track last year [5]. The basic task is as follows: given a topic and an ordered set of relevant documents segmented into sentences, return sentences that are both relevant to the topic and novel given what has already been seen. This task models an application where a user is skimming a set of documents, and the system highlights new, on-topic information.

There are two problems that participants must solve in the novelty track. The first is identifying relevant sentences, which is essentially a passage retrieval task. Sentence retrieval differs from document retrieval because there is much less text to work with, and identifying a relevant sentence may involve examining the sentence in the context of those surrounding it. We have specified the unit of retrieval as the sentence in order to standardize the task across a variety of passage retrieval approaches, as well as to simplify the evaluation.

The second problem is that of identifying those relevant sentences that contain new information. The operational definition of "new" is information that has not appeared previously in this topic's set of documents. In other words, we allow the system to assume that the user is most concerned about finding new information in this particular set of documents, and is tolerant of reading information he already knows because of his background knowledge. Since each sentence adds to the user's knowledge, and later sentences are to be retrieved only if they contain new information, novelty retrieval resembles a filtering task.

To allow participants to focus on the filtering and passage retrieval aspects separately, this year the track offered four tasks. The base task was to identify all relevant and novel sentences in the documents. The other tasks provided varying amounts of relevant and novel sentences as training data. Some groups which chose to focus on passage retrieval alone did only relevant sentence retrieval in the first task.

2 Input Data

Last year, the track used 50 topics from TRECs 6, 7, and 8, along with relevant documents in rank order according to a top-performing manual TREC run. The assessors' judgments for those topics were remarkable in that almost no sentences were judged to be relevant, despite the documents themselves being relevant. As a consequence, nearly every relevant sentence was novel. This was due in large part to assessor disagreement (the assessors were not the original topic authors) and drift (the document judgments were all made several years ago).

To both solve the assessor drift problem and to achieve greater redundancy in the test data, this year we constructed fifty new topics on a collection of three contemporaneous newswires. For each topic, the assessor composed the topic, selected 25 relevant documents by searching the collection, and labeled the relevant and novel sentences in the documents.

As an added twist, 28 of the topics concerned events such as the bombing at the 1996 Olympics in Atlanta, while the remaining topics focused on opinions about controversial subjects such as cloning, gun control, and same-sex marriages. The topic type was indicated in the topic description by a <toptype> tag.

The documents for the novelty track were taken from the AQUAINT collection. This collection is unique in that it contains three news sources from overlapping time periods: New York Times News Service (Jun 1998 – Sep 2000), AP (also Jun 1998 – Sep 2000), and Xinhua News Service (Jan 1996 – Sep 2000). We intended that this collection would exhibit greater redundancy and thus less novel information, increasing the realism of the task. The assessors, in creating their topics, searched the AQUAINT collection using WebPRISE, NIST's IR system, and collected 25 documents which they deemed to be relevant to the topic.

Once selected, the documents were ordered chronologically. (Chronological ordering is achieved trivially in the AQUAINT collection by sorting document IDs.) This is a significant change from last year's task, in which they were ordered according to retrieval status value in a particular TREC ad hoc run. Last year's ordering was motivated by the idea of seeking novel information in a ranked list of documents, whereas this year, the task more closely resembles reading new documents over time. This approach seems to make more sense when working with news articles, since background information tends to occur more completely in earlier articles and is summarized more briefly as time goes on and new information is reported. With relevance ranking, one can identify novel sentences but there is no sense of which document should come first.

The documents were then split into sentences, each sentence receiving an identifier, and all sentences were concatenated together to produce the document set for a topic.

3 Task Definition

This year, there were four tasks:

Task 1. Given the set of 25 relevant documents for the topic, identify all relevant and novel sentences. (This was the same as last year's task.)

Task 2. Given the relevant sentences in all 25 documents, identify all novel sentences.

Task 3. Given the relevant and novel sentences in the first 5 documents **only**, find the relevant and novel sentences in the remaining 20 documents.

Task 4. Given the relevant sentences from all 25 documents, and the novel sentences from the first 5 documents, find the novel sentences in the last 20 documents.

These four tasks allowed the participants to test their approaches to novelty detection given different levels of training: none, partial, or complete relevance information, and none or partial novelty information.

Participants were provided with the topics, the set of sentence-segmented documents, and the chronological order for those documents. For tasks 2-4, training data in the form of relevant and novel "sentence qrels" were also given. The data were released and results were submitted in stages to limit "leakage" of training data between tasks. Depending on the task, the system was to output the identifiers of sentences which the system determined to contain relevant and/or novel relevant information.

4 Evaluation

4.1 Creation of truth data

Judgments were created by having NIST assessors manually perform the task. From the concatenated document set, the assessor selected the relevant sentences, then selected those relevant sentences that were novel. Each topic was independently judged by two different assessors, the topic author and a "secondary" assessor, so that the effects of different human opinions could be assessed.

4.2 Analysis of truth data

Since the novelty task requires systems to automatically select the same sentences that were selected manually by the assessors, it is important to analyze the characteristics of the manually-created truth data in order to better understand the system results. In particular, there were several concerns raised by the peculiarities of last year's data.

1. What percentage of the sentences were marked relevant, and how does this vary across topics and across assessors?

2. Did the quantity of relevant and new information improve from last year? In particular, are more sentences relevant, and are fewer relevant sentences novel?

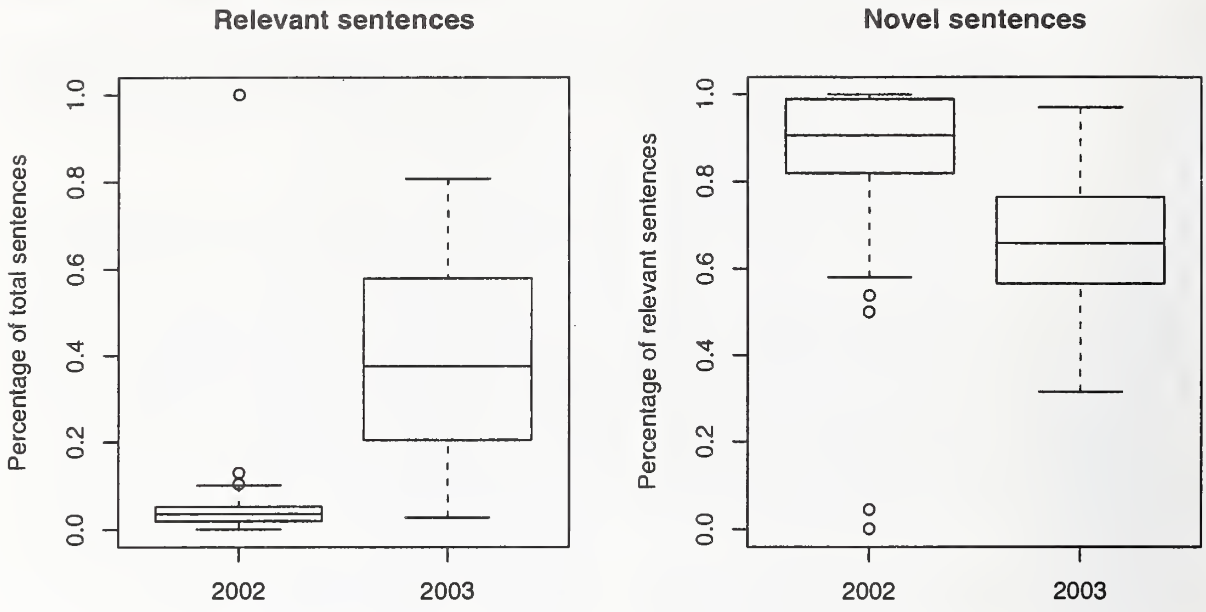


Figure 1: Percentage of relevant and novel sentences (both primary and secondary assessors), compared to 2002 (both minimum and maximum assessors).

3. How different are the results of the secondary assessor from the primary assessor who authored the topic and selected the documents?

4. Is there any difference between "event topics" and "opinion topics", in terms of amounts of relevant and new information?

Table 1 shows the number of relevant and novel sentences selected for each topic by each of the two assessors who worked on that topic. The column marked "assr-1" precedes the results for the primary assessor, whereas "assr-2" precedes those of the secondary assessor. The column marked "rel" is the number of sentences selected as relevant; the next column, "%total", is the percentage of the total set of sentences for that topic that were selected as relevant. The column marked "new" gives the number of sentences selected as novel; the next column, "%rel", is the percentage of relevant sentences that were marked novel. The column "sents" gives the total number of sentences for that topic, and "type" indicates whether the topic is about an event (E) or about opinions on a subject (O).

One of the most striking aspects of Table 1 is the difference in relevant and new percentages from last year. The median percentage of relevant sentences is 37.56%, compared with about 2% last year. For novel sentences, the median is 65.91%, compared with 93% last year. Figure 1 illustrates the range of relevant and novel sentences, and compares it to the 2002 data. Whereas last year, almost no sentences were selected as relevant, and as a result nearly every relevant sentence was novel, this year the distributions of relevant and novel sentences are much more reasonable.

The analysis of assessor effects is complicated by the fact that only four of the seven assessors (B, C, D, and E) acted as both primary and secondary assessors. Assessor A only judged as a primary assessor, and assessors F and G only judged as secondary assessors (i.e., they judged other assessors topics, but did not author their own).

As we might expect, there is a large effect from the assessors. For relevant sentence selection, this effect is more significant than either topic type or judgment round. The four assessors who judged topics in both rounds (B, C, D, and E) were quite different from each other, but judged similarly from the first round to the second. For novel sentences, it's a different story; differences between assessors are more pronounced in the first round, but in the second they are all quite similar to each other. Overall, the number of novel sentences selected is more uniform across

Table 1: Analysis of relevant and novel sentences by topic

Topic	type	sents	assr-1	rel	%total	new	%rel	assr-2	rel	%total	new	%rel
N1	O	880	A	184	20.91	151	82.07	F	457	51.93	265	57.99
N2	E	500	D	78	15.6	43	55.13	B	170	34.0	58	34.12
N3	E	932	C	596	63.95	331	55.54	E	248	26.61	152	61.29
N4	E	928	B	438	47.2	265	60.5	D	113	12.18	72	63.72
N5	E	1662	B	259	15.58	219	84.56	G	293	17.63	246	83.96
N6	E	424	B	317	74.76	233	73.5	C	294	69.34	192	65.31
N7	E	306	E	95	31.05	79	83.16	D	99	32.35	61	61.62
N8	E	659	D	158	23.98	107	67.72	G	349	52.96	210	60.17
N9	E	637	B	160	25.12	62	38.75	F	263	41.29	205	77.95
N10	E	257	C	191	74.32	159	83.25	F	174	67.7	139	79.89
N11	E	393	C	148	37.66	108	72.97	G	130	33.08	107	82.31
N12	O	1044	C	729	69.83	579	79.42	D	76	7.28	62	81.58
N13	O	941	A	205	21.79	166	80.98	F	457	48.57	198	43.33
N14	O	1129	D	93	8.24	42	45.16	G	191	16.92	122	63.87
N15	O	649	C	522	80.43	421	80.65	B	378	58.24	214	56.61
N16	E	500	E	179	35.8	119	66.48	F	274	54.8	183	66.79
N17	O	1106	B	792	71.61	488	61.62	G	724	65.46	524	72.38
N18	O	1238	B	537	43.38	429	79.89	D	98	7.92	47	47.96
N19	O	867	D	62	7.15	37	59.68	C	423	48.79	253	59.81
N20	O	886	D	69	7.79	41	59.42	B	228	25.73	169	74.12
N21	O	932	E	340	36.48	194	57.06	G	317	34.01	265	83.6
N22	O	841	D	84	9.99	52	61.9	F	401	47.68	295	73.57
N23	O	896	E	346	38.62	254	73.41	D	134	14.96	86	64.18
N24	O	968	E	160	16.53	76	47.5	B	220	22.73	101	45.91
N25	O	701	D	19	2.71	16	84.21	C	301	42.94	249	82.72
N26	O	911	C	661	72.56	283	42.81	E	178	19.54	137	76.97
N27	O	962	C	730	75.88	577	79.04	E	273	28.38	229	83.88
N28	O	978	B	371	37.93	261	70.35	E	109	11.15	80	73.39
N29	O	861	D	65	7.55	52	80.0	B	69	8.01	39	56.52
N30	O	900	C	445	49.44	307	68.99	G	497	55.22	386	77.67
N31	O	1220	C	985	80.74	752	76.35	D	74	6.07	48	64.86
N32	O	1078	B	216	20.04	100	46.3	C	684	63.45	475	69.44
N33	E	680	C	526	77.35	376	71.48	G	441	64.85	297	67.35
N34	E	1030	E	475	46.12	217	45.68	D	106	10.29	78	73.58
N35	E	399	E	221	55.39	77	34.84	B	253	63.41	95	37.55
N36	E	355	C	167	47.04	162	97.01	F	239	67.32	183	76.57
N37	E	547	D	76	13.89	52	68.42	G	263	48.08	196	74.52
N38	O	1127	D	140	12.42	96	68.57	F	252	22.36	188	74.6
N39	E	590	B	211	35.76	128	60.66	E	221	37.46	151	68.33
N40	E	533	B	307	57.6	183	59.61	E	209	39.21	145	69.38
N41	E	672	C	535	79.61	403	75.33	B	297	44.2	99	33.33
N42	E	1119	C	400	35.75	340	85.0	B	414	37.0	140	33.82
N43	E	224	E	122	54.46	67	54.92	C	142	63.39	116	81.69
N44	E	585	C	432	73.85	266	61.57	D	55	9.4	34	61.82
N45	E	1054	E	262	24.86	124	47.33	F	412	39.09	254	61.65
N46	E	549	E	322	58.65	123	38.2	C	342	62.3	165	48.25
N47	E	601	C	420	69.88	290	69.05	B	142	23.63	86	60.56
N48	E	1075	D	98	9.12	53	54.08	E	245	22.79	156	63.67
N49	E	684	D	209	30.56	66	31.58	C	225	32.89	147	65.33
N50	E	810	E	400	49.38	200	50.0	C	485	59.88	182	37.53

assessors than relevant sentences. Figure 2 illustrates these differences.

Last year, we found that the assessors tended to pick consecutive groups of sentences as relevant, despite being instructed otherwise. This year, we did not restrict them from selecting consecutive sentences, instead allowing them to select whatever they felt was necessary. As might be expected, this along with the greater amount of relevant sentences chosen resulted in a much higher occurrence of consecutive relevant sentences. On average, 84% of relevant sentences were selected immediately adjacent to another relevant sentence. The median length of a string of consecutive relevant sentences was 2; the mean was 4.252 sentences.

Overall, there was not a large difference between the primary and secondary assessor in terms of the number of relevant and novel sentences selected. Figure 3(a) shows that the secondary assessors tended to be a little more restrictive in their judgments, but this difference is not statistically significant. This implies that the marked difference in judgment patterns we see between this year and last is not only due to an assessor effect. Having more recent documents and topics, and allowing the assessors to select the relevant documents, probably also played a role.

There is a larger difference between event and opinion topics. Figure 3(b) illustrates this. Opinion topics tended to have a lower percentage of relevant and a higher percentage of novel sentences than events. The higher percentage of novel sentences is actually due to the lower percentage of relevant sentences. The difference is statistically significant for relevant sentences, but not for novel ones.

While it may be the case that having multiple news sources from the same time period increased redundancy over last year's topics, having stories from two or three wires did not make a significant difference in the number of novel sentences. Only one topic (10) drew stories from a single news source; all others involved either two or three sources. On average, 63.61% of relevant sentences were novel for topics with two sources, and 64.73% for those with three. Both of these are less than the new percentage for topic 10 (83.25%), but with only one topic we can't make any conclusions.

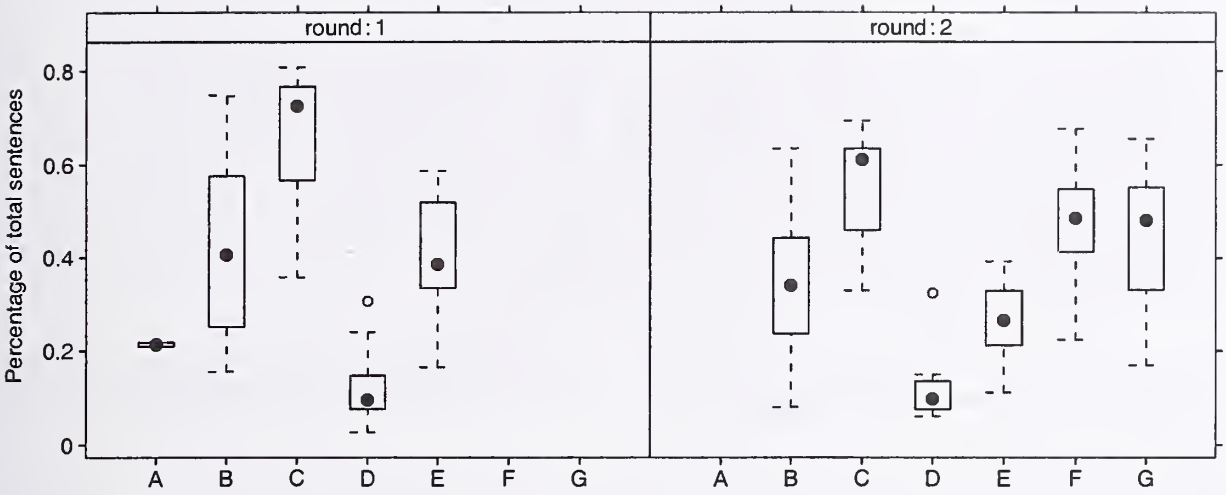
To summarize, the topics and judgments are much improved over last year. While there are differences in judging between the two assessment rounds, and between the different topic types, once again differences between assessors are dominant. Differences are more marked for relevant sentence selection than for novelty, indicating that there is a real difference between these two tasks.

4.3 Scoring

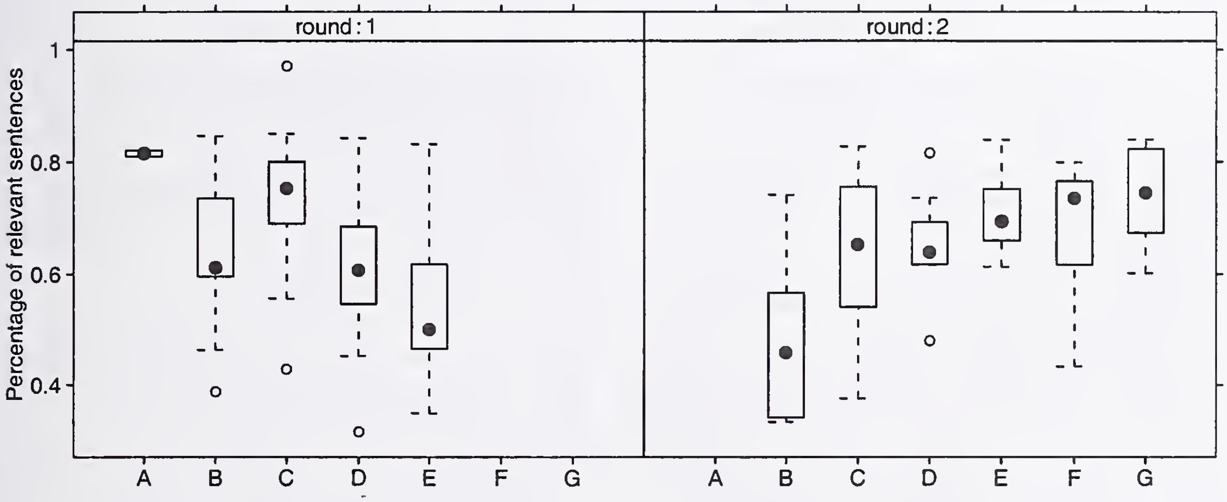
The sentences selected manually by the NIST assessors were considered the truth data. In contrast to last year, where concerns about assessors selecting groups of sentences for context drove the evaluation to use the assessor with the fewest selected relevant sentences (the so-called "minimum assessor"), this year the judgments by the topic author were taken as the truth data. The judgments by the secondary assessor were taken as a human baseline performance in the task.

Because relevant and novel sentences are returned as an unranked set in the novelty track, we cannot use traditional measures of ranked retrieval effectiveness such as mean average precision. The track guidelines specified the F measure as the primary evaluation measure for the track. The F measure (from van Rijsbergen's E measure) is itself derived from set precision and recall. For the novelty track, the "set" in question is the set of retrieved sentences (rather than documents as in the retrieval case). Relevant and novel sentence retrieval are evaluated separately. Let M be the number of matched sentences, i.e., the number of sentences selected by both the assessor and the system, A be the number of sentences selected by the assessor, and S be the number of sentences selected by the system. Then sentence set recall is M/A and precision is M/S.

As previous filtering tracks have demonstrated, set-based recall and precision do not average well, especially when the assessor set sizes vary widely across topics. Consider the following example as an illustration of the problems. One topic has hundreds of relevant sentences and the system retrieves 1 relevant sentence. The second topic has 1 relevant sentence and the system retrieves hundreds of sentences. The average for both recall and precision over these two topics is approximately .5 (the scores on the first topic are 1.0 for precision and essentially 0.0 for recall, and the scores for the second topic are the reverse), even though the system did precisely the wrong thing. While most real submissions won't exhibit this extreme behavior, the fact remains that recall and precision averaged over a set of topics is not a good diagnostic indicator of system performance. There is also the problem of how to define precision when the system returns no sentences ($S = 0$). Not counting that question in the evaluation for that run means differ-

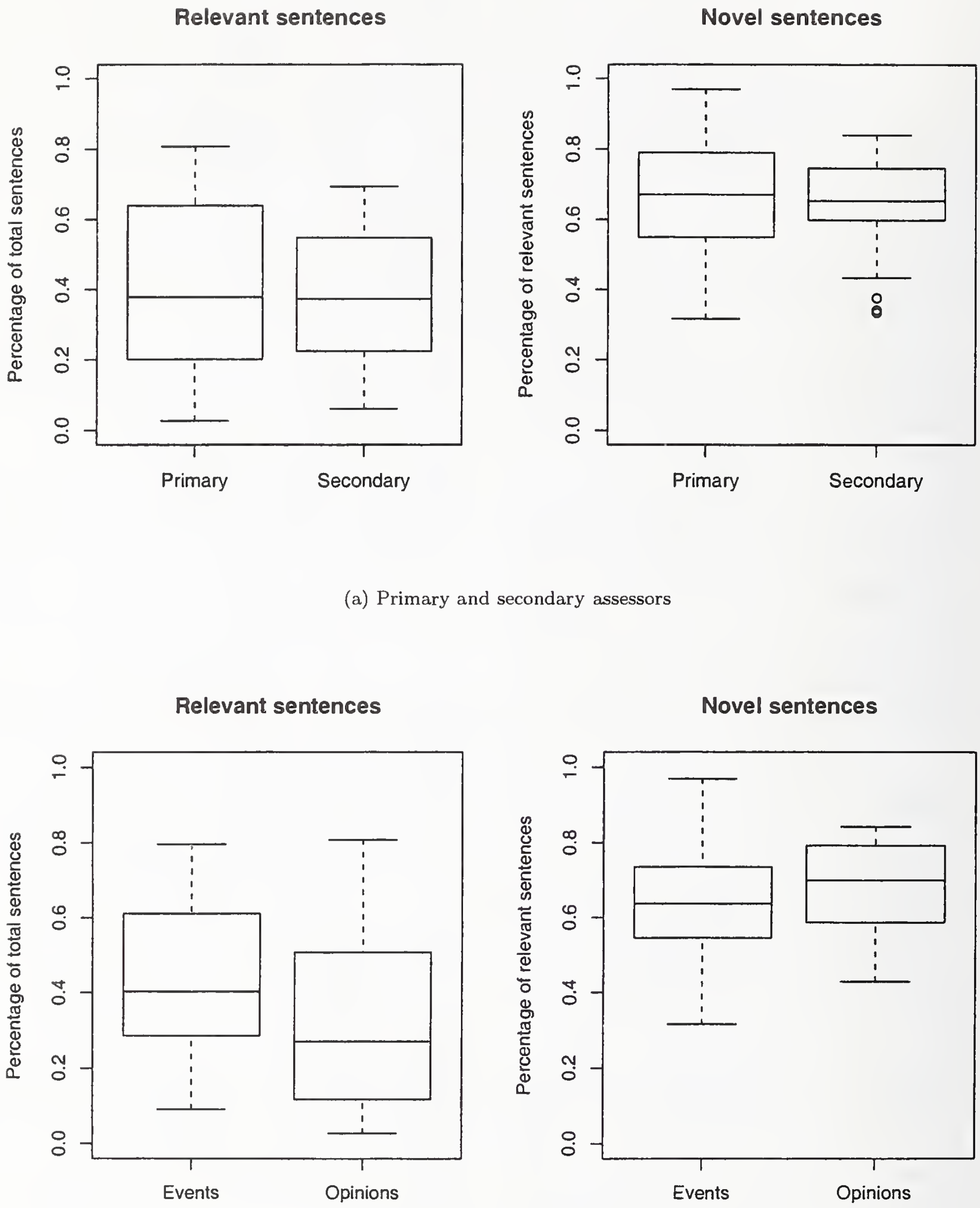


(a) Relevant sentences



(b) New sentences

Figure 2: Assessor effects.



(a) Primary and secondary assessors

(b) Event and opinion topics

Figure 3: Differences between assessment rounds and topic types.

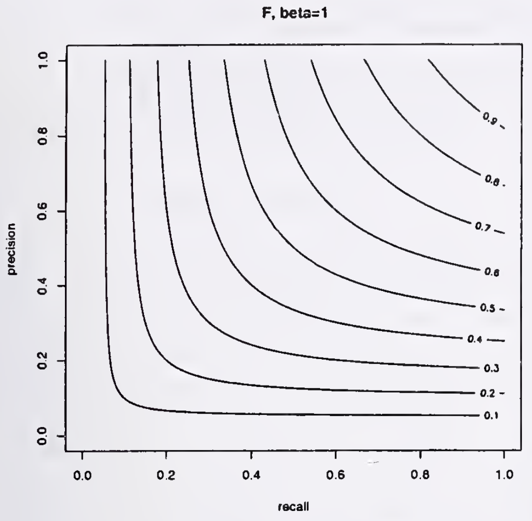


Figure 4: The F measure, plotted according to its precision and recall components. The lines show contours at intervals of 0.1 points of F.

ent systems are evaluated over different numbers of topics, while defining precision to be either 1 or 0 is extreme. (The average scores given in Appendix A defined precision to be 0 when $S = 0$ since that seems the least evil choice.)

To avoid these problems, the primary measure for novelty track runs is the F measure. This measure is a function of set recall and precision, together with a parameter β which determines the relative importance of recall and precision. A β value of 1, indicating equal weight, is used in the novelty track. $F_{\beta=1}$ is given as:

$$\mathrm{F} = \frac{2 \times \mathrm{P} \times \mathrm{R}}{\mathrm{P} + \mathrm{R}}$$

Alternatively, this can be formulated as

$$\mathrm{F} = \frac{2 \times (\#\ \text{relevant sentences retrieved})}{(\#\ \text{retrieved sentences}) + (\#\ \text{relevant sentences})}$$

For any choice of β, F lies in the range $[0, 1]$, and the average of the F measure is meaningful even when the judgment sets sizes vary widely. For example, the F measure in the scenario above is essentially 0, an intuitively appropriate score for such behavior. Using the F measure also deals with the problem of what to do when the system returns no sentences since recall is 0 and the F measure is legitimately 0 regardless of what precision is defined to be.

Note, however, that two runs with equal F scores do not indicate equal precision and recall. Figure 4 illustrates the shape of the F measure in precision-recall space. An F score of 0.5, for example, can reflect a range of precision and recall scores. Thus, two runs with equal F scores may be performing quite differently, and a difference in F scores can be due to changes in precision, recall, or both.

5 Participants

Table 2 lists the 14 groups that participated in the TREC 2003 novelty track. All but one group attempted the first task, and nearly every group tried every task. The rest of this section contains short summaries submitted by most of the groups about their approaches to the novelty task. For more details, please refer to the group's complete paper in the proceedings.

In general, most groups took a similar approach to the problem. Relevant sentences were selected by measuring similarity to the topic, and novel sentences by dissimilarty to past sentences. As can be seen from the following descriptions, there is a tremendous variation in how "the topic" and "past sentences" are modeled, and in how similarity is computed when sentences are involved. Many groups tried variations on term expansion to improve sentence similarity, some with more success than others.

5.1 CCS/University of Maryland [1]

For the 2003 DUC task of forming a summary based on the relevant and novel sentences, we tested a system based on a Hidden Markov Model (HMM). In this work, we use variations of this system on the tasks of the TREC Novelty Track. Our information retrieval system couples a query handler, a document clusterer, and a summary generator with a convenient user interface. Our summarization system uses an HMM to find relevant sentences in a document. The HMM has two types of states, corresponding to relevant and non-relevant sentences. The observation sequence scored by the HMM is composed of the number of signature terms and topic terms contained in each sentence. A signature term is a term that statistically occurs more frequently in the document set than in the document collection at large, and a subject term is a signature term which also occurs in the headlines or subject lines of a document. The counts of these terms are normalized within a document to have a mean of zero and variance of one. We determine the relevant sentences in a document based on

Table 2: Organizations participating in the TREC 2003 Novelty Track

		Runs submitted			
	Run prefix	Task 1	Task 2	Task 3	Task 4
Center for Computing Science / U. Maryland	ccsum	5	4	3	5
Chinese Academy of Sciences (CAS-ICT)	ICT	5	5	5	5
Chinese Academy of Sciences (CAS-NLPR)	NLPR	5	5	5	5
CL Research	clr	4	1	5	1
Indian Institute of Technology Bombay	IITB				1
Institut de Recherche en Informatique de Toulouse	IRIT	5	5		
LexiClone, Inc.	lexiclone	1			
Meiji University	Meiji	5	4	4	4
National Taiwan University	NTU	5	5	5	5
Tsinghua University	THU	5	3	4	5
University of Iowa	UIowa	2	5	2	5
University of Maryland Baltimore County	umbc	3	3		
University of Michigan	umich	5	5	5	5
University of Southern California-ISI	ISI	5			

the HMM posterior probability of each sentence being relevant. In particular, we choose the number of sentences to maximize the expected utility, which for TREC is simply the F1 score.

Several methods were explored to find a subset of the relevant sentences that has good coverage but low redundancy. In our multi-document summarization system, we used the QR algorithm on term-sentence matrices. For this work, we explored the use of the singular value decomposition as well as two variants of the QR algorithm.

5.2 Chinese Academy of Sciences (ICT) [11]

The novelty track can be treated as a binary classification problem: relevant sentences vs. irrelevant sentences, or new vs. non-new. In this way, we applied variants of techniques that have been employed for text categorization problem. To retrieve the relevant sentences, we compute the similarity between the topic and sentences using vector space model. The features for each topic are obtained by employing χ^2 statistic and each feature is also weighted using the χ^2 statistic. If the similarity exceeds a certain threshold, the sentence is considered as relevant. In addition, we try several techniques in an attempt to improve the performance. One is that the narrative section in the topic is analyzed to obtain the negative features and negative vector of the topic. We determine the relevance by adding similarity between the negative vector and sentence as a negative factor. The second, the threshold for different docs in each topic is dynamically adjusted according to the doc density, rather than fixed in the whole period. We have implemented the KNN algorithm and Winnow algorithm for classifying the sentences into relevant and irrelevant sentences in the novelty task 3. To detect the new sentences from the relevant sentences, we try several methods, such as Maximum Marginal Relevance (MMR) measure, Winnow algorithm and word overlapping within sentences. What's more, we attempt to detect novelty by computing semantic distance between sentences using WordNet.

5.3 Chinese Academy of Sciences (NLPR) [6]

For finding relevant sentences, we use a new statistical model called "Term Similarity Tree" to make the process of query expansion more flexible and controllable. Then, relevant feedback is used for additional modification for queries. Serveral different methods for similarity computing are developed to improve the performance. They are "simple window", "dynamic window", "active window". The key notion is that the window-based method can ensure that the closer the query words in sentences, the higher the similarity value. Finally, dynamic thresholds are used for different topics, which usually brings 1% increase of average F measure. For finding new sentences, We define a value called "New Information Degree" (NID) to present whether a sentence includes new information related to the former sentences. If the value of

NID is big, this sentence is reserved, or it will be discard. There are two different ways to define NID of the latter sentence related to the former sentence. One is based on idf value of terms and the other is based on bi-gram sequences.

5.4 CL Research [8]

The CL Research system parses and processes text into an XML representation, tagging the text with discourse, noun, verb, and preposition characteristics. The topic characterizations (titles, descriptions, and/or narratives) and the relevant documents provided by NIST were processed in this way. Componential analysis of the degree to which topic characterizations corresponded to sentences was used as the basis for determining relevance, using various scoring metrics. Similar componential analysis was used to compare each relevant sentence with all those that preceded it in order to assess novelty. Several variables were used as the basis for different runs under the different tasks (which also provided prior information that could be exploited), providing useful experimental results that will inform selection among alternatives for approaching the novelty task.

5.5 IRIT-SIG [2]

In TREC 2003, IRIT improved the strategy that was introduced in TREC 2002. A sentence is considered as relevant if it matches the topic with a certain level of coverage. This coverage depends on the category of the terms used in the texts. Three types of terms were defined for TREC 2002 highly relevant, lowly relevant and non-relevant (like stop words). In TREC 2003 we introduced a new class of terms: highly non-relevant terms. Terms from this category are extracted from the narrative parts of the queries that describe what will be a non-relevant document. A negative weight can be assigned to these words.

With regard to the novelty part, a sentence is considered as novel if its similarity with each of the previously processed and selected-as-novel sentences does not exceed a certain threshold. In addition, this sentence should not be too similar to a virtual sentence made of the n best-matching sentences.

5.6 University of Southern California-ISI

To identify opinion sentences, we used unigrams to indicate subjectivity. In addition to three baseline algorithms, we employed two sets of subjectivity-indicating words (either positive or negative valence, with appropriate strengths). One set was collected manually and extended with WordNet synonyms. The other was learned automatically from the Wall Street Journal. The words' relative scores and the algorithm's cutoff parameters were determined in a series of experiments. To our surprise the TREC results showed that one of our baselines (indicating that every sentence carries an opinion) actually beat the algorithm using the manually collected words. To identify event sentences, we adopted a standard IR procedure, treating each sentence as a separate document. For each event topic, we used all its non-stop words as query to extract event sentences. Again, the cutoff parameter was determined by experiment. We were happy to see that this method worked relatively well.

5.7 LexiClone [4]

For the sake of convenience we decided that on the word-per-word level, any language is about 58-59 percent nouns, 20 percent verbs and 20 percent adjectives. Except for prepositions, conjunctions, interjections, pronouns and other parts of speech that make up the remaining 1-2 percent, the rest of the language is a combination of these three dominant elements (or can be reduced to them). LexiClone establishes all possible combinations of nouns, verbs and adjectives for each sentence. We call these combinations "triads". (Actually, a triad is a smallest possible "key" phrase from a sentence.) After that we find sentences that have triads.

5.8 Meiji University [9]

For identifying relevant sentences, we employed following information-filtering-based approach. We regarded sentences as very short documents. Initial profiles, which are made from topic descriptions, are expanded conceptually. Conceptual fuzzy sets, which we proposed previously, are used for conceptual expansion. If the cosine similarity between the expanded profile and a word vector of each sentence exceeds a threshold, the sentence is regarded as relevant. For identifying new sentences, we considered two measures; sentence score and redundancy score. 1) For calculating a sentence score, we used N-window-idf as a time window. Local sentence score is calculable by using document frequency of past N documents. 2) Redundancy score is the maximum value of the similarity with the sentence judged to be novel in the past.

5.9 National Taiwan University [12]

According to the results of TREC 2002, we realized the major challenge issue of recognizing relevant sentences is a lack of information used in similarity computation among sentences. In TREC 2003, NTU attempts to find relevant and novel information based on variants of employing information retrieval (IR) system. We call this methodology IR with reference corpus, which can also be considered an information expansion of sentences. A sentence is considered as a query of a reference corpus, and similarity between sentences is measured in terms of the weighting vectors of document lists ranked by IR systems. Basically, we looked for relevant sentences by comparing their results on a certain information retrieval system. Two sentences are regarded as similar if they are related to the similar document lists returned by IR system. In novelty parts, similar analysis is used to compare each relevant sentence with all those that preceded it to find out novelty. An effectively dynamic threshold setting which is based on what percentage of relevant sentences is within a relevant document is presented.

5.10 Tsinghua University [14]

Research in IR group of Tsinghua University on this year's novelty track mainly focused on four aspects: (1) unsupervised relevance judgment, where QE and pseudo relevance feedback has been used. (2) efficient sentence redundancy computing: we used unsymmetrical sentence "overlap" metric, sub-topic redundancy elimination and sentence clustering. (3) supervised sentence classification, where a SVM classifier has been used and got encouraging results; (4) supervised redundancy threshold learning. A new IR system named TMiner has been built on which all experiments have been performed.

5.11 University of Iowa [3]

Our approach is basically the same as that used last year. We use new named entity and noun phrase triggering, guarded by a dual threshold of sentence similarity and full-document similarity. If the full document is sufficiently similar and the current sentence is sufficiently similar, the number of newly-detected named entities and noun phrases is compared against a minimum threshold and if the minimum is met, the current sentence is declared to be novel. The named entities used include persons, organizations and place names. Relevance is simple term similarity.

5.12 University of Maryland Baltimore County [7]

To find the relevant sentences, we used a method comprising of query expansion and sentence clustering. In the query expansion step, we experimented with two methods, one was to determine highly cooccurring terms by means of a SVD analysis and, the other was by determining meaningful terms as obtained by a language analysis of the narrative section for each topic. The sentences, per topic, were clustered and the top clusters were selected based on similarity scores of the cluster centroids and the expanded query. All the sentences from the selected clusters are chosen as the relevant sentences.

To find the novel sentences, we experimented with two methods. One, based on a text summarization method, was clustering relevant sentences and choosing one sentence each from the selected clusters to make up the set of novel sentences. In the second method, using a sentence-sentence similarity matrix (of relevant sentences), the dissimilarity between sentences was used to determine novel sentences.

5.13 University of Michigan [10]

First we used the MEAD summarization software to compute scores for each sentence on features such as length, position, word overlap with query, title and description. Since we trained maximum entropy classifiers, these scores were then discretized. Once the MEAD features were calculated, discretized and formatted, we used the maxent-2.1.0 software to train our models for novel and relevant sentences.

For tasks 1 and 3, once the maxent models had been trained for classifying novel and relevant sentences and were used to produce a ranked list of sentences as to how likely they were to be novel or relevant, we then chose differing percentage cut offs for each run in an attempt to maximize recall and precision on our devtest data set. For tasks 2 and 4, we noted that the F-measure for a baseline algorithm of submitting all relevant sentences as being novel was quite high. Therefore, we focused on trying various discretizations of our feature scores in order to improve the classifier's performance on the devtest set

6 Results

Figures 5, 7, 8, and 9 show the average F scores in each task. Task 1 scores are shown alongside the "scores" of the secondary assessor, who may be considered to have been performing this task. Within the margin of error of human disagreement, these lines can be thought of as representing the best possible performance. The best systems are performing at this level. Nine runs have novelty F scores of 0 because those runs did not return any novel sentences.

Tasks 1 and 3 show novelty retrieval performance closely tracking relevant retrieval performance. Only a few runs near the bottom of the performance range did better at retrieving novel sentences than relevant ones. This seems somewhat surprising, since while the retrieved set of relevant sentences places a bound on recall for the novel set (since only retrieved sentences can be labeled novel), any level of precision is possible, and thus there isn't any reason why F_{novel} shouldn't exceed $F_{relevant}$. However, to achieve this most systems would have had to make a very large improvement in precision when retrieving novel sentences.

As stated previously, sometimes it can be hard to understand what the F score means in terms of the actual behavior of each run. Figure 6 shows the F scores for task 1, along with each run's corresponding average recall and precision. Note for example the run ISIALL03 (run #11 on the x axis), which retrieved only relevant sentences, and retrieved all of them; for this run, average recall was 1.0 but precision was 0.41. It is very interesting to note that average recall seems to correlate more closely to the F scores, although F is defined to be a harmonic mean between the two. This may mean that within each run, recall was more consistent across topics than was precision.

The scores for tasks 2–4 show how many of the systems can take advantage of training data, both for relevance and novelty. Comparing the graph of tasks 2 and 3, we can see that having more relevance information dramatically improves novelty retrieval effectiveness. Moreover, comparing tasks 2 and 4, we can see that having relevant sentences is more valuable than having novel sentences for training, since the top systems do not improve from task 2 to task 4.

The graphs for tasks 2 and 4 compare the runs against a baseline system which merely returns all the relevant sentences (provided as training data in these tasks) as novel. The best systems are performing above this baseline, indicating that they are being somewhat selective in what they return as novel.

Event topics were easier than opinion topics. Figure 10 illustrates this phenomenon in task 1. Relevant sentence retrieval scores are on the left, novelty retrieval scores on the right. The graphs show the overall average along with the averages for event and opinion topics for each run. Nearly every run did better at events than opinions; the exceptions are UMBC and NTU for relevant sentences, and NTU and one IRIT run for novel sentences.

As the systems receive more relevant sentences as training data, they improve on opinion topics. In task 3 (where systems received some relevant and novel training data), all systems perform as well or better on event topics than on opinions. However, in tasks 2 and 4, where the systems receive complete relevance information, the situation is reversed: all systems do better on opinion topics. Clearly, the systems are less able to identify relevant sentences in opinion topics, but if they know which ones are relevant, they do better on opinion topics than on events. Having a small amount of relevant sentence training data (as in task 3) is not sufficient to boost a system's overall performance.

References

[1] J. M. Conroy, D. M. Dunlavy, and D. P. O'Leary. From TREC to DUC to TREC again. In Voorhees and Harman [13].

[2] T. Dkaki and J. Mothe. TREC novelty track at IRIT-SIG. In Voorhees and Harman [13].

[3] D. Eichmann, P. Srinivasan, M. Light, H. Wang, X. Y. Qiu, R. J. Arens, and A. Sehgal. Experiments in novelty, genes and questions at the University of Iowa. In Voorhees and Harman [13].

[4] I. S. Geller. The role and meaning of predicative and non-predicative definitions in the search for information. In Voorhees and Harman [13].

[5] Donna Harman. Overview of the TREC 2002 novelty track. In E. M. Voorhees and D. K. Harman, editors, *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002)*, NIST Special Publication 500-251, pages 46–55, Gaithersburg, MD, November 2002.

[6] Q. Jin, J. Zhao, and B. Xu. NLPR at TREC 2003: Novelty and robust. In Voorhees and Harman [13].

[7] S. Kallukar, Y. Shi, R.S. Cost, C. Nicholas, A. Java, C. James, S. Rajavaram, V. Shanbhag, S. Bhatkar, and D. Ogle. UMBC at trec 12. In Voorhees and Harman [13].

[8] K. C. Litkowski. Use of metadata for question answering and novelty tasks. In Voorhees and Harman [13].

[9] R. Ohgaya, A. Shimmura, and T. Takagi. Meiji university web and novelty track experiements at TREC 2003. In Voorhees and Harman [13].

[10] J. Otterbacher, H. Qi, A. Hakim, and D. Radev. The University of Michigan at TREC 2003. In Voorhees and Harman [13].

[11] J. Sun, Z. Yang, W. Pan, H. Zhang, B. Wang, and X. Cheng. TREC 2003 novelty and web track at ICT. In Voorhees and Harman [13].

[12] M.-F. Tsai, M.-H. Hsu, and H.-H. Chen. Approach of information retrieval with reference corpus to novelty detection. In Voorhees and Harman [13].

[13] E. M. Voorhees and D. K. Harman, editors. *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*, Gaithersburg, MD, 2003.

[14] M. Zhang, C. Lin, Y. Liu, L. Zhao, and S. Ma. THUIR at TREC 2003: Novelty, robust and web. In Voorhees and Harman [13].

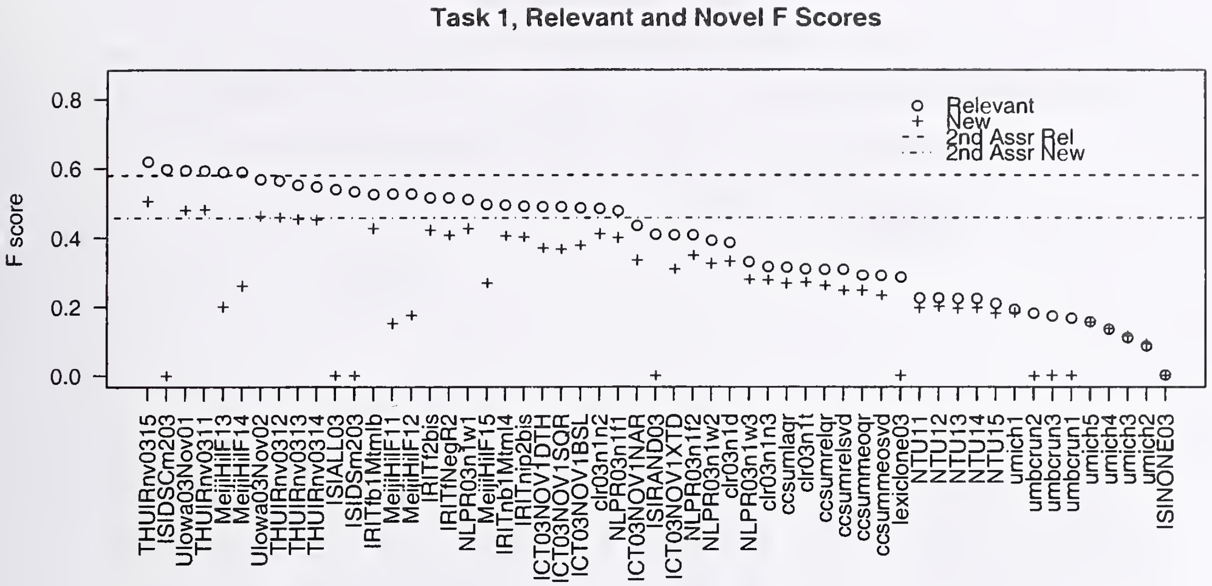


Figure 5: Scores for Task 1, along with the "average score" of the secondary assessor.

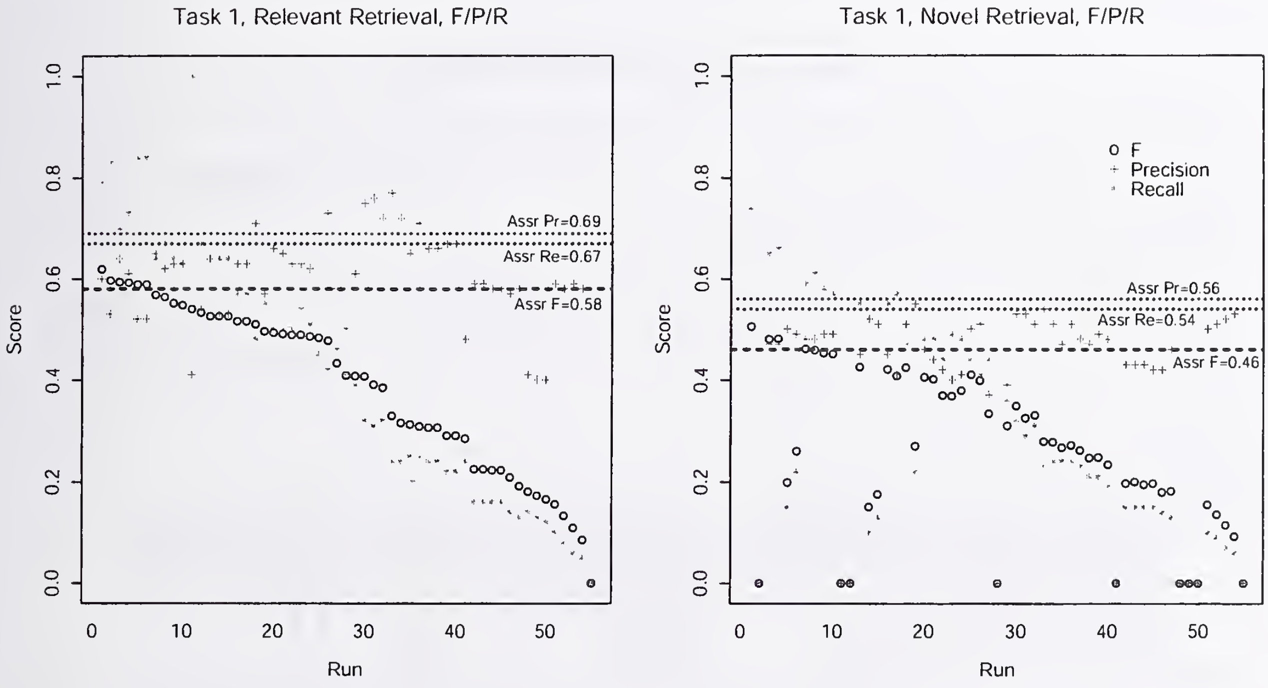


Figure 6: Task 1 relevant and novel F scores, with corresponding precision and recall.

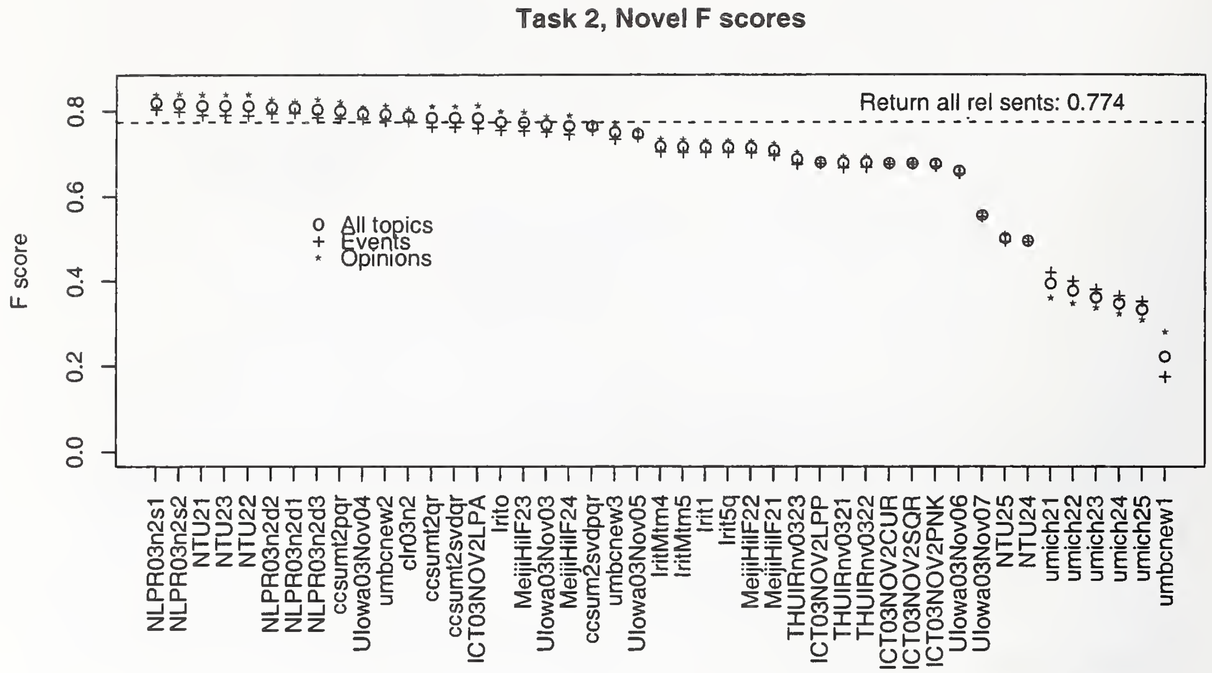


Figure 7: Scores for Task 2, against a baseline of returning all relevant sentences as novel.

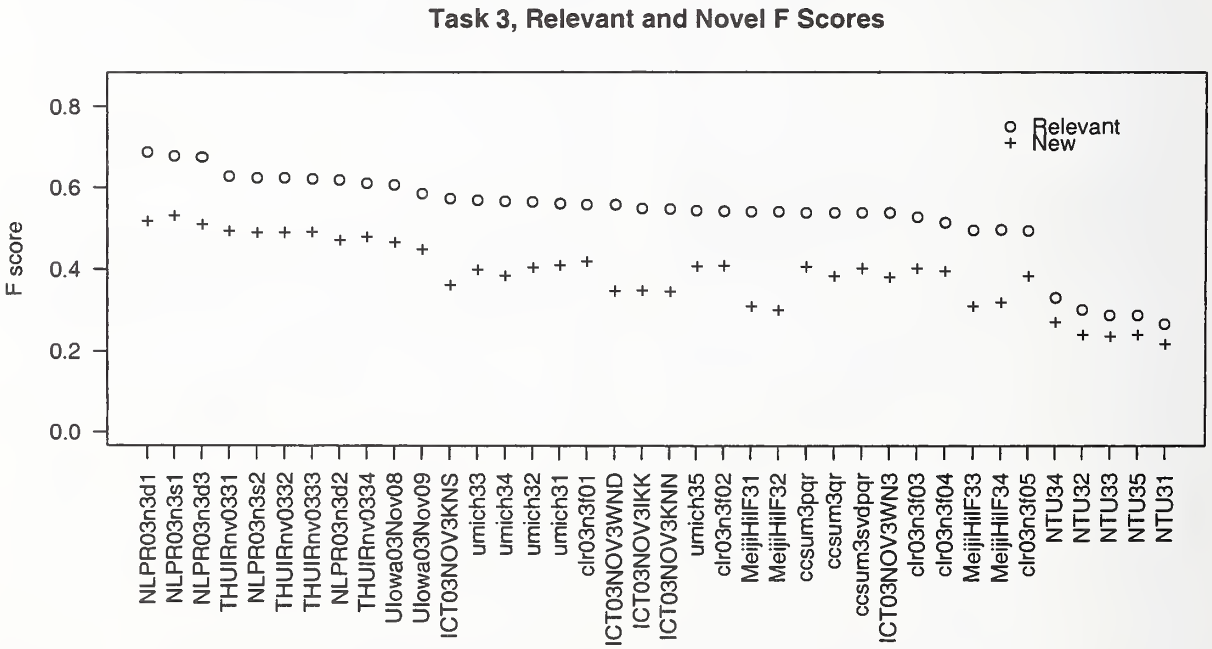


Figure 8: Scores for Task 3.

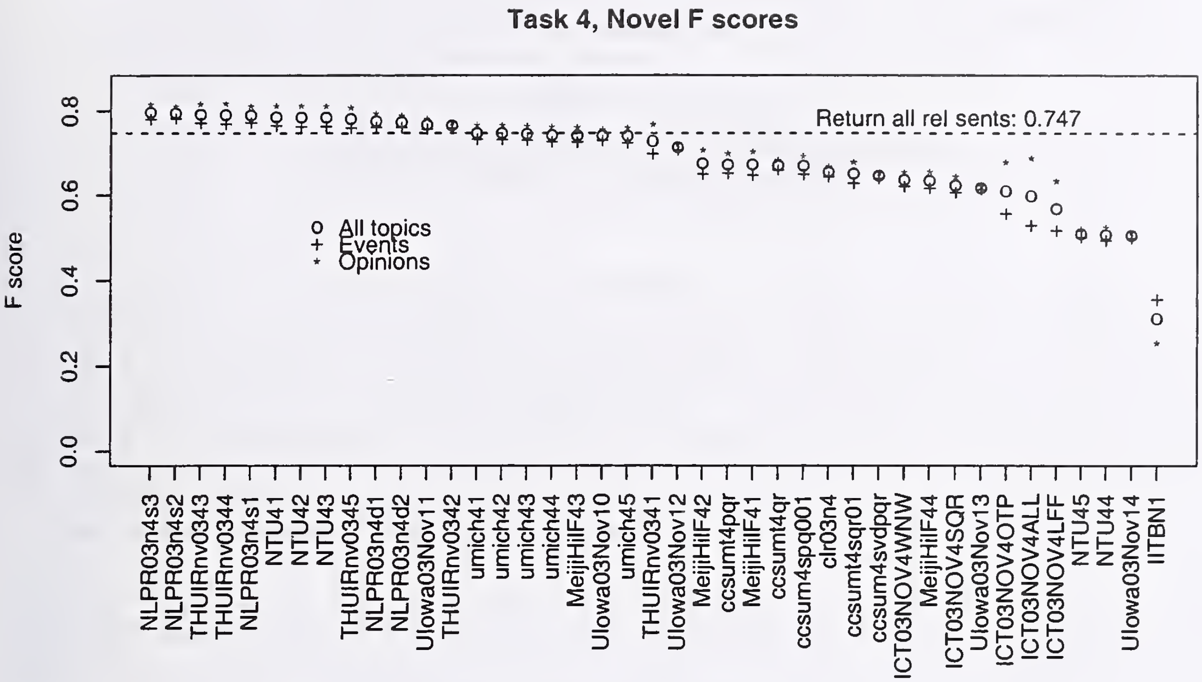


Figure 9: Scores for Task 4, against a baseline of returning all relevant sentences as novel.

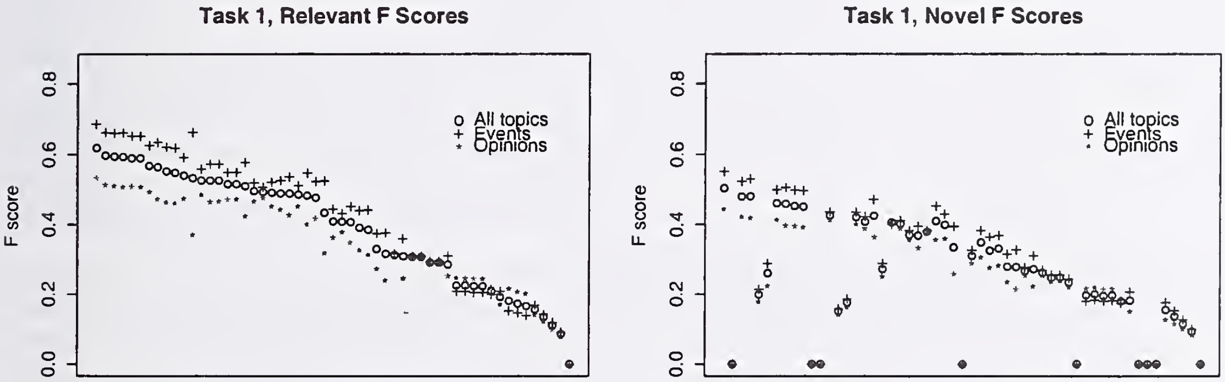


Figure 10: Scores for task 1, broken down by topic type. Runs are along the X axis; the run names have been omitted for readability, but the runs are in the same order as in Figure 5.

Overview of the TREC 2003 Question Answering Track

Ellen M. Voorhees
National Institute of Standards and Technology
Gaithersburg, MD 20899

Abstract

The TREC 2003 question answering track contained two tasks, the passages task and the main task. In the passages task, systems returned a single text snippet in response to factoid questions; the evaluation metric was the number of snippets that contained a correct answer. The main task contained three separate types of questions, factoid questions, list questions, and definition questions. Each of the questions was tagged as to its type and the different question types were evaluated separately. The final score for a main task run was a combination of the scores for the separate question types.

This paper defines the various tasks included in the track and reports the evaluation results. Since the TREC 2003 track was the first time for significant participation in the definition and list subtasks, the paper also examines the reliability of the evaluation for these tasks.

TREC introduced the first question answering (QA) track in TREC-8 (1999). The goal of the track is to foster research on systems that retrieve answers rather than documents in response to a question, with particular emphasis on systems that can function in unrestricted domains. The tasks in the track have evolved over the years to focus research on particular aspects of the problem deemed important to improving the state-of-the-art.

The task in the original QA tracks required systems to return text snippets drawn from a large corpus of newspaper articles in response to closed-class or factoid questions such as *Who invented the paper clip?*. Each response was judged by a human assessor; a response was marked correct if an answer to the question was contained within the snippet. Unfortunately, the relative effectiveness of different systems was masked by the fact that two different snippets could both contain a correct answer while one was a significantly better response then the other [4]. To force systems to demonstrate their ability to locate the actual answer, the TREC 2002 task required systems to return *exact answers*, text strings consisting of a complete answer and nothing else. Strings that contained a right answer with additional text were judged to be "inexact" and did not contribute to a system's score.

Pinpointing the precise extent of an answer is a more difficult problem than finding a text segment that contains an answer, and there are applications of QA technology that do not require this extra step. To provide a forum for research groups interested in these applications, the TREC 2003 track included a "passages" task that allowed text segments containing answers to be returned. The other task in the track, the main task, required exact responses. While the test set of questions for the passages task contained only factoid questions, the main task contained list and definition questions as well as factoid questions. Each question type was evaluated separately, and the final score for a main task run was a combination of the scores for the three questions types.

This paper provides an overview of the results of the TREC 2003 QA track. The first two sections describe the two tasks in the track and present the evaluation results for the tasks. Since the TREC 2003 track was the first time for significant participation in the definition and list subtasks, Section 4 examines the reliability of the evaluation used for these tasks. This analysis demonstrates that the evaluation results for the definition task must be interpreted with care as using different assessors can cause substantial changes in the relative evaluation scores. Using more questions in the definition test set should increase the stability of the evaluation.

1 The Passages Task

The passages task tested a system's ability to find an answer to a factoid question within a relatively short (250 characters) span of text. In contrast to the first years of the QA track, each text span returned by the system was required to be an extract from a document in the corpus—results files submitted to NIST contained the offset from the beginning of the document of the first character in the span plus the span length rather than actual strings. Requiring

Гable 1: Evaluation scores for the best passages task run from each group that submitted a passages run.

Run Tag	Submitter	Accuracy	NIL Prec	NIL Recall
LCCpass03	Language Computer Corp.	0.685	0.381	0.800
nuslamp03a	National University of Singapore (Lee)	0.419	0.156	0.333
uwmtCQ2	University of Waterloo (MultiText)	0.351	—	—
umassql	University of Massachusetts	0.201	—	—
answfind1	Macquarie University	0.191	—	—
Saarland	Saarland University	0.169	0.097	0.367
IITBQA1	Indian Institute of Technology Bombay	0.133	0.045	0.100
clr03p2	CL Research	0.119	0.109	0.233
UAmsT03P1	University of Amsterdam	0.111	0.128	0.333
pircsqa3	Queens College, CUNY	0.097	0.000	0.000
NSIR	University of Michigan	0.085	0.075	0.100

extracts rather than allowing arbitrary 250-character strings resolves many of the issues surrounding correct answers contained within very poor responses. As in previous years, all processing was required to be completely automatic with no changes to the system permitted once the test questions were released.

The document collection used as the source of answers was the same collection used in the TREC 2002 QA track, the AQUAINT Corpus of English News Text. This collection consists of documents from three different sources: the AP newswire from 1998–2000, the New York Times newswire from 1998–2000, and the (English portion of the) Xinhua News Agency from 1996–2000. There are approximately 1,033,000 documents and 3 gigabytes of text in the collection. The test set of questions contained 413 questions drawn from AOL and MSNSearch logs[1]. Thirty of the questions have no known correct answer in the document collection.

A passages task run consisted of exactly one response for each of the test questions. A response was either a specification of a document extract or the string "NIL" used to indicate the system's belief that there was no correct answer in the collection. NIL was the (only) correct response for the 30 questions with no known answer in the document collection. Each extract was independently judged as correct, unsupported, or incorrect by two human assessors. When the two judgments differed, an adjudicator made the final decision. An extract was judged correct if it contained a right answer to the question, if the document from which it was drawn made it clear that it was a right answer, and if the extract was responsive. Extracts that contain multiple entities of the same semantic category as the correct answer but do not indicate which of those entities is the actual answer (e.g., an extract containing multiple names in response to a who question) are not responsive. Extracts that do not include appropriate units (e.g., "500" instead of "500 meters") are also not responsive. Finally, extracts must refer to the famous entity itself and not imitations, copies, and the like to be responsive to questions about a famous entity (e.g., extracts about the Taj Mahal casino are not responsive to questions regarding "the Taj Mahal") [6]. An extract was judged unsupported if it contained a right answer and was responsive, but the document from which it was drawn does not indicate that it is a right answer. Otherwise, an extract was judged as incorrect.

The main evaluation score for a passages task run is *accuracy*, the fraction of questions judged correct. Also reported are the recall and precision of recognizing when no answer exists in the document collection. Precision of recognizing no answer is the ratio of the number of times NIL was returned and correct to the number of times it was returned; recall is the ratio of the number of times NIL was returned and correct to the number of times it was correct (30).

Twenty-one passages task runs from eleven different participating groups were submitted to the QA track. Table 1 gives evaluation results for the most accurate run from each group. The table includes the run tag, the group that submitted the run, and the accuracy, NIL precision, and NIL recall scores. The NIL precision and recall scores are reported as "—" if a run never returned NIL.

[1] The logs from which questions were drawn were generously donated by Abdur Chowdhury of AOL and Sue Dumais of Microsoft Research.

Table 2: Evaluation scores for the runs with the best factoid component.

Run Tag	Submitter	Accuracy	NIL Prec	NIL Recall
LCCmainE03	Language Computer Corp.	0.700	0.381	0.800
lexiclone92	LexiClone	0.622	—	—
nusmml03r1	National University of Singapore (Yang)	0.562	0.160	0.400
isi03a	University of Southern California, ISI	0.337	0.071	0.200
IBM2003a	IBM Research (Prager)	0.298	0.082	0.233
MITCSAIL03b	Massachusetts Institute of Technology	0.295	0.258	0.267
uwbqitekat03	University of Wales, Bangor	0.259	0.092	0.967
Albany03I2	University of Albany	0.240	0.109	0.200
irstqa2003w	ITC-irst	0.235	0.121	0.267
BBN2003B	BBN	0.208	0.068	0.100
FDUT12QA1	Fudan University	0.194	0.077	0.233
ntt2003qam1	NTT Communication Science Labs	0.150	0.090	0.267
MITRE2003A	MITRE Corp.	0.148	0.000	0.000
ICTQA2003C	Chinese Academy of Sciences (CAS-ICT)	0.145	0.105	0.467
UAmsT03M2	University of Amsterdam	0.145	0.273	0.100

2 The Main Task

The main task of the QA track involved three types of questions, factoids, lists, and definitions. Each question was tagged as to its type, and the response formats and evaluation methods differed for each type. The factoid questions for the main task were the same set of 413 questions used in the passages task, and the AQUAINT corpus was used for all subtasks. As in the passages task, completely automatic processing was required.

Fifty-four main task runs from 25 different groups were submitted to the track. Three groups, CL Research, Language Computer Corp., and the University of Amsterdam, submitted both passages and main task runs.

2.1 Factoids

The factoid component of the main task was very similar to the passages task. As noted, the same document and question set were used as in the passages task, and systems returned exactly one response for each factoid question. For the main task, however, systems were required to return an exact answer rather than an extract containing an answer. The answer strings returned by the systems were not required to be an extract from a document; the response for a question was of the form `query-id run-tag doc-id answer-string`. If the system believed there was no correct response in the document collection, `doc-id` was set to NIL and `answer-string` was empty.

Responses were judged as in the passages task except a fourth judgment, not exact, could also be assigned. A judgment was assigned to a response in the following order:

incorrect: the answer string does not contain a right answer or the answer is not responsive;

not supported: the answer string contains a right answer but the document returned does not support that answer;

not exact: the answer string contains a right answer and the document supports that answer, but the string contains more than just the answer (or is missing bits of the answer);

correct: the answer string consists of exactly a right answer and that answer is supported by the document returned.

The score for the factoid component of the main task was accuracy, the fraction of responses judged correct. Table 2 gives evaluation results for the factoid component corresponding to Table 1 for passages. The table shows the most accurate run for the factoid subtask for each of the top 15 groups, and includes the the accuracy, NIL precision, and NIL recall scores.

1915: List the names of chewing gums.

Stimorol
Dirol
Doublemint
Juicy Fruit
Orbit
Trident
Dentyne
Big Red
Winterfresh
Spearmint
Freedent
Chiclets
Double Bubble
Bazooka
Hubba Bubba
Nicorette

Figure 1: Answer list for list question 1915 — names of chewing gums found within the AQUAINT corpus.

2.2 Lists

The list task was offered in both the TREC 2001 and the TREC 2002 QA tracks. However, participation in the task has been quite limited as groups concentrated on the main (factoid) task. Encouraging additional participation in other subtasks was the primary motivation for a combined main task in TREC 2003.

The list task requires systems to assemble an answer from information located in multiple documents. In TREC, a list question asks for different instances of a particular kind of information to be retrieved, such as *List the names of chewing gums*. List questions can be thought of as a shorthand for asking the same factoid question multiple times; the set of answers that satisfy the factoid question is the appropriate response for the list question. A system's response to a list question was an unordered set of [*document-id*, *answer-string*] pairs such that each *answer-string* was considered an instance of the requested type.

Unlike the previous two times the list task was run in TREC, this year's list questions did not specify a target number of instances to return. Instead, system were expected to return all of the correct, distinct answers contained in the document collection. Different questions had different numbers of distinct answers, ranging from a low of 3 (*What Chinese provinces have a McDonald's restaurant?*) to a high of 44 (*Which countries were visited by first lady Hillary Clinton?*).

The 37 list questions used in the list subtask were constructed by NIST assessors. The assessors were instructed to construct questions whose answers would be a list of entities (people, places, dates, numbers) such that the list would not likely be found in a reference work such as a gazetteer or almanac. They searched the document collection using the PRISE search engine to find as complete a list of instances as possible. A single document could contain multiple instances, and the same instance might be repeated in multiple documents. After the participants' results were returned to NIST, the assessor who created the question judged the set of responses for that question. If a system found a correct response that the assessor had not found during question development, the system's answer was added to the list of known instances for that question. The final list of known instances was considered the correct answer to the question and was used to score the systems' responses. Figure 1 shows the final list of answers for the chewing gum question.

Judgments of incorrect, not supported, inexact, or correct were made individually for each [*document-id*, *answer-string*] pair as for the factoid subtask. The assessor was given one run's list at a time, and while judging for correctness he also marked a set of responses as distinct. The assessor arbitrarily chose any one of a set of equivalent responses to mark as the distinct one, and marked the remainder as not distinct. Only correct responses could be marked distinct.

The instance precision (IP) and instance recall (IR) for a list question can be computed from the final answer list and the assessor judgments. Let S by the size of the final answer list (i.e., the number of known answers), D be the number of correct, distinct responses returned by the system, and N be the total number of responses returned by the system. Then $IP = D/N$ and $IR = D/S$. Precision and recall were then combined using the F measure with equal weight given to recall and precision:

$$F = \frac{2 * IP * IR}{(IP + IR)}$$

The score for the list component was the average F score over the 37 list questions. Table 3 gives the average F scores for 15 of the main task submissions. The table gives the run with the best list component score for each of the top 15 groups.

Table 3: Average F scores for the list question subtask. Scores are given for the best run from the top 15 groups.

Run Tag	Submitter	F
LCCmainS03	Language Computer Corp.	0.396
nusmml03r2	National University of Singapore (Yang)	0.319
MITCSAIL03c	Massachusetts Institute of Technology	0.134
isi03a	University of Southern California, ISI	0.118
BBN2003B	BBN	0.097
Albany03I4	University of Albany	0.096
ICTQA2003A	Chinese Academy of Sciences (CAS-ICT)	0.091
FDUT12QA1	Fudan University	0.088
IBM2003c	IBM Research (Prager)	0.077
irstqa2003w	ITC-irst	0.076
MITRE2003A	Mitre Corp.	0.069
UAmsT03M1	University of Amsterdam	0.054
CMUJAV2003	Carnegie Mellon University (Javelin)	0.052
lexiclone92	LexiClone	0.048
ntt2003qaml	NTT Communication Science Labs	0.040

2.3 Definitions

Definition questions are questions such as *Who is Colin Powell?* or *What is mold?*. Definition questions occur relatively frequently in logs of web search engines [4] suggesting they are an important type of question. However, evaluating systems that answer definition questions is much more difficult than evaluating systems that answer factoid questions because it is no longer useful to judge a system response as simply right or wrong. Assigning partial credit to a system response requires some mechanism for matching the concepts in the desired response to the concepts present in the system's response. The issues are similar to those that arise in the evaluation of machine translation and automatic summarization.

A small pilot evaluation of definition questions was held as part of the ARDA AQUAINT program in the fall of 2002 [5]. The lessons learned from the pilot were used to help define the definitions component of the main task, which was the first first large scale evaluation of definition questions.

The definition question test set contained 50 questions. The questions were drawn from the same set of search engine logs that the factoid questions were drawn from. Assessors selected a question from the log, and searched the document collection for information about the target. The final set of questions contained 30 questions for which the target was a (perhaps fictional) person (e.g., Vlad the Impaler, Andrea Bocceli, Ben Hur), 10 questions for which the target was an organization (e.g., Bausch & Lomb, Friends of the Earth, Freddie Mac), and 10 questions for which the target was some other thing (e.g., a golden parachute, feng shui, TB). If the question had a qualification in the log, the qualification remained in the test set (e.g., *What is Ph in biology?*).

In evaluations such as TREC, questions are asked in isolation. This is not much of an issue for factoid questions, but becomes much more of an issue for definition questions. Without any idea of who the questioner is and why he or she is asking the question it is essentially impossible for a system to decide what level of detail in a response is appropriate—presumably an elementary-school-aged child and a nuclear physicist should receive different answers for at least some questions. To provide some guidance for the system developers, the following scenario was assumed for definition questions:

> The questioner is an adult, a native speaker of English, and an "average" reader of US newspapers. In reading an article, the user has come across a term that they would like to find out more about. They may have some basic idea of what the term means either from the context of the article (for example, a bandicoot must be a type of animal) or basic background knowledge (Ulysses S. Grant was a US president). They are not experts in the domain of the target, and therefore are not seeking esoteric details (e.g., not a zoologist looking to distinguish the different species in genus Perameles).

As in the list task, a system returned an unordered set of [*document-id*, *answer-string*] pairs as a response for a definition question. Each string was presumed to be a facet in the definition of the target. There were no limits placed on either the length of an individual answer string or on the number of pairs in the list, though systems were penalized for retrieving extraneous information.

Judging the quality of the systems' responses was done in two steps. In the first step, all of the answer-strings from all of responses were presented to the assessor in a single (long) list. Using these responses and his own research done during question development, the assessor first created a list of "information nuggets" about the target. An information nugget was defined as a fact for which the assessor could make a binary decision as to whether a response contained the nugget. At the end of this step, the assessor decided which nuggets were vital—nuggets that must appear in a definition for that definition to be good. The assessor went on to the second step once the nugget list was created. In this step the assessor went through each of the system responses in turn and marked where each nugget appeared in the response. If a system returned a particular nugget more than once, it was marked only once.

Figure 2 shows an example of how one response was judged for the question *What is a golden parachute?*. The top of the figure shows the nugget list developed by the assessor with the vital nuggets indicated. The bottom of the figure shows a system response with the nugget numbers of the nuggets that were assigned to the response to the left of the individual strings that match the nugget. For example, the assessor matched nugget 1 to item b, and nugget 4 to item i.

The judging of the systems' responses was designed in this way to make the evaluation depend only on the *content* of a system response, and not on the particular structure of a system response. Assessors ignored wording differences, making conceptual matches between the system responses and their nuggets, not syntactic matches (that's why human assessors were needed!). A single answer string within the system's response was allowed to match multiple nuggets such as for string 'a' in Figure 2). The design does depend on the assessor nuggets truly being atomic so that a nugget isn't split across system's strings. Occasionally in the judging the assessors found that a nugget wasn't really atomic; in these cases they tended to assign the nugget to a string that had the main part of the concept.

Given the judgments as described above, it is straightforward to compute the nugget recall of a response: it is simply the ratio between the number of correctly retrieved nuggets to the number of nuggets on the assessor's list. But the corresponding measure of nugget precision, the ratio between the number of nuggets correctly retrieved to the total number of nuggets retrieved, is problematic since the correct value for the denominator is unknown. A trial evaluation prior to the pilot showed that assessors found enumerating *all* concepts represented in a response to be so difficult as to be unworkable. For example, how many information units are contained in the string "`Oh, Eaton also has a new golden parachute clause in his contract.''`? Using only nugget recall as a final score is untenable since systems would not be rewarded for being selective. Retrieving the entire document collection is guaranteed to give a perfect recall score for every question.

Borrowing from the evaluation of summarization systems [1], we can use length as a crude approximation to precision. A length-based measure captures the intuition that users would prefer the shorter of two definitions that contain the same concepts. The length-based measure used in both the pilot and the TREC track gives a system an allowance of 100 (non-white-space) characters for each correct nugget it retrieved. The precision score is set to one if the response is no longer than this allowance. If the response is longer than the allowance, the precision score is downgraded using the function precision $= 1 - \frac{\text{length}-\text{allowance}}{\text{length}}$.

Remember that the assessors marked some nuggets as vital and the remainder are not vital. The non-vital nuggets act as a "don't care" condition. That is, systems should be penalized for not retrieving vital nuggets, and penalized for retrieving items that are not on the assessor's nugget list at all, but should be neither penalized nor rewarded for retrieving a non-vital nugget. To implement the don't care condition, nugget recall is computed only over vital nuggets, while the character allowance in the precision computation is based on both vital and non-vital nuggets. The recall for the example in Figure 2 is thus 3/3, and the character allowance is 500.

The final score for a definition response was computed using the F-measure as it was for list questions. For the definition questions, however, the β parameter was set to five indicating that recall was five times as important as precision. The value of five is arbitrary, but reflects both the emphasis given to recall by the assessors in the pilot version of the task, and acknowledges the crudeness of the length approximation to true precision. Figure 3 shows the complete computation of the F($\beta = 5$) score for a single definition question. The score for the definition component of the main task was the average F($\beta = 5$) score over the 50 definition questions.

Table 4 gives the average F($\beta = 5$) score for the best scoring run from a group for the top 15 groups. The table also gives the average length of a definition response for the run.

1	vital	Agreement between companies and top executives
2	vital	Provides remuneration to executives who lose jobs
3	vital	Remuneration is usually very generous
4		Encourages execs not to resist takeover beneficial to shareholders
5		Incentive for execs to join companies
6		Arrangement for which IRS can impose excise tax

a) nugget list

Nuggets		Answer Strings
2, 3	a.	`The arrangement, which includes lucrative stock options, a hefty salary, and a "golden parachute" if Gifford is fired,`
1	b.	`Oh, Eaton also has a new golden parachute clause in his contract.`
	c.	`But some, including many of BofA’s top executives, joined the 216 and cashed in their "golden parachute" severance packages.`
	d.	`The big payment that Eyler received in January was intended as a "golden parachute"`
	e.	`Cotsakos’ contract included a golden parachute big enough to make a future sale of the company more likely`
	f.	`syndication, the golden parachute for production companies`
6	g.	`But if he quits or is dismissed during the two years after the merger, he will be paid $24.4 million, with DaimlerChrysler paying the "golden parachute" tax for him and the taxes on the compensation paid to cover the tax.`
	h.	`If he left, On leaving, O’Neill could would be able to collect a golden parachute package providing three years of salary and bonuses, stock and other benefits.`
4	i.	`After the takeover, as jobs disappeared and BofA’s stock tumbled, many saw him as a bumbler who had sold out his bank, walking away with a golden parachute that gives him $5 million a year for the rest of his life.`
	j.	`And after BofA disclosed that he had a golden parachute agreement giving him some $50 million to $100 million if he left following the merger, he sent a voice mail message to bank employees that he intended to stay.`

b) system response

Figure 2: Assessor annotation of a sample response for definition question 1905 *What is a golden parachute?*

Let r be the number of vital nuggets returned in a response;

a be the number of acceptable (non-vital but on list) nuggets returned in a response;

R be the total number of vital nuggets in the assessor's list;

len be of the number of non-white space characters in an answer string summed over all answer strings in the response;

Then

$$\text{recall} = r/R$$

$$\text{allowance} = 100 * (r + a)$$

$$\text{precision} = \begin{cases} 1 & \text{if len} < \text{allowance} \\ 1 - \frac{\text{len} - \text{allowance}}{\text{len}} & \text{otherwise} \end{cases}$$

$$F(\beta = 5) = \frac{26 * \text{precision} * \text{recall}}{25 * \text{precision} + \text{recall}}$$

Figure 3: Computation of the F($\beta = 5$) score for a definition question.

Table 4: Average F($\beta = 5$) scores for the definition questions subtask. Scores are given for the best run from the top 15 groups. Also given is the average length of a response for the run measured in non-white-space characters.

Run Tag	Submitter	F($\beta = 5$)	Ave Length
BBN2003C	BBN	0.555	2059.20
nusmml03r2	National University of Singapore (Yang)	0.473	1478.74
isi03a	University of Southern California, ISI	0.461	1404.78
LCCmainS03	Language Computer Corp.	0.442	1407.82
cuaqdef2003	Univ. of Colorado/Columbia Univ.	0.338	1685.60
irstqa2003d	ITC-irst	0.318	431.26
UAmsT03M1	University of Amsterdam	0.315	2815.08
MITCSAIL03a	Massachusetts Institute of Technology	0.309	620.28
shef12simple	University of Sheffield	0.236	338.42
UIowaQA0303	University of Iowa	0.231	3039.26
CMUJAV2003	Carnegie Mellon University (Javelin)	0.216	182.34
FDUT12QA3	Fudan University	0.192	203.54
piq002	University of Pisa	0.185	89.52
IBM2003b	IBM Research (Prager)	0.177	223.16
ntt2003qam1	NTT Communication Science Labs	0.169	2219.24

Table 5: Component scores and final combined scores for main task runs. Scores are given for the best run from the top 15 groups.

Run Tag	Submitter	Component Score			Final
		Factoid	List	Def	Score
LCCmainS03	Language Computer Corp.	0.700	0.396	0.442	0.559
nusmml03r2	National University of Singapore (Yang)	0.562	0.319	0.473	0.479
lexiclone92	LexiClone	0.622	0.048	0.159	0.363
isi03a	University of Southern California, ISI	0.337	0.118	0.461	0.313
BBN2003C	BBN	0.206	0.097	0.555	0.266
MITCSAIL03a	Massachusetts Institute of Technology	0.293	0.130	0.309	0.256
irstqa2003w	ITC-irst	0.235	0.076	0.317	0.216
IBM2003c	IBM Research (Prager)	0.298	0.077	0.175	0.212
Albany03I2	University of Albany	0.240	0.085	0.146	0.178
FDUT12QA3	Fudan University	0.191	0.086	0.192	0.165
UAmsT03M1	University of Amsterdam	0.136	0.054	0.315	0.160
shef12simple	University of Sheffield	0.138	0.029	0.236	0.135
CMUJAV2003	Carnegie Mellon University (Javelin)	0.133	0.052	0.216	0.134
ICTQA2003C	Chinese Academy of Sciences (CAS-ICT)	0.145	0.091	0.149	0.133
uwbqitekat03	University of Wales, Bangor	0.259	0.000	0.000	0.130

2.4 Combined results

The final score for a main task run was computed as a weighted average of the three component scores:

$$\text{FinalScore} = 1/2 * \text{FactoidScore} + 1/4 * \text{ListScore} + 1/4 * \text{DefScore}.$$

Since each of the component scores ranges between 0 and 1, the final score is also in that range. The final score emphasizes the factoid component, which represented the largest number of questions and is the task people are most familiar with. The weight for the other components was made large enough to encourage participation in those subtasks.

Table 5 shows the combined scores for the best run for each of the top 15 groups.

3 Question Answering Techniques

The overall approach taken for answering factoid questions has remained unchanged for the past several years. Systems generally determine the expected answer type of the question, retrieve documents or passages likely to contain answers to the question using important question words and related terms as the query, and then perform a match between the question words and retrieved passages to extract the answer. While the overall approach has remained the same, individual groups continue to refine their techniques for these three steps, increasing the coverage and accuracy of their systems.

A similar approach is used to answer list questions. Indeed, most groups used exactly their factoid-answering system for list questions, changing only the number of responses returned as the answer. The main issue was determining the number of responses to return. Systems whose matching phase creates a question-independent score for each passage returned all answers whose score was above an empirically determined threshold. Other systems returned all answers whose scores were within an empirically determined fraction of the top result's score.

Answering definition questions generally involved using different techniques than those used for factoid questions. Since the definition task emphasized nugget recall and did not require "exact" answers, most systems first retrieved passages about the target using a recall-oriented search. Subsequent processing reduced the amount of material returned. Many systems used pattern-matching to locate definition-content in text. These patterns, such as looking for copular constructions and appositives, were either hand-constructed or learned from a training corpus. Systems also looked to eliminate redundant information, using either word overlap measures or document summarization techniques. The output from this step was then returned as the definition of the target.

4 Analysis of the Evaluation

Since this was the first year for the definition task, and the first year for significant participation in the list task, it is appropriate to assess how well the definition and list question evaluations measure system effectiveness. Of course, in many respects this is a chicken-or-the-egg proposition since the whole point of the evaluation is to define what constitutes "good system effectiveness". In general, a good evaluation will assign higher scores to responses that are intuitively "better" and lower scores to responses that are intuitively "worse". Evaluation results should also provide guidance to system builders as to how their system can be improved.

There are at least two aspects to the quality of an evaluation, fidelity and reliability. Fidelity is the extent to which the evaluation measures what it is intended to measure. Since an evaluation task is an abstraction of a real user's task, fidelity is the extent to which the abstraction captures (some of) the issues of the real task. Reliability is the extent to which an evaluation result can be trusted. Since TREC evaluations are comparative, reliability amounts to the likelihood that an evaluation ranks a better system ahead of a worse system.

This section analyzes the definition and list question evaluations with respect to fidelity and reliability. The first subsection addresses the fidelity of the definition task by examining the effect varying the relative importance of recall and precision has on the evaluation results. The following two subsections empirically estimate the size of the difference required between scores to confidently conclude that two runs are different.

4.1 Definition task fidelity

The F measure is actually a family of measures based on the value of β that controls the relative importance of recall and precision. The general form of the F measure is

$$F = \frac{\beta^2 PR}{(\beta^2 + 1)P + R}$$

Such parameterization allows the measure to be finely tuned to the expectations of the user, but also means that the expectations of the user must be known to properly evaluate the results.

The pilot evaluation of definition questions contained an additional evaluation of the system responses not included in the TREC track [5]. In this "holistic" evaluation, the assessor assigned one score between 0 and 10 to the content of the entire response. This data provided a target for the more quantitative evaluation adopted by the TREC track. Using a β value of five gave a good correlation between the systems ranked by quantitative score and the systems ranked by the holistic score.

The pilot was small, however, and it is unclear whether the mythical average user would so strongly prefer recall. Such a strong emphasis on recall may not encourage systems to be selective enough in the amount of information they return. The results in Table 4 show a trend—but not a strict ordering—for longer responses to receive higher scores. One way to determine how selective the definition question systems are is to compare the results to a baseline run that returns all the sentences in the corpus that contain the target of the definition question. A slightly smarter baseline that returns a sentence only if its overlap with an already-retrieved sentence is small enough was implemented by Jinxi Xu of BBN and the results given to NIST. The results of this sentence baseline were judged by the assessors as part of the regular definition question judging.

Table 6 gives the F(β) evaluation scores for the set of 15 runs from Table 4, plus the sentence baseline (SENT-BASE), for $\beta = 1 \ldots 5$. The table is sorted by decreasing F($\beta = 5$) scores as in Table 4. The sentence baseline is very effective when recall receives strong emphasis. For F($\beta = 5$), the baseline is the second ranked run in the table and ranked 4 of 55 over all runs (only the three BBN runs have a higher F($\beta = 5$) score). For F($\beta = 1$), the baseline is ranked eleventh in the table and 25 out of 55 overall. Other runs also experience a change in relative effectiveness as the importance of recall varies. (Note that the best run for a group for a β value other than five may not be shown in the table.)

The results in Table 6 do not show which F(β) is a better measure for the definition task, only that the evaluation of runs differs depending on the relative importance given to recall and precision. The pilot evaluation demonstrated that there are at least some people for whom recall plays a very dominant role, and for them the F($\beta = 5$) measure is appropriate. Other users may be better represented by the measures that reward the more selective systems.

Table 6: Average F(β) scores for the definition questions subtask. Scores are given for each of the runs in Table 4 plus a sentence-based baseline run. The β parameter varies from 1 (recall and precision have equal importance) to 5 (recall five times as important as precision). The table is sorted by the F($\beta = 5$) scores.

	F(β) Score				
Run Tag	$\beta = 1$	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
BBN2003C	0.310	0.423	0.493	0.532	0.555
SENT-BASE	0.205	0.315	0.400	0.456	0.493
nusmml03r2	0.261	0.360	0.421	0.454	0.473
isi03a	0.270	0.353	0.409	0.442	0.461
LCCmainS03	0.332	0.374	0.408	0.429	0.442
cuaqdef2003	0.187	0.256	0.299	0.324	0.338
irstqa2003d	0.310	0.310	0.314	0.316	0.318
UAmsT03M1	0.163	0.215	0.259	0.292	0.315
MITCSAIL03a	0.296	0.298	0.304	0.307	0.309
shef12simple	0.195	0.211	0.224	0.232	0.236
UIowaQA0303	0.156	0.188	0.210	0.223	0.231
CMUJAV2003	0.246	0.223	0.218	0.217	0.216
FDUT12QA3	0.214	0.196	0.193	0.192	0.192
piq002	0.234	0.198	0.189	0.186	0.185
IBM2003b	0.209	0.186	0.180	0.178	0.177
ntt2003qam1	0.145	0.151	0.159	0.165	0.169

4.2 Definition task reliability

In the evaluation methodology used in TREC, one system is considered to be more effective than another if the evaluation score computed for the output of the first system is greater than the evaluation score computed for the output of the second system. However since *all* measurements have some (unknown) error associated with them, there is always a chance that such a comparison can lead to the wrong result. An analysis of the reliability of an evaluation establishes bounds for how likely it is for a comparison to be in error.

The error in a definition question evaluation score arises from a variety of sources of noise. One source is mistakes made during the judgment process: the assessors are humans and humans will make mistakes, especially given the rate at which the NIST assessors are asked to process results. Another source of noise is the fact that humans have differing opinions as to the quality of a response. These differences of opinions are not mistakes, but legitimate differences in what the assessor considers to be acceptable. A third source of noise is the sample of questions used in the evaluation. Since in general system performance depends on the question that is asked, it may be that a better overall system evaluates as worse than another because of the particular sample of questions used in the test set. None of these sources of error is unique to the definition question task. All three sources exist in all of the evaluations in TREC.

The amount of noise contributed by mistakes in judging has been difficult to measure in other TREC tasks because results are pooled. That is, identical responses from different runs are represented only once in what the assessor judges, so there is no opportunity to be inconsistent. The results are not pooled for the definition (and list) task, however, because the assessor must judge the entire response as a unit for this task. Since some main task submissions differed only in the factoid component of the task, some of the 55 responses a definition assessor was asked to judge were exact copies of another response. These pairs of identical, separately judged responses provide the data for estimating an error rate due to assessor mistakes.

The main task runs contained 14 pairs of identical definition components (i.e., exactly the same for all 50 definition questions). The distribution of the number of questions that were judged differently over these 14 pairs was as follows: 0 questions judged differently, 1 pair; 1 question judged differently, 3 pairs; 2 questions, 1 pair; 3 questions, 4 pairs; 4 questions, 2 pairs; 8 questions, 2 pairs; 10 questions, 1 pair. Across the whole set of 14 pairs, 19 questions had some difference in judging, spread relatively uniformly across individual assessors. The differences in the average F($\beta = 5$) scores for the pairs of identical runs ranged from a low of 0.0 to a high of 0.043, and averaged 0.013.

The definition task was a brand new and more difficult task for the assessors. There is reason to believe the number

1	vital	provides remuneration to executives who lose jobs
2	vital	assures officials of rich compensation if lose job due to takeover
3	vital	contract agreement between companies and their top executives
4		aids in hiring and retention
5		encourages officials not to resist a merger
6		IRS can impose taxes

Figure 4: Information nuggets created by second assessor for question 1905 *What is a golden parachute?*.

of judging errors will decrease now that the assessors have experience with the task and the assessment system software is changed to better support this task. However, the number of errors will never be reduced to zero. Since runs that were absolutely identical had differences in average scores of 0.043 in this year's evaluation, distinct runs that differ by less than this amount (such as the `nusmml03r2`, `isi03a`, and `LCCmainS03` runs in Table 4) clearly must be considered equivalently effective.

Another source of noise is the differences in scores caused by differing opinions of assessors. Since assessor opinions as to correctness are known to differ, each question is judged by a single assessor so the judgments are internally consistent (modulo mistakes). However, to quantify the effect different opinions have on definition question scores, each question was independently judged by a second assessor. Since the second assessor did not research the target of the question during question development as the original assessor had, the second assessor was given the initial list of nuggets the original assessor created during question development as a starting point. The second assessor was free to modify that list in any way (add, delete, or alter nuggets) as he proceeded in the two stages of judging the question. The set of information nuggets created by the second assessor for the golden parachute question of Figure 2 is given in Figure 4.

Not surprisingly, there were definite differences across assessors. Some assessors listed many more nuggets—and many more vital nuggets—than other assessors. There was no universal agreement as to what kinds of information should be returned for, say, a "who is" question. Different assessors looked for different information for the same question, and the same assessor looked for different information for different questions. For example, a birth date was usually wanted for an important historical figure, but not for a contemporary sports figure.

To measure the effect of the differences of opinion of different assessors on the evaluation, the set of runs were scored using each of the two sets of judgments and ranked by decreasing average F($\beta = 5$) scores. The distance between the two rankings of runs was computed using a correlation measure based on Kendall's τ [2]. The τ score between the two rankings of runs was 0.848, which corresponds to 113 pairwise differences between the rankings (out of a possible 1485 pairwise difference since there were 55 runs being ranked). While most of the changes in the rankings were between two runs with small differences in F($\beta = 5$) scores as computed using the original assessor's judgments, eight of the differences were between runs with F scores that differed by more than 0.1. The largest difference in original F($\beta = 5$) scores between two runs that evaluated differently when using different judgments was 0.123

It would be nice to have a critical value for τ such that correlations greater than the critical value guarantee a quality evaluation. Unfortunately, no such value can exist since τ values depend on the set of runs being compared. In practice, we have considered correlations greater than 0.9 to indicate an acceptably stable evaluation [3]. The correlation between the rankings of the definition questions is decidedly smaller than 0.9, and the eight swaps with a difference of greater than 0.1 is especially worrisome. These findings suggest that the evaluation results of this first version of the definition evaluation need to be interpreted with great caution, and that steps should be taken in future years to increase the stability of the evaluation.

One way to increase the stability of an evaluation is to increase the number of questions that scores are computed over. Using larger numbers of questions has two effects: first, the sample of questions is larger and thus more likely to include different categories of questions, and second the law of large numbers says the averages will be closer to the true means. We can use the runs submitted to the track to empirically determine the relationship between the number of questions in a test set, the observed difference in scores (δ), and the likelihood that a single comparison of two definition tasks runs leads to the correct conclusion. Once established, the relationship can be used to derive the minimum difference in scores required for a certain level of confidence in the results given there are 50 questions in the test set. In theory, the same relationship can also predict the number of questions to have in the test set given

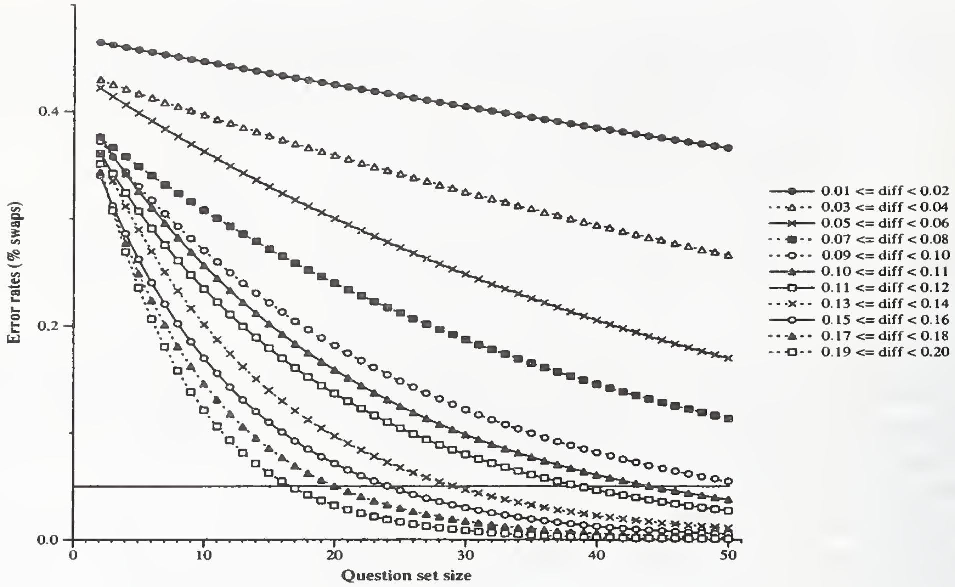


Figure 5: Stability of definition question evaluation as a function of question set size.

that you want a certain level of confidence in the comparison at a given minimum difference in scores. However, the amount of extrapolation involved in that computation makes the prediction of dubious value.

The core of the procedure for establishing the relationship is comparing the effectiveness of a pair runs on two disjoint question sets of equal size to see if the two sets disagree as to which of the runs is better. We define the error rate as the percentage of comparisons that result in a swap. Since the definition component used 50 questions, we can directly compute the error rate for question set sizes up to 25 questions. By fitting curves to the values observed for question set sizes up to 25, we can extrapolate the error rates to question sets up to 50 questions.

When calculating the error rate, the difference between two runs' F($\beta = 5$) scores is categorized into one of 21 bins based on the size of the difference. The first bin contains runs with a difference of less than 0.01 (including no difference at all). The next bin contains runs whose difference is at least 0.01 but less than 0.02. The limits for the remaining bins increase by increments of 0.01, with the last bin containing all runs with a difference of at least 0.2.

Each question set size from 1 to 25 is treated as a separate experiment. Within an experiment, we randomly select two disjoint sets of questions of the required size. We compute the F($\beta = 5$) score over both question sets for all runs (using the original assessor judgments), then count the number of times we see a swap for all pairs of runs using the bins to segregate the counts by size of the difference in scores. The entire procedure is repeated 50 times (i.e., we perform 50 trials), with the counts of the number of swaps kept as running totals over all trials. The ratio of the number of swaps to the total number of cases that land in a bin is the error rate for that bin.

The error rates computed from this procedure are then used to fit curves of the form $ErrorRate = A_1 e^{-A_2 S}$ where A_1 and A_2 are parameters to be estimated and S is the size of the question set. A different curve is fit for each different bin. The input to the curve-fitting procedure used only question set sizes greater than 1 since a single question is known to be very noisy. The largest bin (all differences greater than 0.2) is very flat and therefore difficult to fit, so no curve was fit for it.

The resulting extrapolated error rate curves are plotted in Figure 5. In the figure, the question set size is plotted on the x-axis and the error rate is plotted on the y-axis. The horizontal line in the graph in Figure 5 is drawn at an error rate of 5 %, a level of confidence commonly used in experimental designs. For question set sizes of 50 questions, there needs to be an absolute difference of at least 0.1 in the F($\beta = 5$) scores before the error rate is less than 5 %. Using this standard, swaps caused by different assessor opinions when the difference in F score is less than 0.1 are to be expected since the runs being compared must be considered equally effective. It is the eight swaps among runs with greater differences that are the concern.

Requiring a difference of at least 0.1 (or 0.123) in F($\beta = 5$) scores before considering two evaluation results to be different is necessary to have confidence in the conclusions, but also results in a fairly insensitive test. For example, if we take the run that ranks in the middle of the runs in Table 4, and add ±0.1 to its score, its rank would range

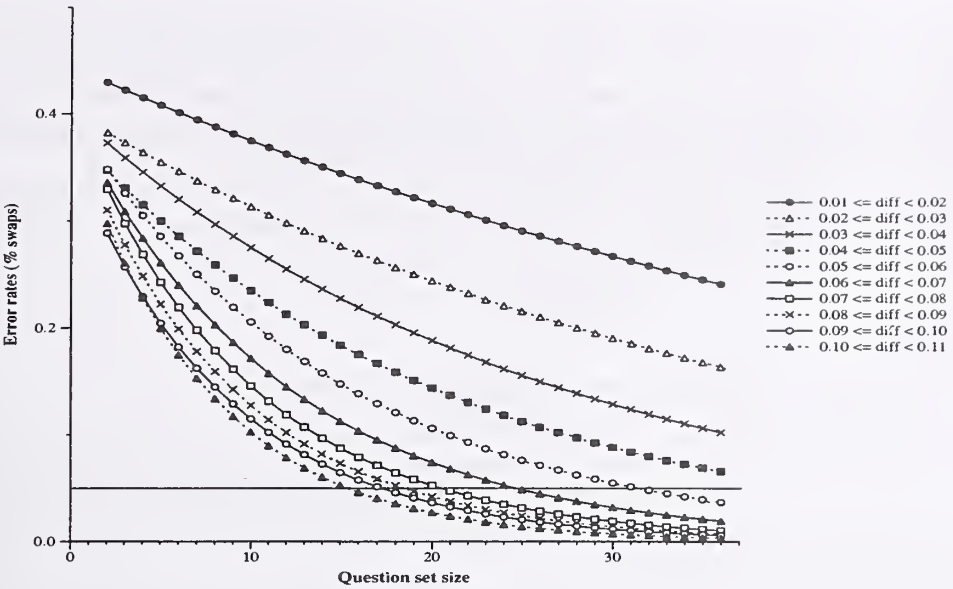


Figure 6: Stability of list question evaluation as a function of question set size.

anywhere from fifth to twelfth. More questions are needed in the test set to increase the sensitivity while remaining equally confident in the result.

4.3 List task reliability

As in the definition question evaluation, list task responses must also be judged as a unit, so an estimate of the error attributable to assessing mistakes can be derived from direct comparison of scores for identical submissions. There were ten pairs of runs that had identical list task components. Over these ten pairs of runs, one pair differed in the score assigned to three questions, four pairs differed in the scores assigned to two questions, four pairs differed in the scores assigned to one question, and one pair was assigned the same score for all questions. The largest difference in the average F score for a pair was 0.004, and the average difference across the ten pairs was 0.002.

The list questions were each judged by only one assessor, so no comparisons across assessors are possible. However, Figure 6 shows the plot of extrapolated error rates by question set size for the list task. Since the list component had only 37 questions, the error rates were directly computed up to question set size 18, and then extrapolated to question set size 36. The plot shows that the error rates decrease quickly as question set size increases: at 36 questions the error rate is less than 5% for a difference in F scores of at least 0.05.

These two tests show that this year's list task results are extremely stable. Remember, however, that the sensitivity analysis is dependent on the runs used to compute it. One reason the list task results appear as stable as they do is because in general the scores for the list task are very low. Only 11 of the 54 main task submissions had an average score for the list component that was greater than 0.1. Only 9 of the 37 questions had a median F score greater than 0.0. Clearly if the scores are similar regardless of which question you choose (in this case, all close to 0), you don't need many questions to converge to the average score.

5 Future of the QA Track

The TREC 2003 QA track offered the first large-scale evaluations of list and definition questions. The results demonstrated that these are challenging tasks for systems, and that they present challenges for evaluation. The proposed task for the TREC 2004 QA track will address these challenges by keeping a combined task, but reorganizing it somewhat so that more definition questions can be accommodated.

There will be approximately 100 targets for a definition. For each target, NIST assessors will define a set of factoid questions and a separate set of list questions. There will also be an implied "tell me more" question, which is to be interpreted as "Tell me other interesting things about this target I don't know enough to ask directly". This last question is roughly equivalent to the definition questions in the TREC 2003 task. As an example, if the target were an author, the factoid questions might ask for things such as birth date, date of death, and nationality. A list question may ask for the names of the author's works. Finally, the answer to the last question may include such items as one of the author's books won a Newberry prize, and the author teaches at XYZ University.

Organized this way, the task allows for factoid and list questions as well as more definition questions. The organization also provides context for the factoid and list questions, an important element that the track has not yet successfully incorporated [4].

Acknowledgements

My thanks to Chris Buckley for fitting the error rate curves shown in Figures 5 and 6.

References

[1] Donna Harman and Paul Over. The DUC summarization evaluations. In *Proceedings of the International Conference on Human Language Technology*, 2002.

[2] Ellen M. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. *Information Processing and Management*, 36:697–716, 2000.

[3] Ellen M. Voorhees. Evaluation by highly relevant documents. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 74–82, 2001.

[4] Ellen M. Voorhees. Overview of the TREC 2001 question answering track. In E.M. Voorhees and D.K. Harman, editors, *Proceedings of the Tenth Text REtreival Conference (TREC 2001)*, pages 42–51, 2002.

[5] Ellen M. Voorhees. Evaluating answers to definition questions. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003), Volume 2*, pages 109–111, May 2003.

[6] Ellen M. Voorhees and Dawn M. Tice. The TREC-8 question answering track evaluation. In E.M. Voorhees and D.K. Harman, editors, *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, pages 83–105, 2000. NIST Special Publication 500-246. Electronic version available at `http://trec.nist.gov/pubs.html`.

Overview of the TREC 2003 Robust Retrieval Track

Ellen M. Voorhees
National Institute of Standards and Technology
Gaithersburg, MD 20899

Abstract

The robust retrieval track is a new track in TREC 2003. The goal of the track is to improve the consistency of retrieval technology by focusing on poorly performing topics. In addition, the track brings back a classic, ad hoc retrieval task to TREC that provides a natural home for new participants.

An important component of effectiveness for commercial retrieval systems is the ability of the system to return reasonable results for every topic. Users remember abject failures. A relatively few such failures cause the user to mistrust the system and discontinue use. Yet the standard retrieval evaluation paradigm based on averages over sets of topics does not significantly penalize systems for failed topics. The robust retrieval track looks to improve the consistency of retrieval technology by focusing on poorly performing topics.

The task within the track was a traditional ad hoc task. An ad hoc task in TREC investigates the performance of systems that search a static set of documents using previously-unseen topics. For each topic, participants create a query and submit a ranking of the top 1000 documents for that topic. In addition to the standard evaluation by `trec_eval`, each run was also evaluated using two new effectiveness measures that focus on the effectiveness of the least-well-performing topics.

This paper presents an overview of the results of the track. The first section provides more details regarding the task and defines the new evaluation measures. The following section presents the systems' retrieval results, while Section 3 examines the new evaluation measures. Systems compare differently when evaluated on the new measures then when evaluated on standard measures such as MAP, suggesting that the new measures capture a different aspect of retrieval behavior. However, the measures are less stable than the traditional measures, and the marigin of error associated with the new measures is large relative to the differences in scores observed in the track.

1 The Robust Retrieval Task

As noted above, the task within the robust retrieval track was a traditional ad hoc task. The topic set consisted of a total of 100 topics, 50 old topics taken from TREC topics 301–450 (TRECs 6–8) and 50 new topics. The document collection was the set of documents on TREC disks 4 and 5, minus the *Congressional Record*, since that is what was used for TRECs 6–8. This document set contains approximately 528,000 documents and 1,904 MB of text.

Since the focus of the track is on poorly performing topics, we wanted to ensure that there were topics that are generally difficult for systems to answer in the test set. We could not (purposely) construct a difficult topic set using only new topics since it is notoriously hard to predict whether or not a topic will be difficult a priori [5]. Instead, we used the effectiveness of the retrieval runs in TRECs 6–8 to construct a topic set of known-to-be-difficult topics. For each of topics 301–450, NIST created a box plot of the average precision scores for all runs (both automatic and manual) submitted to the ad hoc task in that topic's TREC. NIST then selected topics with low median average precision scores but with at least one (there was usually more than one) high outlier. The requirement for at least one system doing well on the topic was designed to eliminate flawed topics from the topic set. The set of old topics selected for the robust track is given in Figure 1.

While using old topics allowed NIST to construct a test set with certain properties, it also meant that full relevance data for these topics was available to the participants, and that systems were likely developed using those topics. NIST therefore created 50 new topics using the standard topic creation process as a type of control group. The 50 new topics are numbered 601–650. Since we could not control how the old topics had been used in the past, the assumption was that the old topics were fully exploited in any way desired in the construction of a participants' retrieval system. In

303	322	344	353	363	378	394	408	426	439
307	325	345	354	367	379	397	409	427	442
310	330	346	355	372	383	399	414	433	443
314	336	347	356	374	389	401	416	435	445
320	341	350	362	375	393	404	419	436	448

Figure 1: The set of old topics used in the robust track.

other words, participants were allowed to explicitly train on the 50 old topics in the test set if they desired to. The only restriction placed on the use of relevance data for the 50 old topics was that the relevance judgments could not be used during the processing of the submitted runs. This precluded such things as true (rather than pseudo) relevance feedback and computing weights based on the known relevant set.

The existing relevance judgments were used for the old topics; no new judgments of any kind were made for these topics. The new topics were judged by creating pools from all runs submitted to the track and using the top 125 documents per run. There was an average of 959 documents judged for each new topic. The assessors made three-way judgments of not relevant, relevant, or highly relevant for the new topics. Seven of the 50 new topics had no highly relevant documents, and another 14 topics had fewer than 5 highly relevant documents. All the evaluation results reported for the track consider both relevant and highly relevant documents as the relevant set since there are no highly relevant judgments for the old set. The number of relevant documents per topic for the old topic set ranged from a low of 5 to a high of 361 and an average of 88. For the new topic set, the minimum number of relevant was 4, the maximum was 115, and the average was 33.

While no new judgments were made for the old topics, we did form pools for those topics (using the top 100 retrieved per run) to examine the coverage of the original judgment set. Across the set of 50 old topics, an average of 61.4 % (minimum 43.2 %, maximum 79.7 %) of the documents in the pools created using robust track runs were judged. A relatively low number of judged documents is to be expected since the old topics were chosen because they were difficult, and there is known to be less overlap among the retrieved sets for difficult topics than for easier topics. Across the 78 runs that were submitted to the track, there was an average of 0.4 unjudged documents in the top 10 documents retrieved and 11.6 unjudged documents in the top 100 retrieved. These averages are inflated by a set of five runs that had very poor effectiveness (a cursory examination confirmed that the poor effectiveness was caused by retrieving documents that were indeed not relevant). Without these five runs, there was an average of 0.2 unjudged documents in the top 10 documents retrieved and 8.7 unjudged documents in the top 100 retrieved. There is still a tendency toward poorer runs having larger numbers of unjudged documents in the retrieved set, but such a bias is expected and is caused by poorer runs retrieving different, really-not-relevant documents.

Runs were evaluated using `trec_eval`, with average scores computed over the set of 50 old topics, the set of 50 new topics, and the combined set of 100 topics. Two additional measures were computed over the same three topic sets. The first measure was the percentage of topics that retrieved no relevant documents in the top ten retrieved. If one accepts "no relevant documents in the top ten retrieved" as an adequate definition of poorly performing topic, then this is a direct measure of the behavior of interest and is therefore a very intuitive and easily understood measure. It has the drawback of being a very coarse measure. That is, there are relatively few discrete values the measure can assume in theory, and the actual range of values seen in practice is much smaller than the theoretical range.

The second measure was suggested by Chris Buckley. One of the initial proposals for a measure for the track was to compute the mean of the average precision scores (MAP) for the system's worst X topics (as measured by average precision) rather than the entire set of topics as `trec_eval` does. In an attempt to pick a suitable X—big enough to make the measure stable but small enough to emphasize the poorly performing topics—the mean average precision over the worst X topics, MAP(X), was plotted as a function of X for several runs. Chris suggested that instead of picking a single point on the curve to use as the measure, to use the area underneath the MAP(X) vs. X curve as the measure. Just as MAP (the area underneath the recall-precision curve) emphasizes high precision but has a recall component, the area under the MAP(X) vs. X curve measure emphasizes the worst-performing topics, but also gives a general measure of quality. The measure as implemented for the track computes the area under the MAP(X) vs. X curve, but limits X to the worst quarter topics. That is, X is set from $1 \ldots 12$ for the 50-topic sets and $1 \ldots 25$ for the combined set. This measure is not exactly intuitive (it doesn't even have a better name than "area underneath the MAP(X) vs. X curve" yet), but it incorporates much more information than the percentage of topics with no relevant

Table 1: Groups participating in the robust track.

Chinese Academy of Sciences (CAS-NLPR)	Tsinghua University (Ma)
Fondazione Ugo Bordoni	University of Amsterdam
Hummingbird	University of Glasgow
Johns Hopkins University/APL	University of Illinois at Chicago
OcE Technologies	University of Illinois at Urbana-Champaign
Queens College, CUNY	University of Melbourne
Rutgers University (Neu)	University of Waterloo (MultiText)
Sabir Research, Inc.	Virginia Tech

in the top 10 retrieved. Note that since the measure is computed over the individual system's worst X topics, different systems' scores are computed over a different set of topics in general.

2 Retrieval Results

The robust track received a total of 78 runs from the 16 groups listed in Table 1. All of the runs submitted to the track were automatic runs. Participants were allowed to submit up to 5 runs. One of the runs was required to use only the description portion of the topic statements; the other runs could use any portion of the topic statements. There was a noticeable difference in effectiveness depending on the portion of the topic statement used: runs using all of the topic statement were better than those using selected fields, and runs using only the title field were worse than those using other portions. The retrieval results reported here are restricted to the runs that used just the description portion of the topic since that was the required run. There were 44 description-only runs submitted to the track.

Table 2 gives the evaluation scores for one run for each of the groups that submitted a description-only run (one group did not submit such a run by mistake). The table gives the scores for the four main measures used in the track as computed over the old topics only, the new topics only, and the combined set of 100 topics. The four measures are mean average precision (MAP), the average of precision at 10 documents retrieved (P10), the percentage of topics with no relevant in the top 10 retrieved (%no), and the area underneath the MAP(X) vs. X curve (area). The run shown in the table is the run with the highest MAP score as computed over the combined topic set; the table is sorted by this same value.

As expected given the way the topic set was constructed, the results show that as a set the 50 old topics are clearly much more difficult than the 50 new topics. The scores for all measures and all runs are better, usually much better, for the new topics than for the old. While all systems score better on the new set than the old, the amount of improvement is not uniform, so the relative ordering of systems is different for the two topic sets. We can quantify how different the relative orderings are by computing the Kendall τ correlation between system rankings using each of the topics sets in turn. A system ranking is an ordering of the runs by decreasing score of an effectiveness measure. The Kendall τ correlation measures the similarity between two rankings as a function of the number of pairwise swaps needed to turn one ranking into the other. The τ ranges between -1.0 and 1.0 where the expected correlation between two randomly generated rankings is 0.0 [2]. Table 3 shows the system rankings for the 44 description-only runs for each of the four evaluation measures of Table 2 for both topics sets. The ranking for the old topic set is given on the top and the ranking for the new topic set on the bottom. Each run is represented by a single character in the rankings. When two runs have a tied score for one measure they are ranked according to their MAP scores for that topic set. The last column in Table 3 gives the Kendall correlation between the two rankings. The τ values confirm that the rankings are different. The precise cause for the differences cannot be determined from this data since there are (at least) two confounded factors: different systems doing different amounts of training on the old topics and different systems being relatively more effective for difficult topics.

Are current retrieval systems handling the difficult topics better now than when the topics first appeared? We can give an approximate answer to this question by comparing the median and maximum scores obtained for each topic when computed over the set of runs submitted to the TREC in which the topic first appeared and the set of runs submitted to the robust track. Figure 2 shows this comparison using average precision as the evaluation measure. Since there were few description-only runs submitted to the previous ad hoc tasks, the sets of runs used to compute

Table 2: Evaluation results for the best description-only run per group as measured by MAP over the combined topic set. Runs are ordered by MAP over the combined topic set. Values given are the mean average precision (MAP), precision at rank 10 averaged over topics (P10), the percentage of topics with no relevant in the top ten retrieved (%no), and the area underneath the MAP(X) vs. X curve (area) as computed for the set of 50 old topics, the set of 50 new topics, and the combined set of 100 topics.

	Old Topic Set				New Topic Set				Combined Topic Set			
Tag	MAP	P10	%no	area	MAP	P10	%no	area	MAP	P10	%no	area
pircRBd2	0.177	0.382	12	0.009	0.403	0.532	4	0.082	0.290	0.457	8	0.022
uwmtCR0	0.150	0.370	14	0.011	0.403	0.536	8	0.052	0.276	0.453	11	0.018
aplrob03d	0.162	0.290	28	0.005	0.383	0.496	16	0.027	0.273	0.393	22	0.008
humR03de	0.148	0.352	24	0.004	0.377	0.484	14	0.023	0.263	0.418	19	0.009
VTDokrcgp5	0.130	0.314	18	0.005	0.382	0.502	12	0.023	0.256	0.408	15	0.008
fub03InOLe3	0.134	0.338	24	0.007	0.370	0.488	10	0.015	0.252	0.413	17	0.008
UIUC03Rd3	0.125	0.282	26	0.003	0.375	0.498	16	0.019	0.250	0.390	21	0.006
Sel78QE	0.124	0.292	32	0.003	0.363	0.452	18	0.007	0.243	0.372	25	0.003
THUIRr0305	0.117	0.312	16	0.009	0.370	0.508	8	0.048	0.243	0.410	12	0.015
SABIR03BF	0.106	0.220	34	0.003	0.346	0.464	12	0.030	0.226	0.342	23	0.006
UAmsT03RDesc	0.107	0.264	14	0.006	0.306	0.442	16	0.014	0.206	0.353	15	0.008
oce03noXbmD	0.092	0.240	24	0.003	0.305	0.446	16	0.013	0.199	0.343	20	0.005
MU03rob01	0.089	0.268	18	0.004	0.297	0.448	10	0.021	0.193	0.358	14	0.009
NLPR03vb50	0.095	0.334	6	0.005	0.259	0.460	8	0.013	0.177	0.397	7	0.007
rutcor0375	0.028	0.104	48	0.000	0.129	0.208	36	0.001	0.078	0.156	42	0.000

Table 3: System rankings and corresponding Kendall τ scores for the old and new topic sets.

Measure	Rankings $\frac{\text{Old Set}}{\text{New Set}}$	τ
MAP	WXCVoDLAqBHIFErhJnimNjpGlkegfMdRUOTQKSPcbZaY qWoVXCrLnljIEBmiHNADFpGhMJfegdkUORTQKSPcZbaY	0.772
P10	WXoLqIFERQPHVrjGpTSJhiCNgnDBmAMOlkUdefKcZbaY oWXqVjrFnClBImLGENpJMHeRQPifAhOUgkDTSdKZcbaY	0.562
% no	DRQPTSWXoGkgjArpOKqMJLBHIEmUFhnldCViNefZbcaY WVXpqojGMJgRQPIFfdOTSrlEBAeKLNDCnmHhkUiZcbaY	0.427
area	qojWpBIADXkEHMCgdRrGJFVhLOTfQmnileUSKPNcZabY WVqXojBApfDeCdJMGrLlOmFNngIkURKEHTQihPSZcbaY	0.560

the median and maximum average precision scores consisted of all automatic runs (i.e., runs using any combination of the fields in the topic statement). In the figure the median scores are plotted using filled symbols while the maximum scores are plotted using hollow symbols. The values computed using the set of runs submitted to the TREC in which the topic first appeared, called the Original TREC, are plotted as ovals; the values computed using robust track runs are plotted as triangles. The topics are sorted by decreasing median average precision score as computed using the robust track submissions. Median effectiveness for the robust track runs is generally better than for the original TREC runs, though for about 10 topics the original runs have a better median. The difference between medians is generally small (with a few notable exceptions). The maximum scores have larger differences and there are more topics for which the original runs had the better maximum score than for the robust runs. There were more different systems contributing to the Original TREC runs set than for the robust track runs set which may account for the better maximum scores. Nonetheless, it is clear that this old topic set remains a difficult set of topics.

Many of the participants used the robust track as a place to try new techniques for general ad hoc retrieval, without particularly focusing on the question of poorly performing topics. Both of the two groups with top-scoring runs, Queens College, CUNY and the Multitext group at the University of Waterloo, expanded the query using terms extracted from the Web (and possibly other document sets). Other groups experimented with new retrieval mod-

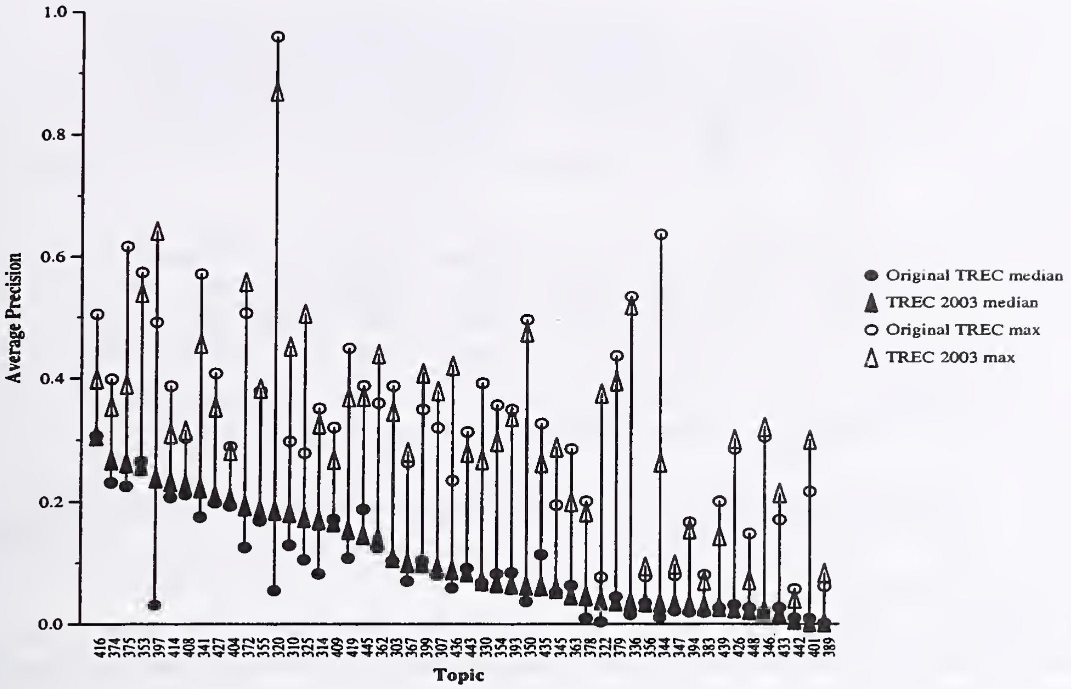


Figure 2: Median and maximum per-topic average precision scores for the old set of topics as computed using the runs submitted to the first TREC the topic was used in (Original TREC) and the TREC 2003 robust track runs (TREC 2003).

els or ranking functions (CAS-NLPR, Tsinghua University, University of Glasgow, University of Illinois Urbana-Champaign, Virginia Tech); with weighting schemes (OcE, Rutgers University, University of Illinois at Chicago, University of Melbourne); and with tokenization techniques (John Hopkins/APL, University of Amsterdam).

Almost all groups tried some version of query expansion based on pseudo-feedback. The query expansion improved average effectiveness, but did not help (and frequently hurt) the worst performing topics except when the expansion was done using a different corpus. This is not particularly surprising since the poorly performing topics are unlikely to have relevant documents in the top retrieved documents, and thus the feedback is as likely to harm as to help the results. After the qrels were published, the group from Fondazione Ugo Bordoni ran a series of experiments to see if they could predict when expansion would be beneficial based on an estimate of the MAP score of the initial retrieval result. When expanding only if the prediction determined it would be beneficial, they were able to both increase the MAP score and decrease the number of topics with no relevant retrieved as compared to their baseline.

Other approaches to increasing the effectiveness of the poorly performing topics included per-topic merging of results from different component runs and reordering the similarity-ranked list to maximize the number of retrieved-set document clusters with representatives in the top 10 ranks. Johns Hopkins/APL found modest success in decreasing the number of poorly performing topics by merging multiple runs, but also found that their results were far below the optimum theoretically obtainable from merging. Hummingbird tried to increase the diversity of the documents in the top 10 ranks by clustering the retrieved set and reranking the top 100 documents such that the top 10 documents were from different clusters. Unfortunately, the reranking did not lead to a significant increase in the number of topics with a relevant document in the top 10 retrieved.

3 Effectiveness Measures

One of the common themes of the participants' results was that query expansion improved MAP scores while not improving or even degrading the effectiveness of the worst topics. This demonstrates that MAP scores are essentially unaffected by the poorly performing topics. Mathematically, a poorly performing topic would have to improve dramatically to affect the MAP score since the magnitude of the MAP score is so much larger than an individual poorly

MAP	WXVoqCLrDABIEnHFjilmhNJpGefMgkdUORTQKSPcZbaY
P10	WoXqVLFIjrEHRQPGCpnJBmNlihMTSgADOkUefdKZcbaY
% no	RQPWXDTSoGgjpqMOrAJkKVIBEFdLlHmUnhCefNiZcbaY
area	WqoXjpVBADGMOLJFdCrIkRgmfEnleHUTQNhiKPSZcbaY

a) Rankings computed using combined topic set

	Old Topics			New Topics			All Topics		
	P10	% no	area	P10	% no	area	P10	% no	area
MAP	0.560	0.171	0.558	0.753	0.334	0.588	0.592	0.180	0.584
P10		0.433	0.444		0.463	0.535		0.397	0.493
% no			0.393			0.518			0.457

b) Kendall τ scores computed between rankings for all pairs of measures

Figure 3: Agreement among system rankings produced by different measures.

performing topic's average precision score.

This section examines the behavior of the two new measures that were introduced in the track. It shows that the new measures do emphasize poorly performing topics as designed, but because their scores are based on relatively few topics, they are more unstable than traditional measures and the margin of error associated with the new measures is large relative to the differences in scores observed in the track. More reliable measures are needed to support research on developing techniques for consistent retrieval.

3.1 Agreement among measures

One way to show that different measures emphasize different factors is to examine whether they rank systems differently. We can produce systems rankings as above (using description-only runs), except that now instead of comparing rankings produced using different topic sets, we compare rankings produced using different evaluation measures. Figure 3 shows the agreement among system rankings for MAP, the average of precision at 10 documents retrieved, the percentage of topics with no relevant in the top 10 retrieved, and the area under the MAP(X) vs. X curve as computed over the set of old topics, the set of new topics, and the combined set of 100 topics. The system rankings themselves as computed over the combined set of topics are given at the top of the figure. The bottom of the figure shows the Kendall τ score computed between the rankings for each pair of measures.

The correlations are quite low, providing support for the contention that the measures are affected by different aspects of retrieval behavior. The correlation between MAP and the percentage of topics with no relevant documents in the top 10 documents is only slightly better than chance. While in theory such a low correlation with MAP means only that the two measures are emphasizing different aspects of retrieval, MAP has been shown to be an effective, stable measure [1] so in practice a low correlation with MAP can be a sign of an unstable measure. The stability of the new measures is investigated below.

The area under the MAP(X) vs. X curve measure depends on the greatest value that X assumes. This value reflects the trade-off in emphasis given to the worst-performing topics and the overall effectiveness of the system. The graphs in Figure 4 illustrate how the relative effectiveness among systems changes as X changes. The graphs plot MAP(X) vs. X using the combined topic set for a subset of the runs shown in Table 2. The left side of the figure shows the plot for all 100 values of X, and the right side of the figure shows the same plot restricted to $X = 1 \ldots 25$ so more detail can be seen. The value of the official area measure is the area underneath the curve plotted in the right side of the figure.

The graphs in Figure 4 make it clear that the relative order of systems ranked by their area scores does change depending on the maximal value of X. For example, the THUIRr0305 run has the best area score when $X \leq 6$, and is ranked third until approximately $X = 65$. However, the area measure is not highly sensitive to the maximal value of X, provided X is greater than about 10. We created system rankings based on the value of the area measure using the combined topic set and all description-only runs as the maximal value of X varied from 1 (i.e., the worst topic

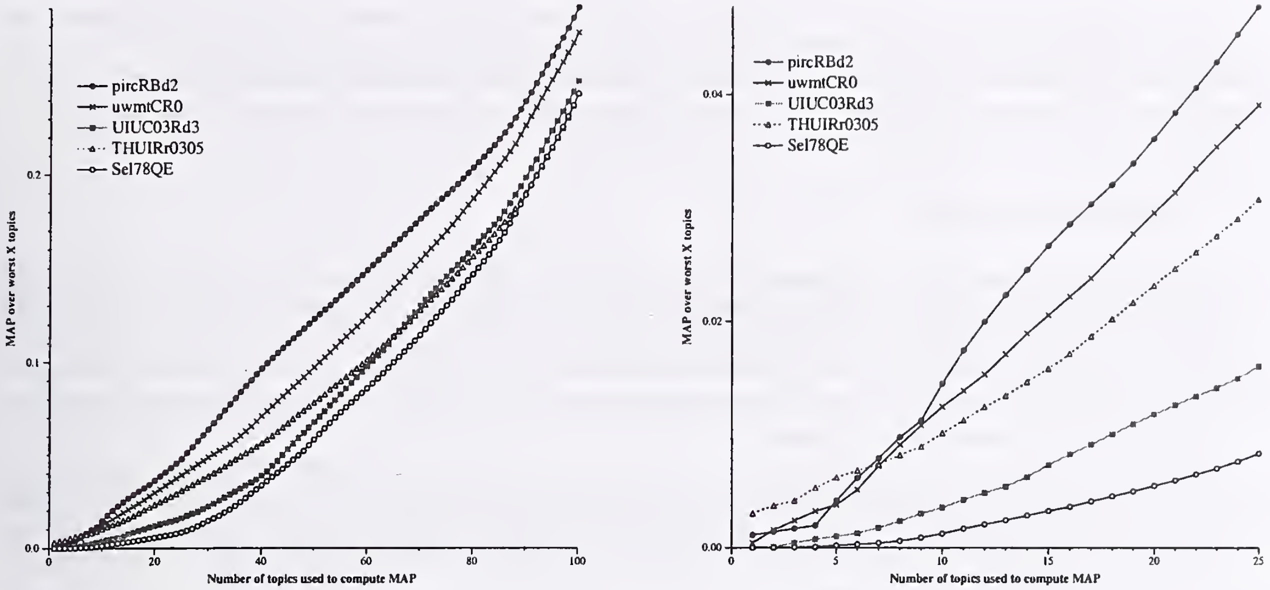


Figure 4: Plot of MAP(X) vs. X. The graph on the left side of the figure shows the entire range of $X = 1 \ldots 100$; the graph on the right is restricted to $X = 1 \ldots 25$ so more detail can be seen.

Table 4: Error rate and proportion of ties for different measures.

	Error Rate (%)	Proportion of Ties
MAP	1.4	0.171
P10	2.6	0.224
% no	9.1	0.090
area	8.4	0.040

determines the score) to 25. The Kendall τ correlations for $X < 10$ are small—in the 0.4 range when $X < 5$—but this is to be expected since measures based on the effectiveness of very few topics are known to be unstable. For $X > 10$, the τ values were greater then 0.85, and were generally greater than 0.95 when the X values being compared were within 5 of one another.

3.2 Stability of measures

The stability of the evaluation measures for topic sets containing 50 topics can be examined using a procedure similar to the one introduced by Buckley and Voorhees [1]. This procedure computes an error rate for an evaluation measure by counting how often the measure disagrees with respect to which of two systems being compared is preferred. Larger error rates imply a less stable measure.

We generated 1000 different test sets of size 50 topics each by randomly selecting 50 topics from the set of 100 topics used in the track. We evaluated all 78 runs submitted to the track on each of the 1000 test sets. For all pairs of runs A and B, we counted the number of test sets for which A evaluated as better than B ($A > B$), B evaluated as better than A ($B > A$), and A and B evaluated as equivalent ($A = B$). Two runs were considered equivalent if the difference in their scores was less than 5 % of the larger score. The error rate is defined as the sum over all run pairs of $\min(A > B, B > A)$, divided by the total number of comparisons. The proportion of ties, $A = B$ divided by the total number of comparisons is also of interest since it indicates how much discrimination power a measure has. A measure with a low error rate but a high proportion of ties has little power.

Table 4 shows the error rate and proportion of ties computed for the four different measures. The numbers for MAP and P10 are close to the numbers reported by Buckley and Voorhees despite the different collection and slightly different methodology. As suspected, the error rates for the two new measures are substantially greater than for MAP and P10, though the proportion of ties for the new measures is substantially smaller than for the traditional measures.

The relative instability of the area and topics-with-no-relevant-retrieved measures is not difficult to understand. Numerically, a very low proportion of ties is likely to increase the error rate—the more decisions you make the more likely some of them are wrong, especially since fewer ties implies finer distinctions. In addition, the new measures are defined over a subset of the topics in the test set. For a test set of a given size, the score for the new measures will always be based on fewer topics than for the traditional measures.

3.3 Sensitivity of measures

While the higher error rates for the new measures are understandable, they do mean that there is much more uncertainty associated with a comparison of two systems when using one of these measures. Voorhees and Buckley introduced a procedure to empirically determine the relationship between the number of topics in a test set, the observed difference in scores of a particular measure (called Δ), and the likelihood that a single comparison of two runs leads to the correct conclusion [4]. Once established, the relationship can be used to derive the minimum difference in scores required for a certain level of confidence in the conclusion.

With 100 topics in the robust track test set, we can directly compute the relationship for topic set sizes up to 50 topics. Robust track runs should require somewhat smaller Δ's for the same level of confidence since they contain 100 topics. Voorhees and Buckley's original procedure used extrapolation to derive minimum differences for topic set sizes larger than those that could be directly computed, but extrapolation is not appropriate for the new measures since their values depend directly on the number of topics in the test set.

For topic sets of size 50, a run needs at least 11 fewer topics with no relevant in the top 10 retrieved to have 95% confidence that it is better than a second run. Over the 1000 topic sets of size 50 generated to estimate the error rate and comparing all pairs of runs submitted to the track, only 11.0 % of the comparisons had a difference at least this large. This is a small percentage that confirms that the measure is only able to distinguish grossly different systems. The area measure could distinguish even fewer systems. For the area measure, the minimum Δ computed for 95 % confidence was 0.025; only 4.6% of the comparisons across all run pairs and the 1000 test sets had a difference in area score greater than 0.025.

Note that the best area *score* (not difference) obtained by a run in the robust track over the old set of 50 topics was 0.0203, so all robust track runs would be considered to be in a single equivalence class if only the old set of topics were used. This set of topics is known to be difficult, and all systems did sufficiently poorly on it that the area measure is not sensitive enough to distinguish one run from another. The best score obtained by a robust track run over the 50 new topics was 0.1062 with 38.6 % of the comparisons between pairs of systems having a difference greater than 0.025, so the measure can distinguish among systems for this topic set. But the new topic set appears to be unusually good: over the 1000 randomly selected 50-topic test sets, the best area score obtained by any run was only 0.043, and as stated above only 4.6 % of the comparisons across all run pairs had a difference greater than 0.025. The topics-with-no-relevant-retrieved measure was much less affected by the particular topic set. For the old topic set, 13.9 % of run pairs had a difference of at least 11 topics; for the new topic set, 11.4 % of run pairs had a difference of at least 11 topics; and over the 1000 randomly selected sets, 11.0 % of run pairs had a difference of at least 11 topics.

4 Conclusion

The TREC 2003 robust retrieval tracks was an initial effort to improve the consistency of retrieval performance by focusing on poorly performing topics. The results of the track provide strong confirmation that average values of traditional effectiveness measures do not reflect poorly performing topics. New measures introduced in the track do emphasize systems' worst topics as designed. The new measures are defined over a subset of the topics in the test set, however, causing them to be much less stable than traditional measures for a given test set size. In turn, the instability causes the margin of error associated with the measures to be large relative to the differences in scores commonly observed.

The robust track will continue in TREC 2004. The current plan for the track is to repeat this year's task using the same fifty old topics (they remain difficult topics) and another set of 50 new topics. A new aspect of the evaluation in the track will be to test whether a system can predict which topics it will perform most poorly on. A similar evaluation strategy in the TREC 2002 question answering track demonstrated that accurately predicting whether a correct answer was retrieved is a challenging problem [3].

References

[1] Chris Buckley and Ellen M. Voorhees. Evaluating evaluation measure stability. In N. Belkin, P. Ingwersen, and M.K. Leong, editors, *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 33–40, 2000.

[2] Alan Stuart. Kendall's tau. In Samuel Kotz and Norman L. Johnson, editors, *Encyclopedia of Statistical Sciences*, volume 4, pages 367–369. John Wiley & Sons, 1983.

[3] Ellen M. Voorhees. Overview of the TREC 2002 question answering track. In *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002)*, number NIST Special Publication 500-251, pages 57–68, 2002.

[4] Ellen M. Voorhees and Chris Buckley. The effect of topic set size on retrieval experiment error. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 316–323, 2002.

[5] Ellen M. Voorhees and Donna Harman. Overview of the sixth Text REtrieval Conference (TREC-6). In E.M. Voorhees and D.K. Harman, editors, *Proceedings of the Sixth Text REtrieval Conference (TREC-6)*, pages 1–24, August 1998. NIST Special Publication 500-240. Electronic version available at http://trec.nist.gov/pubs.html.

Overview of the TREC 2003 Web Track

Nick Craswell and David Hawking
CSIRO ICT Centre
Canberra, Australia
nick.craswell@csiro.au and david.hawking@csiro.au

Ross Wilkinson and Mingfang Wu
CSIRO ICT Centre
Melbourne, Australia
ross.wilkinson@csiro.au and mingfang.wu@csiro.au

March 22, 2004

Abstract

The TREC 2003 web track consisted of both a non-interactive stream and an interactive stream. Both streams worked with the .GOV test collection. The non-interactive stream continued an investigation into the importance of homepages in Web ranking, via both a Topic Distillation task and a Navigational task. In the topic distillation task, systems were expected to return a list of the homepages of sites relevant to each of a series of broad queries. This differs from previous homepage experiments in that queries may have multiple correct answers. The navigational task required systems to return a particular desired web page as early as possible in the ranking in response to queries. In half of the queries, the target answer was the homepage of a site and the query was derived from the name of the site (Homepage finding) while in the other half, the target answers were not homepages and the queries were derived from the name of the page (Named page finding). The two types of query were arbitrarily mixed and not identified.

The interactive stream focused on human participation in a topic distillation task over the .GOV collection. Studies conducted by the two participating groups compared a search engine using automatic topic distillation features with the same engine with those features disabled in order to determine whether the automatic topic distillation features assisted the users in the performance of their tasks and whether humans could achieve better results than the automatic system.

Part I
Non-interactive Experiments

1 Introduction

The non-interactive stream of the TREC 2003 Web Track centred on two tasks: a topic distillation task and a navigational task. The tasks use the 18 gigabyte, 1.25 million document partial crawl of the .gov domain, distributed on CD-ROM as the .GOV collection[1]. A full description of this year's track guidelines is available in a scparatc document in these proceedings.

[1] Seehttp://es.csiro.au/TRECWeb/

2 Tasks

This year's tasks represent two types of search where it is important for the system to be able to return homepages.

A homepage is designed to be the main page of a site. The homepage is important because it is often the first page users will see. It provides an introduction to the site, who created it and what it contains. It usually links to other pages in the site and provides access to other site functions such as search. The homepage URL is often given as the URL of the whole site, as in 'the TREC site is at `http://trec.nist.gov/`'.

In previous years Track participants have developed effective methods for homepage finding. Link-based ranking methods including anchortext propagation are useful, because homepages tend to have higher inlink counts than non-homepages. URL-based ranking is also useful, since homepages tend to have short URLs. This year, such evidence might be expected to be useful in processing topic distillation queries and for some of the navigational queries. In this year's navigational task, the hompepage queries were mixed with an equal number of queries designed to find named pages which were not site homepages.

2.1 Topic Distillation Task

The topic distillation task involves finding relevant homepages, given a broad query. The need underlying the query 'cotton industry' might be 'give me an overview of `.gov` sites about the cotton industry, by listing their homepages'. See Figure 1.

This differs from an adhoc-style interpretation 'give me all pages in `.gov` about the cotton industry'. Adhoc querying would facilitate direct access to a much larger number of pages, but topic distillation gives a better overview of which sites exist and therefore which government labs, groups and programs have sites.

It also differs from past Web track homepage finding tasks in that queries do not identify a specific site. To illustrate the difference, a homepage finding query in the 'cotton industry' area might be 'cotton pathology research unit'.

The topics were numbered 1–50. A good homepage will correspond to a site which:

- Is principally devoted to the topic,
- Provides credible information on the topic, and
- Is not part of a larger site also principally devoted to the topic

This requires judges to understand the structure of the site in question and the quality of information offered, and identify its homepage. We have more emphasis on homepageness than in last year's topic distillation task, and also have used broader queries to ensure that at least some sites exist.

Because many topics had less than 10 results, we abandoned the precision at 10 measure (P@10). The main measure was R-Precision (P@n where n is the number of relevant documents for the current topic).

2.2 Navigational Task

The navigational task is also known as the 'home/named page finding task'. Each query involves finding a particular page, which is a homepage in 50% of queries (participants did not know which queries were for homepages and which were for non-homepages). The query asks for the page by name. For example, when looking for the homepage `http://www.tva.gov` the user might type the query 'TVA-Tennessee Valley Authority'. When looking for the page `http://www.usdoj.gov/crt/ada/enforce.htm`, they might type the query 'ADA Enforcement'. See Figure 2.

Within the framework of this task, a number of research questions can be addressed, including:

1. Do systems tuned for homepage finding also work well on the named page finding task?

```
<top>
<num> Number: TD7
<title>cotton industry</title>
<desc>Description:
Where can I find information about growing, harvesting cotton
and turning it into cloth?
</top>

-------
Answers
-------

Cotton Pathology Research Unit
 cpru.usda.gov/

FAS Cotton Group
 ffas.usda.gov/cots/cotton.html

The Western Cotton Research Laboratory
 nps.ars.usda.gov/locations/locations.htm?modecode=53-44-05-00

Office of Textiles and Apparel
 otexa.ita.doc.gov/

U.S. Cotton Data Sets
 wizard.arsusda.gov/cotton/ars2.html

USDA Cotton Program
 www.ams.usda.gov/cotton/

USDA Cotton Briefing Room
 www.ers.usda.gov/Briefing/Cotton/
 www.ers.usda.gov/briefing/cotton/

USDA Key Topics -- Cotton
 www.ers.usda.gov/Topics/view.asp?T=101206
 www.ers.usda.gov/topics/view.asp?T=101206

Safety and Health Topics: Textiles (links to pages on cotton dust)
 www.osha-slc.gov/SLTC/textiles/

Southwestern Cotton Ginning Research Laboratory
 www.swcgrl.ars.usda.gov/
 www.swcgrl.ars.usda.gov/indextxt.htm
```

Figure 1: Example distillation topic with official qrels. Qrels give an overview of cotton sites in the .GOV corpus, therefore cotton activities in US Government.

```
<top>
<num> Number: NP151
<desc> Description:
ADA Enforcement
</top>

Answer: www.usdoj.gov/crt/ada/enforce.htm

--

<top>
<num> Number: NP161
<desc> Description:
TVA-Tennessee Valley Authority
</top>

Answer: www.tva.gov/
```

Figure 2: Example navigational topics with official qrel pages. The first is a 'named page' the second a 'homepage'.

2. Which techniques which have proven successful in homepage finding are also effective in named page finding?

3. Is it possible to identify homepage finding queries within a query stream?

Topics were numbered 151–450, with 150 homepage and 150 named page queries. For measures, we use the mean reciprocal rank of the first correct answer (MRR) and the proportion of queries where the correct answer appears in the top 10 (S@10). The number 10 was chosen for success rate calculation because search engines often provide 10 answers in the first page of search results.

3 Results

3.1 Topic Distillation Results

Across the 50 topics, 516 pages were judged relevant (average of 10.32 pages per query). Results for the best run submitted by each group are in Table 1. A full listing of runs is in Table 3.

Because there were only a few good answers for each query, system scores were low. This also seemed to reduce the stability of the results. The list of top 5 groups depends on how we sort Table 3. The top groups by R-Precision are in Table 1. Top groups by MAP were: CSIRO, Hummingbird, Neuchatel, UAmsterdam and UGlasgow. Top groups by P@10 were: Hummingbird, CSIRO, IBM Haifa, MSR Asia and UGlasgow.

Here we briefly summarize the information available about the experiments conducted by the top five groups (based on R-Precision).

CSIRO Documents scored via a linear combination of link indegree, anchortext propagation, URL Length and BM25. Linear combination (and BM25 parameters) were tuned using a home page finding query set (same tuning as navigational csiro03ki02). Stemming improved R-Precision by a further 0.0198.

Hummingbird Documents were given additional weight if their URL looked like a homepage URL, and also based on query word/phrase occurences in HTML markup such as title. There was no use of link counts or anchor text. Stemming had little effect.

UAmsterdam Used different representations and retrieval models. Okapi worked well on documents, titles and anchors. Language modelling worked very well on anchors and less well on documents and titles. Anchor text was important. Snowball stemming was used in all runs.

Copernic URL information was important (length and presence of query terms). Representations were each treated differently and included documents, extracted summaries, text with formatting, URL and title. First results were from a boolean AND query, followed by OR results. Porter stemming was used in all runs.

USunderland Used a novel document representation based on automatically assigned word senses as opposed to terms. The ranking algorithm consisted of a variation of Kleinberg's model of hubs and authorities in association with a number of vector space techniques including TF*IDF, and Cosine Similarity.

Based on information from these and other participants:

- Referring anchor text was important.
- Stemming was often helpful. Query expansion (blind feedback) was usually not necessary.
- URL information and link structure were helpful in several cases.
- Topic distillation was noted to bear some relationship to homepage finding, in terms of "what works".

Table 1: Best distillation run for each group, by R-Precision. The codes D, A, L indicate the use of document structure (D), Anchor text (A) and Link structure (L). Measures are R-Precision, mean average precision and precision at 10. (See appendix for a table of all runs.)

	R-Prec	MAP	P@10	Group	Run	D	A	L
1.	0.1636	0.1543	0.1240	csiro	csiro03td03	D	A	L
2.	0.1485	0.1387	0.1280	hummingbird	humTD03upl	D	_	_
3.	0.1432	0.1344	0.0980	uamsterdam	UAmsT03WtOk3	D	A	_
4.	0.1430	0.1325	0.0980	copernic	copTdRun5	D	_	_
5.	0.1407	0.1114	0.0940	usunderland	SBUNIQUE	D	_	L
6.	0.1391	0.1336	0.1140	uglasgow	uogtd4cahs	_	A	_
7.	0.1357	0.1371	0.0880	neuchatelu	UniNEtd4	D	A	L
8.	0.1354	0.1027	0.1160	microsoftasia	MSRA4002	D	A	L
9.	0.1262	0.1131	0.1060	tsinghuau	THUIRtd0305	D	A	L
10.	0.1173	0.1091	0.1220	ibmhaifa	JuruNoQDiff	D	A	L
11.	0.1096	0.0897	0.0920	umelbourne	MU03td01	D	A	_
12.	0.0918	0.0698	0.0920	meijiu	meijihilw1	D	_	_
13.	0.0906	0.0848	0.0760	vatech	VTtdgp5055	_	A	_
14.	0.0818	0.0799	0.0640	fub	fub03IneBBt	_	_	_
15.	0.0784	0.0818	0.0720	irit-sig	Merc1ti	_	_	_
16.	0.0769	0.0660	0.0440	kasetsartu	KUCONTENT	_	_	_
17.	0.0754	0.0728	0.0520	cas-ict	ICTWebTD12A	_	_	_
18.	0.0736	0.1016	0.0760	indianau	widittdb1	_	_	_
19.	0.0699	0.0896	0.0700	ajouu	ajouai0301	_	_	_
20.	0.0590	0.0691	0.0640	uillinoisuc	UIUC03W2s	_	_	_
21.	0.0395	0.0343	0.0280	lehighu	03wume206	_	A	L
22.	0.0281	0.0226	0.0320	umarylandbc	C2B	_	_	L
23.	0.0007	0.0001	0.0000	saarlandu	topics0	_	_	L

The small number of good answers for each query caused the measures to be much less stable than they have been in previous years. For P@10, systems should differ by at least 0.035 in absolute score for a

95% confidence in the difference, using the method of Voorhees and Buckley (SIGIR 2002). An absolute increase of 0.035 represents a 27.3% relative improvement over the top score, which is quite large. For R-Precision, the minimum absolute difference is 0.09. These differences represent quite a large swath of the full range of scores achieved.

The reason for this instability is as follows. Since more than half of the topics have less than 10 relevant, P@10 is always less than 1.0 for those topics, and so there's this overall degradation in the average due to those topics. The scores are also somewhat quantized, you don't see a continuous range of P@10. On the other hand, R-Precision has even more severe quantization, even though it averages better. Moreover, when there are less than 10 relevant documents, P@10 is an easier measure for systems to do well on than R-Precision, because they have more chances to get those documents.

3.2 Navigational Results

Judging involved identifying answers which appear at more than one URL. For example, the page `http://bernie.house.gov/imf/imf.asp` also appears at `http://www.bernie.house.gov/imf/imf.asp` so both were identified as correct answers for query 335. This process identified 19 queries with 2 URLs and 12 with more than 2.

Table 2: Best navigational run for each group, by mean reciprocal rank (MRR) of the first correct answer. The codes D, A, L indicate the use of document structure (D), Anchor text (A) and Link structure (L). Measures are MRR and the success rate at 10 as a percentage. (See appendix for a table of all runs.)

	MRR	S@10	Group	Run	D	A	L
1.	0.727	89.3	cmu	LmrEstUrl	D	A	-
2.	0.702	84.0	csiro	csiro03ki03	D	A	L
3.	0.688	84.7	neuchatelu	UniNEnp4	D	A	-
4.	0.665	87.0	iit	iit03sau	D	A	-
5.	0.651	84.3	microsoftasia	MSRANP1	D	A	-
6.	0.615	79.3	uglasgow	uogki2ca	-	A	-
7.	0.586	79.3	copernic	copNpRun1	D	-	-
8.	0.568	79.0	cas-ict	ICTWebKI12C	D	A	-
9.	0.561	81.0	tsinghuau	THUIRpf0301	-	A	-
10.	0.545	77.3	hummingbird	humNP03up	D	-	-
11.	0.530	75.0	umelbourne	MU03np4	D	A	-
12.	0.519	71.3	uamsterdam	UAmsT03WnLM3	D	A	-
13.	0.400	66.3	indianau	widitpff1	D	A	-
14.	0.374	59.0	vatech	VTnhpgp42	-	A	-
15.	0.350	53.7	rmit	RMITSEG3	D	-	-
16.	0.323	55.3	saarlandu	homepages0	-	-	-
17.	0.291	48.3	ajouu	ajouai0309	D	-	-
18.	0.120	16.3	ualaska	irttgrep	-	-	-
19.	0.067	9.3	lehighu	03wume298	-	A	-

The best run from each group is listed in Table 2. The full list of navigational runs is listed in Table 4.

Here we briefly summarize the information available about the experiments conducted by the top five groups (based on MRR).

CMU In order of MRR effectiveness, language models were formed from: in-link text, title text, full document text, image alternate text, modified size font text, meta keyword/description and a 3-character-gram URL. Best two runs used prior probabilities with URL classes from UTwente, trained on previous HP and NP tasks.

CSIRO Documents scored via a linear combination as described in the distillation section. Tunings were with a HP query set, NP query set and combined HP/NP query set. The NP tuning was most

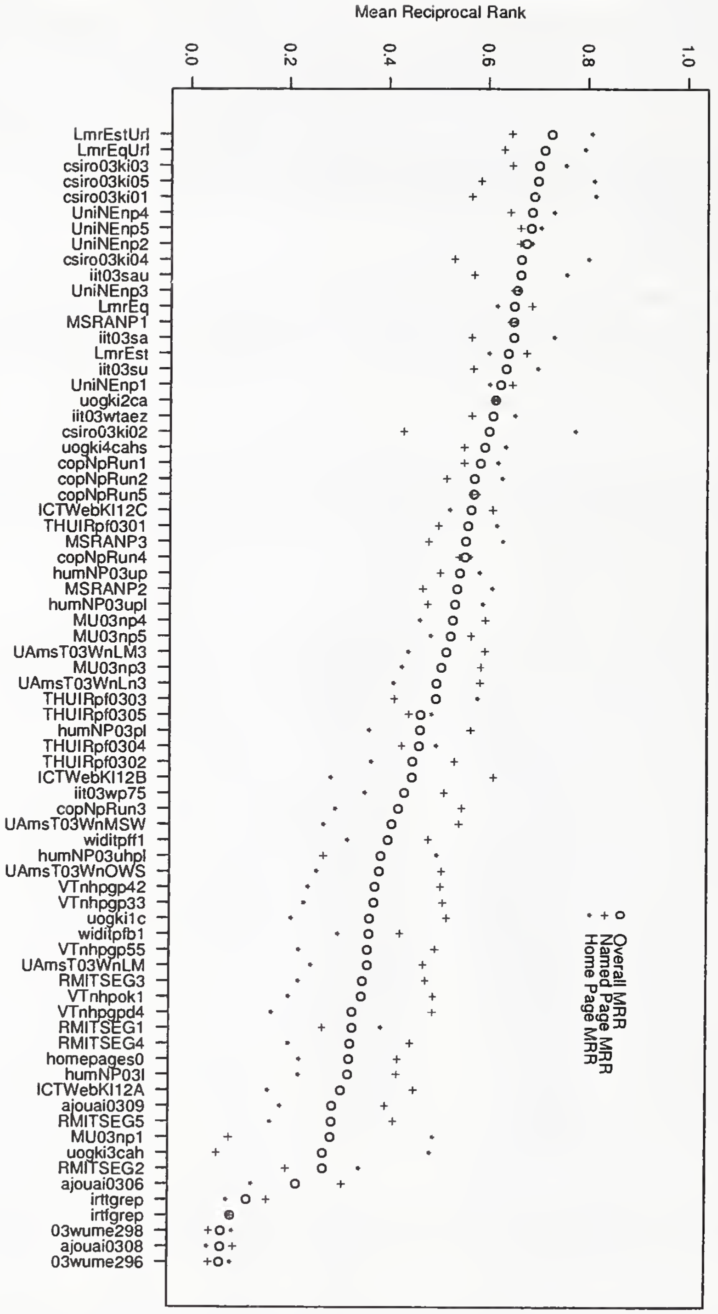


Figure 3: Breakdown of home page vs named page performance.

effective. Incorporating HP tuning was always harmful (in the combined tuning and in combinations via interleaving and CombSUM).

Neuchatel Document surrogates were 1) title with anchors from the current document, 2) title with anchors from referring documents and 3) various other segments of referring documents (title, h1, big). Scoring was with Okapi plus a proximity scoring function. Tuning was of the relative weight of surrogates, Okapi b parameter and the degree of proximity upweighting.

IIT Document surrogates were full text, title and anchors. Fusion was by CombMNZ with exponential z-score normalisation. A query task classification system was also employed, based on 32 words indicative of home page search such as 'home' or 'homepage'.

MSR Asia Document surrogates such as anchors and titles were employed, whose combination was tuned based on TREC-2002. Proximity information was used.

Based on information from these and other participants:

- It seems useful to consider different representations/surrogates based on document structure and including referring anchor text.
- Link structure gave mixed results.
- Stemming was not necessary for most participants.
- Several participants reported improvements based on proximity information, spans, phrases and fuzzy phrases.
- Few or no big wins from query classification (HP vs NP), but some promising attempts.

The scores for the navigational task are very stable. For the MRR measure on the full topic set, an absolute difference of 0.04 is enough to give 95% confidence. A 0.1 difference is reliable with as few as 100 topics.

There is only a little bit of a difference between named pages and homepages. If one uses the named page set alone, one needs to see an absolute difference between 0.06 and 0.07 for 95% confidence. For home pages, it's 0.06. If you pick 150 topics randomly from the larger set, it's 0.07, so the topic type does matter for the evaluation.

4 Conclusion

This year's topic distillation task was considered to be much more representative of real Web search than last year's. We also ran our first mixed query task, identifying approaches which work for home page and named page finding. Several groups attempted classification to differentiate between query types. More comprehensive mixed query tasks — possibly including topic distillation, topic relevance, service finding and navigational queries — seem to offer fertile ground for future evaluations.

Part II
Interactive Experiments

5 Introduction

For TREC 2003, the interactive track was a sub-track of the web track. The topic distillation, one of the web track tasks, has been selected as the interactive track task. The main motivation of the interactive sub-track was to investigate the role of human searchers in the topic distillation task.

6 Tasks

Eight search topics were selected from the topic set as used by the web topic distillation task. For each of these topics, a search scenario was provided in order to provide the participants a context for their search activity - it was not intended to boost the content available for searching.

1. Title: cotton industry

   Search task: You are to construct a resource list for high school students who are interested in cotton industry, from growing, harvesting cotton and turning it into cloth.

2. Title: folk art folk music

   Search task: Assume that you are an art teacher of a high school. You are about to introduce your students to U.S. folk art and folk music. Please prepare a list of bookmarks for your students for study materials.

3. Title: children's literature

   Search task: The teachers from your local primary school are spending a lot of their time on the web to search for materials on children's literature. Please help the teachers by setting up a children's literature web guide which points to useful websites for young readers/writers.

4. Title: wireless communications

   Search task: You are invited to give a presentation on wireless communication to university students. Please prepare a list of bookmarks as a handout to your audience. The bookmarks should cover information on existing and planned uses, research/technology, regulations and legislative interest.

5. Title: arctic exploration

   Search task: Assume that you are a high school student and working on an arctic exploration project. You are asked to collect some resources from the web for your project team on what kinds of exploration of the arctic are underway, especially of glaciers and ice.

6. Title: weather hazards and extremes

   Search task: Assume that you are a high school student and working on a project regarding the study of natural/weather hazards and extremes. You are asked to collect some resources from the web for your project team.

7. Title: electric automobiles

   Search task: You are going to give a seminar on the progress in producing/developing electric automobiles, and you will mention some online resources on this topic. Please prepare a list of bookmarks as a handout to your audience.

8. Title: Bilingual education

   Search task: You are a volunteer of your local community. You are asked to help to create a guide to all online information on bilingual education that may be of interest to your local residents.

7 Search Systems

NIST provided the access to its server with the two versions of "Panoptic search engine". One version of the engine is optimized for the topic distillation task by balancing relevance and homepageness. The other, content-oriented, version is Okapi-based and retruns page in descending order of likely relevance. However, participants were free to use any appropriate search engine. To keep consistent with the automatic topic distillation task, all searches and browses were restricted to the .GOV collection.

8 Experimental Protocol

Participants were free to use any experimental protocol that suits their experimental purpose. However, the guidelines suggested an experiment design proposed by Rutgers group. Similar to the experiment designs in past interactive tracks, this design allows the comparison of two systems or system variants. This year's design divides the eight topics into two blocks, with varying order of topics within each block. This design requires a minimum 16 searchers, each searcher needs to search a block of four topics on one system and another block on another system.

9 Evaluation

The saved lists from each search session were gathered and sent to NIST for assessment. The assessment was based on four criteria: relevance, depth, coverage, and repetition. The assessors were asked to answer the following questions/statement on a five-point Likert scale.

Relevance: The page is relevant for the topic.

1 = Agree strongly, 2 = Agree slightly, 3 = Neutral, 4 = Disagree slightly, 5 = Disagree strongly

Depth: Is the page too broad, too narrow or at the right level of detail for the topic?

1 = Too broad, 2 = Bit broad, 3 = Right level, 4 = Bit narrow, 5 = Too narrow

Coverage: The set of saved entry points covers all the different aspects of the topic.

1 = Agree strongly, 2 = Agree slightly, 3 = Neutral, 4 = Disagree slightly, 5 = Disagree strongly

Repetition: How much repetition/overlap is there within the set of saved entry points?

1 = None, 2 = Minimal, 3 = Some, 4 = A lot of, 5 = Way too much

Instruments for the collection of searcher background and subjective evaluation of search systems were also provided and suggested by the Guidelines.

10 Overview of Results

Two groups, CSIRO and Rutgers, participated in this sub-track. Here is a brief description of their testing hypotheses and initial findings.

10.1 CSIRO

CSIRO investigated the effectiveness of a task tailored delivery method to assist searchers evaluate and thus select key resource pages as described by the topic distillation task.

In their baseline interface, they used the delivery interface from the Panoptic topic distillation engine which provides searchers with a ranked list of potential relevant key resource pages. In their testing interface, they designed a site summary interface and a sitemap interface to explicitly support searchers to judge whether a site is relevant and whether a page is at the right scope.

From their initial analysis, they found that their searchers preferred the testing interface and perceived that they fulfilled their task better by using the testing interface than the baseline interface. However, they didn't find any significant difference between the two interfaces on searcher's performance in terms of relevance, depth, coverage and repetition. By further examining searchers' behavior, they found that the interface for grouping documents into sites changed search behavior: searchers tend to assess a number of pages from the same site by reading their summary information before they selected a page to read further. Also in the post-system questionnaire, the searchers strongly stated that the grouping interface was useful for them to select an entry point to search. However, confounded by many other factors, it is not clear whether this behavior would be beneficial to the overall task.

Comparing the searchers' performance of the baseline interface with that of the corresponding automatic system, they found a significant improvement in terms of relevance, depth and precision. That indicates that engagement of searcher's effort has a positive effect on the system performance.

10.2 Rutgers

The Rutgers group investigated the role that the layout of search results plays in supporting human searchers executing topic distillation tasks. Success was measured in terms of accuracy and precision, operationalized as coverage and overlap, so the searcher was expected to find documents that provide information on as many distinct aspects of the assigned topic as possible, with as little overlap between them as possible. Their hypothesis was that using the structure of the domain and of the document corpus in order to organize the search output, would help identify aspects of the search topic in different sub-domains of the document collection, would reduce the searchers' cognitive load and would produce better results than the classic hit list. They tested this hypothesis by using two user interfaces for the Panoptic search engine, one with a simple list output, and the second with documents clustered based on common URL elements.

Their initial analysis shows that although it does not produce better coverage than the linear interface, the hierarchical interface seems to be conducive to less effort for the searcher: fewer iterations, shorter search sessions, fewer documents seen, selected and viewed. With regards to subjective measures, users perceived the hierarchical one as easier to use and better at supporting the topic distillation task. These results were not statistically significant. What was significant is that the subjects perceived the two systems equally easy to learn and that they preferred the hierarchical display.

One advantage of the structured output, as suggested by the objective measures and highlighted by the users' comments, is the support for investigating different sub-domains of a document collection and consequently different aspects of a topic. The searcher does not need to make a cognitive effort to separate the search results into sub-domain, so the layout makes the interaction easier and more pleasant and more accurately supports the searcher's judgment on task completion.

Appendix A: All non-interactive runs

Table 3: All distillation runs

	R-Prec	MAP	P@10	Group	Run	D	A	L
1.	0.1636	0.1543	0.1240	csiro	csiro03td03	D	A	L
2.	0.1485	0.1387	0.1280	hummingbird	humTD03upl	D	_	_
3.	0.1438	0.1354	0.1120	csiro	csiro03td01	D	A	L
4.	0.1432	0.1344	0.0980	uamsterdam	UAmsT03WtOk3	D	A	_
5.	0.1430	0.1325	0.0980	copernic	copTdRun5	D	_	_
6.	0.1407	0.1114	0.0940	usunderland	SBUNIQUE	D	_	L
7.	0.1391	0.1336	0.1140	uglasgow	uogtd4cahs	_	A	_
8.	0.1361	0.1284	0.1080	uglasgow	uogtd5cass	_	A	L
9.	0.1357	0.1371	0.0880	neuchatelu	UniNEtd4	D	A	L
10.	0.1354	0.1027	0.1160	microsoftasia	MSRA4002	D	A	L
11.	0.1333	0.1166	0.1020	usunderland	TBBASE	D	_	L
12.	0.1328	0.1198	0.1240	hummingbird	humTD03up	D	_	_
13.	0.1325	0.1273	0.1020	uglasgow	uogtd2ca	_	A	_
14.	0.1283	0.1259	0.1020	usunderland	SBBASE	D	_	L
15.	0.1282	0.1107	0.0960	copernic	copTdRun2	D	_	_
16.	0.1278	0.0948	0.0880	usunderland	TBUNIQUE	D	_	L
17.	0.1262	0.1131	0.1060	tsinghuau	THUIRtd0305	D	A	L
18.	0.1259	0.1187	0.0980	copernic	copTdRun3	D	_	_
19.	0.1217	0.1258	0.1080	csiro	csiro03td05	D	A	L
20.	0.1183	0.1180	0.0960	copernic	copTdRun1	D	_	_
21.	0.1173	0.1091	0.1220	ibmhaifa	JuruNoQDiff	D	A	L
22.	0.1162	0.1272	0.1060	csiro	csiro03td02	D	A	L
23.	0.1096	0.0897	0.0920	umelbourne	MU03td01	D	A	_
24.	0.1086	0.1127	0.0860	uamsterdam	UAmsT03WtOkC	_	_	_
25.	0.1079	0.1008	0.1220	ibmhaifa	JuruFull	D	A	L
26.	0.1078	0.0824	0.1100	microsoftasia	MSRA1002	D	A	_
27.	0.1069	0.0978	0.1020	hummingbird	humTD03uhpl	D	_	_
28.	0.1061	0.1022	0.1220	ibmhaifa	JuruNoCohes	D	A	L
29.	0.1060	0.0993	0.0660	copernic	copTdRun4	D	_	_
30.	0.1056	0.1019	0.0840	uamsterdam	UAmsT03WtLM3	D	A	_
31.	0.1053	0.1099	0.0820	uglasgow	uogtd3cas	_	A	_
32.	0.1052	0.0946	0.1140	microsoftasia	MSRA4003	D	A	L
33.	0.1046	0.1285	0.0980	neuchatelu	UniNEtd1	D	A	L
34.	0.1036	0.0764	0.0800	tsinghuau	THUIRtd0301	D	A	L
35.	0.1027	0.0699	0.0960	microsoftasia	MSRA1001	D	A	_
36.	0.1021	0.0902	0.0840	neuchatelu	UniNEtd5	D	_	L
37.	0.1016	0.0933	0.1040	microsoftasia	MSRA3	D	A	L
38.	0.1004	0.1041	0.0880	ibmhaifa	JuruNoAnchor	D	_	L
39.	0.0995	0.0982	0.0860	ibmhaifa	JuruNoSS	D	A	_
40.	0.0994	0.0763	0.0840	tsinghuau	THUIRtd0302	D	A	L
41.	0.0988	0.1004	0.1200	csiro	csiro03td04	D	_	L
42.	0.0918	0.0698	0.0920	meijiu	meijihilw1	D	_	_
43.	0.0906	0.0848	0.0760	vatech	VTtdgp5055	_	A	_
44.	0.0902	0.0652	0.1060	meijiu	meijihilw3	D	_	L
45.	0.0899	0.1003	0.0560	hummingbird	humTD03pl	D	_	_
46.	0.0823	0.0862	0.0760	uamsterdam	UAmsT03WtOkI	_	_	L
47.	0.0823	0.0898	0.0660	vatech	VTtdgp52	_	A	_
48.	0.0818	0.0799	0.0640	fub	fub03IneBBt	_	_	_
49.	0.0818	0.0818	0.0620	fub	fub03IneBMt	_	_	_
50.	0.0811	0.0864	0.0760	neuchatelu	UniNEtd2	D	A	_
51.	0.0811	0.0864	0.0760	neuchatelu	UniNEtd3	D	A	L
52.	0.0786	0.0646	0.0620	tsinghuau	THUIRtd0303	D	A	L

Table 3: All distillation runs (continued)

	R-Prec	MAP	P@10	Group	Run	D	A	L
53.	0.0786	0.0778	0.0620	fub	fub03InLBt	_	_	_
54.	0.0784	0.0818	0.0720	irit-sig	Merc1ti	_	_	_
55.	0.0783	0.0870	0.0760	irit-sig	Merc2tm	_	_	_
56.	0.0783	0.0775	0.0640	fub	fub03InBMt	_	_	_
57.	0.0769	0.0660	0.0440	kasetsartu	KUCONTENT	_	_	_
58.	0.0754	0.0728	0.0520	cas-ict	ICTWebTD12A	_	_	_
59.	0.0736	0.1016	0.0760	indianau	widittdb1	_	_	_
60.	0.0730	0.0886	0.0680	uglasgow	uogtd1c	_	_	_
61.	0.0728	0.0806	0.0580	umelbourne	MU03td05	D	_	_
62.	0.0716	0.0810	0.0580	fub	fub03InLBo1t	_	_	_
63.	0.0699	0.0896	0.0700	ajouu	ajouai0301	_	_	_
64.	0.0692	0.0558	0.0600	tsinghuau	THUIRtd0304	D	A	L
65.	0.0687	0.0486	0.0700	meijiu	meijihilw4	D	_	L
66.	0.0669	0.0845	0.0680	irit-sig	Merc2tp	_	_	_
67.	0.0655	0.0699	0.0540	vatech	VTtdgp33	_	A	_
68.	0.0652	0.0648	0.0580	ajouu	ajouai0305	D	_	_
69.	0.0648	0.0748	0.0620	vatech	VTtdok4	_	A	_
70.	0.0634	0.0687	0.0880	indianau	widittdb1r1	_	_	L
71.	0.0632	0.0639	0.0380	cas-ict	ICTWebTD12B	_	_	_
72.	0.0626	0.0787	0.0980	indianau	widittdf1r1	D	A	L
73.	0.0625	0.0647	0.0400	cas-ict	ICTWebTD12C	D	A	_
74.	0.0614	0.0486	0.0700	meijiu	meijihilw2	D	_	_
75.	0.0594	0.0733	0.0620	vatech	VTtdgp41	_	A	_
76.	0.0590	0.0691	0.0640	uillinoisuc	UIUC03W2s	_	_	_
77.	0.0590	0.0611	0.0580	uillinoisuc	UIUC03Wp	_	_	L
78.	0.0588	0.0773	0.0880	indianau	widittdf1r2	D	A	L
79.	0.0568	0.0631	0.0540	uillinoisuc	UIUC03Wu1	_	_	L
80.	0.0566	0.0627	0.0540	uillinoisuc	UIUC03Wb	_	_	_
81.	0.0559	0.0636	0.0400	umelbourne	MU03td04	D	A	_
82.	0.0553	0.0616	0.0540	uillinoisuc	UIUC03Wu2	_	_	L
83.	0.0537	0.0650	0.0660	umelbourne	MU03td03	D	A	_
84.	0.0523	0.0352	0.0620	meijiu	meijihilw5	D	_	L
85.	0.0433	0.0555	0.0400	irit-sig	Merc1td	_	_	_
86.	0.0395	0.0343	0.0280	lehighu	03wume206	_	A	L
87.	0.0391	0.0412	0.0280	uamsterdam	UAmsT03WtLMI	_	_	L
88.	0.0361	0.0512	0.0440	hummingbird	humTD03l	_	_	_
89.	0.0281	0.0226	0.0320	umarylandbc	C2B	_	_	L
90.	0.0230	0.0222	0.0200	umarylandbc	C2A	_	_	L
91.	0.0204	0.0225	0.0180	lehighu	03wume359	_	A	L
92.	0.0181	0.0250	0.0160	ajouu	ajouai0302	D	_	L
93.	0.0007	0.0001	0.0000	saarlandu	topics0	_	_	L

Table 4: All navigational runs

	MRR	S@10	Group	Run	D	A	L
1.	0.727	89.3	cmu	LmrEstUrl	D	A	_
2.	0.713	88.0	cmu	LmrEqUrl	D	A	_
3.	0.702	84.0	csiro	csiro03ki03	D	A	L
4.	0.699	81.0	csiro	csiro03ki05	D	A	L
5.	0.692	83.7	csiro	csiro03ki01	D	A	L
6.	0.688	84.7	neuchatelu	UniNEnp4	D	A	_
7.	0.686	84.7	neuchatelu	UniNEnp5	D	A	_
8.	0.676	84.0	neuchatelu	UniNEnp2	D	A	_

Table 4: All navigational runs

	MRR	S@10	Group	Run	D	A	L
9.	0.667	86.3	csiro	csiro03ki04	D	A	L
10.	0.665	87.0	iit	iit03sau	D	A	_
11.	0.658	83.7	neuchatelu	UniNEnp3	D	A	_
12.	0.652	83.3	cmu	LmrEq	D	A	_
13.	0.651	84.3	microsoftasia	MSRANP1	D	A	_
14.	0.651	86.7	iit	iit03sa	D	A	_
15.	0.640	83.3	cmu	LmrEst	D	A	_
16.	0.636	85.7	iit	iit03su	D	A	_
17.	0.626	82.3	neuchatelu	UniNEnp1	D	A	_
18.	0.615	79.3	uglasgow	uogki2ca	_	A	_
19.	0.611	84.0	iit	iit03wtaez	D	A	_
20.	0.603	77.7	csiro	csiro03ki02	D	A	L
21.	0.595	75.7	uglasgow	uogki4cahs	_	A	_
22.	0.586	79.3	copernic	copNpRun1	D	_	_
23.	0.574	77.0	copernic	copNpRun2	D	_	_
24.	0.572	75.7	copernic	copNpRun5	D	_	_
25.	0.568	79.0	cas-ict	ICTWebKI12C	D	A	_
26.	0.561	81.0	tsinghuau	THUIRpf0301	_	A	_
27.	0.556	72.7	microsoftasia	MSRANP3	D	A	_
28.	0.555	74.7	copernic	copNpRun4	D	_	_
29.	0.545	77.3	hummingbird	humNP03up	D	_	_
30.	0.540	71.3	microsoftasia	MSRANP2	D	A	_
31.	0.535	77.7	hummingbird	humNP03upl	D	_	_
32.	0.530	75.0	umelbourne	MU03np4	D	A	_
33.	0.527	76.0	umelbourne	MU03np5	D	A	_
34.	0.519	71.3	uamsterdam	UAmsT03WnLM3	D	A	_
35.	0.508	76.7	umelbourne	MU03np3	D	A	_
36.	0.498	72.7	uamsterdam	UAmsT03WnLn3	D	A	_
37.	0.496	64.3	tsinghuau	THUIRpf0303	_	A	_
38.	0.466	63.7	tsinghuau	THUIRpf0305	_	A	_
39.	0.465	68.3	hummingbird	humNP03pl	D	_	_
40.	0.463	62.7	tsinghuau	THUIRpf0304	_	A	_
41.	0.450	75.3	tsinghuau	THUIRpf0302	_	A	_
42.	0.449	65.7	cas-ict	ICTWebKI12B	D	A	_
43.	0.433	67.0	iit	iit03wp75	_	_	_
44.	0.421	61.3	copernic	copNpRun3	D	_	_
45.	0.407	63.0	uamsterdam	UAmsT03WnMSW	_	_	_
46.	0.400	66.3	indianau	widitpff1	D	A	_
47.	0.386	56.7	hummingbird	humNP03uhpl	D	_	_
48.	0.383	59.3	uamsterdam	UAmsT03WnOWS	_	_	_
49.	0.374	59.0	vatech	VTnhpgp42	_	A	_
50.	0.372	59.0	vatech	VTnhpgp33	_	A	_
51.	0.363	55.7	uglasgow	uogki1c	_	_	_
52.	0.362	60.0	indianau	widitpfb1	_	_	_
53.	0.359	56.7	uamsterdam	UAmsT03WnLM	_	_	_
54.	0.359	57.7	vatech	VTnhpgp55	_	A	_
55.	0.350	53.7	rmit	RMITSEG3	D	_	_
56.	0.348	55.7	vatech	VTnhpok1	_	A	_
57.	0.330	51.3	vatech	VTnhpgpd4	_	A	_
58.	0.329	55.3	rmit	RMITSEG1	D	_	_
59.	0.325	54.0	rmit	RMITSEG4	D	_	_
60.	0.323	55.3	saarlandu	homepages0	_	_	_
61.	0.321	54.3	hummingbird	humNP03l	_	_	_
62.	0.308	54.0	cas-ict	ICTWebKI12A	_	_	_
63.	0.291	48.3	ajouu	ajouai0309	D	_	_

Table 4: All navigational runs

	MRR	S@10	Group	Run	D	A	L
64.	0.290	48.0	rmit	RMITSEG5	D	-	-
65.	0.288	40.0	umelbourne	MU03np1	D	A	-
66.	0.273	39.0	uglasgow	uogki3cah	-	A	-
67.	0.272	43.7	rmit	RMITSEG2	D	-	-
68.	0.220	41.3	ajouu	ajouai0306	D	-	-
69.	0.120	16.3	ualaska	irttgrep	-	-	-
70.	0.087	17.3	ualaska	irtfgrep	-	-	-
71.	0.067	9.3	lehighu	03wume298	-	A	-
72.	0.067	11.7	ajouu	ajouai0308	D	-	L
73.	0.065	8.7	lehighu	03wume296	-	A	-

Recognizing Gene and Protein Function in MEDLINE Abstracts

Richard D. Hull, Larry F. Waldman[†]
Axontologic, Inc.
12565 Research Parkway, Suite 300
Orlando, FL 32826
hull@axontologic.com

Abstract

Identification of genes and proteins that affect biological function in humans and other organisms is a critical step in the discovery of new medicinal therapies. Automatic recognition of MEDLINE abstracts that describe gene/protein function would be of tremendous benefit to researchers in industry, government, and academia. Our approach uses simple syntax and domain semantics to both identify sentences from MEDLINE abstracts that suggest gene function and to rank those abstracts by a measure of how many appropriate function instances they contain.

Introduction

Identification of genes and proteins that affect biological function in humans and other organisms is a critical step in the discovery of new medicinal therapies. Automatic recognition of MEDLINE abstracts that describe gene/protein function would be of tremendous benefit to researchers in industry, government, and academia. For example, drug discovery projects often begin with the identification of one or more disease-associated protein targets. Pharmaceutical biologists spend a large portion of their time combing the literature for research articles discussing novel protein targets. Automating this element of their jobs has the potential to result in accelerating the discovery process and reducing costs.

The first Genomics Track of the 12th Text Retrieval Conference (TREC-12) was designed to provide a forum for those interested in developing systems capable of addressing the challenges posed by automatic recognition of gene and protein function. Axontologic and twenty-four other organizations from academia, government and industry participated in the primary task of the Track during the summer and fall of 2003. The official results of all of the participants are available on the TREC website.[1]

This paper begins with an overview of the primary task of the Genomics Tracks. It describes our natural language processing inspired approach and discusses the results of our system. Finally, our conclusions are presented.

Primary Task

The primary task of the TREC-12 Genomics Track began with the release of the Track training set comprised of the task documents, 50 training topics and the training relevancy judgments for those 50 topics. These training resources were made available to the participants in May of 2003. Participants were encouraged to use the training set to understand the nature of the task and develop their systems.

The task documents or corpus contained over 525,000 MEDLINE abstracts indexed between April 2002 and April 2003. The 50 training topics (or queries) were the gene names from 50 LocusLink records. The National

[†]Larry Waldman, School of Computer Science, Carnegie Mellon University, larrywaldman@cmu.edu.

Library of Medicine's LocusLink database "provides a searchable interface to curated sequence and descriptive information about genetic loci."[2] One component of a LocusLink record is its GeneRIF (Gene References into Function), a list of concise statements of the gene's function with associated MEDLINE references. The relevancy judgments for the 50 training topics were those MEDLINE abstracts referenced in the corresponding LocusLink GeneRIFs.

Fifty test topics were released in late June. Each topic contained the gene names from a LocusLink record that was not part of the training set. Participants had until August 4th to submit up to two test runs comprised of a ranked list of at most 1000 MEDLINE abstract identifiers for each of the 50 topics. The results of these runs were judged by the TREC evaluators and returned to the participants two weeks later.

Methodology

Our goal was to explore the use of simple syntax and domain semantics for recognizing gene/protein function and to lay the foundation for competition in future TREC events. Our system is automatic and combines domain independent processing elements with a domain-specific lexicon.

After review of the training set it became clear that there were three challenges to be addressed. First, the gene names provided in the training topics were not comprehensive, i.e., some of the abstracts in the answer set did not mention the given names. Second, using gene names resulted in many false positives, because many of the retrieved documents did not discuss gene function. Third, LocusLink records are organism-specific, therefore, the system had to filter out abstracts that discuss gene function in other organisms. A fourth issue dealing with false negatives could not be addressed directly within the context of the task.

We used the MG (Managing Gigabytes) system[3] to index the MEDLINE corpus. A strategy for heuristic query expansion was developed using a generative grammar. Query terms were expanded using simple grammar rules to handle hyphenation, complex punctuation and common gene name variations, e.g., "alpha-1a adrenergic receptor" vs. "adrenergic receptor alpha-1a." The expanded set of gene names was then fed to MG to retrieve a list of candidate abstracts from the corpus. We limited the number of retrieved documents to the first 5000 in our test runs. The ranking produced by MG was saved for later use.

A proprietary lexicon of terms indicative of gene or protein function was developed by analyzing the training data and other public-domain sources of functional information for genes including the Gene Ontology. The lexicon includes verbs (cleave, inhibit, etc.), nominalizations (activation, regulation, etc.) and adverbs.

The ranked abstracts were parsed into sentences and the sentences were examined to locate query terms and function terms. The abstracts were scored using a function of the frequency of query term/function term pairs found in an abstract's sentences and their proximity to each other, i.e., a gene term adjacent to a function term ("caspase-3 cleaves…") is scored higher than a gene term/function term pair separated by many intervening words. The system differentiates between cases where the gene term is in the subject or object position of the function term and it handles passive constructions.

An additional check was made to verify that the organism of the query was contained in the MeSH terms of each returned abstract. Abstracts were ranked by their functional scores. If no query term/functional term pairs were found in an abstract, then the abstract was ranked using the original MG ranking, after all those abstracts containing query/function pairs.

Results

Axontologic submitted two runs labeled *axon1* and *axon2* – the second run had a slightly more liberal query expansion grammar. The official results are shown in Tables 1 and 2. The results for the two runs were very close with axon2 doing slightly better in average precision (0.3173 compared to *axon1*'s 0.3118).

Table 1. *Axon1* Results.

Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.6372	At 5 docs	0.3200
0.10	0.5764	At 10 docs	0.2400
0.20	0.4725	At 15 docs	0.2120
0.30	0.3930	At 20 docs	0.1890
0.40	0.3420	At 30 docs	0.1493
0.50	0.3020	At 100 docs	0.0742
0.60	0.2504	At 200 docs	0.0437
0.70	0.2268	At 500 docs	0.0203
0.80	0.2070	At 1000 docs	0.0104
0.90	0.1699	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.1468		
Average precision over all relevant docs			
non-interpolated	0.3118	Exact	0.2935

Table 2. *Axon2* Results.

Recall Level Precision Averages		Document Level Averages	
Recall	Precision		Precision
0.00	0.6361	At 5 docs	0.3200
0.10	0.5764	At 10 docs	0.2500
0.20	0.4697	At 15 docs	0.2187
0.30	0.4056	At 20 docs	0.1930
0.40	0.3543	At 30 docs	0.1520
0.50	0.3141	At 100 docs	0.0748
0.60	0.2598	At 200 docs	0.0433
0.70	0.2394	At 500 docs	0.0201
0.80	0.2100	At 1000 docs	0.0105
0.90	0.1748	R-Precision (precision after R docs retrieved (where R is the number of relevant documents))	
1.00	0.1518		
Average precision over all relevant docs			
non-interpolated	0.3173	Exact	0.2959

Discussion

While these results were competitive, there is much more to be done. There were seven queries that axon2 did not meet the median average precision. Six of the seven queries involved problems with the names of the genes given in the topic. For example, in topic 4, the official gene name is given as "guanine nucleotide binding protein (G protein), alpha activating activity polypeptide, olfactory type". There were two MEDLINE abstracts in the answer set for this query, 11901355 and 12037684. The gene names used in these two abstracts are synonyms for the official gene name that were not included in the topic: G-protein alpha(olf), AGalpha(olf), G-protein Golf, and Golf.

Topic 38 held a similar problem for our system. The topic used name "MIP-1-alpha", but we missed answer abstracts that used lexical variations MIP-1 alpha, MIP-1alpha, and (MIP)-1alpha. Our query expansion grammar did not correctly handle cases with multiple dashes.

Clearly the problem of gene name variability is a fundamental issue, much as word choice is in traditional information retrieval. If the MG system did not retrieve an answer abstract, then there was no way for later processing components to compensate for that. We plan to improve our query expansion grammar and to add additional gene synonyms from public domain sources.

On the positive side, there were five topics that one or both of *axon1* and *axon2* returned the best average precision score. Of these five, *axon1* returned an average precision score that was several times greater than the median in topics 10, 14 and 42. To understand what happened with these query topics, we have compared the original MG ranking to the final *axon1* ranking for the answer abstracts (see Tables 3, 4 and 5). In nearly every case, the rankings of target abstracts were improved by the use of function terms and in many cases the improvements were dramatic.

Table 3. MG and *Axon1* Rankings for Topic 10.

PMID	MG	Axon1
11750880	23	21
11756417	36	1
11913997	108	11
11961237	92	9
12135673	21	3
12187073	252	57
12359731	41	18
12468916	424	69

For example, four target abstracts for Topic 14, PMID's 11865975, 12167626, 12234259 and 12354983, were improved into the top 10 from their original rankings of 687, 1419, 1739, and 3217 respectively. MG returned the maximum 5000 articles for this topic.

Table 4. MG and *Axon1* Rankings for Topic 14.

PMID	MG	Axon1
11865975	687	6
11920569	1082	74
11983915	2169	18
12086670	219	11
12167626	1419	1
12189556	4071	64
12218115	806	20
12234259	1739	9
12270125	-	-
12370314	889	38
12374983	3217	4
12393617	-	-

The strength of our method is in its ability to promote abstracts that discuss the function of the gene over those abstracts that simply mention the gene. In those cases where there are a large number of abstracts containing the topic gene names, our method has the potential to make significant improvements.

Table 5. MG and *Axon1* Rankings for Topic 42.

PMID	MG	Axon1
11756426	743	1
11809755	683	121
11823458	24	108
11877420	891	7
11912192	502	192
11912196	1892	42
11948811	1549	61
11972038	27	37
12068009	90	31
12087104	677	123
12093166	266	126
12101040	222	39
12230982	643	209
12239221	543	198
12431992	488	105
12493631	761	224
12515826	259	9

Conclusions

The Genomics track of TREC-12 provided an environment for developing, testing and evaluating computational methods for ranking abstracts by how well those abstracts describe the function of genes. Participating in TREC has increased our understanding of the problems researchers face and our solutions to those problems will be incorporated in Axontologic's future products.

We intend to bolster the query expansion component of the system to address the problem of not recognizing gene name synonyms in answer abstracts. We are currently investigating more sophisticated methods of scoring that utilize finite state transducers and shallow syntactic parsing models. It is our belief that even simple methods, however, can be of value to the biomedical community given the large numbers of false negatives found by our system and that of the other competitors.

References

[1] Text REtrieval Conference (TREC) Homepage. http://trec.nist.gov/.

[2] Ward, J. Gene Indexing. NLM Tech Bull. 2002 Sep-Oct;(328):e6.

[3] Managing Gigabytes: Compressing and Indexing Documents and Images, Witten, I. H., Moffat, A, and Bell, T. C., Morgan Kaufmann Publishing, San Francisco, 1999.

TREC2003 QA at BBN: Answering Definitional Questions

Jinxi Xu, Ana Licuanan and Ralph Weischedel

BBN Technologies

50 Moulton Street

Cambridge, MA 02138

1 INTRODUCTION

In TREC 2003, we focused on definitional questions. For factoid and list questions, we simply re-used our TREC 2002 system with some modifications.

For definitional QA, we adopted a hybrid approach that combines several complementary technology components. Information retrieval (IR) was used to retrieve from the corpus the relevant documents for each question. Various linguistic and extraction tools were used to analyze the retrieved texts and to extract various types of kernel facts from which the answer to the question is generated. These tools include name finding, parsing, co-reference resolution, proposition extraction, relation extraction and extraction of structured patterns. All text analysis functions except structured pattern extraction were carried out by Serif, a state of the art information extraction engine (Ramshaw, et al, 2001) from BBN.

Section 2 summarizes our submission for factoid and list qeustion answering (QA). The rest of the paper focuses on defintional questions. Section 4 concludes this work.

2 FACTOID AND LIST QUESTIONS

The factoid system is the same as our system for TREC 2002 (Xu, et al, 2003), except for a couple of modifications. One modification is to boost the score for answers that occurred multiple times in the corpus. This is similar to previous studies (e.g. Clarke, et al, 2001) that employed redundancy information to improve QA performance. Assuming the occurrences of an answer a are $a_1, a_2, \ldots a_n$, which are ranked by IR score, then the final score for a is

$$score(a_1) + c \sum_{1<i<=n} score(a_i)$$

We set c=0.001.

The other modification is to use additional information to validate answers. Specifically,

- **Validation based on question type**: For questions of the form "What X is ...", the system uses WordNet to verify that the question type is a hypernym of the answer. (For example, "Tagalog" is a valid answer for "What language ..." because "language" is a hyperym of "Tagalog".)
- **Validation of answers for questions looking for a date**: We required certain constraints on date answers based on the form of the question. For example, "When (was|did) ... " questions are likely to refer to a specific date in the past, and a good answer candidate should contain a year. "When is ..." questions may refer to either a specific

date in the past (in which case they should contain a year), or a relative date, such as "the day after X". "What month ..." questions should contain a month, etc.

- **Validation of answers for questions looking for a measurement**: We wrote patterns to detect four common measurement questions: dimension, duration, speed and temperature. Valid candidate answers are expected to have both a quantity and a unit of the appropriate type.

- **Validation of answers for questions looking for an author**: Specific patterns were written to extract "who-wrote" answers from the text.

- **Validation of answers for questions looking for an inventor**: Specific patterns were written to extract "who-invented" answers from the text.

- **Validation based on verb-argument**: We used WordNet to match verbs (for example, "Who killed X?" = "Y shot X"). In addition, we refined the scoring of slot matching to take into account the number of "filler" words that appear in between the question words.

List questions were processed like factoid questions, except that answers were ranked and the list of answers was truncated when the score of an answer drops below 90% of that of the best one.

We submitted three runs, BBN2003A, BBN2003B and BBN2003C. The differences are:

- BBN2003A. The Web was not used in answer finding.

- BBN2003B. For factoid questions, answers were found from both the TREC corpus and the Web. For list questions, it is the same as BBN2003A, except the constant *c* in the score computing formula was increased to 0.1.

- BBN2003C. For factoid questions, it is the same as BBN2003B, except for one difference. If an answer was found from both the TREC corpus and the Web, its score was boosted. For list questions, it is the same as BBN2003B.

Our technique to use the Web for QA is documented in our TREC 2002 work (Xu et al, 2003). Table 1 shows the NIST scores for the three runs. Two observations can be made. First, using the Web improved factoid QA. This is not surprising given previous TREC results by our group as well as by other groups. Compared with our TREC 2002 results, however, the impact of using the Web on QA performance is much smaller, improving accuracy from 0.177 to 0.208, a 3% improvement absolute. In comparison, for TREC 2002, using the Web produced a much larger improvement (10% absolute) in accuracy. More work is needed to determine if the reduced benefit of using the Web is due to changes in the characteristics of the questions or due to the modifications we made to last year's system. Second, using a large *c* significantly improved the performance of the list questions (from 0.087 to 0.097). This indicates that taking advantage of answer redundancy is more crucial for list questions than for factoid questions.

	BBN2003A	BBN2003B	BBN2003C
Factoid	0.177	0.208	0.206
List	0.087	0.097	0.097

Table 1: Scores for factoid and list questions

3 DEFINITIONAL QUESTIONS

3.1 System Overview

Our system processed a definitional question in a number of steps. First, question classification identified the question type, i.e. whether a question is a *who* or a *what* question. The distinction is necessary because some subsequent processing treats the two types of questions differently. Also in this step, the question target was extracted from the question text, by stripping of "What is", "Who is" etc.

Second, information retrieval pulled documents about the question target from the TREC corpus. This was achieved by treating the question target as an IR query. BBN's HMM IR system (Miller, et al, 1999) was used for this purpose. For each question, the top 1000 documents were retrieved.

Third, heuristics were applied to the sentences in the retrieved documents to determine if they mention the question target. Sentences that do not mention the question target were dropped.

Fourth, *kernel facts* that mention the question target were extracted from sentences by a variety of linguistic processing and information extraction tools. A kernel fact is usually a phrase extracted from a sentence. The purpose of using kernel facts is twofold: to minimize irrelevant materials in the answer and to facilitate redundancy detection.

Fifth, all kernel facts were ranked by their type and their similarity to the *profile* of the question. The question profile is a word centroid that models the importance of different words in answering the question.

Finally, heuristics were applied to detect redundant kernel facts. Up to a cap on the total answer length, facts that survived redundancy detection were output as the answer to the question.

3.2 Checking if a Sentence Mentions a Question Target

First, we check if a document mentions a question target at all. If a document does not mention a question target, we drop the whole document from consideration. For *who* questions, we require a document to contain a word sequence "*F...L*" (*F* and *L* are the first and last names of the question target) and the distance between *F* and *L* is less than 3. The purpose is to match "George Bush" with "George Walker Bush". We assume the first and last names are the first and last word of the question target respectively. For *what* questions, we require the document to contain the exact string of the question target, except that plural forms were converted to singular before string comparison.

If a sentence contains a noun phrase that either matches the question target directly (via string comparison) or indirectly (through co-reference), we think it contains the question target. We used Serif (Ramshaw et al, 2001) for co-ref resolution and parsing. For *who* questions, only the last name was used in string comparison.

3.3 Extraction of Kernel facts

3.3.1 Appositives and Copula Constructions

An example appositive is the phrase "George Bush, the US President". An example copula is the sentence "George Bush is the US President." In both cases, the phrase "the US President" is a definition for "George Bush". Appositives and copulas were extracted from the parse trees of the sentences based on simple rules using Serif.

3.3.2 Propositions

Propositions represent an approximation of predicate-argument structures and take the form: predicate (role$_1$: arg$_1$, ... , role$_n$: arg$_n$). In the context of this work, the predicate is typically a verb. Arguments can be either an entity or another proposition. The most common roles include logical subject, logical object, and object of a prepositional phrase modifying the predicate. For example, "Smith went to Spain" is represented as went(logical subject: Smith, PP-to: Spain). Propositions were extracted from parse trees using Serif.

We classified propositions into special propositions and ordinary ones. We manually created a list of predicate-argument structures that we thought were particularly important in defining an entity. For example, "<PERSON> was born on <DATE>" is one of the predicate -argument structures for persons. Propositions that matched one of such pre-defined structures were classified as special while others were classified as ordinary.

3.3.3 Structured Patterns

We handcrafted over 40 rules to extract structured patterns that are typically used in defining a term. Similar techniques were also used by Columbia University (Blair-Goldensohn, et al, 2004) and Language Computer Corporation (Harabagiu, et al, 2004) in their TREC 2003 work. For example, one such rule is "<TERM> ,? (is|was)? also? <RB>? called|named|known+as <NP>". Applied to a parsed sentence, the rule will match the question target (<TERM>), optionally followed by a coma, optionally followed by "is" or "was", optionally followed by "also", optionally by an adverb (<RB>), followed by "called", "named" or "known as" and followed by a noun phrase (<NP>). In the pattern, the "?" denotes optional, "+" concatenation, and "|" alternative. If the question is "What are tsunamis?", the pattern will extract the phrase "Tsunamis, also known as tidal waves" from the sentence "Tsunamis, also known as tidal waves, are caused by earthquakes."

3.3.4 Relations

As discussed in Section 3.3.2, propositions simply consist of lexical predicates. Since different lexical predicates can represent the same underlying relation, normalizing these propositions into relations that are commonly found in an ontology, where possible, is obviously desirable for definitional QA.

In this work, relations were extracted by Serif. Serif can extract the 24 binary relations defined in the ACE guidelines (Linguistic Data Consortium, 2002). Using lexicalized patterns, Serif extracts those relations from the propositions. For example, the relation role/general-staff("Gunter Blobel", "Rockefeller University") will be extracted from the sentence "Dr. Gunter Blobel of The Rockefeller University won the Nobel Prize for medicine today for protein research that shed new light on diseases including cystic fibrosis and early development of kidney stones.".

The QA guidelines require the answer to a definitional question to be a list of textual strings rather relations. We mapped a relation extracted by Serif to a phrase by finding the smallest phrase in the parse tree that contains a mention of the question target and the other argument of the relation. For the above example, the extracted phrase would be "Dr. Gunter Blobel of The Rockefeller University".

3.3.5 Sentences

In addition to the above types of kernel facts, we used full sentences as fall back facts in order to deal with sentences from which none of the above-mentioned types of kernel facts can be extracted.

3.4 Ranking the Kernel Facts

The ranking order of the kernel facts is based on two factors: their type and their similarity to the profile of the question. Appositives and copulas were ranked at the top, then structured patterns, then special propositions, then relations and finally ordinary propositions and sentences. Within each type, kernel facts were ranked based on their similarity to the question profile. The similarity is the tf.idf score where both the kernel fact and the question profile were treated as a bag of words. We used the tf.idf function described by Allan et al, 2000.

The question profile was created in three possible ways. First, we searched for existing definitions of the question target from a number of sources. The resources include: WordNet glossaries, Merriam-Webster dictionary (www.m-w.com), the Columbia Encyclopedia (online at www.bartleby.com), Wikipedia (www.wikipedia.com), the biography dictionary at www.s9.com and Google. To search for biographies on Google, we used the person name and the word "biography" as a query (e.g. "George Bush, biography"). A simple rule-based classifier was used to weed out false hits. If definitions were found from these sources, the centroid (i.e,. vector of words and frequencies) of the retrieved definitions was used as the question profile.

If no definitions were found from the above sources, we considered two options. If the question is a *who* question, we used the centroid of a collection of 17,000 short biographies from www.s9.com as the question profile. Our hope is that using a large number of human created biographies, we can predict what words are important for biography generation. If the question is a *what* question, we used the centriod of all kernel facts about the question target as the question profile. The assumption here is that the most frequently co-occurring words with the question target are the most informative words for answering the question. Similar techniques have been used in definitional QA (Blair-Goldsenshon, et al, 2003) and summarization (Radev, et al, 2000).

3.5 Redundancy Removal

The goal of redundancy removal is to determine if the information in a kernel fact *f* is covered by a set of kernel facts *S* that are already in the answer. Three methods were used to decide if *f* is redundant with respect to *S*:

- If *f* is a proposition, we check if one of the facts in *S* is equivalent to *f*. Two propositions are considered equivalent if they share the same predicate (e.g., verb) and same head noun for each of the arguments. If such a fact is found, *f* is considered redundant.
- If *f* is a structured pattern, we check how many facts in *S* were extracted using the same underlying rule. If two or more facts were found, we consider *f* redundant.
- Otherwise, we check the percentage of content words in *f* that have appeared at least once in the facts in *S*. If the percentage is very high (i.e., >0.7), we consider *f* redundant.

3.6 Results and Discussion

The algorithm to generate an answer for a definitional question is:

1. Set the answer set *S*={ }
2. Rank all kernel facts based on their similarity to the question profile regardless of their type. Iterate over all facts: In each iteration, discard a fact if it is redundant with respect to *S*. Otherwise, add the fact to *S*. Go to the next step if *S* has *m* facts.
3. Rank all remaining facts by type (the primary field) and then by similarity (the secondary field). Iterate over the ranked facts: In each iteration, add a fact to *S* if it is not redundant. Go to step 5 if the size (i.e., the number of non-space characters) of *S* is greater than *max_answer_size* or the number of sentences and ordinary propositions in *S* is greater than *n*.
4. If *S* is empty, rank all sentences in the top 1,000 retrieved documents based on the number of shared words between a sentence and the question target. Add the top 20 sentences to *S*.
5. Return *S* as the answer to the question.

We submitted three official runs, BBN2003A, BBN2003B and BBN2003C. For all three runs, *max_answer_size*=4,000 bytes. For BBN2003A, *m*=0 and *n*=5. For BBN2003B, *m*=0 and *n*=20. For BBN2003C, *m*=5 and *n*=10. The parameters were empirically set based on a set of about 79 development questions. Table 2 shows the results of the three runs. Overall, our results are satisfactory, given the median and best scores of all runs provided by NIST. In fact, BBN2003C achieved the highest score for definitional QA at TREC 2003.

Shortly after submitting the official runs and after discussion with NIST, we submitted a baseline run. The goal of the baseline was to give every group (including us) a chance to calibrate the results of their official runs. For each question, the baseline sequentially selected from the top 1,000 documents the sentences that mention the question target. The same heuristic in Section 3.2 was used to check if a sentence mentions the question target. As a fallback, if no sentences were found to mention the question target, all sentences in the top 1,000 documents were selected and ranked by the number of shared words between the question target and the sentences. For answer generation, we iterated over the selected sentences. For each sentence, we checked the percentage of words in the sentence that had occurred in previous sentences in the output answer. If the percentage was greater than 70%, the sentence was considered redundant and was skipped. Otherwise, the sentence was appended to the answer. The iteration continued until all sentences were considered or the answer length (i.e. the number of non-space characters in the answer) is greater than 4,000. Note that we applied a large length threshold because the F-score favors recall over precision. The baseline run was assessed by NIST in the same way as the official runs.

As shown in Table 2, the baseline performed surprisingly well, with an F-score 0.49. In fact, it outperformed all runs NIST received except BBN2003A, B&C. Our official runs (BBN2003A, B&C) are higher than the baseline, but the improvements are modest. One possible explanation for the unexpectedly good baseline is that the current state of the art of definitional QA is immature. The other is that with $\beta=5$ the F-score is overly recall-oriented and as such was "fooled" by the long answers produced by the baseline.

BBN2003A	BBN2003B	BBN2003C	Baseline
0.521	0.520	0.555	0.49

Table 2: Results for Definitional QA

Table 3 shows the scores for *who* and *what* questions for BBN2003C. The average score for *who* questions is somewhat better that that of *what* questions, but due to the relatively small number of questions, it is hard to determine if the difference is statistically meaningful.

TYPE	Number of questions	NIST F score
Who questions	30	0.577
What questions	20	0.522
Total	50	0.555

Table 3: BBN2003C score breakdown based on types of definitional questions

Figure 1 shows the score distribution over the 50 definitional questions sorted by F score. Quite a few questions (10) get a score of zero or close to zero. An initial analysis shows that a major source of failures is faulty assumptions we made in interpreting the question target. One example is "What is Ph in biology?". Our system assumed the literal string "Ph in biology" is the question target and tried to find it in text. Understandably, it failed. Another example is "Who is Akbar the Great?". Our system assumed the last name is "Great". These problems can be fixed..

Another major source of errors is erroneous redundancy removal. For example, for the question "Who is Ari Fleischer?", the inclusion of the kernel fact "Ari Fleischer, Dole's former spokesman who now works for Bush" in the answer masks the fact "Ari Fleischer, a Bush spokesman". The latter was considered to be redundant because all the words in it appeared in the former one. We hope better redundancy detection strategies will overcome such problems.

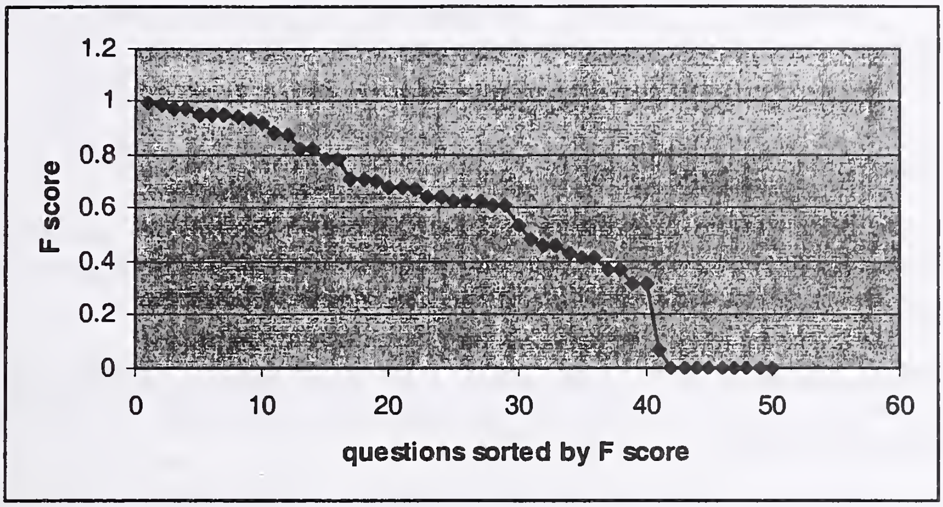


Figure 1: Score distribution over 50 definitional questions for BBN2003C

A question by question analysis shows that when a question obtains a bad F score, it is usually due to recall rather than precision. In fact, for BBN2003C, if we assume perfect precision for all questions, it would merely increase the average F score from 0.555 to 0.614. However, if we assume perfect recall for all questions, it would increase the score to 0.797. This imbalance is understandable because the F-metric used for TREC 2003 QA emphasizes recall by a factor of five over precision.

4 CONCLUSIONS

In TREC 2003 QA, we focused on definitional questions. Our approach combines a number of complementary technologies, including information retrieval and various linguistic and extraction tools (e.g., parsing, proposition recognition, pattern matching and relation extraction) for analyzing text. Our results for definitional questions are excellent compared with the results of other groups. However, much work remains as our results are only modestly better than a baseline that did little more than sentence selection using IR.

References

Allan, J., Callan, J., Feng, F., and Malin, D. 2000. "INQUERY at TREC8." In *TREC8 Proceedings*, Special publication by NIST, 2000.

Blair-Goldensohn, S., McKeown, K., and Schlaikjer, A., 2003. "Answering Definitional Questions: A Hybrid Approach." To appear in Maybury, M., editor, *New Directions in Question Answering*, AAAI Press. Chapter 13, during 2004.

Blair-Goldensohn, S., McKeown, K., and Schlaikjer, A., 2004. "A Hybrid Approach for QA Track Definitional Questions." To appear in *TREC 2003 Proceedings*, Special publication by NIST, 2004.

Clarke, C., Cormack, G., and Lynam, T., 2001. "Exploiting redundancy in question answering." In *Proceedings of SIGIR*, 2001.

Harabagiu, S., Moldovan, D., Clark, C., Bowden, M., Williams, J., and Bensley, J., 2004. "Answer Mining by Combining Extraction Techniques with Abductive Reasoning". To appear in *TREC 2003 Proceedings*, Special publication by NIST, 2004.

Linguistic Data Consortium, 2002. "ACE Phase 2: Information for LDC Annotators", http://www.ldc.upenn.edu/Projects/ACE2/.

Miller, D., Leek, T., and Schwartz, R., 1999. "A hidden markov model information retrieval system." In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1999.

Radev, D., Jing, H., and Budzikowska, M., 2000. "Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies." In *ANLP/NAACL Workshop on Summarization*, Seattle, WA, 2000.

Ramshaw, L., Boschee, E., Bratus, S., Miller, S., Stone, R., Weischedel, R., and Zamanian, A., 2001. "Experiments in Multi-Modal Automatic Content Extraction." In *Proceedings of Human Language Technology Conference*, San Diego, CA, March 18-21, 2001.

Xu, J., Licuanan, A., May, J., Miller, S. and Weischedel, R., 2003. "TREC2002 QA at BBN: Answer Selection and Confidence Estimation." In *TREC 2002 Proceedings*, Special publication by NIST, 2003.

REGEN: Retrieval and Extraction of Genomics Data

Rocio Guillén, Tasnim Ferdous
Computer Science Department, California State University San Marcos
email:{rguillen,ferdo001}@csusm.edu

Abstract

In this paper we present REGEN (Retrieval and Extraction of GENomics Data) a natural language processing system that retrieves and extracts information from Genomics data. These two tasks are independent of each other in the sense that the retrieved documents are not input to the extraction task. The retrieval task is based on a combination of exact-match and partial-match searching. The extraction task uses syntactic and semantic cues as patterns to generate candidate GENERIFs. We are currently generating just one candidate.

1. Introduction

One of the goals of the TExt Retrieval Conference (TREC) is to encourage research in information retrieval from large test collections. The nature of the tasks, data and evaluation procedures is experimental. TREC activity is organized into "tracks" of common interest, such as question-answering, multi-lingual IR, Web searching, and interactive retrieval. Because a great deal of genomics information resources are available one of the new tracks for 2003 was the TREC Genomics track focused on the retrieval of texts to aid users to acquire new knowledge in a sub-area of biology linked with genomics information. We have implemented a search engine for information retrieval and an information extraction procedure to participate in TREC 2003 in the Genomics Track. The Genomics Track consisted of two tasks namely primary and secondary. The focus of the primary task was on ad-hoc information retrieval and the focus of the secondary task was on information extraction. The Information Retrieval (IR) approach used for the search engine was the Extended Boolean model with partial matching and term weighting for the ranking (1,2,3,4,5). For the extraction we analyzed the MEDLINE abstracts provided as training set looking for syntactic and semantic cues we could use as patterns for generating GENERIFs candidates.

2. Architecture

We have built a system with two independent modules, namely retrieval and extraction. The overall architecture of the system is presented in Figure 1.

2.1 Retrieval

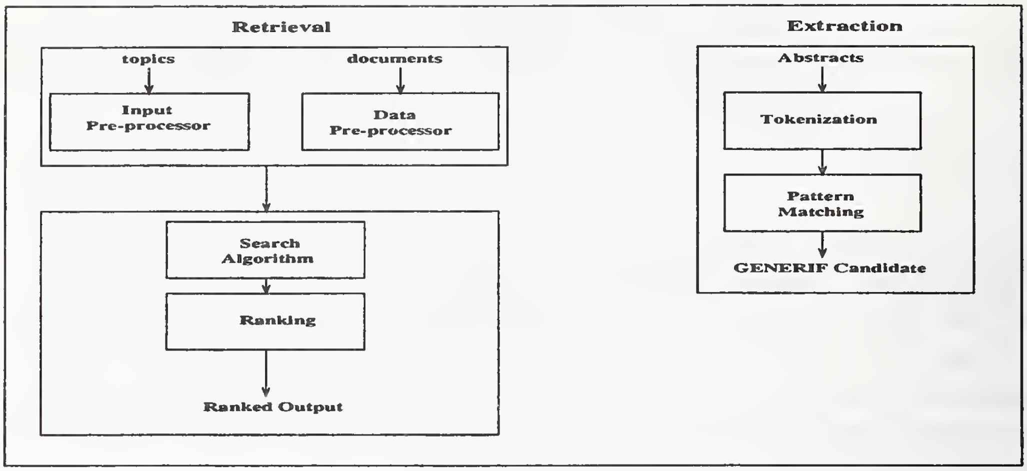


Figure 1 REGEN

Pre-processing of both the documents and queries was necessary to normalize, expand and modify data. We implemented a pre-processor for the document collection (Data Pre-processor) and one pre-processor for the topics or queries (Input Pre-processor). The Data pre-processor takes as input the document collection in XML format. and outputs two files called trec_XML.con and trec_XML.idx. The file called trec_XML.con stores the pre-processed documents and the file called trec_XML.idx keeps pointers to different positions of the trec_XML.con file. The purpose is to process information included in one of the following tags: MedlineID, PMID, ArticleTitle, AbstractText, DescriptorName, and NameOfSubstance. The Input pre-processor reads the topics from the topic file and then reorganizes them by eliminating redundant fields, LocusLink ID and name type, from the file. The basic steps carried out are the following:

1. Read a topic, which is a string of characters from the input file.
2. Write topic number in a new file.
3. Skip second substring and read third substring.
4. Replace third substring read for Organism name with appropriate names.
5. Separate Gene name and products using "—" as marker.

Mapping the name of species into corresponding MeSH equivalent names modifies the third substring of the topic file, which represents the Organism name. Thus,

∗ the string Human replaces Homo Sapiens.
∗ the string Mice replaces Mus musculus.
∗ the string Rat replaces Rattus norvegicus.

∗ the string Drosophila replaces Drosophila melanogaster.

The topics thus processed are further expanded applying some specific rules, presented next, to generate queries used by the search engine. The rules used were obtained after carrying out experiments to find patterns in the topics, and are consistent for all the topics. These rules are applied on gene names with OFFICIAL_GENE_NAME type only. The reason behind this restriction is that we observed that mainly the gene names of this type are rather complex and need to be split for further processing.

2.1.1 Rules

We came up with ten rules for normalizing and expanding queries. The rules reflect the patterns we found by carrying out different experiments and analyzing the results generated.

Rule 1. A gene name with a pattern like <cyclin-dependent kinase inhibitor 1A (p21, Cip1)> is split into two strings. The first string, s1, corresponds to all the characters appearing before “(", i.e., "cyclin- dependent kinase inhibitor 1A". The second string corresponds to "(" , ")", i.e., "(p21, Cip1)", which includes two substrings separated by a comma. The second string is split into s2 that corresponds to "p21" and s3 that corresponds to "Cip1", and the gene name is represented as <s1(s2,s3)>. Where, each string will be considered as the new query ""cyclin-dependent kinase inhibitor 1A p21 Cip1".

Rule 2. A gene name with a pattern like <glycine receptor, alpha 1 (startle disease/hyperekplexia, stiff man syndrome)> can be represented as four strings <s1, s2 (s3, s4)>. s1 corresponds to "glycine receptor", s2 corresponds to "alpha 1", s3 corresponds to "startle disease/hyperekplexia", and s4 corresponds to "stiff man syndrome". Such pattern can be further expanded as s1, s1 + s2, s1 + s3, and s1 + s4.

Rule 3. A gene name like <luteinizing hormone/choriogonadotropin receptor> can be represented as the pattern <s1 s2/s3 s4>. s1 corresponds to "luteinizing", s2 corresponds to "hormone", s3 corresponds to "choriogonadotropin" and s4 corresponds to "receptor". The pattern is further expanded as s1 + s2 + s4, and s1 + s3 + s4. Therefore, the newly generated queries are "luteinizing hormone receptor" and "luteinizing choriogonadotropin receptor".

Rule 4. A gene name with a pattern like <phospholipase C, gamma 1> can be represented as <s1, s2> where s1 corresponds to "phospholipase C" and s2 corresponds to "gamma 1". The pattern is further expanded to s1 + s2 and reduced to s1. Therefore, the new generated queries are "phospholipase C gamma 1" and "phospholipase C".

Rule 5. Any single number (N) or literal (L) in a gene name or a newly generated query is concatenated with the previous word. If a gene name or a newly generated query is in the pattern <Calcineurin B> that is <s1 L> where s1 corresponds to "Calcineurin" and L corresponds to "B" then the new pattern is s1L. That is, the newly generated query is "CalcineurinB". If a gene name or a newly generated query is in the pattern < alpha 1> that is <s1 N> where s1 is "alpha" and N is "1" then the new pattern becomes s1N. This means that the newly generated query is "alphal".

Rule 6. When the pattern is a list of names separated by comma and within parentheses then each of the elements in the list becomes a query itself. For instance, a gene name with the pattern <Tachykinin (Substance P, Neurokinin A, Neuropeptide K, Neuropeptide gamma)> can be represented as <s1 (s2, s3, s4, s5, s6.................. sn)>. s1 corresponds to "Tachykinin", s2 to "Substance P", s3 to "Neurokinin A", s4 to "Neuropeptide K", and s5 to "Neuropeptide gamma". Therefore the newly generated queries are s1, s2. s3. s4, and s5.

Rule 7. When the pattern is a list of names separated by comma like <major histocompatibility complex, class II, DQ beta 1> then it can be represented as <s1, s2, s3, ..., sn>. This will generate new queries s1 that corresponds to "major histocompatibility complex", s1 + s2 where s2 corresponds to "class II", s1 + s3 where s3 corresponds to "DQ beta 1". In general, the new queries are s1 + sj, where j = 2, ", n. Therefore, for the gene name shown above the newly generated queries will be "major histocompatibility complex", "major histocompatibility complex class II" and "major histocompatibility complex DQ beta 1".

Rule 8. A gene name like <Janus kinase 2 (a protein tyrosine kinase)> can be represented as the pattern <s1 (s2)>. s1 corresponds to "Janus kinase 2", and s2 corresponds to "a protein tyrosine kinase". Therefore, the newly generated queries are "Janus kinase 2" and "a protein tyrosine kinase".

Rule 9. A gene name like <metallothionein 3 (growth inhibitory factor (neurotrophic))> can be represented as the pattern <s1 (s2 (s3))>. s1 corresponds to "metallothionein 3", s2 corresponds to "growth inhibitory factor" and s3 corresponds to "neurotrophic". Therefore, the newly generated queries are "metallothieonein 3", "growth inhibitory factor", and "neurotrophic".

Rule 10. If there is a literal (L) and a number (N) or a number (N) and a literal (L) after any string then they are concatenated together. For instance, if the gene name is <Alpha 1A> it can be represented as the pattern <s1 NL> where s1 corresponds to "Alpha", N corresponds to "1" and L corresponds to "A" then the new pattern is <s1NL>. That means that the newly generated query is "Alpha1A". Similarly, the gene name <Alpha A1> can be represented as the pattern <s1 LN>. Therefore, the newly generated query is "AlphaA1".

2.1.2 Applying Rules

The procedure for applying these rules is summarized as follows. Start by reading the first gene name that is the OFFICIAL_GENE_NAME from the processed topic file. The gene name is a term composed of different subterms. For term T, let t1, t2, t3 be different subterms. Thus term T is decomposed using the following rules:

a. If the pattern is t1(t2,t3,t4,..) then replace it with the pattern t1/t2/t3/t4.
b. If the pattern is t,t1,t2,t3 then replace it with the pattern t, t1/t2/t3 .

Each pattern [t1/t2/t3] generated is a term (denoted by E). Thus,
a. t1/t2/t3 is replaced with E1.
b. t, t1/t2/t3 which is replaced with E1, E2.

Let the term T be decomposed as E1 [,] E2 E3. The expanded terms generated are as follows.
a. T1, where T1 $\in$ E1 and there is a comma after E1.
b. T1 T2 T3 ..., where T1 $\in$ E1, T2 $\in$ E2 and T3 $\in$ E3.

Lastly, concatenate all numerals, single letters, any sequence of one numeral and one letter.

2.1.3 Search

We used the extended Boolean model for our retrieval system with partial matching and term weighting. The original search procedure developed works as the combination of three tests defined as follows.

Test1: Check whether there is a match between the name of the organism in the topic of interest and the contents of <DescriptorName> tag in the document.
Test2: Check whether there is a match between the gene name or any of its symbol or alias symbol in the topic and the contents of the <ArticleTitle> tag and/or <AbstractText> tag in the document.
Test3: Check whether there is a match between the gene name or any of its symbol or alias symbol in the topic and the contents of the <NameOfSubstance> tag.

If (Test1 AND Test2 AND Test3) is true for a document given a topic then that document is considered relevant to that specific topic. We used partial matching when the search did not retrieve at least three relevant documents for a given topic. In this case a document is considered relevant if (Test1 AND (Test2 OR Test 3)) is true.

The algorithm for the different phases of the search is given below.

```
Until all queries are processed
1. Read a query (topic) Q.
2. For each document D in the collection.
    Do
    2.1 Search for the organism name of Q in Descriptor_Name tag of D.
        If there is a match, for each term T in Q except the first one,
          2.2 Search for T in the ArticleTitle and AbstractText of D.
              If there is a match, for each term T in Q except the first one,
                2.3 Search for T in the Name of Substance.
                    If there is an match then D is relevant to Q.
```

Other factors considered in the retrieval module are the following.

1. Multi-word spanning: This feature allows to search for a string in the query as two strings in the document and vice versa. For instance, *histocompatibilty* can be searched as *histo compatibility* and vice versa.

2. 80% similarity rule: Using this feature two strings can be said to be similar if they match 80%. The 80% is calculated taking the length of the strings. For instance, *METALLOTHIONEINIII* and *METALLOTHIONEIN3* are similar using this rule.

3. Stemming: Only rules for plurals were implemented.

4. Expansion of abbreviations: If a gene name has its abbreviation in the topic then it is expanded using the different combinations of the gene name and the abbreviation of the gene name. For instance, the gene name <*luteinizing hormone receptor*> and its abbreviation <*LHR*> was used to expand the queries as <*Luteinizing HR*>, <*Leuteinizing Hormone R*>, <*LH Receptor*> and <*L Hormone Receptor*>.

2.1.4 Scoring

After a document is determined to be relevant, the scoring is done based on frequency and on weights. We first score a document based on frequency. We assign one point each time any of the gene name, its symbol, alias symbol, product or alias product appears in the document. Next, we score a document by assigning some constant weights based on its performance. The different constant weights used are Title_weight, Partial_weight, Exact_weight and Complete_weight. Title_weight is given to a document, which has either the gene name or its symbol or alias symbol, or the product or alias product in the title, i.e., within the ArticleTitle tag. Partial_weight is given to a document where some sub-strings of the gene name, its symbol, alias symbol, product or alias product is fully matched and the remaining is partially matched. Exact_weight is used for the documents that contain the gene name, its symbol, alias symbol, product or alias product exactly the way they are in the NameOfSubstance. Complete_weight was assigned to the documents for which the outcome of the three tests mentioned earlier was successful for determining relevancy. This weight was assigned to maintain some distinction between the documents retrieved by the original retrieval and the documents that were retrieved using partial- match. Except for the Title_weight no other constant weights are added to the documents retrieved with just partial-match. After scoring all the documents, the top 100 documents are selected as the relevant documents for the result.

2.2 Experiments

We carried out six experiments in which we modified the search procedure to improve recall and precision. The first five experiments focused on improving recall and the last experiment, which itself consisted of several tests (experiments), focused on improving precision.

2.2.1 Experiments for recall

The first experiment used only the three tests of relevancy. For the second experiment we added multi-word spanning and the 80% similarity rule. For the third experiment we added stemming. Next we implemented expansion of abbreviations. Lastly, we implemented partial matching. We observed a monotonous increment of recall in every new experiment which is shown in Figure 2.

2.2.2 Experiments for precision

To improve precision it was understood that we needed to rank the relevant documents higher than the non-relevant ones. We assigned different types of weights as discussed in section 3. We carried out experiments with different sets of values to determine the set that generated the best average precision for the training data. A comparison of the different sets of values is presented in Table 1.

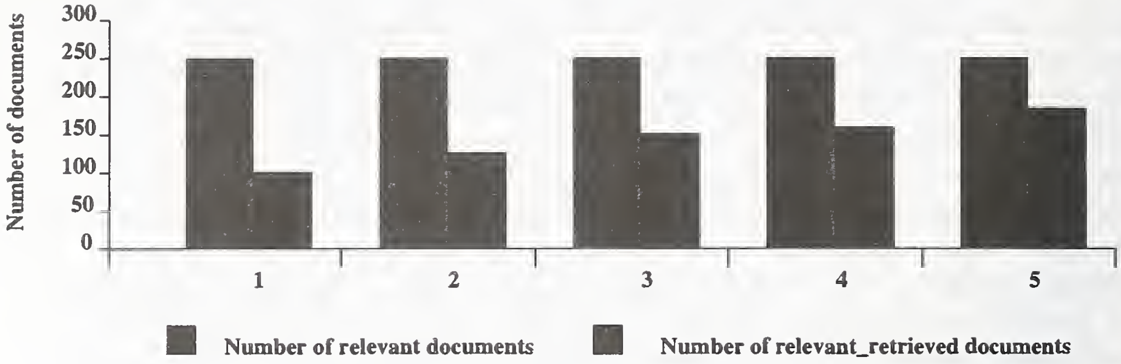


Figure 2. Experiments for improving recall

Value Set	Title Weight	Partial Weight	Exact Weight	Complete Weight	Average Precision
1	3	7	7	3	0.47
2	7	5	10	7	0.452
3	3	5	5	3	0.456
4	3	4	5	2	0.394
5	3	5	7	5	0.393

Table 1: Comparing average precision for different scores.

3. Extraction Module

The main purpose of participating in the secondary task was to carry out preliminary studies and experiments with the resources we have available at our site. Currently, the extraction module has two components, namely tokenization and pattern-matching.

The first step before extracting and generating GENERIF candidates was to download the abstracts from MEDLINE. Next we created XML-like tags for each abstract to make the tokenization more efficient. The tags we added are the following: 1) a number tag <N< and </N> that is a sequential number from 1 to 139; 2) a title tag <TI> and </TI>; 3) an author tag <AU> and </AU>; 4) an address tag <AD> and </AD>; 4) a text tag <TX> and </TX> for the abstract itself; and 5) an identifier tag <ID> and </ID> for the PMID. An example is shown below.

```
<N> 1 </N><TI>Regulation of Fas-associated death domain interactions
by the death effector domain identified by a modified reverse ... </TI>
<AU> Thomas LR, Stillman DJ, Thorburn A.</AU><AD>Department of Cancer
Biology, ...</AD><TX>The adapter protein FADD consists of two protein
... </TX> <ID> PMID: 12107169 [PubMed - indexed for MEDLINE] </ID>
```

Next, we analyzed the abstracts in terms of the GENERIFs given, looking for syntactic and semantic cues that could be used as patterns to extract information for generating GENERIF candidates. The GENERIF given was compared with the title of the abstract to find similarities and differences. In many instances, there was an exact match with the title of the abstract. Thus, the title became a default GENERIF candidate. If the match was not an exact match differences were checked. We observed that some of the differences occurred with verbs. For instance, a verb in the GENERIF given was nominalized in the title or abstract text and viceversa; a verb appeared in passive form in the GENERIF given whereas it was in active form in the title or abstract text and viceversa.

Then we examined the abstract text and we found verbs, nouns and adverbs potentially useful for our task. Among the verbs are "demonstrate", "activate", "promote", "induce", "show", "suggest", "provide", "indicate", "identify", "regulate", "conclude", "reveal", and "mediate". The nouns that we determined were more relevant for extraction purposes were "data", "results" and "study" when they appeared with verbs such as "demonstrate", "show", "indicate" and "suggest". Adverbs such as "therefore", "together", and "thus" appeared to be an important link between the GENERIF given and the abstract text. In other words, these adverbs are relevant to those who generated the GENERIFs given. The result of this analysis was a set of patterns that we used for generating GENERIFs. We used a small subset of patterns in the actual implementation. The patterns in this subset are "therefore", "indicate" and "together". The latter when it occurs at the beginning of a sentence. The beginning of a sentence is marked by a "." followed by a string whose first character is an uppercase letter. A sentence is the set of strings between two ".".

3.1 Implementation

The extraction module has two main functions a tokenizer and a pattern search. The tokenizer scans the abstract to identify boundaries of words and tags. We are interested in identifying tag boundaries first. Once a tag is identified, it is further processed to determine whether it is a title or the abstract text. The other tags are ignored except for the tag that identifies the sequential number which is used in the output. A default GENERIF is generated directly from the title. The tokenizer then scans the abstract text to identify boundaries of words.

The pattern search compares each of the words with the subset of patterns. If a match is found a GENERIF is generated from the rest of the sentence whenever the pattern is an adverb. The GENERIF will include the pattern whenever the pattern is a verb. If there is no match we use the default GENERIF as the GENERIF candidate. We did not use probabilities nor weights to rank candidates because the subset of patterns is small. In most cases the default GENERIF was kept.

3.2 Evaluation

Results of the evaluation show that our preliminary approach did well or poorly. In average the figures for the Classic Dice, Modified unigram Dice, Modified bigram Dice and Modified bigram Dice phrases are the same or slightly above the median. We conclude that the number of GENERIFs generated by the patterns was relatively small which is likely due to the fact that the number of patterns used was small. Therefore, we need to carry more experiments with a larger set of patterns, among other tasks, to determine whether syntactic cues are useful in information extraction of genomics data.

4. Conclusion

Preliminary results and evaluation have shown that our system has the potential to become an efficient and accurate system for retrieval and extraction of genomics data. We are carrying out research to implement a different information retrieval model and exploring natural language processing approaches for the information extraction task.

5. References

1. D. Grossman and O. Frieder. Information Retrieval: Algorithms and Heuristics, Kluwer Academic Press, 1998.
2. C. J. van RIJSBERGEN. Information Retrieval, London ; Boston: Butterworths, 1979.
3. Ricardo Baeza - Yates and Berthier Ribeiro-Neto, Modern Information Retrieval , Addison-Wesley, 1999.
4. Gerard Salton and Edward A. Fox, and Harry Wu. Extended Boolean information retrieval. Communication of the ACM, 26(11): 1022-1036, November 1983.
5. William Hersh, Information Retrieval: A Health and Biomedical Perspective. Springer Verlag. 2003.

TREC12 HARD Track at ISCAS

Zeng Wu, Lin Du, Le Sun, Shiwei Ye*
Institute of Software, Chinese Academy of Science
Wuzeng01@iscas.cn; ldu@sonata.iscas.ac.cn
sunle@iscas.cn
Graduate School of Chinese Academy of Science.
Shwye@gscas.ac.cn

Abstract

Statistical model in retrieval has been shown to perform well empirically. Extended Boolean model has been widely used in business system for its easiness to be complemented and not bad results. In this paper, a statistical model and modified Boolean model and natural language processing techniques, shallow query understanding techniques are used and results show that even with very limited training corpus, an appropriate statistical model can greatly improve the performance. .

Keywords: TREC12, HARD, Boolean, statistical model, gradient decrease

1. Introduction

The HARD, which means high accuracy retrieval from documents, is a new track in TREC12. The goal of HARD is to achieve high accuracy retrieval from documents by leveraging additional information about researcher and/or the search context, through techniques such as passage retrieval, and using very targeted interaction with the searcher.

The key point of this track is to choose the most relevant text "granularity", the difficulty in choosing which may lead to that Ad-Hoc Track dwindled in the few last years, according to the purpose and genre metadata in the topic. The granularity may be a total document, a passage, or even a sentence. It is different from traditional full text retrieval. If one part, maybe a passage or several sentences, in a document best suit the topic, but the total score of this document is not high, from the point of HARD, this document should be the best document in the list. For every document in the collection, calculating the relevance between a topic and all the granularity parts individually is ideally. But it will cost so much computation. So the process is divided into two steps as Q/A system does.

The rest of the paper proceeds as follows. Section 2 outlines some background and related work. Section 3 introduces how to generate query words automatically. Section 4 explains the baseline run. Section 5 presents the final run. The evaluation is concluded in section 6. Section 7 is the conclusion and future work part.

2. Background and Related Work

The problem of HARD retrieval system design can be thought of as a problem in the combination of the following steps. The first step is to get key words from the topic, the second is to get relevant by using retrieval model and then to rank them, the last is to locate topic information in the documents. This system is built totally by our site.

The purpose of previous TREC Ad Hoc track is to increase individual topic effectiveness. To generate query more accurately, some ideas can be learned from that community. In William. S. Cooper [2], they come up with such model like the following:

$$\log O(R \mid Q, D) \approx c_0 + \sum_{i=1}^{6} c_i X_i$$

They get coefficient by fitting the equation to the empirical data by means of a logistic regression analysis. (Hosmer & Lemesshow 1989) The statistical clues, $X_{i,}$ are all based on the conventional frequency counts in query and document instead of using thesauri, parsing, phrase discovery, disambiguation, and other natural language processing or AI-like approaches. To the contrary, they felt it is a virtue of regression procedures explored here that they are more hospitable than most to the incorporation additional clues. However, an astute use of simple stem and document frequency information lift s one to a high plateau of effectiveness.

As mentioned above, search with information about searcher and search context can lift the results. So, the relevance feedback is also studied in this experiment. Relevance feedback, despite its long history in information retrieval research, has not been successfully adopted. The closest feature found in some search systems is “find more documents like this”. Query expansion techniques have been used in a number of systems to suggest additional search terms, with limited success. There are many reasons for the apparent failure of relevance feedback. The primary one is the difficulty of getting users to provide the relevance information. Simply providing “relevant” and “not relevant” button in the interface does not seem to provide enough incentive for user. For this reason, researchers are investigating techniques to infer relevance through passive measures such as time spent browsing page or number of links followed from the a page. Another reason is that identifying the correct context is not simple. Experiments [13] have shown that if a user can indicate relevant sections or even phrases in a document, relevance is more accurate.

To resolve these problems, the sophisticated interface design and good algorithm for inferring context are required.

In the second step, there is a lot of literature on approaches to information retrieval, we will not survey them all here. The focus, here, is on the modification of extended Boolean model and the statistical model. The modified Boolean model is used in baseline run.

The standard Boolean retrieval has following limitations

1) It gives counterintuitive results for certain types of queries.
2) It has no provision for ranking documents.
3) During the indexing process, it is necessary to decide whether a particular document is either relevant or non relevant with respect to a given index term.
4) It has no provision for assigning importance factors or weights to query terms.

The P-norm model is proposed as alternative to the Boolean model. P-norm model has the ability to consider weighted query terms and provides a ranking of retrieved documents in order of decreasing relevance.

However, the statistical model look information retrieval as a problem in the combination of statistical clues. The design objective is to achieve as high a level of retrieval effectiveness as possible, consistent with reasonable theoretical and computational simplicity. In the final run, a statistical model is constructed. Then the compare between two models is evaluated.

The TREC Q/A track is designed to take a step closer to information retrieval rather than document retrieval. The researchers in this field mostly use statistical method. Abraham Ittycheriah applied Machine Translation ideas to the Q/A [3]. Because they have sufficient rules and weights, the answers are created from learning their known question and answer pairs in the open domain.

In getting the answer to a query, researchers usually use tagging or parsing tools to tag the query, then get the critical information including answer concepts, which are identified by categorizing queries using a method similar in spirit to extracting the named entities [8], the named focuses [9], and question-answer tokens [10]. Then the same procedure put on the Documents, and it will cost so much time. This process is always off line. Lastly, using different matching methods to generate the answer.

In retrieval, a query using the phrase such as "white house" is much more likely to be satisfied by a document using those two words in sequence than by one that has them separately. For some words may have distinctive meaning in the context of another word or in a larger phrase. This approach, however, requires that all sentences, whether in documents or in queries be segmented into phrases. This depends on the identity of the previous word generated. David R. H. Miller [5] bring forwarded three states Hidden Markov Model to identify two words phrase..

All the models mentioned above are built based on the statistical foundations which mean overwhelming majority of documents paired with relevant queries are available. In practices, it is usually difficult to come by.

As mentioned above, the HARD track has some familiarity with Ad Hoc track, Q/A track, and Interactive track. Some useful ideas and techniiques can be learned from these tracks based on both HARD requirements and the resources available in hand.

3. Automatic Construct Key Terms

To take part in the HARD track, the system is built completely by us on the RedHat Linux platform. To make search on disk file more conveniently Berkeley DB-4.2 is introduced in the system.

In every topic, the sentence is generated from the <title> field, <descr> field and <narr> field. Then use Brill Tagger tool to tag it. In HARD topics, most sentences from <narr> field have such word like "on topic", "off topic". If no such phrases in the sentences, it will have negative words like "neither, nor, no". In this system, the sentences having such words negative sentences is called negative sentences and words extracted from these sentences are called negative words. Others are called positive sentences and words from these sentences are called positive words. After tagging these sentences, the words not tagged as "NNP ", "NN","NNS", "VBD","VBN" are abandoned, except the last word in sentences. For in examining the sample tagged files, the last word, a noun word from human, always is tagged as CD. Some words tagged as ADJ may in fact have some meanings, but limited to our resources, they can not be identified and be thrown off.

Now a word list named positive and a negative word list is constructed. In either list, every word is not a stop word and has been stemmed. It has a remark telling it from title field or <narr> field or <descry> field. It is clear that word from title field has more importance in retrieval. As for the same words in the same list or in the two lists, we also include it as if they were different words. The relevance between query q and document d can be calculated according to the following equation

$$w = positive\_c * \sum w_{+} - negative\_c * \sum w_{-} + length\_c * doc\_length + location\_c * \frac{doclength}{\sum word\_location} +$$

$$qlentgth\_c * query\_length$$

1) w₊ is the weight of word in positive list; w₋ is the weight of word in the negative list.

$$w_{+} = (0.5 + \frac{0.5 * freq_q}{doclength}) * \log_2(\frac{N-n}{n}) \qquad (*)$$

$$w_{-} = (0.5 + \frac{0.5 * freq_q}{doclength}) * \log_2(\frac{N-n}{n})$$

$freq_q$ is the count of the word occurring in the document. doclength is count of all of the words in document d, n is count of this word occurring in all of the collection , N is the count of all of the document in the collection.
To get this equation, the equation is modified according to the document [2]. W_{-} or w_{+} is the weight of w_i in negative list or in the positive list

2) word_location is the offset where the word occur in the document.

3) positive_c and negative_c, length_c, location_c, qlength_c are our statistical model parameters. But for the limited training corpus by hand, which is only the training topics provided by HARD, and the importance of these five parameters, the last three parameters are omitted.

4 Metadata and Clarification Form

4.1 CF

The clarification forms contains the following fields. The first field is composed of the title of the topic. The second field is composed of the words extracted from the sentences in the <descr> fields and on-topic words of the <narr> field. The third field is a list of negative words, which are extracted from off-topic section of the <narr> fields. If there is no negative word, this field is empty.

4.2 Metadata

For the tag RELATED-TEXT of the metadata, the relevant words extracted from the documents are added to the queries. If GENRE equals to ADMINSTRATIVE, the documents in HARDGOV corpus is returned. If is I-REACTION, the documents in the HARDGOV is not retrieved. For the metadata PURPOSE, if it is ANSWER, the simple method of Q/A is used. .The following section is to do with the circumstance that the GRANULARITY is passage.

All other tags are not processed.

5.Baseline run

In our baseline run, assign positive_c = negative_c = 1;

For a certain topic, we rank the document according to the followings:

1) Calculate the weight according equation (1)
2) Ignore the document whose weight less than 0;
3) Rank the document according weight got in 1)
4) If the count of ranked documents is more than 1000, choose the first 1000 documents
5) For each document from ranked highest to ranked lowest, get the raw document content. For every word in positive list, the first location where the word occurred is the offset value shown in results file. The length is document length minus offset.

6.Final run

5.1 train positive_c, negative_c .

There is a statistical relation between the topic-document relevance and total positive word weights, negative word weights, words location, the context of words in relevant document, which can be used when the metadata granularity is passage. Because having involved the document length in get words weight, the normalization is not considered in this step. But there is only document and topic No in training relevance document and no other resources are available, so the model is simplified to two parameters as mention above.

In training relevance document, the document is remarked as 1 or 0.5 or 0, which display the document is hard-relevant, soft-relevant and non-relevant. So document remarked as 1 is more relevant than remarked as 0.5 and 0. It is the same with the document remarked with 0.5 and 0. Then to get such expressions like the following

For every certain topic.

$$w_i > w_j \forall i, j$$

When document i is remarked as 1, as HARD marked, document j is remarked 0.5 or 0.

When document i is remarked as 0.5, as HARD marked, document j is remarked 0.

We can simply write W in equation 1 as

$$w_i = positive\_c * w_{+i} - negative\_c * w_-$$

Because positive_c and negative_c is constant so the constant term is integrated in the two sides of equation, then

$$positive\_c * w_{+} - negative\_c * w_{-} > 0 \, (2)$$

For one topic a list of such equations is got, and for total topics, it consist of a complete list of equations .Now a appropriate value for positive_c and negative_c is set to make expression (2) true in training corpus. We use Gradient Decrease Algorithm, commonly used in numerical calculation to get them.

5.2 locate the information

For all the words in the positive list and the negative list, the place of the first sentence which obtain the positive word is the offset value required in result file. The offset is the file length minus the offset.

5.3 work done especially for the request of some metadata

If GENRE equals to ADMINSTRATIVE, the documents in HARDGOV corpus is returned. If is I-REACTION, the documents in the HARDGOV is not retrieved.

For the tag RELATED-TEXT of the metadata, the words only already in the word list is extracted from the documents are added to the queries ignoring the fact that it has been in the list.

If the tag GRANULARITY is passage, the retrieval processing is composed of two stages: document retrieval and passage-level ranking. The document retrieval first gets all the relevant documents. Initially, the summary of a document is zero. From top passage to end one, if it contains a word in the positive list, the sum is added with one. If a passage contains a word in the negative list, the score is decreased with one. In the end, the sum of every passage is acquired. Then the maximum of them divide by the doc length is the new score of the doc. Then the doc list is ranked according to the new score.

7.Evaluation and Results

The Hard-rel judgment means that the document is relevant *and* it satisfies the appropriate metadata. The Soft-rel judgment means that document is relevant to the topic but that it does not satisfy the appropriate metadata. It either does not satisfy the PURPOSE, GENRE, or the FAMILIARITY items (the others are not document-level items).

In constructing the model, we do not count in the idf value in the topic, as the formula * show. For we think our model is a modified Boolean model and the word is noun in the sentences, which has a substantial meaning. The more they occur, the more important they are. And we have a desire to see what is happening without obeying classical theory. But from the tables, this thought does not accord with the fact.

There is a relation between the first score and last score in the retrieval, but we divide it subjectively.

In re-scoring the doc, there should be a similar expression with the expression 1. But results are even worse when we manually check them. The reason is that the amount of the sample points is not sufficient. For the same reason, the parameter in form 1 is not precise enough. The model cannot satisfyingly predict the future.

The training corpus in our site is nothing but the corpus provided by the HARD, so to make parameter trained precise enough, only four topics are chosen for testing the results. It is not sufficient.

Document level retrieval results.

Table 1

Hard-rel criteria	Baseline run	Final run
Average precision	0.0324	0.0715
R-Precision	0.0679	0.1210

Table 2

Soft-rel criteria	Baseline run	Final run
Average precision	0.0368	0.0858
R-Precision	0.0702	0.1406

The following is an operational definition of passage recall and precision as used in the evaluation. For each relevant passage allocate a string representing all of the character positions contained within the relevant passage (i.e., a relevant passage of length 100 has a string of length 100 allocated). Each passage in the retrieved set marks those character positions in the relevant passages that it overlaps with. A character position can be marked at most once, regardless of how many different retrieved passages contain it. (Retrieved passages may overlap, but relevant passages do not overlap.) The passage recall is then defined as the average over all relevant passages of the fraction of the passage that is marked. The passage precision is defined as the total number of marked character positions divided by the total number of characters in the retrieved set. The F score is defined in the same way as for documents, assigning equal weight to recall and precision: F = (2*prec*recall)/(prec+recall) where F is defined to be 0 if prec+recall is 0. We included the F score because set-based recall and precision average extremely poorly but F averages well. R-precision also averages well.

In all of the above, a document is treated as a (potentially long) passage. That is, for topics where the granularity is "document" the relevant passage starts at the beginning of the document and is as long as the document. (These are represented in the judgment file as passages with -1 offset and -1 length, but are treated as described above.) For any topic, a retrieved document (i.e., where offset and length are -1) is again just a passage with offset 0 and length the length of the document.

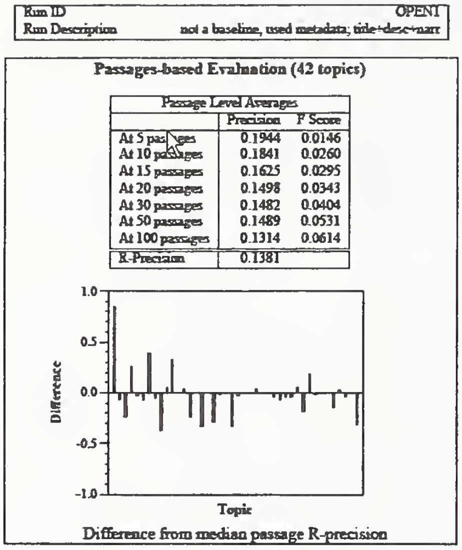
Using the above definition of passage recall, passage recall and standard document level recall are identical when both retrieved and relevant passages are whole documents. That is not true for this definition of passage precision. Passage precision will be greater when a shorter irrelevant document is retrieved as compared to when a longer irrelevant document is retrieved. This makes sense, but is different from standard document level precision.

The following table is passage level results.

Table 3

	R-precision
OPEN	0.0954
OPEN1	0.1381

From the tables, the statistical elements partly overcome the model default in the baseline run.

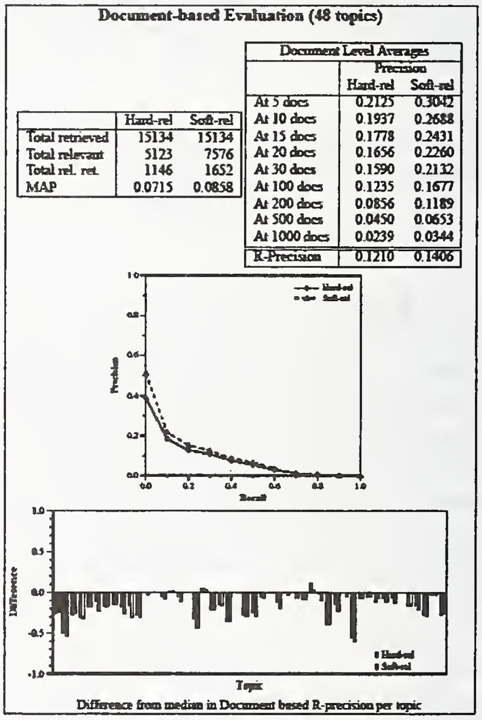


Run ID	OPEN1
Run Description	not a baseline, used metadata; title+desc+narr

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.1944	0.0146
At 10 passages	0.1841	0.0260
At 15 passages	0.1625	0.0295
At 20 passages	0.1498	0.0343
At 30 passages	0.1482	0.0404
At 50 passages	0.1489	0.0531
At 100 passages	0.1314	0.0614
R-Precision	0.1381	

Difference from median passage R-precision



Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	15134	15134
Total relevant	5123	7576
Total rel. ret.	1146	1652
MAP	0.0715	0.0858

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.2125	0.3042
At 10 docs	0.1937	0.2688
At 15 docs	0.1778	0.2431
At 20 docs	0.1656	0.2260
At 30 docs	0.1590	0.2132
At 100 docs	0.1235	0.1677
At 200 docs	0.0856	0.1189
At 500 docs	0.0450	0.0653
At 1000 docs	0.0239	0.0344
R-Precision	0.1210	0.1406

Difference from median in Document based R-precision per topic

8 Conclusion and Future work

The statistical model is effective, for using the same system, the latter results are twice better as much as the former.

The idf value is important whenever using any kind of retrieval model. It at least does not do any bad to the results.

Given more time and hands and more corpus, the equation (1) can be expanded in containing such elements as the doc length, query length, the word location occurring in the doc. And we will use Conjunctive Gradient Decrease or use MLP, using simulate anneal to relieve local minimum. From the contrast of the baseline run and final run, we are sure of performing better.

9.Acknowledge

This work is supported by China 863 Project(Grant No. 2001AA114040) and the National Science Fund of China under contact 60203007.

Reference

[1] E. Fox, S. Betrabet, M. Koushik "Extended Boolean Models" In "*Information Retrieval Data Structure & Algorithms*" pp393-418 1992

[2] William. S. Cooper, Aitao Chen, "TREC-3 working note: Experiments in the Probabilistic Retrieval of Full Text Documents"

[3] Abraham Ittycheriah, Salim Roukos, "TREC-11 working note: IBM's Statistical Question Answering System –TREC-11".

[4] Jinxi Xu, Ana Licuanan, Jonathan May, Scott Miller and Ralph Weischedel "TREC-11 working note: TREC 2002 QA at BBN: Answer Selection and Confidence Estimation"

[5] David R. H. Miller, Tim Leek, Richard M. Schwartz, "A Hidden Markov Model Information Retrieval System" In "*Annual ACM Conference on Research and Development in Information Retrieval Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval" B*erkeley, California, United States,pp214-221 1999

[6] Donna Harman "Ranking Algorithms" In "*Information Retrieval Data Structure & Algorithms*" pp363-392 1992

[7] Adam Berger, John Lafferty, "Information Retrieval as Statistical Translation" In *"Proceedings of the 22nd ACM Conference on Research and Development in Information Retrieval((SIGIR'99)",* pages 222--229, 1999

[8] Daniel Pack, Clifford Weinstein "The Use of Dynamic Segment Scoring for Language-Independent Question Answering" In *"HLT 2001 Presentations"*

[9] Adam Berger, Vibhu Mittal "Query-relevant summarization using FAQs" In *"Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL- http://citeseer.nj.nec.com/341776.html" 2000*

[11] Lynn Carlson, Daniel Marcu, Mary Ellen Okurowski. "Building a discourse-tagged corpus in the framework of rhetorical structure theory" In *Proceedings of the 2nd SIGDIAL workshop on discourse and dialogue, Eurospeech, Aalborg , Denmark*

[12] W. Bruce Croft, Stephen Cronen-Townsend, Victor Lavrenko "Relevance Feedback and Personalization: A Language Modeling Perspective" *In Proceedings of the DELOS-NSF Workshop on Personalization and Recommender Systems in Digital Libraries, pages 49--54, 2001.*

NLPR at TREC 2003: Novelty and Robust

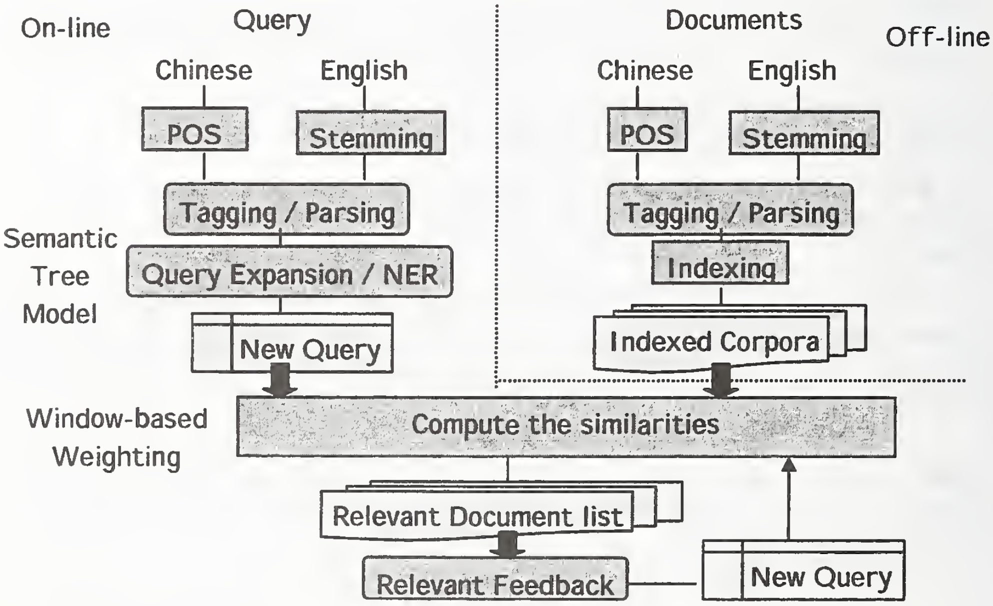
Qianli Jin, Jun Zhao, Bo Xu
National Laboratory of Pattern Recognition,
Institute of Automation, Chinese Academy of Science
Beijing, China 100080
{ qljin, jzhao, bxu }@nlpr.ia.ac.cn

It is the first time that the Chinese Information Processing group of NLPR participates in TREC. Our goal in this year is to test our IR system and get some experience about the TREC evaluation. So, we select two retrieval tasks: Novelty Track and Robust Track. We build a new IR system based on two key technologies: Window-based weighting method and Semantic Tree Model for query expansion. In this paper, the IR system and some new technologies are described first, and then some detail work and results in Novelty and Robust Track are listed.

1 IR System and new technologies

1.1 The Architecture of IR system

Our IR system is both for English Retrieval and Chinese Retrieval. Some maim parts of the system are shown in the following.



There are many modules in the system, such as POS, stemming, tagging, NER, Query Expansion and etc. Most of them are traditional and common, except that there are two new technologies: Window-based Weighting and Semantic Tree Model. In the following two parts, they are detail introduced.

1.2 Window-based Weighting

The key algorithm of an IR system is similarity computing between queries and documents. Till now, the most popular algorithm is the inner product of vectors, and the vectors can be built by using weighting technologies, such as binary weight, tf-idf, query expansion, relevant feedback and etc. In other words, most of the existing algorithms are based on vector computing. However, this method usually gets limited precision, because sometimes, a vector can not represent a query properly. For instance, if we have the following query and two documents:

Query: **"Can radio waves from radio towers or car phones affect brain cancer occurrence?"**

Document A: "John claimed his <u>brain</u> <u>cancer</u> was caused by the <u>wave</u> from his cellular <u>phone</u>. That claim, put forth in a lawsuit, has no basis in accepted scientific fact."

Document B: "I was listening to the <u>radio</u>, when the <u>tower</u> collapsed. I ran several blocks before my <u>brain</u> kicked in, and saw that another <u>wave</u> of people started running towards a police <u>car</u>."

We definitely know that Document A is relevant to the Query, while Document B is not. But if binary word vector model is used, the similarity value between Document B and the query is larger than the one between Document A and the query. We also try some other weighting technologies such as tf-idf, query expansion and etc., but find that in this case, based on vector computing, Document B seems more "relevant" to the query than Document A.

In order to solve this problem, we develop two key notions as follows:

1. *Query words appearing closely in the document provide more contributions to the similarity value than the ones appearing separately. The closer the query words in a document, the larger the similarity value between the query and the document.*

2. *Some query words, like named entities and baseNP are called "Core Words", while the other words are called "Surrounding Words". "Core Words" are much more important than "Surrounding Words", and should have special status in the retrieval processing (i.e. having larger weights).*

Based on the above two key notions, we developed three window-based models for the application of information retrieval. They are called "Simple Window-based Model", "Dynamic Window-based Model" and "Core-window-based Model", from the simplest model to the most complex one.

1.2.1 Model One: Simple Window-based Model

As our first key notion, the closer the query words in a document, the larger the similarity value between the query and the document. So, we introduce a window in the retrieval processing. When the query words co-occur in the window, a larger similarity weight is provided. First we put all the words of the document into the word sequence orderly, like the Figure 1. Each sub-sequence with *d* continual words is included in a *d*-width window.

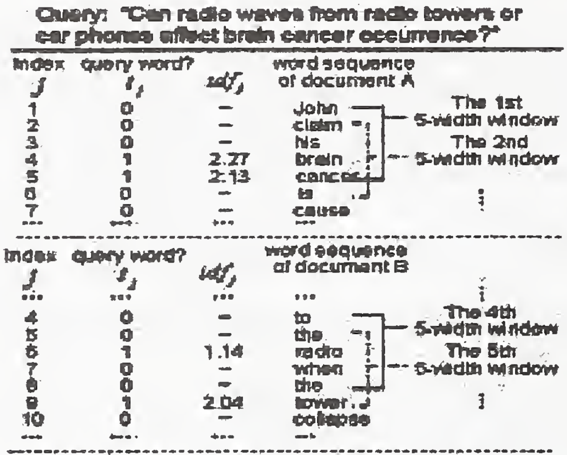


Figure 1: Example for window-based method

Let N denote the number of the words in the whole sequence, and *d* denote the width of the window. Then the similarity value ($Sim1(q,d)$) between query and document can be represented as follows:

$$Sim1(q,d) = \sum_{i=1}^{N-d} SWin(i,i+d) \qquad ......(1)$$

$$SWin(i,i+d) = [\sum_{j=i}^{i+d} t_j * idf_j] * [\sum_{j=i}^{i+d} t_j] ......(2)$$

where $SWin(i,i+d)$ denotes the similarity value between the query and the *d*-width window from the *i*th word to (*i+d*)th word in the whole sequence in the Simple Window-based Model. t_j is the binary signal of the *j*th word in the whole sequence. Here, t_j is equal to ONE if the *j*th word is a query word, otherwise, it is equal to ZERO. And idf_j is the inverse document frequency of the *j*th word in the sequence. The final similarity value between a query and a document is the sum of the similarity values of all the windows. The similarity value in a single window, represented as Formula 2, consists of two items. The first one $[\sum_{j=i}^{i+d} t_j * idf_j]$ is just like the traditional tf-idf method. And the second one $[\sum_{j=i}^{i+d} t_j]$ provides more weight, when more than one query word appeared in the corresponding window.

1.2.2 Model Two: Dynamic Window-based Model

In Simple Window-based Model, we give larger weight to the window, which includes more than one query word. But what is the distribution of these query words in the window? They can be separate or conjoint. If these query words in the window are conjoint, they maybe form a phrase. As we all know, phrases usually are less ambiguous than words. So, we should give the conjoint query words larger weight than the separate query words in the window. Another problem in Model One is that it is difficult to decide the width of window in real applications. A fixed window width cannot be suitable for all queries. In order to solve the above two problems, Model Two is proposed, which is called Dynamic Window-based Model. In the new Model, a dynamic window width called "TightWin" is developed to modify the original fixed window.

Define: TightWin is the smallest window width, which can overlay all the query words in the original window.

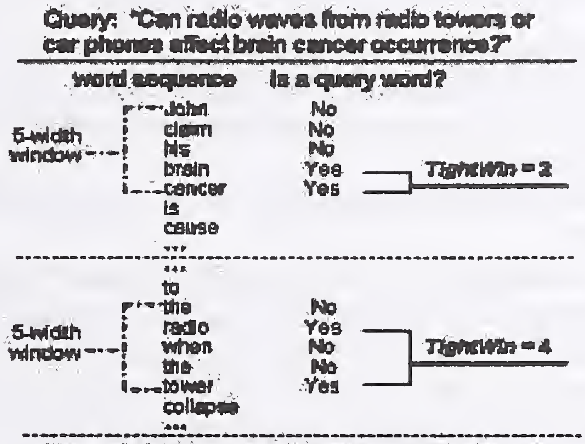


Figure 2: Example for the value of *TightWin*

In Figure 2, we give several examples about the value of TightWin. If the query words distribute separately in the original window, the value of TightWin is large. And if they are conjoint, the TightWin is small. So, we should give a large weight when TightWin is small.

Let N denote the number of the words in the whole sequence, and *d* denote the width of the window. Then the similarity values between query and document in Model Two ($Sim2(q,d)$) can be represented as follows:

$$Sim2(q,d) = \sum_{i=1}^{N-d} DWin(i, i+d) \qquad ......(3)$$

$$DWin(i,i+d) = SWin(i,i+d) * \left(\frac{[\sum_{j=i}^{i+d} t_j]}{TightWin}\right)^p = [\sum_{j=i}^{i+d} t_j * idf_j] * [\sum_{j=i}^{i+d} t_j] * \left(\frac{[\sum_{j=i}^{i+d} t_j]}{TightWin}\right)^p ......(4)$$

where $DWin(i,i+d)$ denotes the similarity value between the query and the *d*-width window from the *i*th word to (*i*+*d*)th word in the whole sequence in Dynamic Window-based Model. t_j is the binary signal of the *j*th word in the whole sequence. Here, t_j is equal to ONE if the *j*th word is a query word, otherwise, it is equal to ZERO. And idf_j is the inverse document frequency of the *j*th word in the sequence. *TightWin* is defined above, and *p* is a parameter, which is larger than zero.

Compared with Model One, Model Two has an additional item $(\frac{[\sum_{j=i}^{i+d} t_j]}{TightWin})^p$, which provides adjustment to the original fixed window. The conjoint query words provide more contributions to the final similarity value.

1.2.3 Model Three: Core Window-based Model

In the above two models, when query words appear closely in the document, they will be given larger weight. In some cases, it may bring some problems. Take a look at the above example query again.

> **"Can radio waves from radio towers or car phones affect brain cancer occurrence?"**

When the query words "radio waves" and "brain cancer" appear closely in a document, we can say that this document is most likely relevant to the query. But, when the query words "car phone" and "affect" appear closely in a document, we are not sure whether it is relevant. So, based our second key notion, we parse the query sentence and classify the query words into two groups. They are "Core Words" and "Surrounding Words" defined as follows.

Define:

(1) The query words, which represent the main meaning of the query, such as baseNP and Named Entities, are called "Core Words".

(2) The query words, which are not core words, are called "Surrounding Word".

(3) A window is called "Active Window", if and only if it includes Core Words.

Obviously, Core Words are much more important than Surrounding Word. So, Active Window should have larger weight than the common window.

Let N denote the number of the words in the whole sequence, and *d* denote the width of the window. Then the similarity value between query and document in Model Three ($Sim(q,d)$) can be represented as follows:

$$Sim3(q,d) = \sum_{i=1}^{N-d} CWin(i,i+d) \qquad ......(5)$$

$$CWin(i,i+d)=DWin(i,i+d)*[\sum_{j=i}^{i+d}t_j^*]=[\sum_{j=i}^{i+d}t_j*idf_j]*[\sum_{j=i}^{i+d}t_j]*(\frac{[\sum_{j=i}^{i+d}t_j]}{TightWin})^p*[\sum_{j=i}^{i+d}t_j^*]^m \ldots(6)$$

where $CWin(i,i+d)$ denotes the similarity value between the query and the *d*-width window from the *i*th word to (*i*+*d*)th word in the whole sequence in Core Window-based Model. t_j is the binary signal of the *j*th word in the whole sequence. Here, t_j is equal to ONE if the *j*th word is a query word, otherwise, it is equal to ZERO. t_j^* is another binary signal of the *j*th word in the whole sequence for Core Words. t_j^* is equal to ONE if the *j*th word is a Core Word, otherwise, it is equal to ZERO. And idf_j is the inverse document frequency of the *j*th word in the sequence. *TightWin* is defined in 1.2.2, and *m* is a parameter, which is larger than zero. Compared with Model Two, Model Three has an additional item$[\sum_{j=i}^{i+d}t_j^*]^m$, which focuses on the Core Words in the window. Only the active window has contributions to the final similarity value. The more the core words in the window, the larger the similarity value.

Detail evaluations of window-based weighting are included in the **Reference [1]**.

1.3 Semantic Tree Model for Query Expansion

The key problem of query expansion is to compute the similarities between terms and the original query. In other words, the original query can be regard as a point in the semantic space, and the goal of query expansion is to select some additional terms, which have the closest meaning to the point. So, as the first step, like most of the former methods, we need to compute the prior similarities between the terms. And we use Term Similarity Trees to represent and estimate the similarities between terms, which can cluster the terms according to their meaning. Then, we use the TSTM to expand queries.

1.3.1 Grow Term Similarity Trees based on prior similarities between terms

1.3.1.1 Build elements of Term Similarity Tree

Let $PSim\ (q,p)$ denote the prior similarity between the term *q* and *p*. For a given term *q*, use a tree to represent the sorted similarities in descending order between *q* and all the other words, like the left part of the following Figure.

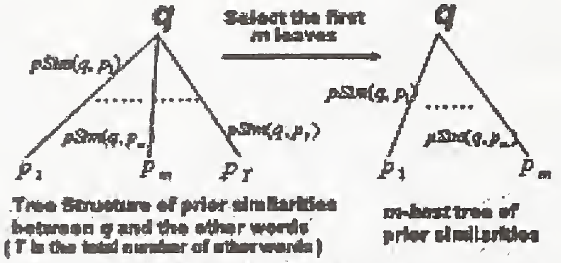

q
Select the first m leaves
q
pSim(q, p1)
pSim(q, pm)
pSim(q, pT)
p1
pm
pT
pSim(q, p1)
pSim(q, pm)
p1
pm
Tree Structure of prior similarities between q and the other words
(T is the total number of other words)
m-best tree of prior similarities

Figure 1: Build *m-best* tree of prior similarities

Note that the weight of the branch between q and p_m is $PSim(q, p_m)$, and:

$$PSim\ (q, p_1) \geq \ldots \geq PSim\ (q, p_m) \geq \ldots .$$

Then, we keep the first *m* leaves and discard the rest to build *m-best* tree of prior similarities, like the right part of the Figure. There are several methods for computing the prior similarities between terms. Here, we use normalized local co-occurrence algorithm to estimate them.

1.3.1.2 Grow Term Similarity Tree

Let $Q = (q_1, q_2, \ldots, q_i, \ldots, q_K)$ denote the original query including *K* terms, where q_i is the *i*th term in query. Then, we grow the Term Similarity Tree of query *Q* as $TSTM\ (Q, v, m)$, where *v* denotes the expanded level and *m* indicates that each element in term similarity tree is an *m-best* tree of prior similarities. The term similarity tree of query *Q* is shown in the following Figure.

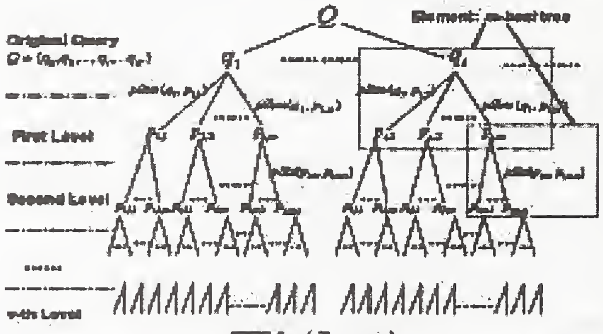


Figure 2: Term Similarity Tree of Original Query $Q=(q_1,q_2,\ldots q_i \ldots q_K)$ with *v* levels and *m-best* tree per element

Each part framed by a quadrangle in the Figure denotes an element of TSTM, which is an *m-best* tree of prior similarities. Using the multi-level term similarity tree, we can easily compute the semantic similarity between two terms (one query term and one other term), no matter whether they co-occur in the training corpora. Note that each term in the query, like q_i, has its own sub-tree, whose root is the query-term itself. We define:

(a) A path between the query-term q_i and its leave-term p is the route from the root node q_i to the leave node p.
(b) The weight of a path between the query-term q_i and its leave-term p is the product of all the branch weights (prior similarity) on the path from q_i to p.
(c) The shortest path between the query-term q_i and its leave-term p is the path between q_i and p, which has the largest weight.
(d) The similarity between the query-term q_i and its leave-term p is defined as the weight of their shortest path.

For instance, we can compute similarity between two terms q_1 and $p_{1,i,j}$ as:

$$Sim\ (q_1, p_{1,i,j}) = weight - of - shortest - path\ (q_1, p_{1,i,j})$$
$$= PSim\ (q_1, p_{1,i}) \times PSim\ (p_{1,i}, p_{1,i,j}) \ldots\ldots \quad (7)$$

1.3.2 Query Expansion based on TSTM

Let $Q = (q_1, q_2, \ldots, q_i, \ldots, q_K)$ denote the original query including K terms, where q_i is the ith term in query. The $TSTM\ (Q, v, m)$ is the term similarity tree of query Q.

The term w is expanded to query Q, when w satisfies two conditions illustrated as the following two formulae:

$$\begin{cases} Sim\ (Q, w) = \sum_{i=1}^{K} Sim\ (q_i, w) \geq cv \\ Overlay\ (TSTM\ (Q, v, m), w) \geq percent\ \times K \end{cases} \ldots\ldots \quad (8)$$

where $Sim(Q, w)$ is the similarity between the query Q and the term w. $Sim(q_i, w)$ is the similarity between the term q_i and w, which can be estimated as formula (7). And cv is the threshold value of similarity.

$Overlay(TSTM(Q, v, m), w)$ denotes the occurrence times of term w in the sub-trees of $TSTM(Q, v, m)$. It means that in the total K sub-trees of Q's term similarity tree, how many sub-trees include (overlay) the term w. $percent$ is the threshold value of overlay degree.

The first discrimination function in formula (8) is used to estimate the similarities between the term w and the query terms $q_1, q_2, \ldots, q_i, \ldots, q_K$ respectively. And the second discrimination function in formula (2) is used to estimate the similarity between the term w and the whole query Q. The query Q can be regard as a point in the semantic space, so we need to know whether the term w is close in meaning to this point, not just close to the independent meaning of each q_i.

Detail evaluations of Semantic Tree Model for query expansion are included in the **Reference [2]**.

2 Novelty Track

The Novelty Track is designed to investigate systems' abilities to locate relevant and new information within a set of documents relevant to a TREC topic. The goal of the track is to find out relevant/new sentences, instead of documents.

2.1 Relevant

Considering sentences have few words than documents, query expansion is much more important. So, we use the following process to deal with relevant retrieval:

A) Two Stages Query Expansion. In the first stage, we use Term Similarity Model to expand queries. In the second stage, we use Relevant Feedback to modify and expand queries again to improve the retrieval result. Usually, top 20% sentences are used for relevant feedback, after the first retrieval.

B) We use two different methods to compute similarities between queries and sentences. The first is the traditional tf-idf method. And the second is window-based method to ensure that the closer the query words in sentences, the higher the similarity value. Actually, this method is the expansion of N-gram model (because window-based method does not require the query words appearing directly continual).

C) We use an existing method called 'pivoted document length normalization' to normalize sentence length. (see the **Reference [3]**)

D) The similarity values of different topics are usually different. The main reason is that queries have different length and query words have different characteristics (i.e. idf). So, it's unreasonable to use a simple and fixed threshold for every topic. Here, we developed one dynamic threshold for one topic, based on the probabilistic characteristics of similarity values between this topic and sentences.

2.2 Novelty (new)

Having relevant sentences, we have another task to filter out repeated information. We define a value called 'New Information Degree'(NID) to present whether a sentence includes new information related to the former sentences. If the value of NID is big, this sentence is reserved, or it will be discarded. There are two different ways to define NID of the latter sentence related to the former sentence.

$$\text{NID\_1} = 1 - \frac{\text{Sum the 'idf' value of words appeared in both sentences}}{\text{Sum the 'idf' value of words appeared in latter sentences}}$$

$$\text{NID\_2} = 1 - \frac{\text{The number of matched bi-gram word sequences}}{\text{Total number of bi-gram word sequences in latter sentences}}$$

Usually, if the value of NID is bigger than 10%-20%, the latter sentence will be considered useful (including new information).

2.3 Official Results

Dyn: Dynamic Threshold **Sta:** Static Threshold
Win: Core Window-based weighting method **RF:** Relevant Feedback
Leng: Length Normalization **Tf-idf:** tf-idf weighting
NID_1 / NID_2: defined above in 2.2 **QE:** Query Expansion

Task 1 Table

ID TAG	Algorithms	Relevant Results Average F Measure	Novelty Results Average F Measure
NLPR03n1w1	Dyn-Win-RF	0.510	0.425
NLPR03n1f1	Tf-idf-leng-RF	0.477	0.399
NLPR03n1f2	Tf-idf-leng-RF	0.407	0.349
NLPR03n1w2	Dyn-Win-RF	0.391	0.325
NLPR03n1w3	Dyn-Win-RF	0.330	0.279

Task 2 Table

ID TAG	Algorithms	Average F Measure
NLPR03n2d1	NID_1, Dyn	0.807
NLPR03n2s1	NID_1, Sta	0.819
NLPR03n2d2	NID_2, Dyn	0.808
NLPR03n2s2	NID_2, Sta	0.817
NLPR03n2d3	NID_1+2, Dyn	0.803

Task 3 Table

ID TAG	Algorithms	**Relevant** Average F	**Novelty** Average F
NLPR03n3d1	RF, Win, leng, NID_2, Dyn	0.687	0.518
NLPR03n3s1	RF,Win, leng, NID_1, Sta	0.677	0.532

NLPR03n3d3	RF,Win, leng, NID_1, Dyn	0.674	0.509
NLPR03n3d2	QE,RF,tf-idf, leng, NID_2,Dyn	0.618	0.472
NLPR03n3s2	QE,RF, tf-idf, leng, NID_1,Sta	0.624	0.489

Task 4 Table

ID TAG	Algorithms	Average F Measure
NLPR03n4d1	NID_1,Dyn	0.775
NLPR03n4s1	NID_1, Sta	0.789
NLPR03n4s2	NID_1+2, Sta	0.794
NLPR03n4s3	NID_2, Sta	0.796
NLPR03n4d2	NID_2. Dyn	0.773

Form the above results, we find that the technologies of window-based weighting method and relevant feedback are useful. In task 2,3,4, we all get very big values of average F measure, but in task 1, it is small. The reason is that in task 1, we use TREC Novelty 2002 data as our training corpora, which have quite few relevant sentences. However, the data for 2003 have many relevant sentences (even more than 50% for some topics), so we should use small thresholds. The big thresholds from the training corpora (2002 data) make good precision and poor recall (also poor F measure). In task 2-4, we use a part of the 2003 data as the training corpora and get better F values.

3 Robust Track

As a mature IR system, the robustness is quite important. The goal of the Robust Track is to improve the consistency of retrieval technology by focusing on poorly performing topics. So, in this track, we improve our system based on the following two points:

1) Considering that even the worst topic should have an acceptable result, a robust algorithm of similarity should be produced to improve precision for each topic.
2) Though at most 1000 documents will be accepted by NIST per topic, we suppose that most users only look through the former part of retrieval result (Maybe 2 or 3 screens). So, for each run, we submit several dozens of retrieved documents. We should make sure that the true relevant documents have the top similarity values.

3.1 Processing

In order to improve the robustness described above, we use a combined algorithm, including four technologies. The processing of our system is as follows:

A) Query expansion based on Term Similarity Trees. It is described in the first part of this paper and also in the Reference [1].
B) Window-based weighting methods for computing similarities. It is described in the first part of this paper and also in the Reference [2].
C) Length normalization (see the Reference [3])

D) Dynamic Threshold. The similarity values of different topics are usually different. The main reason is that queries have different length and query words have different characteristics (i.e. df). So, it's unreasonable to use a simple and fixed threshold for every topic. Here, we developed one dynamic threshold for one topic, based on the probabilistic characteristics of similarity values between this topic and documents.

3.2 Official Results

ID Tag	Algorithms	Retrieved documents	Average Precision (non-interpolated)	Number of topics with no relevant in top 10
NLPR03vb25	Dynamic Window-based weighting	25	0.1516	7%
NLPR03vb10	Simple Window-based weighting	10	0.1055	7%
NLPR03w16	Core Window-based weighting	16	0.1153	10%
NLPR03vb50	Dynamic Window-based weighting	50	0.1770	7%
NLPR03w49	Core Window-based weighting	49	0.2434	10%

From the above table, we can see that we get a low value of average precision. The reason is that for each topic, only several dozens of retrieved documents are returned, not 1000. We suppose that most users only look through the former part of retrieval result and a robust IR system should ensure that users can find relevant documents at the top 10 or 20. Based on the experiments, the window-based methods, especially core window based method, outperform most traditional tf-idf weighting method. Detail evaluation and discussion are given in the paper of reference [1].

Reference

[1] Qianli Jin, Jun Zhao, Bo Xu, *Window-based Method for Information Retrieval*, 2004, The First International Joint Conference on Natural Language Processing

[2] Qianli Jin, Jun Zhao, Bo Xu, *Query Expansion based on Term Similarity Tree*, 2003 International Conference on Natural Language Processing and Knowledge Engineering, IEEE.

[3] Amit Singhal, Chris Buckley, Mandar Mitra, *Pivoted Document Length Normalization*, SIGIR 1996.

TREC-2003 Novelty and Web Track at ICT

Jian Sun, Zhe Yang, Wenfeng Pan, Huaping Zhang, Bin Wang, Xueqi Cheng
Institute of Computing Technology (ICT), Chinese Academy of Sciences, P.R. China
sunjian@ict.ac.cn

1 Overview

In this paper, we will present our approaches and experiments on the following two tracks of TREC-2003: Novelty track and Web track.

The novelty track can be treated as a binary classification problem: relevant vs. irrelevant sentences, or new vs. non-new. In this way, we applied variants of techniques that have been employed for text categorization. To retrieve the relevant sentences, we compute the similarity between the topic and sentences using vector space model (VSM). In addition, we tried several techniques in an attempt to improve the performance: using narrative field and adopting dynamic threshold for different documents. We also have implemented the KNN algorithm and Winnow algorithm for classifying the sentences into relevant and irrelevant in the novelty task 3. To detect the new sentences, we used Maximum Marginal Relevance (MMR) measure, Winnow algorithm and so on. In addition, we attempted to detect novelty by computing semantic distance between sentences using WordNet.

For the Web track, we improved the basic SMART system, and the *Lnu-Ltu* weighting method was introduced into the system. The improved system has been proved to be effective in last year's task. In addition, we implemented a simple retrieval system using the probability model that is adopted by Okapi.

The structure of the paper is as follows: The section 2 reports the approaches and experiments in novelty track. The section 3 describes the experiments in web track. Finally, in section 4, we conclude by summarizing our experiments and presenting the future work.

2 Novelty Track

In the novelty track, there are four tasks which vary the kinds of data available to the systems and the kinds of results that need to be returned. In what follows, we will describe our approach to each task together with results of TREC experiments.

2.1 Task1 of Novelty Track

2.1.1 Relevant Sentences Retrieval

We retrieved the relevant sentences by comparing the topic to them using VSM and also applied several techniques in an attempt to improve the performance.

2.1.1.1 Vector Space Model (VSM)

In the VSM, the feature selection is required to decrease the dimensionality and improve the efficiency of classification. We used a χ^2 statistic, which measures the lack of independence between term *t* and topic *c*, to select features [Yang and Pedersen, 1997] for each topic in our novelty track.

We denote the feature set, which is obtained using χ^2 statistic, as F_1. Finally, the feature set F of one topic can be obtained using the following formula:

$$F = F_1 \cup (TW \cap \overline{S}) \qquad (2.1)$$

where TW is the set of title words of the topic, $\overline{S}$ is the complement of stop words. The weighting of each term we used is also χ^2 statistic.

After the feature selection and weighting, both the topic and the sentence are represented as weighted vectors. The sentence is scored based on its similarity (in our experiments, we use cosine function) to the topic vector. If the similarity score is greater than a specific threshold δ, the sentence is regarded as relevant.

2.1.1.2 Dynamic threshold (DTH)

We observe that, in general, there is a specific time period during which the topic occurs. For example in topic N5 titled "*World Cup soccer*", the docs are mainly from June 03, 1998 to July 12, 1998. We call the specific time period "*focus*". Before and after the *focus*, there are fewer reports on the topic. As a result, we think the number of relevant sentences in docs not in *focus* is fewer than that in docs in *focus*. The observation motivates us to dynamically adjust the threshold according to the time of the document. Before and after the *focus* of the topic, the threshold is increased. Our strategy on the dynamic threshold for each topic is to: (1) obtain the starting and ending time of the all documents; (2) divide the time period into 4 equal time periods; (3) map each document to one of the 4 periods and obtain the number of docs in that period, *n;* (4) compute the document density, *density=n/N*, for each period. N is the number of docs.

The threshold of each period is as follows:

$$\delta = \begin{cases} \delta_0 - 0.05, & density > 0.75 \\ \delta_0, & density > 0.5 \\ \delta_0 + 0.05, & density > 0.25 \\ \delta_0 + 0.1, & density > 0 \end{cases} \tag{2.2}$$

where δ_0 is the basic threshold, which is obtained using the TREC-2002 Novelty data.

2.1.1.3 Enlarging the data (XTD)

There are at most 25 relevant documents for each topic. We want to know whether the data is enough or not. Therefore we investigated whether the performance can be improved by enlarging the text data. We retrieved more documents (75 docs for each topic) from AQUAINT collection using the SMART system. The query for each topic was the content words in the title field. As a result, there will be 100 documents for each topic. We then used the same feature selection method to determine the terms and their weighting.

2.1.1.4 Using Narrative Field (NAR)

We know that the narrative field describes the information requirement in detail: (1) what is we need (e.g. *References to Dolly's children are relevant if Dolly's name is included.*) and (2) what is irrelevant (e.g. *Mention of Polly and Molly are not relevant*). Traditional methods usually did not use the information (we call it *negative information*) in (2). In order to utilize the negative information, we first obtained the negative features by selecting the content words in the narrative field that tell us what information is not needed, and then built a negative vector for each topic.

We determine the relevance of the sentence by computing the following similarity as follows:

$$sim(topic, s_i) = sim(V_{si}, V_{tp}) - sim(V_{si}, V_{tn}) \tag{2.3}$$

where V_{si} denotes the vector of sentence *i*, V_{tp} the (positive) vector of topic, V_{tn} the negative vector of topic.

2.1.1.5 Features based on local co-occurrence (CUR)

We also tried another method of extracting the features. For example, we adopted the approach, *local co-occurrence*, proposed by [Zhang. etc, 2002]. The fixed window we used was -2 ~ 2. The weighting was also χ^2 statistic. One official run using the method was submitted.

2.1.2 Novelty Detection

As for the novelty detection, we applied two methods: word overlapping between two sentences and maximum marginal relevance (MMR).

2.1.2.1 Word Overlapping between two sentences

The word overlapping (OLP) between two sentences, s_i and s_j, is similar to [Zhang. etc, 2002]. The method did not use the similarity between topic and sentence. Since the method is simple, we did not use it in task1. In task2, we submitted one run that used the method.

2.1.2.2 Maximum Marginal Relevance (MMR)

Another approach to select novelty sentences from relevant sentences is Maximum Marginal Relevance (MMR) [Carbonell and Goldstein, 1998] measure, which is given by:

$$MMR(s_i) = \lambda Sim_1(V_{si}, V_{tp}) - (1-\lambda)\max_{S_j \in R} Sim_2(V_{si}, V_{sj})] \qquad (2.4)$$

where R is the set of selected relevant sentences, and Sim_1 and Sim_2 are similarity metrics.

To obtain the novelty sentences from the relevant set, we first ranked the relevant sentences according to one of above measures and then selected a specific percentage from them.

2.1.3 Submitted Results

We submitted five runs for task 1. The five runs were:

1) ICT03NOV1BSL: local co-occurrence,threshold=0.48; MMR
2) ICT03NOV1SQR: χ^2 statistic, threshold=0.40; MMR
3) ICT03NOV1NAR: NAR, local co-occurrence, threshold=0.20; MMR
4) ICT03NOV1XTD: XTD, χ^2 statistic, threshold=0.20; MMR
5) ICT03NOV1DTH: DTH, local co-occurrence, threshold=0.46; MMR

The results of our official runs at TREC-2003 Novelty Task 1 were shown in Table 2.1. We observed that the run that adopted the dynamic threshold achieved the best performance of five runs.

Table 2.1 Performance of Official Run of Novelty Task 1

Run#	Relevant Part			Novelty Part		
	P	R	F	P	R	F
ICT03NOV1BSL	0.62	0.51	0.486	0.41	0.48	0.379
ICT03NOV1SQR	0.63	0.54	0.489	0.40	0.49	0.368
ICT03NOV1NAR	0.58	0.46	0.434	0.37	0.44	0.334
ICT03NOV1XTD	0.61	0.39	0.408	0.39	0.36	0.310
ICT03NOV1DTH	0.63	0.50	0.489	0.42	0.44	0.370

2.2 Task2 of Novelty Track

For the novelty detection of task 2, we select a specific percentage of relevant sentences as new sentences. We believe the percentage of new sentences in relevant sentences decreases as the ID of document increases for each topic. To model the intuition, we applied a simple method, *dynamic percentage*, as follows:

1) set the average percentage p_v , which is obtained using the TREC-2002 data, of new sentences in relevant sentences
2) set the percentage p_1 of new sentences in relevant sentences of 1th document: $p_1 = p_v + 12.5\%$
3) set the percentage p_i of new sentences in relevant sentences of ith document: $p_i = p_1 - i*1\%$

2.2.1 Submitted Results

The five runs we submitted were:

1) ICT03NOV2SQR: χ^2 statistic, MMR
2) ICT03NOV2CUR: local co-occurrence, MMR
3) ICT03NOV2PNK: χ^2 statistic, NAR described in Section 2.1.1.4, MMR
4) ICT03NOV2LPP: OLP described in Section 2.1.2.1, dynamic percentage
5) ICT03NOV2LPA: OLP described in Section 2.1.2.1, fixed percentage

Table 2.2 showed our official runs at TREC-2003 Novelty Task 2. Comparing the results of the five runs, we noticed that the performance of run ICT03NOV2LPA was better than that of other runs. Actually, the run only applied the simplest techniques, i.e. counting the words that occur in both sentences.

2.3 Task3 of Novelty Track

For task 3, we concentrated on the retrieval of relevant sentences. We implemented KNN algorithm and Winnow algorithm for selecting the relevant sentences. The method of novelty detection was similar to that of task 1.

2.3.1 KNN algorithm for Retrieval of Relevant Sentences

In the task 3, we examined the KNN algorithm at the sentence level. And two strategies were taken to predict the class (relevant or irrelevant) of a sentence. One was that the prediction will be the class that has the largest number of members in the *k* nearest neighbors. The other was that the class with maximal average similarity will be the winner. These two strategies were denoted as KNN1 and KNN2, respectively.

2.3.2 Winnow algorithm for Retrieval of Relevant Sentences

The Winnow [Dagan, 19997] algorithm has been shown that it functions well in text domain. In the experiments presented here, we used it at the sentence level.

In this experiment, the strength of the feature is taken to indicate only the presence or absence of it in the sentence, that is, it is either 1 or 0.

2.3.3 Submitted Results

The five runs we submitted were:

1) ICT03NOV3KNN: all content words as features, KNN1; MMR
2) ICT03NOV3IKK: all content words as features, KNN2; MMR
3) ICT03NOV3KNS: χ^2 statistic, KNN2; MMR
4) ICT03NOV3WND: Winnow; MMR, dynamic percentage
5) ICT03NOV3WN3: Winnow; MMR, fixed percentage

The Table 2.3 showed our official runs at TREC-2003 Novelty Task 3. Comparing the first three runs, the run ICT03NOV3KNS that applied feature selection achieved better results. Comparing the last two runs with the first three runs, we observed that the precision increased using the Winnow algorithm while the recall decreased. As for the novelty detection, we made a mistake in the official run and the first relevant sentence in the 6th doc was taken to be the first relevant sentence in the topic by us. The mistake resulted in the bad performance of the novelty detection. We believe that we can improve the performance of novelty track further in future work.

Table 2.2 Performance of Official Run of Novelty Task 2

Run#	Novelty Part		
	P	R	F
ICT03NOV2SQR	0.65	0.74	0.677
ICT03NOV2CUR	0.65	0.73	0.677
ICT03NOV2PNK	0.65	0.73	0.676
ICT03NOV2LPP	0.65	0.74	0.679
ICT03NOV2LPA	0.73	0.87	0.783

Table 2.3 Performance of Official Run of Novelty Task 3

Run#	Relevant Part			Novelty Part		
	P	R	F	P	R	F
ICT03NOV3KNN	0.57	0.56	0.547	0.35	0.37	0.346
ICT03NOV3IKK	0.57	0.58	0.548	0.36	0.39	0.348
ICT03NOV3KNS	0.60	0.58	0.572	0.37	0.39	0.362
ICT03NOV3WND	0.65	0.53	0.557	0.39	0.35	0.346
ICT03NOV3WN3	0.68	0.49	0.537	0.43	0.41	0.381

2.4 Task4 of Novelty Track

To detect the new sentences from the relevant sentences for the task 4, we applied several methods, such as MMR measure and Winnow algorithm. In addition, we attempted to detect novelty by computing semantic distance between two sentences using WordNet.

2.4.1 Winnow Algorithm for Novelty Detection

Since we know the new sentences and non-new sentences in the relevance for the task 4, we can train a classifier using Winnow algorithm. The classifier represents a sentence as a set of features F = $\{f_1, f_2 \ldots f_m\}$. The number of active features in the sentence we used was 5. We compute the strength of each feature as follows:

(1) compute the similarity between the sentence and the topic, f_1';

(2) compute the similarities between the sentence and all of those thatoccurred before it using VSM and obtain the two biggest similarities, f_2' and f_3';

(3) compute the word overlapping between the sentence and all of those that occurred before it and obtain the two biggest overlapping, f_4' and f_5';

(4) compute the strength of each feature as $f_1 = f_1$', $f_2 = 1 - f_2$', $f_3 = 1 - f_3$', $f_4 = 1 - f_4$' and $f_5 = 1 - f_5$', respectively.

The weight vector was estimated on training data using Winnow algorithm. After the weight vector was obtained, the Winnow algorithm was used to predict the novel sentence.

2.4.2 Computing Semantic Distance using WordNet for Novelty Detection

2.4.2.1 Motivation

Compared with the traditional IR, the Novelty track returned ranked only new and relevant sentences rather than a large amount of relevant document. The information content within a sentence is very small. Traditional methods can be used in sentence level; however, we think it is not the best choice if we only focus on word form, as almost all words occur once within a sentence. For example, there are two sentences in the N12 topic:

1) *Daily we read news stories about dissatisfaction with managed care, Medicare fraud and overbilling.*
2) *Eighty percent agreed with this.*

Both sentences described opinions on universal healthcare. However, it's difficult to detect relevance or novelty between them, since the words "dissatisfaction" and "agreed" seem irrelevant in terms of word form.

2.4.2.2 WordNet-based Semantic Distance between Words

We assume that the distance between same words is zero. If the two words have different word form, we compute the distance $Dist(w_1,w_2)$ as described in [Jiang and Conrath, 1997].

We have found that in many cases, it is still far from requirement if we only cover hypernym between words. Based on the above work, we introduce more word relations. They mainly include similarity and derivation between words. For example, "friendly' is derived from "friend" and "friendly" is similar to "amicably". We assigned the distance between such words with 0.5.

Apart from above relations, distance between any other words is set to be a large value.

2.4.2.3 Computation on Semantic Distance between Sentences

Before semantic distance between sentences, we define word-sentence semantic distance (WSSD) to be the minimum distance between the word w and words within the sentence S. Therefore, we estimate $WSSD(w, S)$ with the following formula:

$$WSSD(w, S) = \min \{Dist(w,w_i) | w_i \in S\} \quad (2.5)$$

where w is a word, S is a sentence and w_i is a word in sentence S.

Based on $WSSD(w, S)$, we define sentence distance SSSD(Sa,Sb) as follows:

$$\text{SSSD(Sa, Sb)} = \frac{\sum_{w_i \in Sa} WSSD\ (w_i, Sb) + \sum_{w_j \in Sa} WSSD\ (w_j, Sa)}{|Sa| + |Sb|} \quad (2.6)$$

where Sa and Sb are sentences; |Sa| and |Sb| are word numbers in sentence, respectively.

2.4.2.4 Novelty Detection

For novelty detection, we consider the following features:

1) *f1*: Semantic distance between a relevant sentence S and topic T.Let it be $SSSD(S, T)$. Naturally, S is more likely to be new if $SSSD(S, T)$ is less.
2) *f2*: minimal semantic distance from S to previous valid context P. Let it be $SSSD(S, P)$.
3) *f3*: Word overlapping from sentence S to topic T. Let it be *Overlapping*(S, T). For two sentence Si and Sj, we define word overlapping with:

$$\textit{Overlapping}(\text{Si,Sj}) = \frac{|\text{Si} \cap \text{Sj}|}{|\text{Si}|} \quad (2.7)$$

Similarly, larger *Overlapping*(S, T) indicates that the sentence is more relevant to the topic.

4) *f4*: Word overlapping with previous valid context P. Let it be *Overlapping*(S, T). Less *Overlapping*(S, T) tends to be new sentence.
5) *f5*: Is S a head sentence? Let it be *IsHead*(S). *IsHead*(S) is equal to 1 if S is a head sentence in the paragraph. Otherwise, it is 0.

Then we can define a 5-tuple feature vector as $F=(f_1, f_2, f_3, f_4, f_5)=(1- SSSD(S,T), SSSD(S,P),$ $Overlapping(S,T)$, 1-*Overlapping*(S,P), *IsHead*(S)). After defining the features, we apply Winnow algorithm to estimate weight.

For the task 4, the novel sentences in the first 5 documents were treated as positive training set, while other relevant sentences were treated as negative training set. After the weight vector was obtained, the Winnow algorithm predicted the novel sentence as described in Section 2.4.1.

2.4.3 Submitted Results

We submitted two runs (ICT03NOV4SQR and ICT03NOV4WNW) using the methods described in Section 2.4.1. The MMR was adopted in run ICT03NOV4SQR and the Winnow algorithm was adopted in run ICT03NOV4WNW. The three runs (ICT03NOV4ALL, ICT03NOV4LFF and ICT03NOV4OTP) adopted the methods described in Section 2.4.2. The detailed feature vectors of the three runs were shown in Table 2.4:

Table 2.4 The three runs and their features

Run#	Feature vector
ICT03NOV4OTP	(f_3, f_4)
ICT03NOV4LFF	(f_1, f_2, f_3, f_4)
ICT03NOV4ALL	$(f_1, f_2, f_3, f_4, f_5)$

Table 2.5 Performance of Official Run of Novelty Task 4

Run#	P	R	F
ICT03NOV4ALL	0.60	0.68	0.598
ICT03NOV4LFF	0.59	0.64	0.568
ICT03NOV4OTP	0.59	0.70	0.610
ICT03NOV4SQR	0.61	0.66	0.623
ICT03NOV4WNW	0.65	0.72	0.636

The Table 2.5 showed our official runs at TREC-2003 Novelty Task 4. From the results, the run ICT03NOV4WNW achieved better performance than other runs. Although the computation of semantic distance using WordNet did not show any improvement in our official runs, we think that there are still potential improvements if we can make good use of prior knowledge and other information imbedded in the sentence.

3 Web Track

3.1 Introduction

This year, Web track consists of two subtasks: the Named/Home Page Finding task, and the Topic Distillation task. The former task is introduced to investigate methods for effective navigational search, with a mixture of home page and named page queries: finding a particular page desired by the user. This task involves a mixture of tasks from two previous years: home page finding and named page finding. In both cases, there is only one target page and user's queries are often the name of the page. For the Topic Distillation task, it is introduced to investigate methods for finding a set of the best home pages given a broad query. For this task, the key is to find as many different websites (represented by their entry pages) as possible within the first ten results. The test collection of this year's Web track is *.Gov* data set as the last year.

As the last year, our retrieval system was based on SMART. We modified the basic SMART system, and the *Lnu-Ltu* weighting method was introduced into the system. This system has been proved to be effective in last year's task. In addition, we implemented a simple retrieval system using the probability model that is adopted by Okapi.

3.2 Named/Home Page Finding

As introduced in above section, the goal of named page finding task is to find the page that named as user's query. For the home page finding task, the difference is that home page finding queries are restricted to home pages. Since in named/home page finding task the user explains his goal explicitly, every word in the query is more important than that be in ad-hoc task, which is a tough reason to request nearly all the words in the query appearing in the relevant document and content should be emphasized. The run ICTWebKI12A is an original result retrieved by content, which determine other runs' performance.

As the task's name, the target page always has a name being similar with the query, and the title of web page is also a very important component. To some extent, the anchor text is also the page's name: it is the index by which users can visit the page from other pages. This useful structure information is the key of the improvement of performance. In the run ICTWebKI12B, we combined the scores of content, title and the anchor text into a unified measure using the linear interpolation, and obtained a better result than the original result.

In this year's task, home page finding topics are mixed with named page finding topics. If we can divide these topics, the URL can be used in the home page finding task. For this purpose, we used a simple strategy: the topic described as entity, such as a special person, a special location or a special organization, was judged to be a home page finding topic. We used two different combining methods to the divided topics: for the named page finding topics, content, anchor text and title was used, for the home page finding topics, URL length was added to be an important factor which computes the probability of a page to be a home page. The run ICTWebKI12C divided the topics into NP topics and HP topics as described, and obtained a better result than ICTWebKI12B.That is to say: our simple method to identify home page finding task is useful. The Table 2.6below showed the results of three runs for this year's task.

Table 2.6 Results of Named/home Page Finding Task in TREC-2003

RunId	MRR	Founded Answers	Not Found
ICTWebKI12A	0.308	207/300	93/300
ICTWebKI12B	0.449	247/300	53/300
ICTWebKI12C	0.568	265/300	35/300

3.3 Topic Distillation

As described in the TREC-2003 Web Track Guideline, to be judged a "key resource", the page returned should be a good entry point to a website which: 1) is principally devoted to the topic, 2) provides credible information on the topic and 3) is not part of a larger site also principally devoted to the topic.

In TREC-2002, almost all the participants reached the same conclusion: The structure information will hurt the effect of retrieve. Our further experiments proved it again: Only anchor text or title can only get 0.03 for P@10, and the performance of retrieve almost can not benefit from such a result. In TREC-2002, our original retrieval result get 0.2360 at P@10, that is to say that there are more than 2 relevant documents in the first ten results. Such a good result motivated us to adopt the pseudo relevance feedback. Assuming the first five or ten results as relevance documents, we used Rocchio method to expend the original queries, and then retrieved with the new queries and obtained the final results. In our experiments, a small number of relevant documents assumed (such as 5) and moderate expended query terms (such as 15) can get a considerable improvement. These experiments suppose us to use pseudo relevance feedback in this year's task. But in this year's task, no query was offered like before: only narratives were given. We must create queries manually and it leads to disastrous original results.

Pseudo relevance feedback is influenced by the original results, and bad original results lead to worse final results. The table 2.7 below showed the different results of pseudo relevance feedback based on different original results.

Table 2.7 Feedback results and Original results in TREC-2002 and TREC-2003

Run	Average precision	P@10
Orig_2002	0.1620	0.2306
Fdb_2002	0.1748	0.2510
Orig_2003	0.0728	0.0520
Fdb_2003	0.0639	0.0380

Table 2.8 Multiple retrieval systems in TREC-2003

Run	Average precision	P@10
1.VSM	0.0728	0.0520
2.VSM with stemming	0.0571	0.0480
3.Okapi	0.0441	0.0320
4.Okapi with stemming	0.0440	0.0480
5.1+2+3+4	0.1036	0.0536

In addition, we used multiple retrieval systems vote mechanism. We combined the results of four different retrieval systems: VSM with stemming, VSM without stemming, probability model with stemming and probability model without stemming. As the results shown in Table 2.8, VSM system without stemming represented better than the others, and multiple retrieval systems vote mechanism showed its contribution for improving the performance.

4. Summary and Future Work

This paper presented our work in the Novelty track and Web track evaluated at TREC-2003. In the Novelty track, we applied the KNN algorithm and Winnow algorithm to retrieve relevant sentences. To detect the new sentences, we tried several methods, including semantic distance between sentences using WordNet, MMR measure, Winnow algorithm and so on. We also conducted our experiments on: (1) the use of χ^2 statistic for feature selection, (2) dynamic threshold for different document to retrieve relevant sentence, (3) the use of narrative field in topic, and (4) dynamic percentage of relevant sentences as novelty sentences within a document.

In the experiments, we observed that the performance of novelty detection greatly depends on the system's ability to retrieve the relevant sentences. The χ^2 statistic for feature selection and dynamic threshold for different document to retrieve relevant sentence were shown to be effective in the track.

The experiments on the Web track showed that the vote mechanism can improve the performance and the VSM retrieval systems without stemming always work a little better than those systems with stemming.

Our future work includes: (1) Further studying the problem of how to expand the information of sentence level using WordNet or other resources for similarity computation; (2) Exploiting the use of type information of topic; (3) Investigating how to determine the number of features for each topic.

References

J. Allan, C. Wade and A. Bolivar. Retrieval and Novelty Detection at the Sentence Level, Proceedings of the 26th annual international ACM SIGIR, pp314-321, 2003

T. Ault and Yiming Yang. kNN, Rocchio and Metrics for Information Filtering at TREC 10. Proceedings of the Tenth Text Retrieval Conference, pp 84-93. 2001

J. Carbonell and J. Goldstein, The use of MMR, Diversity-based Reranking for reordering document and producing summaries. Proceeding of SIGIR 98, pp335-336, 1998

Dagan, Ido, Yael Karov and Dan Roth. Mistake-driven learning in text categorization, in Proceedings of Second Conference on Empirical Methods in Natural Language Processing (EMNLP-2), 1997

Rocchio, J. J. Relevance Feedback in Information Retrieval. In The SMART Retrieval system, Prentice-Hall, Englewood NJ. 232-241,1971

Jay J. Jiang and David W. Conrath, Semantic similarity based on corpus statistics and lexical taxonomy. In proceedings of International Conference Research on Computational Linguistics, 1997.

H. Kazawa, T. Hirao, H. Isosaki and E. Maeda. A machine learning approach for QA and Novelty Tracks: NTT system description. To appear in the Proceedings of the Eleventh Text Retrieval Conference, 2002

G.Salton,Chris Buckley. 1990. Improving Retrieval Performance By Relevance Feedback. JASIS 41.288-297.

Thijs Westerveld, Wessel Kraaij, and Djoerd Hiemstra. 2001. RetrievingWeb Pages using Content, Links, URLs and Anchors, In The Tenth Text REtrieval Conference (TREC 10), pp663, 2001

Yang, Y., Pedersen J.P. A Comparative Study on Feature Selection in Text Categorization Proceedings of the Fourteenth International Conference on Machine Learning (ICML'97), pp412-420, 1997

M. Zhang, R. Song, C. Lin, S. Ma, Z. Jiang, Y. Jin, Y. Liu, L. Zhao and S. Ma. Expansion-Based Technologies in Finding Relevant and New Information: THU TREC2002 novelty track experiments. To appear in the Proceedings of the 11th TREC Conference, 2002

TREC 2003 Question Answering Track at CAS-ICT

Yi Chang, Hongbo Xu, Shuo Bai
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
{changyi, hbxu}@software.ict.ac.cn

Abstract

In our system, we make use of Chunk information to analyze the question. A multilevel method is fulfilled to retrieve candidate Bi-sentences. As to answer selecting, we proposed a voting method. We figure out the performance of each module of our system, and our study shows that 65.54% information has lost in document retrieval and Bi-sentence retrieval.

Keyword: Chunk, Bi-sentence, multilevel retrieval, voting

1. Introduction

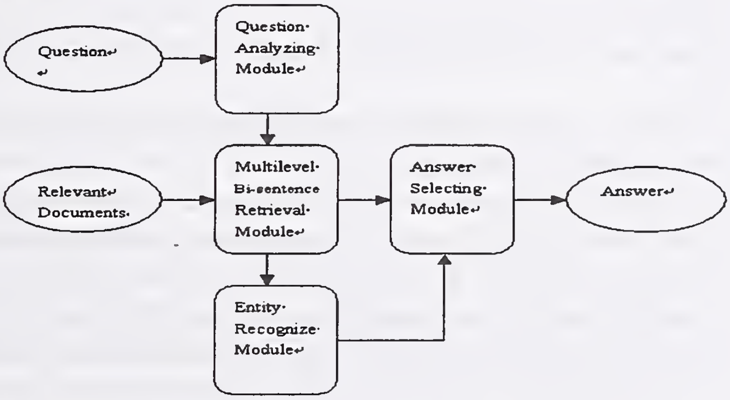
It is the third time we take part in the TREC-QA track. We undertook the new main subtask and submitted three runs for evaluation.

Our QA system incorporates several useful tools. The first is LT_CHUNK, which is developed at University of Edinburgh. We use LT_CHUNK to get the chunks of each sentence and the POS tags of different words. The second is GATE, developed by University of Sheffield. We use GATE to identify some Named Entities.

In our previous QA system, we tried different kinds of elaborate algorithms, but the results were not satisfactory, and we didn't make it clear what is the performance of each step in our system. So we try to figure it out this year, and our study shows: 65.54% information has lost in document retrieval and Bi-sentence retrieval.

2. System Description

Our system contains four major modules, namely Question Analyzing Module, Multilevel Bi-sentence Retrieval Module, Entity Recognizing Module and Answer Selecting Module. However, for those definition questions, the Entity Recognizing Module is unnecessary. The system architecture is represented as below.



To answer each question, Question Analyzing Module makes use of NLP techniques to identify the right type of information that the question requires. We select top 50 out of the 1000 given relevant documents to find the candidate answer from them. Since there exists too much redundant information in these documents, Multilevel Bi-sentence Retrieval Module matches the question with the 50 relevant documents at different levels and retrieves some top ranked Bi-sentences to find the candidate answer from them. Entity Recognizing Module identities the candidate entities from the selected Bi-sentences, and Answering Selecting Module selects the answer in a voting method.

2.1 Question Analyzing Module

Since most of the factoid and list questions ask for Named Entity (NE) as the response, Question Analyzing Module tries to identify the required NE type during parsing a question.

Some question words can map to the required NE types directly, e.g. "who"(PERSON), "where"(LOCATION), "how many"(NUMBER).

However, for most of the questions whose question word is "which" or "what", we need to find the Core Noun to help us to identify the required NE type. Where Core Noun is a noun in each question that indicates the answer. For example:

[1] *Which city is home to Superman?*

[2] *Which past and present NFL players have the last name of Johnson?*

[3] *What type of bee drills holes in wood?*

The Core Noun of question [1] is "city" since the answer to the question is a city, and the Core Noun of question [2] and [3] is respectively "players" and "bee". We identify the required NE type according to a predefined Map Lexicon that consists of hundreds of nouns mapping to the NE type that can be recognized later, e.g. "city"(CITY), "player"(PERSON). We build another Abstract Noun Lexicon that consists of some abstract nouns, e.g. "type", "breed".

The algorithm to find the Core Noun is as follows:

Step 1: Take the last noun in the first Noun Group as Core Noun;

Step 2: If the Core Noun is in Abstract Noun Lexicon, find the last noun in the next Noun Group as Core Noun;

Step 3: If there is no suitable noun that can be found, the Core Noun is empty.

Because there are some questions whose required NE type cannot be recognized later by Entity Identifying Module, e.g. "bee". We regard these questions as "other NE questions" and keep their Core Noun for further matching.

The question whose Core Noun is empty is called "miscellaneous question". For example:

[4] *How did Minnesota get its name?*

[5] *What is tequila made from?*

Applying some syntactic pattems to find its answer is a practical method, but in our current QA system we simply answer NIL.

Processing the definition question is rather simple, and we just extract the phrase to be explained.

2.2 Multilevel Bi-sentence Retrieval Module

A Bi-sentence is a pair of consecutive sentences, and a Phrase is a sequence of keywords or one keyword in a question, where a Keyword is a word in the question but not in our Stop Word list.

The goal of Multilevel Bi-sentence Retrieval Module is to find some top ranked Bi-sentences most relevant to a question. According to our training results on TREC-11 QA corpus, we select top 20 Bi-sentences for the factoid and definition questions and top 50 Bi-sentences for the list question.

We assume: 1) Bi-sentences that can match a phrase of a question are more relevant than those only can match separate keywords. 2) Bi-sentences that can match a phase of a question in raw form are more relevant than those only can match in stemmed form.

Since the strictly matching will decrease the recall rate of the Bi-sentences retrieval, we parse a Bi-sentence for several times and match the question at different levels to improve the precision rate without decreasing the recall rate.

For factoid and list question, our system applies a four-level method to select candidate Bi-sentences. At each level, we define two kinds of substrings, Compulsory Phrase and Assistant Keyword. Compulsory Phrase is a phrase set in which each element is obligatory to match Bi-sentences. Assistant Keyword is a keyword set in which each element is optional to match. Those words not belong to the Compulsory Phrase and Stop Word list are regarded as the elements of the Assistant Keyword. We compute the weight of a Bi-sentence as below:

$$weight\_p = \beta \times count\_a / (count\_q + count\_p)$$

Where $weight\_p$ means the weight of the Bi-sentence, $count\_a$ means the number of matching Assistant Keyword between the question and the Bi-sentence, $count\_q$ means the number of keywords in the question, $count\_p$ means the number of keywords in the Bi-sentence, and β is an experiential parameter. All relevant Bi-sentences are ranked by: the Bi-sentence selected from the higher level has a higher priority, and in the same level, the Bi-sentence with a larger weight has a higher priority. Furthermore, the first level is based on raw matching, while the other three levels are based on stemmed matching.

At the first level, we take the last Noun Group and the last verb in the last Verb Group as the Compulsory Phrase. And those phrases with initial capital on each word are also regarded as the Compulsory Phrase. At the second level, we move the verb from the Compulsory Phrase to the Assistant Keyword because the verb is difficult to match and we don't fulfill any verb expansion. At the third level, we only leave those phrases composed of successive initial capital words as the Compulsory Phrase. At the last level, the Compulsory Phrase is empty, all words belong to the Assistant Keyword, and this is equal to the method in our previous QA system to compute relevance between the question and the candidate Bi-sentences.

For definition question, our system simply takes a two-level method to select a candidate Bi-sentence. At the first level, we take the phrase to be explained as the Compulsory Phrase. A Bi-sentence selected not only matches the Compulsory Phrase but also matches the definition pattern proposed by InsightSoft[1]. In the second level, we just regard each keyword in the question as Assistant Keyword, and then find the most relevant Bi-sentence according to the weight.

2.3 Entity Recognizing Module

We use GATE to recognize some types of Named Entities, such as PERSON, LOCATION, COUNTRY, etc. And we take some new strategies to recognize other types, such as YEAR, COLOR, DISEASE, etc.

For those questions whose required NE type is identified, we recognize the required NE as our candidate answer. However for those questions whose required NE type cannot be identified, we make use of Core Noun to construct the possible phrase as the candidate entity by some syntactic rules.

2.4 Answer Selecting Module

In the top 20 Bi-sentences for each factoid question, we may find more than one suitable Named Entity. How to choose the most suitable NE as the answer in our system is difficult, since the NE type is the only semantic information we used.

We assume that the answer in the top 20 Bi-sentences is most likely to appear several times, so we use a voting method to select the answer. In our experiments on TREC-11 QA corpus, the voting method gets an improvement of 15.58% comparing with the method that simply select the first one as the answer.

We also study the weighted voting method, where the weight of the candidate answer decreases with the rank of the Bi-sentences. According to our experiments on TREC-11 QA corpus, the results are similar to the simple voting method.

For list question, since the answer number of each question is unknown, we choose the entities whose frequencies in voting are beyond a threshold. According to training results on TREC-11 QA corpus, our threshold varies from the required NE types.

For definition question, we simply choose the first Bi-sentence as the answer.

3. Results

We don't try radically different methods in our three runs, but instead we simply make little change to get a steady output. Table 3.1 shows our results. In run A, we answer NIL for all "other NE question" whose required NE type is highly depended on Core Noun, and we only take part of the definition patterns into effect. In run B, we use some looser rules to construct the possible NE for "other NE question", while in run C we use some stricter rules.

RunID	Factoid Accuracy	#Correct (#NIL)	#Inexact	#Unsupported	List Ave_F	Definition Ave_F	Final Score
ICTQA2003A	0.128	53(21)	6	6	0.091	0.142	0.122
ICTQA2003B	0.140	58(17)	5	8	0.089	0.149	0.130
ICTQA2003C	0.145	60(14)	8	10	0.091	0.149	0.133

Table 3.1

4. Error Analysis

Here we just focus our analysis on factoid questions because it is easier to get a quantitative evaluation at each step than that of the other two types of questions. Since we simply answer NIL whenever we cannot find an answer, we don't have too many opinions on the NIL questions. Therefore we only focus our analysis on the question whose answer is not NIL. All of the error analysis is studied from the result of run C.

4.1 Question Analyzing Error

Table 4.1 shows the distribution of the question analysis and the error rate in each question category.

Question type	#Total Questions	#NE identified Questions/#Wrong	#Other NE Questions/#Wrong	#Miscellaneous Questions/#Wrong
Factoid	413	275/5	123/24	15/1
List	37	24/0	13/0	0/0
Definition	50	/	/	50/0

Table 4.1

The performance of our Question Analyzing Module is fairly satisfactory, and the overall accuracy of question analysis is 94%. There are two main reasons for the rest 30 questions incorrectly analyzed: 1) 16.7% (5/30) error is caused by chunk error of LT_CHUNK. 2) 83.3% (25/30) error is that our Question Analyzing algorithm cannot cover all questions.

4.1 Retrieval Error

4.1.1 Document Retrieval Error

According to the our statistical result, there are 275 out of 413 factoid questions whose answers can be got from the top 50 documents. Since there are total 383 questions whose answer is not NIL, the maximum accuracy of document retrieval with top 50 documents is 71.80%. That is, at least 28.20% of 383 questions cannot be answered correctly in document retrieval process.

4.1.2 Bi-sentence Retrieval Error

In our statistics, from the top 20 Bi-sentences, there are only 132 out of 275 factoid questions can be answered correctly based on the answer and the corresponding Document ID. So the maximum accuracy of Bi-sentence retrieval by our Multilevel Bi-sentence Retrieval Module is 48.0%. Also, at least 52.0% of 275 questions cannot be answered correctly in Bi-sentence retrieval process.

4.2 Answer Error

There are the two main reasons for the inexact answer error: 1) 50% (4/8) error is caused by the inexact identification of required NE type; 2) 50% (4/8) is caused by the inexact recognizing of NE.

Since we make use of no more semantic information than NE type, we cannot avoid the unsupported answer error.

According to our manual statistical result listed in Table 4.2, those questions whose required NE type can be identified are easier to answer.

Question Type	#Not NIL Questions	#Correct	Precision	#Inexact	#Unsupported
NE identified Question	254	41	16.14%	5	8
Other NE Question	116	5	4.31%	3	2

Table 4.2

5. Conclusions

Of 383 factoid questions whose answer is not NIL, only 46 are correctly answered in our system, so the precision is 12.01%. The accuracy of document retrieval is 71.80% and Bi-sentence retrieval 48.0%. The accuracy of question analyzing, NE recognizing and NE selecting adds up to 34.85%. That is, 65.54% error results from retrieval process including document retrieval and Bi-sentence retrieval, while only 22.45% error results from question analyzing, NE recognizing and NE selecting.

According to the study above, most information lost during retrieval process. So we hope to focus our further study on seeking better retrieval algorithms, especially Bi-sentence retrieval algorithms.

Acknowledgements

This research is supported by the National High Technology Research and Development Program of China(863 program) under contact of 2002AA142110, the Institute Youth Fund under contact of 20026180-24. We give our thanks to all the people who have contributed to this research and development, in particular Xueqi Cheng, Bin Wang, Dongbo Bu, Li Guo, etc.

References

[1] M.M.Soubbotin, Patterns of Potential Answer Expressions as Clues to the Right Answer, *In the Tenth Text REtrieval Conference (TREC 10),* 2001.

[2] Rohini Srihari and Wei Li, Information Extraction Supported Question Answer, *In the Eighth Text REtrieval Conference (TREC 8),* 1999.

[3] Amit Singhal, Steve Abney, Michiel Bacchiani, Michael Collins, Donald Hindle, Fernando Pereira, AT&T at TREC-8, *In the Eighth Text REtrieval Conference (TREC 8),* 1999.

[4] Dan Moldovan, Sanda Harabagiu, Roxana Girju, Paul Morarescu, Finley Lacatusu, Adrian Novischi, Adriana Badulescu and Orest Bolohan, LCC Tools for Question Answering, *In the Eleventh Text REtrieval Conference (TREC 11),* 2002.

[5] Hongbo Xu, Hao Zhang, Shuo Bai, ICT Experiments in TREC-11 QA Main Task, *In the Eleventh Text REtrieval Conference (TREC 11),* 2002.

Clairvoyance Corporation Experiments in the TREC 2003 High Accuracy Retrieval from Documents (HARD) Track

James G. Shanahan, Jeffrey Bennett, David A. Evans, David A. Hull, Jesse Montgomery

Clairvoyance Corporation, Pittsburgh, PA
{jimi, jfb, dae, hull, jmontgomery}@clairvoyancecorp.com

1. Introduction

The Clairvoyance team participated in the HARD Track, submitting fifteen runs. Our experiments focused primarily on exploiting user feedback through clarification forms for query expansion. We made limited use of the genre and related text metadata. Within the clarification form feedback framework, we explored the cluster hypothesis in the context of relevance feedback. The cluster hypothesis states that closely associated documents tend to be relevant to the same requests [Van Rijsbergen, 1979]. With this in mind we investigated the impact on performance of exploiting user feedback on groups of documents (i.e., organizing the top retrieved documents for a query into intuitive groups through agglomerative clustering or document-centric clustering), as an alternative to a ranked list of titles. This forms the basis for a new blind feedback mechanism (used to expand queries) based upon clusters of documents, as an alternative to blind feedback based upon taking the top *N* ranked documents, an approach that is commonly used.

Though our submitted results suffered from incorrect reference statistics in the baseline run the cluster hypothesis was validated; feedback through our cluster-based clarification form yielded a 20% improvement for mean average precision over blind feedback. The cluster hypothesis was further validated, in a somewhat ideal setting, when expansion was performed automatically using the optimal cluster that was selected using a post-hoc analysis. Here, the boost in performance over blind feedback is 20% and is comparable to the TREC max for this track. Ongoing work is investigating techniques that would automatically select the optimal cluster(s).

2. Approach to High Accuracy Retrieval from Documents (HARD) Task

2.1 HARD Corpus

The HARD evaluation corpus is composed of documents from The New York Times (NYT), Associated Press Worldstream (APW), Xinghua English (XIE),The Congressional Record (CR), and Federal Register (FR). We merged all corpora to form one large corpus over which global statistics, such as inverse document frequency (IDF), were computed. The CLARIT system is based on passage retrieval. Passages are defined at indexing time and are commonly referred to as sub-documents (or SubDocs). The sub-document size is fully configurable, with the default setting (used here) producing passages that range in size from 8 to 20 sentences. The typical default sub-document size is 12 sentences. Using this process, we split the documents in this evaluation corpus into passage-size sub-documents.

2.2 Query Formulation and Blind Feedback

A HARD topic is composed of a *title* field, a *description* field and other metadata. From the other meta-data, we considered only *genre* and *related text* in our experiments. To form a query for a topic we merged the title and description fields. We extracted terms from the query text using the Clarit system natural language processing, giving morphologically-normalized words, phrases, and sub-phrases as index terms [Evans and Lefferts, 1995]. Each query term, *t*, is associated with a weight calculated as follows:

$$Weight(t) = TF(t) * IDF(t) * coefficient(t)$$

where the *coefficient(t)* value is set to 1 and *TF*, the term frequency, is defined as follows:

$$TF(t) = 0.5 + 0.5 * TermFreq(t)$$

where *TermFreq(t)* denotes the number of times the term *t* occurs in the query.

The IDF term, corresponding to the inverse document frequency, is defined as follows:

$$IDF(t) = 1 + \log\left(\frac{SubDocCount}{SubDocCount_t}\right)$$

Where *SubDocCount* is the number of sub-documents in the corpus and *SubDocCount$_t$* corresponds to the number of sub-documents in the corpus that contain the term *t*.

We submit this query to our Clarit retrieval engine and get a ranked list of sub-documents based upon the score between each sub-document and the query. This sub-document score is calculated as follows:

$$Score(SDoc, Query) = \sum_{t \in Query} TF_{SDoc}(t) * IDF(t)^2 * coefficient(t) * TF_{Query}(t)$$

This ranked list of sub-documents is post processed such that sub-documents belonging to a single document are reduced to the original document and its score is set to the score of the highest scoring sub-document.

Subsequently, blind feedback can be used to expand the original query automatically using terms extracted from the top *C* ranked documents. Blind feedback has been shown to improved ah-hoc retrieval performance [Evans and Lefferts, 1995]. A similar process can be used for supervised/directed feedback. Terms are extracted from all sub-documents that score at or above the *C*-th document score. This may lead to using multiple sub-documents from the same document. Extraction of terms from these top sub-documents is performed using Clarit NLP. Term selection is performed using the following steps. Terms are ranked in decreasing order using the *Prob2* weighting scheme, which is defined as follows:

$$Prob2(t) = log(R_t + 1)\ x \left(log(\frac{N - R + 2}{N_t - R_t + 1} - 1) - log(\frac{R + 1}{R_t} - 1) \right)$$

where *N* is the number of sub-documents in the reference corpus, and *N$_t$* is the distribution of *t* in the corpus (i.e., the number of sub-documents that contain the term *t* in the corpus). Similarly, *R* is the number of sub-documents in the top *C* documents, and *R$_t$* is the distribution of the term in the top *C* documents (i.e., the number of sub-documents that contain the term *t* in the top *C* documents). The top *k* terms (highest Prob2 weighted terms), known as the expanded set, are appended to the original query. We set term coefficients in the expanded query as follows:

- terms that occur in both the expanded set and the original query are set to 1.5;
- terms that occur in the query only are set to 1.0; and
- terms that occur in the expanded set only are set to 0.5.

In all our experiments, unless otherwise noted, we set *C* to 6 (documents) and *k* to 30 (terms).

In our post-TREC experiments, we set the coefficients of query terms and new terms through a term normalization algorithm:

- the coefficient of terms that occur in the expanded set and in the original query are set to *(1 * boostFactor) + normalisedProb2*;
- terms that occur in the query only are set to 1.0; and
- terms that occur in the expanded set are set to *normalisedProb2*.

For our experiments, *boostFactor* was set to 2. The *normalisedProb2* factor is calculated as follows:

$$normalisedProb2(t) = \frac{Prob2(t)}{MaxProb2}$$

2.3 Clarification Forms

We explored two types of clarification forms. Both forms presented documents in groups derived from clustering. The first form, called as the *title-based form*, presents the top ranking documents for a query in groups. These groups are formed by clustering the top 100 ranked results using a simple clustering strategy outlined below. Each group is presented using a list of terms, corresponding to typical terms for the group, and a list of documents, where each document is represented using its title and the information source (very similar to the forms used in Scatter-Gather [Hearst and Pederson, 1995]). A clarification form of this type corresponding to Topic 33 is depicted in Figure 1.

The user is presented with ten groups of documents for each topic, where each group consists of five documents: the seed document and its four nearest neighbors. The seed for the first group is set to the top ranking document. This seed is then used to rank the top 100 documents. The top 4 ranking documents, in addition to the seed document, comprise the first group. Other groups are formed in a similar way, where the seed is set to be the top-most ranking document that has not already been used as a seed or as a nearest neighbor.

The user is asked to judge the relevance of each group for a query as being "*On Topic*", "*Not on Topic*", "*Unsure*" or "*Unjudged*". The "*On Topic*" option denotes that at least one document or some of the terms in the group are on topic. The "*Not on Topic*" option means that the user believes none of the group documents or group terms represent the topic well. If uncertain whether the group accurately represents the topic, the user could choose "*Unsure*." And if the user runs out of time or simply fails to judge a group, the default value for a group is "*Unjudged*".

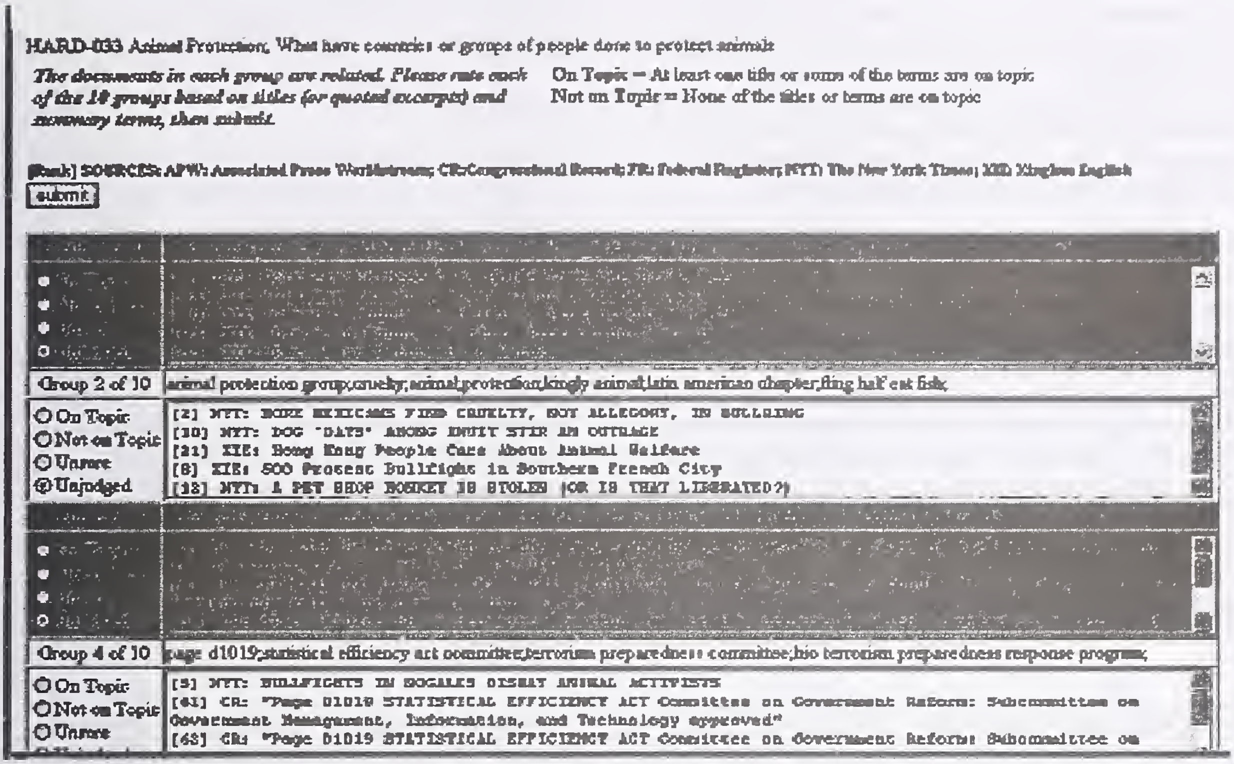
The system uses the group judgments, as directed feedback, to expand the query with terms extracted from the constituent documents of the positively assessed groups. We add all documents from groups that are marked "*On Topic*" to the feedback pool. If the number of groups marked as "*On Topic*" is less than two, we also include the "*Unsure*" groups in the feedback pool. If there are still fewer than two groups, we use the default blind feedback strategy (of 6 documents and 30 terms). If there are enough documents in the feedback pool, we expand the original query using the expansion strategy defined above, where *C* is set to the number of documents in the pool.

We subsequently perform a retrieval over the entire corpus using the expanded query. We post-process this ranked list by front-loading (promoting to the top of the list) all documents in the feedback pool, and excluding documents from groups marked as "*Not On Topic*".

In the second form documents are again organized into groups. Here, each group is represented using a list of terms, corresponding to typical terms from the documents that make up the group. The groups in this form are created by clustering the top-ranking 100 documents for the original query (using a version of the agglomerative average-link clustering algorithm). The terms are extracted from the documents in each cluster using Clarit NLP and the top forty terms (determined by their Prob2 weight) are listed as representative terms for the cluster.

Once again, the user is asked to judge the relevance of each group for the query as being "*Clearly Related*", "*Somewhat Related*", "*Not Related*", "*Unsure*", or "*Unjudged*". The "*Clearly Related*" option denotes that some of the terms in the group were strongly representative of the topic. The "*Somewhat Related*" option signifies that some of the terms in the group were somewhat representative of the topic. The "*Not on Topic*" option indicates that none of the terms was on topic. The other options are self-explanatory.

The system uses the group judgments as a from of directed feedback to expand the query with terms extracted from the constituent documents of the positively assessed groups. We add all documents from groups that are marked "*Clearly Related* " to the feedback pool. If the number of groups marked as "*Clearly Related*" is less than two, we also include the "*Somewhat Related*" groups in the feedback pool. If there are still fewer than two groups, we use the default blind feedback strategy to expand the query. If we have enough documents in the feedback pool, we expand the original query using the expansion strategy defined above, where *C* is set to the number of documents in the pool.



HARD-033 Animal Protection; What have countries or groups of people done to protect animals

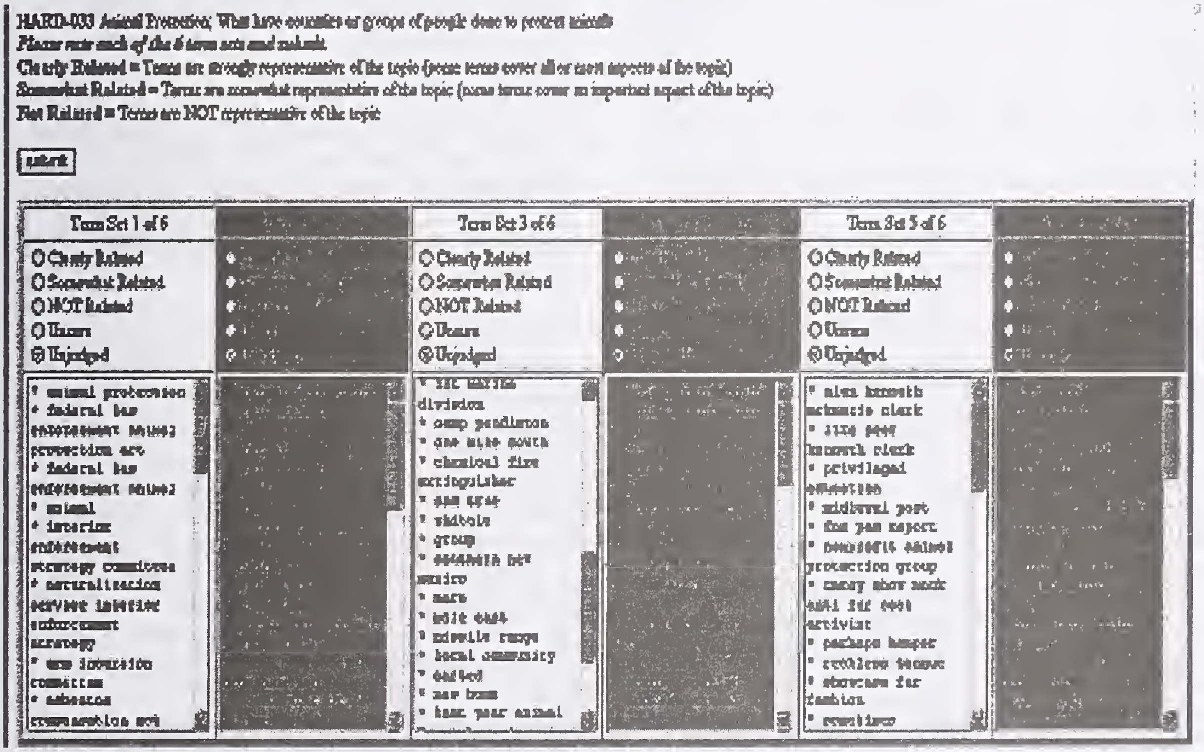
The documents in each group are related. Please rate each of the 10 groups based on titles (or quoted excerpts) and summary terms, then submit.

On Topic = At least one title or some of the terms are on topic
Not on Topic = None of the titles or terms are on topic

[Rank] SOURCES: APW: Associated Press Worldstream; CR: Congressional Record; FR: Federal Register; NYT: The New York Times; XIE: Xinghua English

Group 2 of 10	animal protection group;cruelty;animal protection;kingly animal;latin american chapter;fling half eat fish;
○ On Topic ○ Not on Topic ○ Unsure ◉ Unjudged	[2] NYT: MORE MEXICANS FIND CRUELTY, NOT ALLEGORY, IN BULLRING [10] NYT: DOG "DAYS" AMONG INUIT STIR AN OUTRAGE [21] XIE: Hong Kong People Care About Animal Welfare [8] XIE: 500 Protest Bullfight in Southern French City [13] NYT: A PET SHOP MONKEY IS STOLEN (OR IS THAT LIBERATED?)
Group 4 of 10	page d1019;statistical efficiency act committee;terrorism preparedness committee;bio terrorism preparedness response program;
○ On Topic ○ Not on Topic ○ Unsure	[5] NYT: BULLFIGHTS IN NOGALES DISMAY ANIMAL ACTIVISTS [61] CR: "Page D1019 STATISTICAL EFFICIENCY ACT Committee on Government Reform: Subcommittee on Government Management, Information, and Technology approved" [63] CR: "Page D1019 STATISTICAL EFFICIENCY ACT Committee on Government Reform: Subcommittee on

Figure 1. An example of a title-based clarification form using typical terms and document titles.



HARD-033 Animal Protection; What have countries or groups of people done to protect animals

Please rate each of the 6 term sets and submit.

Clearly Related = Terms are strongly representative of the topic (some terms cover all or most aspects of the topic)
Somewhat Related = Terms are somewhat representative of the topic (some terms cover an important aspect of the topic)
Not Related = Terms are NOT representative of the topic

Term Set 1 of 6		Term Set 3 of 6		Term Set 5 of 6	
○ Clearly Related ○ Somewhat Related ○ NOT Related ○ Unsure ◉ Unjudged		○ Clearly Related ○ Somewhat Related ○ NOT Related ○ Unsure ◉ Unjudged		○ Clearly Related ○ Somewhat Related ○ NOT Related ○ Unsure ◉ Unjudged	
* animal protection * federal law enforcement animal protection act * federal law enforcement animal * animal * interior enforcement strategy committee * naturalization service interior enforcement strategy * [illegible] committee * asbestos compensation act		* 1st marine division * camp pendleton * one mile south * chemical fire extinguisher * [illegible] * [illegible] * group * [illegible] new mexico * [illegible] * [illegible] * missile range * local community * [illegible] * [illegible] * last year animal		* alan kenneth mckenzie clark * [illegible] kenneth clark * privileged education * midlevel port * [illegible] report * nonprofit animal protection group * [illegible] anti fur [illegible] activist * perhaps longer * ruthless [illegible] * showcase for fashion * [illegible]	

Figure 2. An example of a term based clarification form.

We subsequently perform a retrieval over the entire corpus using the expanded query. We post-process this ranked list by upgrading all documents that occur in groups marked as "*Clearly Related*" to the front of the ranked list, while documents that are members of groups marked as "*Not Related*" are excluded from the top 1000 documents list.

2.4 Exploiting Topic Metadata

We selectively used two pieces of topic metadata: genre and related text. The genre metadata field introduces a strong source-specific bias for individual topics. We factor in the topic genre with a post-processing step to re-rank the retrieval results. Essentially, we assign a weight factor ranging from 0 to 1 for each genre/source pair. The weight factor is a rough measure of the relative likelihood that a document from that source would be relevant to the given topic genre. This weight is multiplied by the final retrieval score and the documents are re-ranked accordingly. The final weight table is given below:

Genre\Source	APW	CR	FR	NYT	XIE
Any	1.0	1.0	1.0	1.0	1.0
Administrative	0.2	1.0	1.0	0.2	0.2
I-Reaction	0.5	0.2	0.2	0.5	1.0
Overview	1.0	0.2	0.2	1.0	1.0
Reaction	1.0	0.2	0.2	1.0	1.0

It might have been possible to learn these weights automatically given training data, but we decided that there were not enough training data available to make this a productive exercise.

We used the related text to expand the original query (i.e., we concatenated it to the title and description data). For some experiments, where the related text contained URLs, we fetched the content of the page pointed to by the URL and concatenated that to the original query also.

3. Results

Table 1 presents a list of the experiments we carried out. Each row in this table denotes an experiment and how the original query was constructed (using related text or downloaded related web pages that were pointed to in the related text data), and whether any genre post-processing was done. The experiments labeled *NewBase630*, *BestClusterRun*, and *BestGroupRun* correspond to experiments that were performed after the submission deadline.

Table 2 presents the results of our experiments in terms of mean average precision, exact precision, and precision at 10.

Overall, our submitted runs suffered from incorrect reference statistics in the baseline run, which was also used to generate both clarification forms. The mean average precision (MAP) over the top 1000 documents for this baseline submission was 0.24, denoted as experiment *CLSTD630* in Table 2. This corresponds to the median results for all systems in this track. This baseline run has since been improved to 0.31 (experiment *NewBase630*). Figure 3 presents a comparison of TREC Median and Maximum to our submitted baseline run and our post-TREC baseline run (*NewBase630*).

When we incorporated feedback from the title-based form, the MAP improved to 0.28 from 0.24. The term-based form did not yield any significant improvement. Figure 4 presents a comparison of the TREC Median and Maximum to our submitted baseline run and our best submitted run (*CLAI1NG*).

Our approaches to exploiting the topic metadata (genre and related text) do not yield any improvement. Table 3 presents passage level evaluation results. Our passage level results reflect our document level results in terms of the evaluation measures.

We performed various follow-up experiments where we exploited the cluster hypothesis. Here, our baseline performance corresponded to choosing a single best-performing group/cluster for each topic automatically. When documents were grouped using agglomerative clustering, and the best performing cluster was selected (post-hoc), we attain an overall MAP of 0.37 (*BestClusterRun*). Similarly, when grouping using single-ranked documents (as was used in generating the groups in the title-based form), using the best performing cluster selected, we attain an overall MAP of 0.37 (*BestGroupRun*). These experiments compare very favorably to the TREC maximum MAP of 0.40 for this track.

Submission name	Genre (MetaData)	Related Text (MetaData)	Related Text (Web Pages)
CLAI1NG	No	No	No
CLAI1G	Yes	No	No
CLAI2NG	No	No	No
CLAI2G	Yes	No	No
CLAI2RTNG	No	Yes	No
CLAI2RTG	Yes	Yes	No
CLAI2WRTNG	No	Yes	Yes
CLAI2WRTG	Yes	Yes	Yes
CLAISTDNG	No	No	No
CLAISTDG	Yes	No	No
CLAISTDRTG	Yes	Yes	No
CLAISTDRTNG	No	Yes	No
CLAISTDWRTG	Yes	Yes	Yes
CLAISTDWRTNG	No	Yes	Yes
CLSTD630	No	No	No
NewBase630	No	No	No
BestClusterRun	No	No	No
BestGroupRun	No	No	No

Table 1. Submission and post-submission experiment details. (All runs used Title+Description fields as the query.)

Experiment	Avg Prec	R-Prec	Prec @ 10
CLAI1G	0.2884	0.3229	0.4729
CLAI1NG	0.2917	0.3246	0.4729
CLAI2G	0.2499	0.2861	0.4188
CLAI2NG	0.2514	0.2870	0.4146
CLAI2RTG	0.2218	0.2634	0.4125
CLAI2RTNG	0.2225	0.2633	0.4083
CLAI2WRTG	0.2138	0.2533	0.4104
CLAI2WRTNG	0.2170	0.2569	0.4063
CLAISTDG	0.2309	0.2658	0.3750
CLAISTDNG	0.2345	0.2712	0.3917
CLAISTDRTG	0.2334	0.2686	0.3979
CLAISTDRTNG	0.2285	0.2618	0.4000
CLAISTDWRTG	0.2134	0.2468	0.3729
CLAISTDWRTNG	0.2105	0.2420	0.3646
CLSTD630	0.2341	0.2772	0.3938
NewBase630 (post-TREC run)	0.3069	n/a	n/a
BestClusterRun (post-TREC run)	0.3727	n/a	n/a
BestGroupRun (post-TREC run)	0.3741	n/a	n/a
TREC median	0.2841	0.2994	0.4729
TREC max	0.4069	0.4250	0.6500

Table 2: Document Level Evaluation Results.

Run	F @ 30	R-Prec	Prec @ 10
CLAI1G	0.0905	0.2426	0.3235
CLAI1NG	0.0851	0.2266	0.2886
CLAI2G	0.0814	0.1900	0.2538
CLAI2NG	0.0781	0.1815	0.2201
CLAI2RTG	0.0762	0.1773	0.2507
CLAI2RTNG	0.0728	0.1708	0.2176
CLAI2WRTG	0.0761	0.1733	0.2483
CLAI2WRTNG	0.0728	0.1704	0.2142
CLAISTDG	0.0738	0.1799	0.2230
CLAISTDNG	0.0733	0.1737	0.2056
CLAISTDRTG	0.0826	0.2076	0.2610
CLAISTDRTNG	0.0745	0.1782	0.2246
CLAISTDWRTG	0.0822	0.1963	0.2582
CLAISTDWRTNG	0.0744	0.1700	0.2243
CLSTD630	0.0700	0.1726	0.2100
BestClusterRun (post-TREC run)	n/a	n/a	n/a
BestGroupRun (post-TREC run)	n/a	n/a	n/a
TREC med	0.1000	0.1794	0.2574
TREC max	0.1738	0.3195	0.3973

Table 3: Passage level evaluation results.

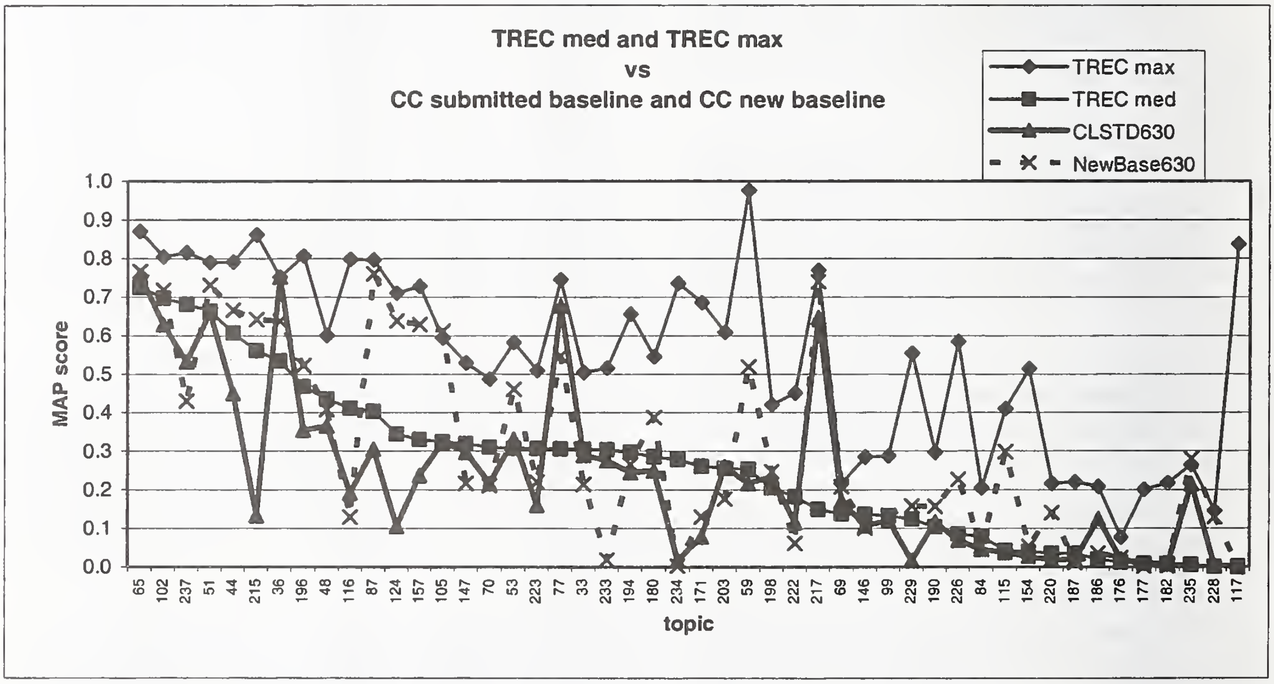


Figure 3: Comparison of TREC Median and Maximum to Clairvoyance's submitted baseline run and Clairvoyance's post-TREC baseline run.

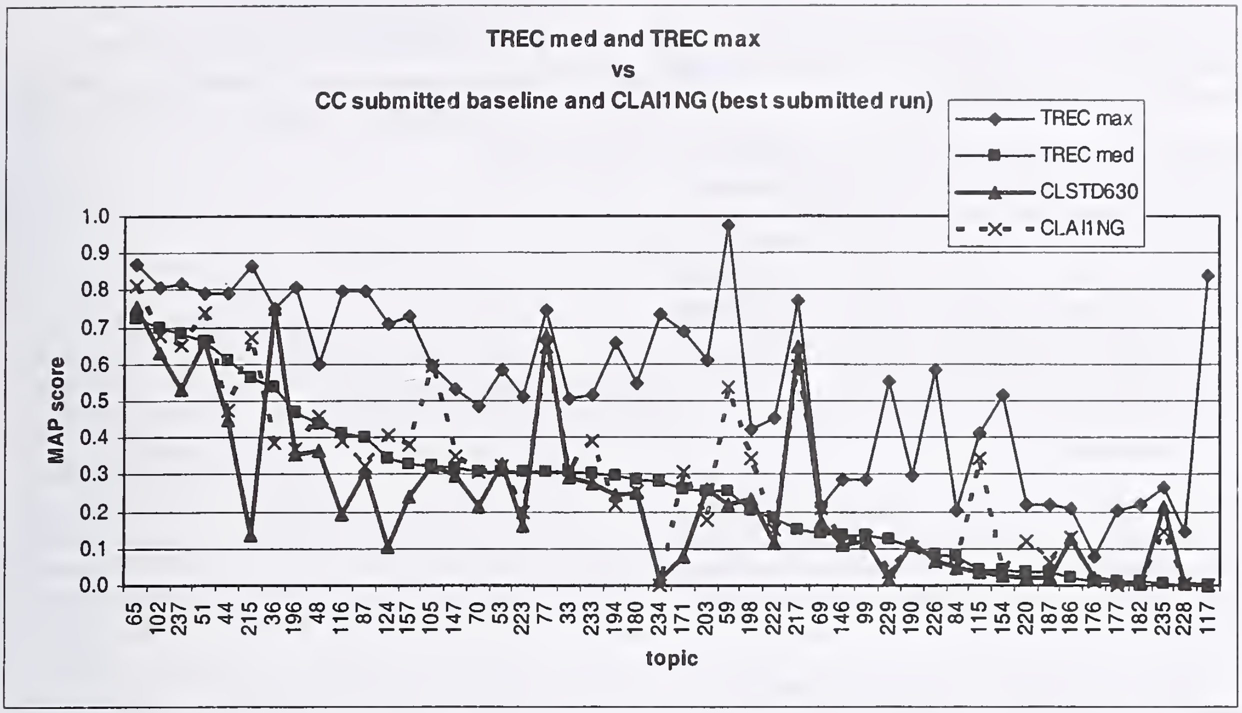


Figure 4: Comparison of TREC Median and Maximum to Clairvoyance's submitted baseline run and Clairvoyance's best submitted run (CLAI1NG).

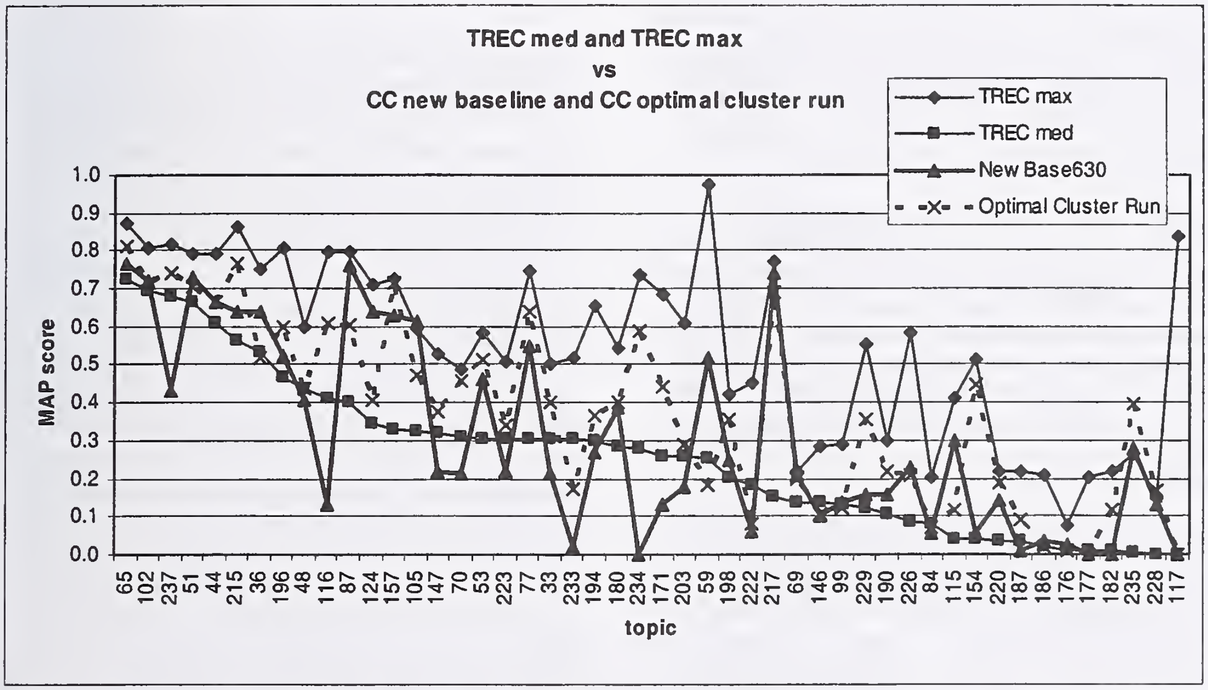


Figure 5: Comparison of TREC Median and Maximum to Clairvoyance's Post-TREC Baseline Run and Clairvoyance's Post-TREC Optimal Cluster Run.

Though we have investigated a number of approaches to automatically selecting a group(s), none of our examined approaches seems to provide consistent results across all topics.

Figure 5 presents a comparison of the TREC Median and Maximum to our post-TREC baseline run and our post-TREC optimal cluster run. It is important to note that this 0.37 MAP performance comes from selecting the best group. Performance can be potentially improved by incorporating more than one group as feedback. We are currently exploring other approached on how to select these groups for feedback automatically.

3. Conclusions

For our HARD experiments we explored the cluster hypothesis in the context of manual feedback through clarification forms and automatic feedback. We compared both of these approaches to blind feedback.

Though our submitted results suffered from incorrect reference statistics in the baseline run, the cluster hypothesis was validated; feedback through our cluster-based (title) form yielded a 20% improvement for mean average precision over blind feedback. While we have demonstrated the benefits of manually selecting the optimal clusters for feedback, a better comparison would be to have the user manually select documents from a ranked list of titles. This could potentially provide comparable results without the burden of clustering.

The cluster hypothesis was also validated, in a somewhat ideal setting, when expansion was performed automatically using the optimal group that was selected using a post-hoc analysis. Here, the boost in performance over blind feedback is 20% and is comparable to the TREC max for this track. Our continuing work is investigating techniques that would automatically select the optimal cluster(s).

References

[1] Evans, David.A., and Robert G. Lefferts. CLARIT–TREC Experiments. Information Processing and Management, Vol.31, No.3, pp.385–395, 1995.

[2] Hearst, Marti A., and Jan O. Pedersen. Reexamining the Cluster Hypothesis: Scatter/Gather on Retrieval Results, Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval.

[3] Van Rijsbergen, C. J. Information Retrieval, Butterworths, London, Second Edition, 1979.

Use of Metadata for Question Answering and Novelty Tasks

Kenneth C. Litkowski
CL Research
9208 Gue Road
Damascus, MD 20872
ken@clres.com

Abstract

CL Research's question-answering system for TREC 2003 was modified away from reliance on database technology to the core underlying technology of using massive XML-tagging for processing both questions and documents. This core technology was then extended to participate in the novelty task. This technology provides many opportuinities for experimenting with various approaches to question answering and novelty determination.

For the QA track, we submitted one run and our overall main task score was 0.075, with scores of 0.070 for factoid questions, 0.000 for list questions, and 0.160 for definition questions. For the passage task, we submitted two runs, our better score was 0.119 for the factoid questions. These scores were all considerably below the medians for these tasks. We have implemented further routines since our official submission, improving our scores to 0.18 and 0.23 for the exact answer and passages tasks, respectively. For the Novelty track, we submitted four runs for task 1, one run for task 2, five runs for task 3, and one run for task 4; our submissions for tasks 2 and 4 were identical. For task 1, our best run received an F-score of 0.483 for relevant sentences and 0.410 for new sentences. For task 2, our F-score was 0.788 for new sentences. For task 3, our best F-score was 0.558 for relevant sentences and 0.419 for new sentences. For task 4, our F-score was 0.655 for new sentences. On average, our F-scores were somewhat above the medians on all tasks. We describe our system and examine our results from the perspective of exploiting the metadata in the XML tags.

1 Introduction

In TREC 2002, CL Research examined the potential of using XML-tagged documents for question answering (Litkowski, 2003a). In particular, we saw how the use of hand-developed XPath expressions could obtain extremely good results when compared with the best sytems. However, as noted, the challenge was the automatic creation of XPath expressions. While this was the focus of our participation in the TREC 2003 questions answering (QA) track, this effort was performed in the context of creating a new Knowledge Management System (KMS) from the GUI interface we used in last year's explorations. KMS was first used in CL Research's participation in text summarization in the 2003 Document Understanding Conference (Litkowski, 2003b). KMS is continuing to evolve as we explore the possibilities of using XML-tagging to perform various natural language processing (NLP) tasks.

Extensible Markup Language (XML) provides a natural mechanism for representing texts, from small snippets like titles through extensive collections of texts. A valid XML document is a tree and we can readily design our entire representation on this tree structure. We can create XML representations for questions, topic descriptions (such as used in the Novelty task), individual documents, and collections of documents. Generally, the representations create nodes for sentences, clauses, phrases, and words. Each node in the tree will generally have associated attribute names and values. A major challenge is determining an appropriate set of metadata (tags, attributes, and values) for different NLP tasks. This paper describes some of our findings emerging from CL Research's participation in TREC.

A key part of the XML design philosophy is the ability to transform an XML file into usable output for display or other purposes (e.g., populating a database). This is accomplished via XML stylesheet language transformations (XSLT). XSLT is based on the creation of XPath expressions, which specify the path from the top of the XML tree to some intermediate or leaf node. XPath expressions are useful in QA, since they provide a simple mechanism for homing in on answers. Just as important, we have found it essential to be able to move about in the XML tree from a particular node, based on relations of one node to others (e.g., moving to a clause containing a noun phrase).

Section 2 presents the TREC QA and Novelty problem descriptions. Section 3 describes the KMS, specifically components for processing texts and for performing particular NLP tasks. Section 4 provides our question answering results, particularly our experience in handling different types of questions. Section 5 describes our novelty experiments, particularly identifying insights about the nature of the task as they emerged. Section 6 describes anticipated next steps for improving the question-answering capability and for using XML-tagged documents in other applications such as information extraction, text summarization, novelty studies, and investigation of linguistic phenomena.

2 Problem Description

The TREC 2003 QA and Novelty tasks used the AQUAINT Corpus of English News Text on two CD-ROMs, (about one million documents), containing documents from *Associated Press Newswire*, *New York Times Newswire*, and *Xinhua News Agency*. These documents were stored with SGML formatting tags (XML compliant).

For the QA track, participants were provided with 500 unseen questions to be answered in a "main" task from the corpus. Participants were given the option of using their own search engine or of using the results of a "generic" search engine. CL Research chose the latter, relying on the top 50 documents retrieved by the search engine. These top documents were provided simultaneously with the questions. The 500 questions included a type: factoid (417, requiring a short, exact answer), list (37, requiring a list of answers), and definition (50, requiring a set of core elements and/or acceptable peripheral information concerning the term to be defined). An additional task, using only the 413 factoid questions, was to submit "passages" containing the answer.

Participants in the main task were required to answer the 413 factoid questions with a single exact answer, containing no extraneous information and supported by a document in the corpus. A valid answer could be NIL, indicating that there was no answer in the document set; NIST included 30 questions for which no answer exists in the collection. For these questions, NIST evaluators next judged whether an answer was correct, inexact, unsupported, or incorrect. The submissions were then scored as percent of correct answers. For the list questions, participants returned a set of answers (e.g., a list of chewing gums); submissions were given F-scores, measuring recall of the possible set of answers and the precision of the answers returned. For definitions questions ("Who is Vlad the Impaler" or "What are fractals"), participants provided a set of answers. These answer sets were also scored with an F-score, measuring whether the answer set contained certain "vital" information and how efficiently peripheral information was captured (based on answer lengths).

CL Research submitted one run for the main task and two runs for the passages task.

For the Novelty track, participants were provided with descriptions of 50 topics (labeled as "event" or "opinion") and a set of 25 documents relevant to each topic. These documents were further broken down into sentences. There were four tasks. For the first task, participants were to identify sentences relevant to the topic and then to identify which of these sentences provided novel or new information. For task 2, participants were given the relevant sentences from all documents and asked to identify those which were new. For task 3, participants were provided with the relevant and new sentences from the first five documents and asked to identify the relevant and new sentences for the remaining 20 documents. For task 4, participants were given the relevant sentences from all documents and the new sentences from the first 5 documents and asked to identify the new sentences in the last 20 documents. The tasks were spread out over a three week period, with participants given additional information at the end of the first week.

CL Research submitted four runs for task 1 (using different amounts of information from the topic description and with a different threshhold for judging relevance), one run for task 2, five runs for task 3 (using different threshholds for taking into account information in the identified relevant sentences), and one run for task 4.

3 The Knowledge Management System

The CL Research KMS consists of two major components, a tasking component and a text processing component. The tasking component sets up the tasks to be performed based on the NLP task (such as question answering, text summarization, or novelty detection). The text processing component processes documents into an XML representation. For question answering, the tasking component will parse and process a question into an XML representation, invoke the text processing component to handle any documents that might contain the answers, and answers the question. For novelty detection, the tasking component will parse and process the title, the description, and the narrative into an XML representation, invoke the text processing component for relevant documents, and then process the sentences in the documents against the topic description to identify relevant and novel sentences.

The text processing component consists of three elements: (1) a sentence splitter that separates the source documents into individual sentences; (2) a parser which takes each sentence and parses it, resulting in a parse tree containing the constituents of the sentence; and (3) a parse tree analyzer that identifies important discourse constituents (sentences and clauses, discourse entities, verbs and prepositions) and creates an XML-tagged version of the document. The remainder of this section details these elements, focusing on document processing; these same elements are used in parsing and processing questions and topic descriptions into XML representations.

3.1 Sentence Splitting

Sentence splitting proceeds as described in previous years (Litkowski, 2002a; Litkowski, 2001). For TREC 2003, we were able to process the full set of 50 documents for each question quite rapidly, unlike previous years where we were more limited in speed by the reliance on database technology, where processing speed slowed exponentially as the database grew. Instead, processing grew linearly with the size of the document collection. Overall, we processed 25,000 documents from which 724,165 sentences were identified and presented to the parser. Thus, we processed an average of 29.0 sentences per document (compared to 25.7 in TREC 2002, 22.8 in TREC-10, 28.9 in TREC-9 and 31.9 in TREC-8). Overall, we had an average of 1448 sentences to consider for each question. For the novelty task, only 1250 documents were processed, at 25 documents for each of 50 topics.

Our sentence splitter has remained largely the same, but some improvements have been made, primarily in the recognition of abbreviations and initials associated with a name. In previous years, this had resulted in improper splitting. In TREC 2003, we observed that the documents for the novelty task, which were provided already split into "sentences", had many sentences improperly split. We rejoined these sentences prior to processing these texts.

3.2 Parser

We continued our use of the Proximity parser, described in more detail in our previous papers (Litkowski, 2002a; Litkowski, 2001). As described there, the parser output consists of bracketed parse trees, with leaf nodes describing the part of speech and lexical entry for each sentence word. Annotations, such as number and tense information, may be included at any node. Usable output was generated by the parser for 99.9 percent of the sentences that were processed.

3.3 Sentence and Discourse Analysis

The sentence parsing in KMS is part of a broader system designed to provide a discourse analysis of an entire text. After each sentence is identified and parsed, its parse tree is traversed in a depth-first recursive function. During this traversal, each non-terminal and terminal node is analyzed, making use of parse tree annotations and other functions and lexical resources that provide "semantic" interpretations of syntactic properties and lexical information.

At the top node in the tree, just prior to iteration over its immediate children, the principal discourse analysis steps are performed. Each sentence is treated as an "event" and added to a list of events that constitute the discourse. We first update data structures used for anaphora resolution. Next, we perform a quick traversal of the parse tree to identify discourse markers (e.g., subordinating conjunctions, relative clause boundaries, and discourse punctuation) and break the sentence down into elementary discourse units. We also identify and maintain a list of the sentence's verbs at this stage, to serve as the bearers of the event for each discourse unit.

After the initial discourse analysis, the focal points in the traversal of the parse tree are the noun phrases. When a noun phrase is encountered, its constituents are examined and its relationship to other sentence constituents are determined. Each noun phrase is added to a list of discourse entities for the entire text, that is, a "history" list. As each noun phrase is encountered, it is compared to discourse entities already on the history list. This comparison first looks for a prior mention, in whole or in part, to determine whether the new entity is a coreferent of a previous entity (particularly valuable for named entities). If the new entity is an anaphor, an anaphoric resolution module is invoked to establish the antecedent. A similar effort is made to find antecedents for definite noun phrases. The noun phrase's constituents are examined for numbers, adjective sequences, possessives (which are also subjected to the anaphoric resolution module), genitive determiners (which are made into separate discourse entities), leading noun sequences, ordinals, and time phrases. Finally, an attempt is made to assign a semantic type to the head noun of the phrase using WordNet or an integrated machine-readable dictionary or thesaurus.

If a noun phrase is part of a prepositional phrase, a special preposition dictionary is invoked in an attempt to disambiguate the preposition and identify its semantic type. This module identifies the attachment point of the preposition and uses information about the syntactic and semantic characteristics of the attachment point and the prepositional object for this disambiguation. The preposition "definitions" in this dictionary are actually function calls that check for such things as literals and hypernymy relations in WordNet. A list of all prepositions encountered in the text is maintained as the text is processed. (See Litkowski (2002b) for further details.)

Predicative adjective phrases, relative clauses, subordinate clauses, and appositives are also flagged as the parse tree is traversed. The attachment points and spans of relative clauses and appositives are noted.

As indicated above, the text analysis module develops four lists at the same time as the semantic relation triples: (1) events (the discourse segments), (2) entities (the discourse entities), (3) verbs, and (3) semantic relations (the prepositions). Each document consists of one or more tagged segments, which may include nested segments. Each discourse entity, verb, and preposition in each segment is then tagged. A segment may also contain untagged text, such as adverbs and punctuation. Each item on each list has an identification number (used in many of the functions of the text analysis module). As indicated above, the discourse analysis assigns attributes to each segment (and subsegment), discourse entity, verb, and preposition.

For segments, the attributes include the sentence number (if the segment is the full sentence), a list of subsegments (if any), the parent segment (if a subsegment), the text of the segment, the discourse markers in the sentence, and a type (e.g., a "definition" sentence or "appositive"). For discourse entities, the attributes include its segment, position in the sentence, syntactic role (subject, object, prepositional object), syntactic characteristics (number, gender, and person), type (anaphor, definite or indefinite), semantic type (such as person, location, or organization), coreferent (if it appeared earlier in the document), whether the noun phrase includes a number or an ordinal, antecedent (for definite noun phrases and anaphors), and a tag indicating the type of question it may answer (such as who, when, where, how many, and how much). For verbs, the attributes include its segment, position in the sentence, the subcategorization type (from a set of 30 types), its arguments, its base form (when inflected), and its grammatical role (when used as an adjective). For prepositions, the attributes include its segment, the type of semantic relation it instantiates (based on disambiguation of the preposition) and its arguments (both the prepositional object and the attachment point of the prepositional phrase).

After all sentences in a document have been processed, the four lists are used to create an XML-tagged version of the document. The XML tagging is performed for each segment within the XML element **segment**, with the attributes listed in the tag opening. The tag content is initialized to the segment text and we proceed to mark up this text according to the text contained within each subsegment, discourse entity (**discent**), verb (**verb**), and preposition (**semrel**) in the segment. As these XML elements are generated, their attributes are added to the tag opening.

The resultant XML-tagged text for individual documents were combined into one overall file of documents, each with a tag for the document number. For TREC, there was an XML file for each of the 500 questions in the QA track and for each of the 50 topics in the novelty track. As of the submission date for the QA track, XML tagging results in nearly a sevenfold expansion of the documents. The 94 MBs of the top 50 documents generated 632 MBs of XML-tagged text.

The tagging process is in continual development. Improvements arise in the first instance from detecting bugs in the parser, extracting more information from the parse results, and detecting bugs in the creation of the lists for sentences, clauses, noun phrases, verbs, and prepostions. In the second instance, improvements arise from improved use of lexical resources for characterizing the various elements. Finally, improvements occur as the system is expanded to cover more linguistic phenomena, particularly in characterizing semantic relations between various discourse elements.

4 Question-Answering Using XML-Tagged Documents

As described for TREC 2002 (Litkowski, 2003a), question-answering against the XML files essentially involves describing a path (XPath) from the top of the document tree (in this case, a file of 50 TREC documents) to a discourse entity (in our case, to a **discent** node) which is returned as the answer. In TREC 2002, we demonstrated that the desired **discent** nodes were present for almost all the questions, with the residual few cases not present because of parsing or processing bugs. Further, we showed that an XPath expression could be developed by hand to extract these answers to a degree better than the best performing system. (This is slightly misleading since the measure used in TREC 2002 was a confidence weighted score that was higher than simply the percentage of correct answers, as used in TREC 2003.) We concluded our discussion last year by indicating that the challenge was the automatic creation of XPath expressions.

As we proceeded into the development of the functionality for XPath expression creation for TREC 2003, we found it necessary to situate the use of the XPath expressions into a broader set of routines. Generally, the overall architecture for answering questions consists of four components: (1) question analysis, (2) creating of a broad XPath expression that will retrieve appropriate sentences, (3) appending to this basic XPath expression a specific XPath expression request for discourse entities (with routines specific to question type), and (4) retrieving candidate answers using the XPath expression and submitting these answers to a post-processng routine (also question-type specific) to evaluate the answers. We have found that some looping through these components may be necessary, based on results that are obtained. For the most part, this involves revising the specific XPath expression, but some looping may also occur in evaluating answers when a pass through the candidates results in a narrowing and subsequent passes can examine the relative quality of different answers.

The complexity of these processes was unexpected, with the result that our implementation was incomplete at the time of our submission. In particular, we were unable to convert many of the details of our candidate answer evaluations (which are question-type specific) into appropriate processing steps making use of the XPath and XML node processing capabilities. We have since implemented many of these routines. Not surprisingly, the complexity of the routines varies with the question type. Several general principles have emerged. Again, not surprisingly, the principles reflect those developed for question answering in our previous work and in the work of other TREC participants. These prinicples are couched within a more general architecture that implements linguistic and semantic processing, rather than simply implementing pattern recognition routines. Although we have not ahcieved the performance of our hand generation of XPath expressions, we are confident that continued development of the XML mechanisms will serve not only question answering, but also other NLP tasks.

We describe our observations to date in the following areas: (1) the question analysis phase, (2) the creation of the basic XPath expression to retrieve sentences, (3) question-type specific XPath expression creation ,and (4) evaluation of candidate answers.

4.1 Question Analysis

We have grounded our analysis of questions on a rigorous hierarchical analysis of dictionary definitions for question words, specifically for **what**, **which**, **who**, **when**, **where**, **why**, and **how**. The definition for **what** is at the top of the hierarchy, "asking for information specifying something", with **which** being a slight variant, "asking for information specifying one or more people or things from a definite set". The others are defined in terms using the **what** primitive: "what person" (**who**), "at what time" (**when**), "at what location" (**where**), "for what reason" (**why**), and "by what means" (**how**). There are three additional varieties of **how**: **how much** ("what amount or price"), **how many** ("what number"), and **how adj** ("used to ask about the degree or extent of something").

Our question analysis identifies the type of question, either directly according to the question word or by analyzing the question string and the semantics of key discourse entities in the question. Thus, "what year" is converted into a **when** question, "what is the length of np" is converted into a **how adj** question, and "what country" is converted into a **which** question (or equivalently, a **whatNP** question). Sentences that have no question elements (which are given a "qelem" attribute in the XML tagging), such as "List types of chewing gum", are converted into **whatNP** questions. Very simple **what** or **who** questions "Who is Vlad the Impaler?" are converted into **whatIsDef** or **whoIsDef** questons.

4.2 Basic Sentence XPath Expression

Each sentence that is processed in KMS is enclosed in the metadata tag **segment**. Clauses also have this tag, but are distinguished from the full sentence via attributes. A full sentence is assigned an ID number and may have an associated list of subsegments, while a clause has a parent. All questions give rise to an XPath expression that selects sentences and begin with

//segment[@sent].

This will retrieve all sentences in all documents in a file.

We next add further restrictions on what sentences we would like to examine in further detail based on the elements of the XML representation of the question, in particular the discourse entities. At this point, we want to remain fairly generous in keeping sentences, so we look for any sentences containing any of the words in the discourse entities in the question, except for stop words. For example, for question 1401 in TREC 2002, "What is the democratic party symbol?", the basic XPath expression is

//segment[@sent and (contains(.,'democratic') or contains(.,'party') or contains('symbol'))]

Identification of the words to be included in this part of the XPath expression is somewhat involved, but provides a first screen to identify sentences where an answer might be found. When evaluating answers for each question type, we determine whether changing "or" or "and" retrieves any sentences, and allow this most restrictive screen if it returns any sentences.

We are continuing to study alternatives to this basic XPath expression, such as using regular expressions, allowing query expansion using synonyms, and weighting the importance of terms.

4.3 Question-Specific XPath Expressions

As indicated above, each question type has its own specific processing to characterize the discourse entity that is sought in the XPath expression. This generally entails adding to the basic XPath expression a specification for a **discent**, with the following appended:

//discent

In all cases, the attributes of the desired discourse entity are specified via qualifiers. Since we have tagged discourse entities with many attributes during text processing, we can make use of this metadata when adding qualifiers.

For the more "complex" question types (**when**, **where**, **how much**, **how many**, and **how adj**), the specification qualifier for the discourse entity is actually simpler, such as

//discent[@tag='when'], or
//discent[@tag='where'].

For a question like "How many time zones", the qualifier would add the head noun "time zones"

//discent[@tag='howmany' and contains(.,'time zones')]

to obtain a discourse entity that has a numeric quantifier.

For the more "basic" question types, **what**, **whatNP**, and **who**, analysis of the question is necessary to identify appropriate qualifiers. For **who** questions, the discourse entity would be required to have a **semtype** attribute with a value of "person". When a question asks for the "first" or the "biggest", the qualifier would require that the discourse entity have an **ordMod** (ordinal modifier) with the same numerical value (**ord**) as required (e.g., "2" for second) or a **degMod** (degree modifer) of the same level (e.g., "est" for superlative).

Many questions would give rise to qualifiers on the discourse entity string, frequently asking that either the entity itself or its antecedent (if a pronoun) contain the key or focal noun in the question. This is done frequently with **what** and **who** questions, where the qualification will retrieve a discourse entity that is essentially identical with what is being asked for, rather than the discourse entity that is the answer. This is done so that in the next (evaluation) step, we examine the context surrounding such a key noun, looking for appositives, verb subjects, or verb or prepositional objects. (Frequently, In our official submission, for example, in a question like "what band", our answer was the word "band", because we had not fully implemented our routines for examining the context surrounding the focal noun.)

Another common qualifier on the discourse entity was a specification of the context, where we would look for a discourse entity in relation to a verb. For example, in "Who sang the Tennessee Waltz" looking for a discourse entity that had a **synrole** attribute of "subj" and preceded the verb either equal to "sing" or with a **base** attribute equal to "sing".

In general, the development of these qualifiers is quite tricky. There is a tradeoff between specifying the characteristics of the actual discourse entity or specifying sufficient qualifiers to obtain a reasonable candidate set of discourse entities that can be evaluated, as described in the next section.

4.4 Question-Specific Evaluation of Candidate Answers

In general, the search query returned a considerable number of candidate answers. We then use question-specific routines to examine the context for text elements that have a specific relation to each candidate answer. (For all questions, as mentioned above, we first determined whether we could convert the "or" to "and" to retrieve sentences containing all the terms in the search query to reduce the number of candidates.)

For **What** and **Who** questions, the answer is sought in discourse entities that stand in a copular or appositive relationship to the candidate answers. The candidate answer is first screened to make sure that it does not contain a mismatch in degree or ordinal from one that is specified in the question (e.g., "highest" or "second"). An answer is first sought in a copular relationship, and if not found, the candidate is examined to determine whether it has a following appositive. For example, "What is the national airline of Spain?", the discourse entity "Spain's state airline" is followed by "Iberia". A similar set of tests is used in answering **WhatIsDef** and **WhoIsDef** questions, with the distinctions that these questions allow multiple answers.

For **WhoIsDef** questions, documents in the collection that were talking about a person with the same last name but a different first name were excluded from consideration. This was particularly important when a candidate answer was an anaphor, so it was important to establish that its antecedent was the person in the question. For these questions, we also looked for particular constructions, such as verb phrases beginning with "won" or "discovered" or prepostional phrases beginning with "of' (identifying a person's affiliation).

For **WhatNP** and **List** questions, the key noun constitutes a hypernym and what is being sought is a hyponym. We examine whether a candidate answer node (i.e., a discourse entity) is itself a hyponym, i.e., a noun phrase such as "Labor Party" in answering the question "What party led Australia from 1983 to 1996?". This includes examining the head noun of a phrase to determine, in WordNet, whether it is a hyponym of the key noun in the question. Another test examines other discourse entities in the same sentence to determine if they have a hypernym that matches the one required by the question (e.g., "Philadelphia" is a city in answering the question "What city is the Liberty Bell currently located in?").

For basic **How** questions, the focus is on the verb (e.g., "die"). The candidate answers are the verbs with the base form ("die" from "died"). If the verbs don't appear in sentences containing other key words from the question, an expanded set of synonyms is used (e.g., "assassinate", "murder", "kill", "shoot"). Then, if the verb form is not one that lexicalizes "death" (a derived form in WordNet), the sentence is examined for semantic components (or frame elements) in prepositional phrases that identify the cause of death (e.g., "died of cancer", "died in a plane crash", "from a ruptured abdominal aneurysm"). Other constraints in the question are examined to ensure that the key noun is the subject of the verb (e.g., making sure an anaphor refers to "Marty Robbins" and not "Jerome Robbins").

For **when** questions, the candidate answers were retrieved simply by asking for discourse entities that had the **tag** "when". For questions that asked specifically for a particular type of time ("what year", "what day", or "what month") (which had previously been recategorized as **When** questions rather than **What** questions), the candidate time entities were scrutinized specifically to ensure they were of the proper type. For these question, when the key noun was a phrase ("Cold Mountain" or "International Volunteers Day"), the sentence containing the candidate answer was examined for the occurrence of each word to give higher scores for those containing more words in the phrase.

For **Where** questions, all candidate answers had either a **tag** value of "where" or a **semtype** of "location". were those tagged as being of type "where" or in which the key noun was contained in WordNet, we tested whether a candidate answer was also present in the WordNet definition. We also tested whether a candidate answer had a hypernym in WordNet that was equivalent to the type of geographic location that was sought (e.g., in questions like "what city").

For **HowMany**, **HowMeas**, and **HowMuch** questions, the candidate answers were discourse entities containing numbers and that had already been tagged as one of these three types during the creation of the XML-tagged files. The specific tag was selected during this process based on the semantic characteristics of the modified noun. This considerably simplified the search for candidate answers.

4.5 TREC 2003 QA Results

We submitted one run for the main QA task and two runs for the passage task. Our overall main task score was 0.075, with scores of 0.070 for factoid questions, 0.000 for list questions, and 0.160 for definition questions. For the passage task, we had some difficulty in matching up our answer to the original text, since our sentence splitting often modified the original text, making it difficult to determine the offset and length of our answer. Our first run contained our computations of the offsets and lengths that we had constructed when the splitting was performed; for this run, our socre was 0.087. We then wrote a script that enabled us to examine the correctness of the offset and length of our passage and to make adjustments where necessary (37 percent of the cases). This second run, while not fully automatic, would have been the correct version if we had submitted the actual passage rather than the offset and length. For this run, our score was was 0.119 for the factoid questions. These scores were all considerably below the medians for these tasks.

As indicated earlier, we had not yet implemented many of the routines described in the previous section at the time of submission. With their implementation and further detailed examination of the results, we can provided an updated assessment of our progress in answering questions using the XML-tagging of documents and the use of XPath specifications for obtaining the exact answer. Tables 1 and 2 show our results for the passages task and the exact answer task, respectively.

Table 1. Factoid Questions (Passage)

Question Type	Number	No docs	Correct	Accuracy	Un Mrr	Adj Mrr	Fnd
How	36	9	3	0.11	0.16	0.21	0.37
HowMany	45	4	22	0.54	0.51	0.56	0.59
HowMeas	45	16	0	0.00	0.04	0.05	0.41
HowMuch	5	1	0	0.00	0.00	0.00	0.00
What	98	33	9	0.14	0.13	0.20	0.49
WhatNP	139	42	24	0.25	0.20	0.28	0.39
When	39	12	9	0.33	0.28	0.40	0.74
Where	1	0	1	1.00	1.00	1.00	1.00
Who	5	2	1	0.33	0.20	0.33	0.67
Total	413	119	69	0.23	0.20	0.28	0.47

Table 2. Factoid Questions (Exact Answer)

Question Type	Number	No docs	Correct	Accuracy	Un Mrr	Adj Mrr	Fnd
How	36	9	2	0.07	0.11	0.14	0.26
HowMany	45	4	21	0.51	0.50	0.55	0.59
HowMeas	45	16	0	0.00	0.02	0.04	0.28
HowMuch	5	1	0	0.00	0.00	0.00	0.00
What	98	33	5	0.08	0.06	0.20	0.20
WhatNP	139	42	17	0.18	0.14	0.31	0.31
When	39	12	8	0.30	0.24	0.63	0.63
Where	1	0	1	1.00	1.00	1.00	1.00
Who	5	2	0	0.00	0.00	0.00	0.33
Total	413	119	54	0.18	0.15	0.22	0.34

In these tables, the first column shows our breakdown of the question types and the second column the number of each question type. The third column shows the number of questions for which there were no answers in the top 50 documents provided by NIST. Since we used this set, we feel that an evaluation of our performance should take into account only those questions for which we have a possibility of obtaining an answer. The fourth column shows the number of correct answers using the NIST criteria. The fifth column shows the accuracy based on the number correct divided by the total number of questions less the number with no documents. The sixth column shows the mean reciprocal rank of our answers for those cases in which an answer was obtained in the top five answers, without adjusting for the questions with no answers. The seventh column shows the mean reciprocal rank, with adjustment for the questions with no answers. The eighth column shows the percentage of cases in which our answer set contained the correct answer (i.e., without regard to the rank of our answer).

Our overall accuracy for the passages task is 0.167 (69/413) and for the exact answer task is 0.131 (54/413). These are somewhat better than our official scores of 0.119 and 0.070, but they would still not affect our low rank among participating teams. We suggest that a better picture of our results is given by our accuracy scores after adjusting for questions with no answers in the top 50 documents. Our score of 0.23 for the passages task would place us in the 4th position, and our score of 0.18 for the exact answer task would place us in the 12th position. We would not suggest this should be used as the measure to be used in ranking teams, but only to indicate that the information retrieval component is important and that, notwithstanding, the basic mechanisms we have employed are viable. In addition, the final column of the tables further indicate the viability of our approach, since they indicate that we are finding the answers, but have further work to boost them in our ranking system.

For all question types, we have not yet implemented the full range of tests that we used in past years when using a database representation of questions and documents. We believe that our further implementation will lead to significant improvements. In addition, we note that many of our routines were implemented for specific question types. Many of these routines can be usefully employed for other question types and can be further generalized. Work in this area is also likely to lead to some overall improvements.

5 Novelty Detection Using XML-Taggged Documents

Our participation in the Novelty track had two main components, one implementing special procedures to handle the various tasks, and the other implementing the procedures for performing the tasks.

In order to allow KMS to process the Novelty texts and build XML representations for them, we first added wrappers to the NIST provided texts to make them XML compliant. We then processed the files with KMS. Similarly, we added wrappers to the topic file to make it XML compliant, and then processed it with KMS, processing as text the title, description, and narrative fields. For tasks 2, 3, and 4, we converted the NIST-provided qrels files of relevant and new sentences into XPath expressions that would select the corresponding sentences from the XML versions of the NIST texts.

5.1 Determination of Relevance

The basic relevance judgment for a sentence was determined by examining its discourse entities and antecedents if a discourse entity had an antecedent (anaphors, coreferents, or definite noun phrases). Each word, except words on a stop list, was compared to the list of words obtained from the topic. The basic criterion for selection of a sentence as relevant was whether a sentence had two or more hits.

For task 1, the basic criterion was applied using different amounts of information, with one run using only the title, one run using the title and the description, and a third run using the title, the description, and the narrative (with words in sentences containing the word "irrelevant" or the phrase "not relevant" excluded from the list). A fourth run used all the information as in the third run but required three or more hits.

For task 3, where relevant sentences were provided for the first five documents, a frequency list was developed for words in discourse entities or antecedents. The total number of words in this list was also determined. For each sentence, a frequency score was computed as the sum of the frequency count for all word in the sentence on the frequency list divided by the total number of words on this list. Then, the basic criterion was modified. Sentences with two or more hits were still selected as relevant, but also sentences with a frequency score greater than a specified level were also selected as relevant. For task 3, five runs were submitted based on different frequency scores, 0.01, 0.02, 0.03, 0.04, and 0.05. The lower scores allow more sentences, thus increasing recall.

5.2 Determination of Novelty

Once a set of sentences had been selected as relevant, they were considered in order to determine novelty. Sentences that were exact duplicates were first eliminated. Next, each discourse entity was evaluated for novelty against an accumulating list of all unique discourse entities encountered thus far. Again, antecedents were used in preference to the actual discourse entity for anaphors, coreferents, and definite noun phrases that had a non-empty antecedent attribute. If a discourse entity from the current sentence being evaluated was not found, it was added to the growing history list and the sentence was accepted as novel. In evaluating a discourse entity, if all of its words were present in a discourse entity already on the history list, the candidate was viewed as old information. If all discourse entities in a sentence were present on the history list, the sentence being evaluated was characterized as overlapping with prior information and thus eliminated from the set of novel sentences.

For task 1, the novel sentences were selected from the relevant sentences that had been determined as described above. For task 2, where all relevant sentences were given, only these were considered in determining novelty; this task thus provides a reasonable characterization of the novelty component by itself. For task 3, the novel sentences will be selected from among a wider than used in task 1, since these were conditioned by the greater recall of relevance sentences. For task 4, we submitted the same results as for task 2, since our novelty routines at this time contained no processing that would take into account sentences that had been identified as being novel.

5.3 Novelty Track Results

For the Novelty track, we submitted four runs for task 1, one run for task 2, five runs for task 3, and one run for task 4; our submissions for tasks 2 and 4 were identical.

Table 1. Task 1 (Relevance)

Run	Precision	Recall	F-Score
clr03n1t	0.71	0.25	0.309
clr03n1d	0.72	0.32	0.385
clr03n1n2	0.69	0.45	0.483
clr03n1n3	0.72	0.24	0.316

Table 2. Task 1 (Novelty)

Run	Precision	Recall	F-Score
clr03n1t	0.51	0.24	0.272
clr03n1d	0.51	0.31	0.331
clr03n1n2	0.50	0.40	0.410
clr03n1n3	0.51	0.24	0.278

For task 1, our best run received an F-score of 0.483 for relevant sentences and 0.410 for new sentences. For all runs, our F-scores were on average higher than the median. When examining the scores by the topic type, we found that on relevance, our F-scores were quite similar, but on novelty, there was a wide difference in our system's performance, achieving a much higher average F-score on event topics. For run clr03n1d, our F-score on event topics was 0.369 and for opinion topics, it was 0.282.

The results show clearly that more information describing the topic is valuable in increasing recall dramatically, while precision is only moderately different. Changing the number of hits for relevance determination clearly decreases the recall significantly, to less than what was achieved with less information.

Table 3. Task 2 (Novelty)

Run	Precision	Recall	F-Score
clr03n2	0.71	0.91	0.788

Task 2 shows that the novelty component of our system is performing at a quite high level. This indicates that when the relevance determination is of high quality, we are able to discriminate novel information quite well. If our system improves its relevance assessment, or if our system operates in an environment where a user can provide relevance feedback, we can expect to identify novel information with considerable fidelity. This would be quite useful for text summarization.

Table 4. Task 3 (Relevance)

Run	Precision	Recall	F-Score
clr03n3f01	0.48	0.84	0.558
clr03n3f02	0.48	0.77	0.541
clr03n3f03	0.48	0.72	0.527
clr03n3f04	0.48	0.68	0.513
clr03n3f05	0.48	0.63	0.493

Table 5. Task 3 (Novelty)

Run	Precision	Recall	F-Score
clr03n3f01	0.33	0.79	0.419
clr03n3f02	0.33	0.73	0.408
clr03n3f03	0.33	0.69	0.401
clr03n3f04	0.33	0.65	0.395
clr03n3f05	0.33	0.61	0.383

Task 3 also indicates the value of relevance feedback, as well as the value of using frequency assessments in improving recall. We also made an additional five runs with the frequency score cutoff at values from 0.06 to 0.10, with the trends shown above occurring with them as well, with lower and lower values for recall, with a resultant degradation in the overall F-score.

Table 6. Task 4 (Novelty)

Run	Precision	Recall	F-Score
clr03n1t	0.53	0.91	0.655

As indicated above, we submitted the same run for task 4 as for task 2; our lower F-score of 0.655 for new sentences is a direct reflection of the decreased performance brought about by the degradation of precision.

6 Summary

Our results on the QA and Novelty tracks indicate that our approach of using massively XML-tagged documents is viable and worth continuing development. There are many opportunities that will be investigated.

References

Litkowski, K. C. (2001). Syntactic Clues and Lexical Resources in Question-Answering. In E. M. Voorhees & D. K. Harman (eds.), *The Ninth Text Retrieval Conference (TREC-9)*. NIST Special Publication 500-249. Gaithersburg, MD., 157-166.

Litkowski, K. C. (2002a). CL Research Experiments in TREC-10 Question-Answering. In E. M. Voorhees & D. K. Harman (eds.), *The Tenth Text Retrieval Conference (TREC 2001)*. NIST Special Publication 500-250. Gaithersburg, MD., 122-131.

Litkowski, K. C. (2002b). Digraph Analysis of Dictionary Preposition Definitions. *Proceedings of the ACL SIGLEX Workshop: Word Sense Disambiguation*. Philadelphia, PA., 9-16.

Litkowski, K. C. (2003a). Question Answering Using XML-Tagged Documents. In E. M. Voorhees & L. P. Buckland (eds.), *The Eleventh Text Retrieval Conference (TREC 2002)*. NIST Special Publication 500-251. Gaithersburg, MD., 122-131.

Litkowski, K. C. (2002a). Text Summarization Using XML-Tagged Documents. In Proceedings of the DUC 2003 Document Understanding Conference.

Approaches to Robust and Web Retrieval

Jaap Kamps **Christof Monz*** **Maarten de Rijke** **Börkur Sigurbjörnsson**
Language & Inference Technology Group
University of Amsterdam
http://lit.science.uva.nl/

Abstract: We describe our participation in the TREC 2003 Robust and Web tracks. For the Robust track, we experimented with the impact of stemming and feedback on the worst scoring topics. Our main finding is the effectiveness of stemming on poorly performing topics, which sheds new light on the role of morphological normalization in information retrieval. For both the home/named page finding and topic distillation tasks of the Web track, we experimented with different document representations and retrieval models. Our main finding is effectiveness of the anchor text index for both tasks, suggesting that compact document representations are a fruitful strategy for scaling-up retrieval systems.

1 Introduction

This year, our aim for the Web track was to experiment with different document representations and retrieval models for the home/named page finding and topic distillation tasks. The Robust track was new in 2003; our aim here was to investigate the impact of blind feedback and stemming on poorly performing topics.

For both tracks, our experiments exploited the home-grown FlexIR document retrieval system [9]. The main goal underlying FlexIR's design is to facilitate flexible experimentation with a wide variety of retrieval components and techniques. FlexIR is implemented in Perl, and supports many types of pre-processing, scoring, indexing, and term-weighting methods.

The rest of this paper is organized as follows. In two (largely self-contained) sections we describe our work for the Robust and Web tracks. Finally, we summarize our findings in a concluding section.

2 Robust Track

After describing the experimental setup for this track, we discuss our runs investigating the impact of blind feedback and stemming on the poorly performing topics.

System Description

All Robust track runs use the FlexIR information retrieval system. We employ a number of techniques:

Tokenization We remove punctuation marks, apply case-folding, and map marked characters into the unmarked tokens. We either index the words themselves, or the stems of the words. We use the Snowball stemming algorithm [13]. Snowball is a small string processing language designed for creating stemming algorithms for use in information retrieval

Retrieval model We use a multinominal language model with Jelinek-Mercer smoothing [4]. For all robust track runs, we use a uniform query term importance weight of 0.15.

Blind feedback Term weights are recomputed by using the standard Rocchio method [12], where we consider the top 10 documents to be relevant and doc-

*Present address: Institute for Advanced Computer Studies, University of Maryland, 3161 A.V. Williams Building, College Park, MD 20742, USA. Email: christof@umiacs.umd.edu.

uments ranked 501–1000 to be non-relevant. We allow at most 20 terms to be added to the original query.

Runs

We conduct two sets of experiments using (1) only the description field of the topics (D-topics), or (2) both the title and description fields (TD-topics). Using the resulting queries, we constructed the following four runs:

Words Language model run on a word-based index. This runs serves as the baseline for our stemming and feedback experiments.

Words+feedback Language model run on a word-based index, using Rocchio blind feedback.

Stems Language model run on the Snowball stemmed index.

Stems+feedback Language model run on the Snowball stemmed index, using Rocchio blind feedback.

Results

Table 1 gives the results of the runs over all 100 robust topics (best scores in boldface). The second column shows the mean average precision, the third the precision at 10 documents, the fourth the percentage of topics with no relevant document in the top 10; the fifth shows the area underneath the MAP(X) versus X curve for the worst 25 topics.

Table 1: Results for the Robust track (D top and TD bottom).

Run identifier	MAP	Prec.10	NoTop10	MAP(X)
Words	0.2065	0.3530	15.0%	0.0076
Words+feedback	0.1970	0.3420	17.0%	0.0059
Stems	**0.2319**	**0.3960**	**14.0%**	**0.0126**
Stems+feedback	0.2068	0.3570	16.0%	0.0098
Words	0.2324	0.4050	9.0%	0.0216
Words+feedback	**0.2452**	0.4110	13.0%	0.0210
Stems	0.2450	**0.4150**	**6.0%**	0.0256
Stems+feedback	0.2373	0.4040	14.0%	**0.0273**

The results of blind feedback are mixed at best. On the one hand feedback helps the overall score for the runs using TD-topics, with a best precision at 10 and a best score for mean average precision. On the other hand feedback hurts the performance on the worst scoring topics. For the runs using D-topics, feedback deteriorates scoring on all measures.

We can regard the T-field of the topics as a "gold standard" experiment on query expansion. If we compare the score of runs using TD-topics with the scores of runs using D-topics, we see an improvement on all measures and runs. In particular the improvement on the weak-scoring topic measures is substantial.

The results for Snowball stemming are positive overall. Stemming helps both the overall performance, with a best score for precision at 10, as well as the performance of the worst scoring topics, with a best score for the percentage of topics with a top 10 relevant document. For runs using D-topics, stemming gives the best score for all measures. The use of both stemming and feedback gives the best score for the area under the MAP(X) curve for the runs using TD-topics, but does not promote performance on the other measures.

We also break down the score over the 50 old topics (in Table 2) and the 50 new topics (in Table 3). Note that the area underneath MAP(X) versus X curve (in the last column) is now calculated for the worst 12 topics. For both the old and new topics, the effectiveness of feedback and stemming is comparable to the effectiveness on all topics. There is, however, a striking difference in the performance between the two types of topics: the new topics

Table 2: Results for the old topics (D top and TD bottom).

Run identifier	MAP	Prec.10	NoTop10	MAP(X)
Words	0.1066	0.2640	**14.0%**	0.0064
Words+feedback	0.0969	0.2460	20.0%	0.0039
Stems	**0.1164**	**0.3020**	18.0%	**0.0108**
Stems+feedback	0.1065	0.2640	18.0%	0.0085
Words	0.1349	0.3180	12.0%	0.0142
Words+feedback	**0.1377**	0.3200	16.0%	0.0143
Stems	0.1327	**0.3300**	**6.0%**	0.0185
Stems+feedback	0.1361	**0.3300**	16.0%	**0.0204**

Table 3: Results for the new topics (D top and TD bottom).

Run identifier	MAP	Prec.10	NoTop10	MAP(X)
Words	0.3064	0.4420	16.0%	0.0142
Words+feedback	0.2971	0.4380	14.0%	0.0105
Stems	**0.3475**	**0.4900**	**10.0%**	**0.0294**
Stems+feedback	0.3071	0.4500	14.0%	0.0216
Words	0.3300	0.4920	**6.0%**	0.0433
Words+feedback	0.3528	**0.5020**	10.0%	0.0368
Stems	**0.3572**	0.5000	**6.0%**	**0.0551**
Stems+feedback	0.3386	0.4780	12.0%	0.0478

give a much higher mean average precision score. This is an obvious consequence of the way the old topics were selected for inclusion in this year's Robust track. As a result, the worst topic measures are dominated by the old topics.

3 Web Track

After describing our experimental setup for this track, we discuss our runs for the home/named page finding task (known-item search), followed by the runs for the topic distillation task (key resource search).

System Description

All Web track runs use the FlexIR information retrieval system. We employ a number of techniques:

Document representation We create indexes for (1) the full documents, (2) the text in the title tags, (3) the anchor texts pointing toward the document. For the anchor texts index, we unfold relative links and normalize URLs, and do not index repeated occurrences of the same anchor text [10].

Tokenization We remove HTML-tags, punctuation marks, apply case-folding, and map marked characters into the unmarked tokens. We either index the free-text without further processing, or use the Snowball stemming algorithm [13].

Retrieval model We use three retrieval models. First, a statistical language model [4] with a uniform query term importance weight of either 0.35 or 0.70. Second, the Okapi weighting scheme [11] with tuning parameters $k = 1.5$ and $b = 0.8$. Third, the `Lnu.ltc` weighting scheme [1] with *slope* at 0.1 or 0.2; the pivot was set to the average number of unique words per document.

Combination We use the standard combination methods such as CombSUM and CombMAX [3], or weighted fusion [14]. We combine either full length runs, or limit the combination to the top n results. Unless indicated otherwise, we normalize the scores before combining them.

Minimal span weighting We calculate a minimally matching span for each document. Intuitively, a minimal matching span is the smallest text excerpt from a document that contains all terms which occur in the query and the document. Minimal span weighting depends on three factors (for details, see [2, 5, 8]).

1. *document similarity*: The document similarity is computed for the whole document, i.e., positional information is not taken into account. Similarity scores are normalized with respect to the maximal similarity score for a query.
2. *span size ratio*: The span size ratio is the number of unique matching terms in the span over the total number of tokens in the span.
3. *matching term ratio*: The matching term ratio is the number of unique matching terms over the number of unique terms in the query, after stop word removal.

In two separate sections, we will now address our runs and results for the home/named page finding task, and the topic distillation task.

3.1 Home/Named Page Finding Task

Runs

We submitted the following five official runs for the home/named page finding task:

UAmsT03WnOWS CombSUM of top 1000 of Okapi on word-based and stemmed full document indexes.

UAmsT03WnLM Language model run ($\lambda = 0.70$) on word-based full document index.

UAmsT03WnLn3 CombMAX on the top 25 of `Lnu.ltc` runs (*slope* = 0.2) on the three stemmed indexes: full documents, titles, and anchor texts.

UAmsT03WnLM3 Weighted fusion of language model runs ($\lambda = 0.70$) on the three word-based indexes: 0.7 full documents, 0.2 titles, and 0.1 anchor texts.

UAmsT03WnMSW Minimal span weighting based on the `Lnu.ltc` run (*slope* = 0.1) on the stemmed full document index.

Results

The results of the official runs for the home/named page finding task are shown in Table 4 (best scores in boldface). The second column gives the mean reciprocal rank, the third the number and percentage of topics with a relevant document in the top 10, the fourth the number and percentage of topics for which no relevant document is found (in the top 50). The language model run combining the non-stemmed documents, titles, and anchors scores best with an average reciprocal rank of 0.5185. The Lnu.ltc weighted combination of the three stemmed indexes scores second best.

Table 4: Results for home/named page finding.

Run identifier	MRR	Top 10	not found
UAmsT03WnOWS	0.3833	178 (59.3%)	70 (23.3%)
UAmsT03WnLM	0.3592	170 (56.7%)	81 (27.0%)
UAmsT03WnLn3	0.4982	**218** (72.7%)	**38** (12.7%)
UAmsT03WnLM3	**0.5185**	214 (71.3%)	46 (15.3%)
UAmsT03WnMSW	0.4073	189 (63.0%)	64 (21.3%)

Table 5 shows the mean average precision of the base runs used in combinations for our official runs. All Lnu.ltc runs use a slope of 0.2, and all language model runs use a uniform term weight of 0.70. Here, we retrieve up to 1,000 documents per topic, leading to slightly higher MRRs than the official runs using a maximum of 50 documents. We see an interesting difference between the three retrieval models: where the Lnu.ltc and Okapi models score best on the full document representation, the language model runs on the anchor text index score more than 20% better than the runs on the full document index. In fact, our best score on a single index is on the language model run on the non-stemmed anchor text index. There is no clear benefit of the use of a stemming algorithm on the mean reciprocal ranks: stemming improves the score for four out of the nine comparative runs.

There is another interesting difference between the retrieval models, which has to do with combination. The combination of Okapi runs on the document stems and words, UAmsT03WnOWS, does not improve over document stems run. The combination of the three stemmed Lnu.ltc runs, run UAmsT03WnLn3, does improve 34.8% over the best scoring stemmed runs. The combination of the three non-stemmed language model runs, UAmsT03WnLM3, improves 16.9% over the best scoring base runs. Finally, the run using the matching-span weighting uses a Lnu.ltc full document base run with a different slope of 0.1 scoring a MRR of 0.2742. The resulting run, UAmsT03WnMSW, improves no less than 48.5% over the underlying base run.

Table 5: MRR for home/named page finding base runs.

Index type		Lnu.ltc	Okapi	LM
Documents	Words	0.3750	0.3795	0.3604
	Stems	0.3697	0.3833	0.3616
Titles	Words	0.2339	0.3421	0.3536
	Stems	0.3655	0.3334	0.3487
Anchors	Words	0.3068	0.3593	**0.4436**
	Stems	0.2934	0.3379	0.4278

Table 6: Results for home page topics.

Run identifier	MRR	Top 10	not found
UAmsT03WnOWS	0.2567	67 (44.7%)	55 (36.7%)
UAmsT03WnLM	0.2462	64 (42.7%)	60 (40.0%)
UAmsT03WnLn3	0.4105	97 (64.7%)	**26** (17.3%)
UAmsT03WnLM3	**0.4402**	**101** (67.3%)	33 (22.0%)
UAmsT03WnMSW	0.2708	73 (48.7%)	53 (35.3%)

We also break down the score over the 150 home page topics (in Table 6) and the 150 named page topics (in Table 7). Here we see a much better performance on the named page topics. This is perhaps unexpected because named page finding is conceived to be a more difficult task than home page finding. The simple explanation is that we decided not to apply special home page finding strategies. Although techniques like slash-counts or URL priors are effective for home page finding [7], they seem to hurt the named page topics considerably. Even without a particular home page bias, home pages can be retrieved with reasonable effectiveness, as is witnessed by our results for the home page topics in Table 6.

Table 7: Results for named page topics.

Run identifier	MRR	Top 10	not found
UAmsT03WnOWS	0.5098	111 (74.0%)	15 (10.0%)
UAmsT03WnLM	0.4721	106 (70.7%)	21 (14.0%)
UAmsT03WnLn3	0.5859	**121** (80.7%)	12 (8.0%)
UAmsT03WnLM3	**0.5969**	113 (75.3%)	13 (8.7%)
UAmsT03WnMSW	0.5438	116 (77.3%)	**11** (7.3%)

3.2 Topic Distillation Task

Runs

We submitted the following five official runs for the topic distillation task:

UAmsT03WtOk3 Weighted fusion of Okapi runs on the three stemmed indexes: 0.7 full documents, 0.2 titles; and 0.1 anchor texts.

UAmsT03WtLM3 Weighted fusion of language model runs on the three stemmed indexes: 0.7 full documents ($\lambda = 0.35$), 0.2 titles ($\lambda = 0.7$), and 0.1 anchor texts ($\lambda = 0.7$). We combine the probabilities without normalization.

UAmsT03WtOkI Weighted fusion of 0.9 Okapi run on the stemmed full document index with 0.1 of a link topology measure. We applied the realized indegree on the top 10 documents [10]. This is a variant of HITS [6] where we consider the fraction of inlinks that is in the local set—roughly a `tf·idf` measure for link topology.

UAmsT03WtLMI Weighted fusion of 0.9 language model run ($\lambda = 0.35$) on the stemmed full document index with 0.1 of the realized indegree of the top 10 documents.

UAmsT03WtOkC Weighted fusion of 0.8 Okapi run on the stemmed full document index with 0.2 of a URL-based reranking. The reranking was done by clustering the found pages by their base URLs, and to only return the page with the lowest slash-count per cluster.

Results

The results of the official runs for the topic distillation task are shown in Table 8 (best scores in boldface). The second column shows the mean average precision, the third to fifth columns show the precision at 10, 20, and 30 documents, respectively. The best score is obtained by UAmsT03WtOk3, the fusion of Okapi runs on the three stemmed indexes. The second best score is obtained by UAmsT03WtOkC, a URL-based clustering of the Okapi full documents run. Before discussing the results of our experiments, we first evaluate the results of the runs used to create our official runs.

Table 8: Results for topic distillation.

Run identifier	MAP	Prec. at 10, 20, 30		
UAmsT03WtOk3	**0.1344**	**0.0980**	**0.0810**	**0.0787**
UAmsT03WtLM3	0.1019	0.0840	0.0630	0.0533
UAmsT03WtOkI	0.0862	0.0760	0.0660	0.0567
UAmsT03WtLMI	0.0412	0.0280	0.0260	0.0267
UAmsT03WtOkC	0.1127	0.0860	0.0650	0.0540

Table 9 shows the results of the base runs used in combination for our official runs. All these runs use the Snowball stemming algorithm [13]. We see a remarkable divergence between the scoring for Okapi and the language model. The Okapi model performs comparable on all the three indexes, documents, titles, and anchors. The language model performs poorly on the document and title indexes, but excels for the anchor text index. The combination of the three Okapi runs, UAmsT03WtOk3, improves significantly over the best underlying run (MAP +38.4%, Precision at 10 +25.6%). The combination of language model runs, UAmsT03WtLM3, uses far from optimal relative weights and, as a result, does not improve over the anchor text run. The runs using the hyperlink graph topology do not result in significant improvement. The Okapi run UAmsT03WtOkI slightly improves its precision at 10 over the document run; whereas the language model run UAmsT03WtLMI slightly decreases its precision at 10 over the document run. Finally, the Okapi run clustering per base URL, UAmsT03WtOkC, does improve over the Okapi document run (MAP +25.1%, Precision at 10 +16.2%).

Table 9: Results for topic distillation stemmed base runs.

Run type	MAP	Prec. at 10, 20, 30		
Doc. Okapi	0.0901	0.0740	0.0580	**0.0527**
Title Okapi	0.0870	0.0780	**0.0590**	0.0453
Anchor Okapi	0.0971	0.0780	0.0560	0.0493
Doc. LM (0.35)	0.0386	0.0300	0.0320	0.0293
Title LM (0.70)	0.0434	0.0480	0.0360	0.0293
Anchor LM (0.70)	**0.1068**	**0.0860**	0.0560	0.0473

4 Conclusions

In this paper we have described our participation in the TREC 2003 Robust and Web tracks.

For the Robust track, we experimented with the impact of stemming and feedback on the worst scoring topics. Our results suggest that blind feedback can help overall performance but does not increase the effectiveness on the lowest scoring topics. Our results also suggest that applying a stemming algorithm does benefit both the overall performance, as well as the performance of the worst scoring topics. This result sheds some new light on the role of morphological normalization in information retrieval.

For the Web track, we saw very similar results for both the home/named page finding task and the topic distillation task. Using the hyperlinks in the collection for creating an anchor text index turns out to be very effective. Also, the use of HTML-structure in the documents to elicit their titles turns out to be effective. Combining these alternative document representations with a standard document index led to our best scores for both tasks.

A further general observation is the effectiveness of compact document representations, such as indexing only document titles, or only anchor texts pointing toward documents. These compact document representations result in performance that meets or exceeds the performance of a massive full document text index. This result suggests that it is feasible to create effective retrieval indexes for even larger web collections, provided that the appropriate document representation is chosen.

Acknowledgments Jaap Kamps was supported by NWO under project numbers 400-20-036 and 612.066.302. Christof Monz was supported by NWO under project numbers 612-13-001 and 220-80-001. Maarten de Rijke was supported by NWO under project numbers 612-13-001, 365-20-005, 612.069.006, 612.000.106, 220-80-001, 612.000.207, and 612.066.302.

References

[1] C. Buckley, A. Singhal, and M. Mitra. New retrieval approaches using SMART: TREC 4. In D. Harman, editor, *The Fourth Text REtrieval Conference (TREC-4)*, pages 25–48. National Institute for Standards and Technology. NIST Special Publication 500-236, 1996.

[2] C. Clarke, G. Cormack, and T. Lynam. Exploiting redundancy in question answering. In D. H. Kraft, W. B. Croft, D. J. Harper, and J. Zobel, editors, *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 358–365. ACM Press, New York NY, USA, 2001.

[3] E. Fox and J. Shaw. Combination of multiple searches. In D. Harman, editor, *The Second Text REtrieval Conference (TREC-2)*, pages 243–252. National Institute for Standards and Technology. NIST Special Publication 500-215, 1994.

[4] D. Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, Center for Telematics and Information Technology, University of Twente, 2001.

[5] V. Jijkoun, G. Mishne, C. Monz, M. de Rijke, S. Schlobach, and O. Tsur. The University of Amsterdam at the TREC 2003 question answering track. In *The Twelfth Text REtrieval Conference (TREC 2003)*. National Institute for Standards and Technology, 2004.

[6] J. M. Kleinberg. Authoritative structures in a hyperlinked environment. *Journal of the ACM*, 46:604–632, 1999.

[7] W. Kraaij, T. Westerveld, and D. Hiemstra. The importance of prior probabilities for entry page search. In K. Järvelin, M. Beaulieu, R. Baeza-Yates, and S. H. Myaeng, editors, *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 27–34. ACM Press, New York NY, USA, 2002.

[8] C. Monz. *From Document Retrieval to Question Answering*. ILLC dissertation series 2003-04, University of Amsterdam, 2003.

[9] C. Monz and M. de Rijke. Shallow morphological analysis in monolingual information retrieval for Dutch, German and Italian. In C. Peters, M. Braschler, J. Gonzalo, and M. Kluck, editors, *Evaluation of Cross-Language Information Retrieval Systems, CLEF 2001*, volume 2406 of *Lecture Notes in Computer Science*, pages 262–277. Springer, 2002.

[10] C. Monz, J. Kamps, and M. de Rijke. The University of Amsterdam at TREC 2002. In E. M. Voorhees and L. P. Buckland, editors, *The Eleventh Text REtrieval Conference (TREC 2002)*, pages 603–614. National Institute for Standards and Technology. NIST Special Publication 500-251, 2003.

[11] S. Robertson, S. Walker, and M. Beaulieu. Experimentation as a way of life: Okapi at TREC. *Information Processing & Management*, 36:95–108, 2000.

[12] J. Rocchio, Jr. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART Retrieval System: Experiments in Automatic Document Processing*, Prentice-Hall Series in Automatic Computation, chapter 14, pages 313–323. Prentice-Hall, Englewood Cliffs NJ, 1971.

[13] Snowball. Stemming algorithms for use in information retrieval, 2003. `http://www.snowball.tartarus.org/`.

[14] C. C. Vogt and G. W. Cottrell. Predicting the performance of linearly combined IR systems. In W. B. Croft, A. Moffat, C. J. van Rijsbergen, R. Wilkinson, and J. Zobel, editors, *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 190–196. ACM Press, New York NY, USA, 1998.

Combining Structural Information and the Use of Priors in Mixed Named-Page and Homepage Finding

Paul Ogilvie and Jamie Callan
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
{pto, callan}@cs.cmu.edu

Abstract

This paper presents Carnegie Mellon University's experiments on the mixed named-page and homepage finding task of the TREC 12 Web Track. Our results were strong; we achieved the success using language models estimated from combining information from document text, in-link text, and information present in the structure of the documents. We also present experiments using expectations about posterior distributions to create class-based prior probabilities. We find that priors do provide a large gain for our official runs, but we do further experiments that show the priors do not always help. Some preliminary analysis shows that the prior probabilities are not providing the desired posterior distributions. In cases where applying the priors harm performance, the observed posterior distributions in the rankings are far off of the desired posterior distributions.

1. Introduction

Documents found on the Internet are rich in structure, and this provides many information sources useful for retrieval. In particular, structural information has been found useful for known-item searches such as homepage finding and named-page finding [Craswell 2001][Kraaij 2002][Ogilvie 2003][Zhang 2002]. A known-item retrieval system should attempt to leverage the structural information in a manner that is consistent with its model and provides an improvement in retrieval performance.

In this paper, we present experiments where we combine structural information using language modeling. We create several document representations for each document using the structural information present in HTML documents. From these document representations, we form language models. We combine the language models using a linear interpolation to form a new language model. This new language model is then used to estimate the probability that the document has generated the query.

It is also possible to leverage query independent information through the use of document priors. Document priors are probabilities or beliefs that the document is relevant to the query independent of any knowledge about the query. In previous homepage finding experiments, [Kraaij 2002] found a prior based on the type of the URL to be a very effective source of information. The URL types ROOT, SUBROOT, FILE, and PATH form four distinct document classes. We hypothesize that these classes will also be useful for a mixed homepage/named-page finding task, and present experiments using these priors. The prior probabilities are estimated from training data which gives us a desired posterior distribution. This desired posterior distribution defines ratios of the document classes what we would like to observe in the rankings. Using Bayes' rule, we can estimate the prior probabilities from training data and corpus statistics. A detailed derivation is provided Section 5.

In the next section, we describe the basic generative language model where we are using information from only one document representation. Section 3 describes combining information from several language models. Section 4 briefly discusses system specifics for the experiments. Section 5 provides a detailed derivation of the priors and posterior distribution we used for the mixed homepage/named-page finding track of the TREC 12 Web Track, and Section 6 describes our official runs and other experiments. Section 7 provides a discussion of the URL priors and their effectiveness in producing the desired posterior distributions over the rankings. We state conclusions in Section 8.

2. Generative Language Models

A unigram language model defines a multinomial probability distribution over all words in the vocabulary of the corpus. These probabilities are interpreted as word generation probabilities, and documents are ranked by their probability of generating the query. This generation probability is computed by taking the product over all query terms of the probability of the query term given the language model [Zhai 2001]:

$$P(Q|\theta_D) = \prod_{i=1}^{|Q|} P(q_i|\theta_D) \tag{1}$$

where q_i is the i^{th} query term of query Q, |Q| is the length of Q, and θ_D is a language model estimated from document D. In the case where we desire a ranking using only one of the document representations, we directly take

$$P(w|\theta_D) = P(w|\theta_{D(i)}) \tag{2}$$

where i indicates a specific document representation.

The language models for an individual document representation can be estimated by smoothing a maximum likelihood estimator (MLE) with a collection-wide document representation model:

$$P(w|\theta_{D(i)}) = \lambda_i P_{MLE}(w|D_i) + (1-\lambda_i)P_{MLE}(w|C_i) \tag{3}$$

where C(i) is the aggregate over all document representations D(i). The MLE for a document representation is:

$$P_{MLE}(w|D_i) = \frac{count(w;D_i)}{|D_i|} \tag{4}$$

The MLE distribution for the C(i) is estimated similarly. It is common to set the linear interpolation parameters λ_1 and λ_2 in Equation 3 using guidance from Dirichlet prior smoothing:

$$\lambda_i = \frac{|D_i|}{|D_i| + \mu_i} \tag{5}$$

where μ_i is a parameter often set emprically or by using cross-validation [Zhai 2002].

3. Combining Information Using Language Models

When we wish to combine information formed from a variety of document representations, we take a linear interpolation of the unigram language models estimated from the individual representations [Ogilvie 2003]. This new language model for a document should be designed so that it closely models what we would expect a user to write as a query when requesting the document. This is different from doing a linear combination of scores from different systems as we directly estimate the probability of a word given the differing language models. This by replacing Equation 2 with:

$$P(w|\theta_D) = \sum_{i=1}^{k} \varphi_i P(w|\theta_{D(i)}) \tag{6}$$

where k is the number of document representations for a document and φ_i is the weight placed on the i^{th} document representation. The φ values must be positive and sum to one.

When we wish to incorporate a prior probability in the ranking, we restate the problem as one of estimating the probability of the document given the query and applying Bayes' rule:

$$P(D|Q) = \frac{P(Q|D)P(D)}{P(Q)} \tag{7}$$

The P(Q) constant can be ignored in ranking, and the $P(\theta_D)$ component is the prior. We estimate P(Q|D) using the generative probability and estimate the priors using the URL class of the web page, giving:

$$P(D|Q) \propto P(Q|\theta_D)P(NP \vee HP|type(D)) \tag{8}$$

where P(NP or HP|$type$(D)) is the prior probability of the page being a named page or homepage given the URL type of the document and $P(Q|\theta_D)$ is the generative probability defined in Equation 1.

4. System Specifics

We use the Lemur toolkit [Lemur] for document indexing and retrieval. For document tokenization we used Inquery's stopword list and the Porter stemmer. The URLs were tokenized on punctuation (., /) and were not stemmed. A shorter stopword list was used for URLs ("http", "www", "com", "gov", "html", etc.). Each document had as many as seven document representations, as shown in Table 1. For every representation except the URL, we formed language models using Dirichlet prior smoothing. The Dirichlet prior parameter was chosen to be close to twice the average length of the representation. This is not an optimal parameter setting, but may not have a large effect on results. See [Zhai 2001][Zhai 2002] for more information. The probability of a word given the document's URL was computed treating the URL and word as a character sequence, then computing a character-based trigram generative probability. The numerator and denominator probabilities in the trigram expansion were estimated using a linear interpolation with the collection model (all URLs in the corpus). The final document scores were computed as the generative probability of the query given the document, taking the linear interpolation over the document representations.

Representation	Description
Alt	Image alternate text
Font	Changed font sizes and headings
Full	Full document text
Link	In-link text
Meta	Meta tags (keyword, description)
Title	Document title
URL	Character trigram on URL

Table 1: Document representations

5. Parameter Estimation

The weights for the document representations (the φ parameters used in Equation 2) were estimated by averaging the MRR (Mean Reciprocal Rank) scores of the individual document representations on the TREC 10 Homepage Finding task and the TREC 11 Named-Page Finding task. Table 2 shows the MRR of the individual representations on the previous known-item tasks and the resulting scaled weights. Table 2 also shows the φ we estimated from the TREC 10 and TREC 11 data. We estimated the φ values by scaling the TREC 10 and TREC 11 columns of Table 2 to sum to one then averaged the two scaled columns.

Representation	TREC 10 Homepage (MRR)	TREC 11 Named-Page (MRR)	φ
Alt	0.186	0.194	0.102
Font	0.155	0.191	0.093
Full	0.300	0.469	0.204
Link	0.515	0.455	0.263
Meta	0.115	0.144	0.069
Title	0.332	0.406	0.198
URL	0.132	0.131	0.071

Table 2: Estimating representation weight from TREC 10 and TREC 11 data

Table 3 shows the actual performance of the document representations on the TREC 12 test data. We can see that the performance of the document representations for named page finding in TREC 11 is similar to their performance in TREC 12. This is not surprising; the TREC 11 and TREC 12 named-page topics were selected in a similar manner and both use the .GOV corpus. However, the performance of the individual document representations TREC 10 homepage finding task is not as predictive for the TREC 12 homepage topics. In particular, the full document text is much less useful for the TREC 12 topics than for the TREC 10 topics. This could be a result of variance or small sample sizes, but we believe it is more likely a result of different corpus characteristics.

Representation	TREC 12 Homepage (MRR)	TREC 12 Named-Page (MRR)	TREC 12 Mixed (MRR)
Alt	0.167	0.171	0.169
Font	0.107	0.233	0.170
Full	0.125	0.394	0.260
Link	0.487	0.467	0.477
Meta	0.160	0.083	0.121
Title	0.284	0.416	0.350
URL	0.079	0.122	0.100

Table 3: Performance of individual representations on TREC 12 and hypothetical φ values estimated from test data

The priors were estimated from TREC 10 data and TREC 11 data. We made the assumption that the URLs of homepages in the .GOV corpus would have similar characteristics to those in the WT10G corpus. In addition to using the test topics for the TREC 10 Homepage Finding task, we also used the 80 training topics provided that year. We leveraged the knowledge that there would be an equal number of homepage topics and named-page topics in the test set by scaling our estimates on the posterior P(*type*|NP or HP) to the same number of topics.

There are simpler methods to devise equivalent numbers for ranking purposes, but in the interests of describing what we did with the numbers we used, we present our actual numbers and computations. These numbers are presented in Table 3, and we provide a justification for the method used in Table 3 below. In these derivations NP denotes a named page, HP denotes a homepage, and *t* is a page type (ROOT, SUBROOT, FILE, or PATH). The method of derivation for priors is similar to the approach used in [Kraaij 2002].

Posterior:

$$\mathrm{P}(t|\mathrm{NP} \vee \mathrm{HP}) = \mathrm{P}(t|\mathrm{NP})\mathrm{P}(\mathrm{NP}) + \mathrm{P}(t|\mathrm{HP})\mathrm{P}(\mathrm{HP}) \tag{9}$$

since NP and HP are disjoint.

$$= \frac{c(t,\mathrm{NP})}{2c(\mathrm{NP})} + \frac{c(t,\mathrm{HP})}{2c(\mathrm{HP})} \tag{10}$$

where c(*t*,NP) denotes the count of named pages of type *t* in the training data and c(NP) denotes the number of named page queries in the training data. In this step we approximated the values with training data. Leveraging the fact that we know that homepages and named-pages are equally likely in the test data, we assumed P(NP) = P(HP) = 0.5. What we're substituting in for P(HP) and P(NP) is actually P(correct document is a HP) and P(correct document is a NP). This is fine here, but will have some implications to the correctness of their use for the priors. With a little rewriting, we get our formula for the estimation of the posterior distribution:

$$= \frac{c(t,\mathrm{NP}) + \frac{c(t,\mathrm{HP})c(\mathrm{NP})}{c(\mathrm{HP})}}{2c(\mathrm{NP})} \tag{11}$$

Prior:

$$\mathrm{P}(\mathrm{NP} \vee \mathrm{HP}|t) = \frac{\mathrm{P}(t|\mathrm{NP} \vee \mathrm{HP})\mathrm{P}(\mathrm{NP} \vee \mathrm{HP})}{\mathrm{P}(t)} \tag{12}$$

by Bayes rule. Note that now we are interpreting P(NP or HP|*t*) as P(this document is a NP or HP|*t*) and not P(correct document is a NP or HP|*t*) as we do not know that the document is correct. We believe the document may be correct, but we do not know. This means that substituting in the P(*t*|NP or HP) we estimated above is not the value we should be using here, but we will assume that it is close to the true value we desire. Doing so gives:

$$\approx \frac{\frac{c(t,\mathrm{NP}) + \frac{c(t,\mathrm{HP})c(\mathrm{NP})}{c(\mathrm{HP})}}{2c(\mathrm{NP})}\mathrm{P}(\mathrm{NP} \vee \mathrm{HP})}{\mathrm{P}(t)} \tag{13}$$

We estimate P(*t*) from the training data and discard the constants P(NP or HP) and 2*c*(NP) as our prior is multiplicative and discarding a multiplicative constant will not affect the rankings:

$$\propto \frac{c(t,\mathrm{NP}) + \frac{c(t,\mathrm{HP})c(\mathrm{NP})}{c(\mathrm{HP})}}{c(t)/|collection|} \tag{14}$$

where *c*(*t*) is the number of documents of type *t* in the collection and |*collection*| is the number of documents in the collection. We can also ignore the constant size of the collection:

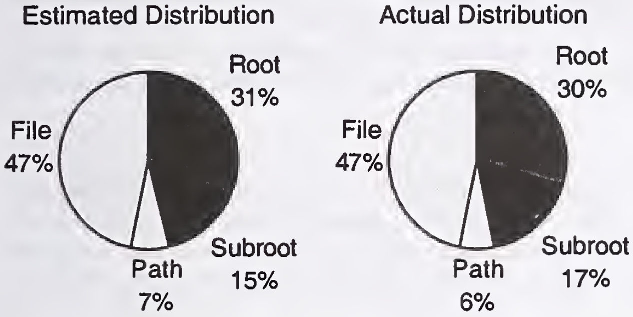
$$\propto \frac{c(t,\mathrm{NP}) + \frac{c(t,\mathrm{HP})c(\mathrm{NP})}{c(\mathrm{HP})}}{c(t)} \tag{15}$$

This gives us the formula we used to estimate the priors.

Class	NP	HP	HP at 170	NP + HP at 170	Posterior	.GOV	Prior
ROOT	5	216	102.0	107.0	0.315	6768	0.0158
SUBROOT	14	75	35.4	49.4	0.145	120923	0.000409
PATH	10	29	13.7	23.7	0.070	65980	0.000359
FILE	141	40	18.9	159.9	0.470	1054082	0.000152
Total	170	360	170.0	340.0	1	1247753	
Computation	From data	From data	HP * 170/360	NP + HP at 170	(NP + HP at 170) / 340	From data	(NP + HP at 170) / .GOV

Table 4: Estimation of prior probabilities and posterior distribution expectations

We note that this method of parameter estimation for the posterior distribution expectation was very accurate given the training data. Figures 1 and 2 show pie charts corresponding to the distribution we estimated and the actual distribution observed in the relevance judgments. We believe it is reasonable to achieve such good estimation of this distribution in practice, as it is a relatively low cost activity to provide assessments for known-item tasks.



Figures 1 and 2: Estimated posterior distribution of page types for correct documents (left) and actual distribution of correct documents (right).

6. Experiments

In this section we describe official and unofficial runs. We submitted four official runs LmrEq, LmrEst, LmrEqUrl, and LmrEstUrl. "Lmr" denotes the Lemur system, "Eq" indicates the φ values were set to be equal to each other, "Est" indicates that the φ were those presented in Table 2, and "Url" signifies that the URL priors presented in Table 4 were applied to the scores. Table 5 summarizes these runs and their performance. LmrFlat and LmrFlatUrl are unofficial runs that do not use document structure or the text of the URL.

Run	Official	Structure	φ	URL Prior	Not found	Found by 10	MRR
LmrEq	YES	YES	equal	NO	8.3 %	83.3 %	0.652
LmrEst	YES	YES	estimated	NO	7.7 %	83.3 %	0.640
LmrFlat	NO	NO	-	NO	11.0 %	78.7 %	0.612
LmrEqUrl	YES	YES	equal	YES	5.3 %	88.0 %	0.713
LmrEstUrl	YES	YES	estimated	YES	4.7 %	89.3 %	0.727
LmrFlatUrl	NO	NO	-	YES	9.3 %	50.7 %	0.315

Table 5: Summary of runs and their performance

Our best performance was achieved when we used the estimated φ in combination with the URL priors. However, when we did not use the URL priors, the estimated φ parameters had worse performance than equally weighted φ values. This raises questions about both the use of URL priors and the method of training the φ values. We recognize that our approach to training the φ values is not optimal or always effective. What we found more interesting was that the URL priors did not help the different runs uniformly, despite the fact that they had similar initial performance. Applying the URL priors to the LmrFlat run severely degraded performance, so this leads us to have questions about the use of URL priors.

7. Discussion of URL Priors

When we apply the priors to the scores returned in a ranking, we do so with the hopes that the reranked lists will match our expectations of the posterior distributions. The fact that the URL priors do help in some cases suggests that we may indeed be observing this behavior when the priors are helping.

To test this hypothesis, we plotted the cumulative distribution of results across all queries. We estimated the probability of a class given a run by counting all documents of a given class up to a rank threshold across all queries and dividing by the number of documents returned by that rank. Figures 4-6 (last page of the paper) show these estimated distributions. The lines without the points are the desired posterior distribution we hoped to achieve by applying the prior probabilities. For LmrEqUrl and LmrEstUrl, we see that the FILE, PATH, and SUBROOT classes are favored more than desired, but the ROOT class is returned less often than we would like. However, we know that performance is increased by applying the priors to the LmrEq and LmrEst runs, and we can see in Figures 4 and 5 that the URL prior brought the results closer to the desired distribution of documents in the ranking.

On the other hand, LmrFlatUrl does not match our expected posterior distribution well (Figure 6). Applying the URL prior to LmrFlat produces very undesirable behavior in the rankings. It is not surprising that the mean reciprocal rank of LmrFlatUrl is much worse than that of LmrFlat. The cause of this behavior is not apparent from the current analysis, but the analysis does suggest a way to assess whether the priors producing a desirable effect on the posterior distributions. This can be done without relevance assessments, so it may be useful during the training and tuning phase of a retrieval experiment.

8. Conclusions

This paper described our TREC mixed homepage/named-page runs. We feel that our performance on this task was very strong. Our basic approach was to use language models estimated using structural information present in the documents to estimate the query generation probability. We found a URL-based prior that others found successful for the TREC 10 homepage finding task was also effective here. We described how we estimated the priors, and provided data analysis as to the effectiveness of the priors.

However, we also demonstrate that the prior probabilities are not producing the desired expected posterior distributions. We provide some analysis suggesting that when the priors harm performance, they produce undesirable effects to the posterior distribution. This analysis is simple to do and may be useful for others when tuning their systems.

For future work, we would like to gain a better understanding of the reason why the priors are not producing desired posterior distributions. One hypothesis is that the distribution of documents in the ranking does not match the distribution of documents in the collection. This may cause any biases present in the original ranking to be present in the reranked results lists. Another cause may be that the scores behave differently for the different classes. In this case, a simple flat prior may not fix the problem. We feel that if we can come up with a solution that produces the desired posterior distribution while preserving as much information in the scores as possible, we may be able to improve on our already strong results.

9. Acknowledgements

This work was sponsored in full by the Advanced Research and Development Activity in Information Technology (ARDA) under its Statistical Language Modeling for Information Retrieval Research Program. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors, and do not necessarily reflect those of the sponsor.

10. References

[Lemur] The Lemur Toolkit. http://www.cs.cmu.edu/~lemur

[Craswell 2001] N. Craswell, D. Hawking, and S. Robertson. Effective site finding using link anchor information. In *Proceedings of the Twenty-Fourth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01)*, 2001.

[Kraaij 2002] W. Kraaij, T. Westerveld, and D. Hiemstra. The importance of prior probabilities for entry page search. In *Proceedings of the Twenty-Fifth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'02)*, 2002.

[Ogilvie 2003] P. Ogilvie and J. Callan. Combining document representations for known-item search. In *Proceedings of the Twenty-Sixth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'03)*, 2003.

[Zhai 2001] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the Twenty-Fourth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01)*, 2001.

[Zhang 2003] M. Zhang, R. Song, C. Lin, S. Ma, Z. Jiang, Y. Jin, Y. Liu, and L. Zhao. THU TREC2002 Web Track Experiments. In *Proceedings of the Eleventh Text Retrieval Conference, TREC 2002*, 2003.

[Zhai 2002] C. Zhai and J. Lafferty. Two-stage language models for information retrieval. In *Proceedings of the Twenty-Fifth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'02)*, 2002.

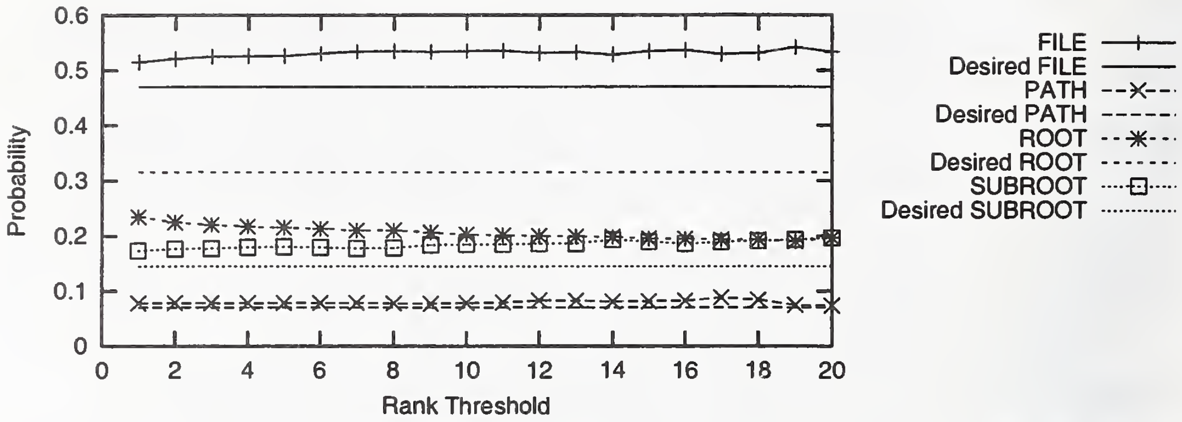


Figure 4: Posterior distribution for **LmrEqUrl**. Applying the URL prior to LmrEq gives result lists are more towards biased towards the FILE and SUBROOT classes than desired, and less biased toward the ROOT class then desired.

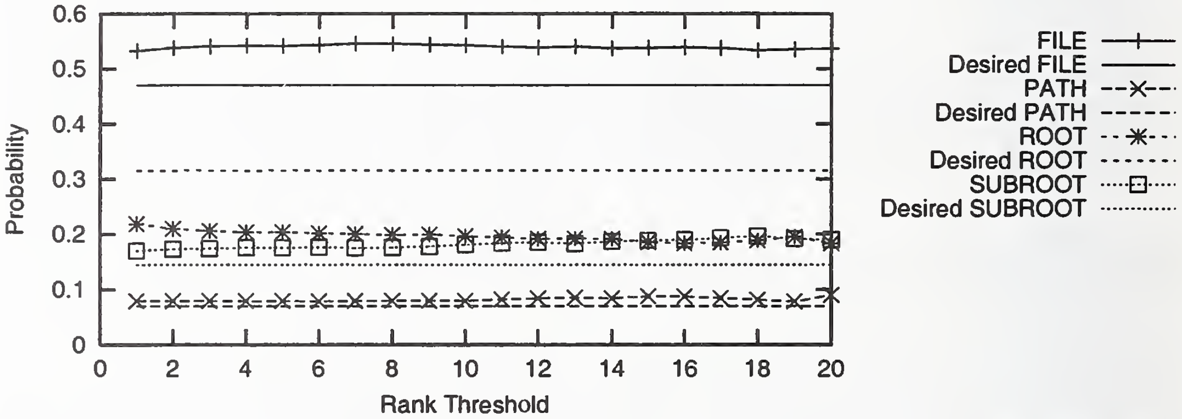


Figure 5: Posterior distribution for **LmrEstUrl**. The trends here are similar to those in Figure 4 for LmrEqUrl.

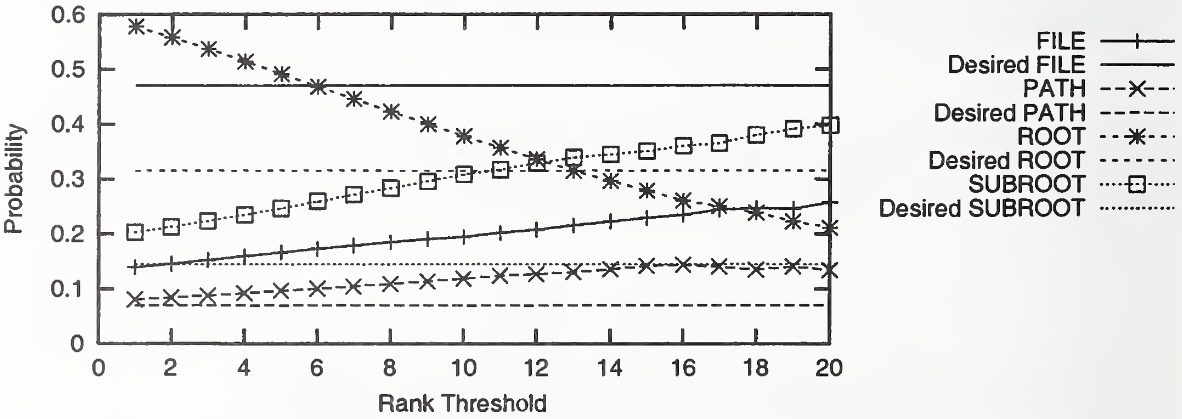


Figure 6: Posterior distribution for **LmrFlatUrl**. Applying the URL priors to LmrFlat results in a heavy bias towards ROOT pages at early ranks which decays rapidly. There is a trend towards an increasing bias toward the SUBROOT class. The FILE class has a much stronger bias against it then desired, and the PATH class has a stronger bias towards it then desired. The bias against the FILE class may account for the poor performance.

A Hybrid Approach for QA Track Definitional Questions

Sasha Blair-Goldensohn **Kathleen R. McKeown** **Andrew Hazen Schlaikjer**
Department of Computer Science
Columbia University
New York, NY 10027
{sashabg,kathy,hazen}@cs.columbia.edu

Abstract

We present an overview of DefScriber, a system developed at Columbia University that combines knowledge-based and statistical methods to answer definitional questions of the form, "What is X?" We discuss how DefScriber was applied to the definition questions in the TREC 2003 QA track main task. We conclude with an analysis of our system's results on the definition questions.[1]

1 Introduction

In recent years, the QA systems in TREC have reached a remarkably high level of performance (Voorhees, 2002). Until this year, however, the task has focused on the short-answer, or *factoid* model, in which the goal is to answer questions for which the correct response is a number, short phrase, or sentence fragment. In this paper, we focus on the newly introduced *definitional* question type and present the results of a system which we have built to answer such questions.

Why build a system that is specific to definitional questions? Consider a student asked to prepare a report on the Hajj, an Islamic religious duty. In the context of short-answer QA, both patience and prescience will be required to elicit the core facts. First, a relatively long list of sub-questions would be required (e.g., "Where is the Hajj carried out?", "How long does it last?", "Who undertakes a Hajj?" etc.). Second, knowing which questions to ask requires knowledge that the questioner likely does not have. That is, the questions that best elicit a description of one thing (e.g., the Hajj) can be quite different from those best suited for finding out about something else (e.g., the Caspian Sea).

Instead, it is useful to have a sytem which can answer "What is X?" questions directly, presenting a comprehensive response which effectively combines the answers to the relevant sub-questions. This capability is a valuable complement to static knowledge sources like encyclopedias, especially in answering questions about an "X" whose meaning may be evolving, or in creating custom answers that focus on particular aspects of a definition.

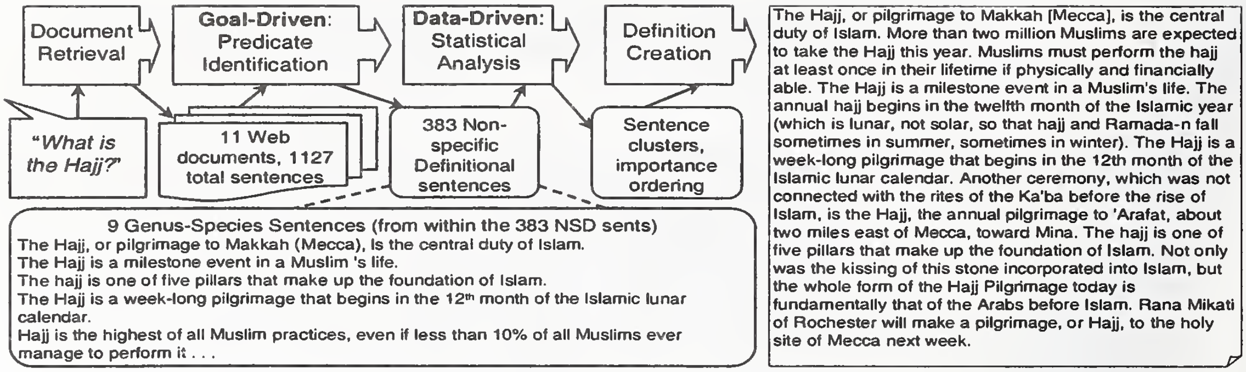
The remainder of this paper presents DefScriber, a definitional QA system implemented at Columbia University. We first present a brief overview of the system's architecture and previous evaluations (more detail on these topics has been reported previously (Blair-Goldensohn et al., 2003)). We then focus on our performance on the 50 definitional questions included in this year's TREC QA main task.

2 Architecture Overview

Figure 1 gives a high-level view of DefScriber's operation, illustrating input and output of each stage. This example traces an actual answer generated for the question "What is the Hajj?"

The input is specified as a question, which feeds into the document retrieval phase. The user may specify which databases to search, a maximum number of documents to retrieve, and the desired answer length.

[1] We did not produce answers to the other, non-definitional, questions.


Document Retrieval
Goal-Driven: Predicate Identification
Data-Driven: Statistical Analysis
Definition Creation
"What is the Hajj?"
11 Web documents, 1127 total sentences
383 Non-specific Definitional sentences
Sentence clusters, importance ordering
9 Genus-Species Sentences (from within the 383 NSD sents)

The Hajj, or pilgrimage to Makkah (Mecca), Is the central duty of Islam.
The Hajj is a milestone event in a Muslim 's life.
The hajj is one of five pillars that make up the foundation of Islam.
The Hajj is a week-long pilgrimage that begins in the 12th month of the Islamic lunar calendar.
Hajj is the highest of all Muslim practices, even if less than 10% of all Muslims ever manage to perform it . . .

The Hajj, or pilgrimage to Makkah [Mecca], is the central duty of Islam. More than two million Muslims are expected to take the Hajj this year. Muslims must perform the hajj at least once in their lifetime if physically and financially able. The Hajj is a milestone event in a Muslim's life. The annual hajj begins in the twelfth month of the Islamic year (which is lunar, not solar, so that hajj and Ramada-n fall sometimes in summer, sometimes in winter). The Hajj is a week-long pilgrimage that begins in the 12th month of the Islamic lunar calendar. Another ceremony, which was not connected with the rites of the Ka'ba before the rise of Islam, is the Hajj, the annual pilgrimage to 'Arafat, about two miles east of Mecca, toward Mina. The hajj is one of five pillars that make up the foundation of Islam. Not only was the kissing of this stone incorporated into Islam, but the whole form of the Hajj Pilgrimage today is fundamentally that of the Arabs before Islam. Rana Mikati of Rochester will make a pilgrimage, or Hajj, to the holy site of Mecca next week.

Figure 1: DefScriber answers "What is the Hajj?"

Currently, DefScriber is able to search the Internet via Google, as well as the TREC-11 and CNS[2] collections, which are indexed locally.

The information retrieval (IR) module uses a fixed set of patterns to identify the term to be defined in the question, and then generates a set of search queries. These queries are sent to the selected search engine in order of decreasing expected precision until a threshold number of documents has been retrieved or the set of queries has been exhausted.

Once documents are retrieved, the primary goal-driven step is performed, with the system examining documents for instances of definitional predicates. Next, a data-driven analysis produces sentence clustering and ordering information. In the last step, a definitional answer is created via sentence extraction, guided by the results of the goal- and data-driven stages.

3 Definitional Predicates: A Goal-Driven Approach

Answering a "What is X?" definitional question and creating a summary of query results for the search term "X" are strongly related problems. Yet, as readers, we have more specific expectations for a definition than for a general-use summary. The idea of definitional predicates is to model these special properties of a definition so the system can use them to create better answers.

3.1 The Predicate Set

Our set of definitional predicates is shown in Table 1. Currently, the system automatically identifies instances of three of these types in text: Genus, Species and Non-specific Definitional (NSD). Research on identifying Target Partition and History instances is ongoing.

An important distinction is that NSD subsumes all of the other more specific predicate types that appear underneath it in Table 1. Thus, identifying NSD text is crucial because it is a cue to the presence of other predicates; it also removes noise and provides a set of useful definitional text which is given as input to data-driven methods even when the text cannot be further classified with a more specific predicate. We chose Genus and Species as the first specific predicates to implement because they are at the core of what definitions are; Related work (Sager and L'Homme, 1994; Swartz, 1997; Sarner and Carberry, 1988) consistently identifies these two concepts as key parts of defining a term.

3.2 Automatic Predicate Identification

To use these predicates in our system, we must identify units of text which contain them. To do this, we first did a manual examination of documents to create sample data annotated with predicates. Using this data, we explored two approaches to identifying predicates. The first uses machine learning to learn a feature-based classifier that predicts when a predicate occurs. The second uses pattern-recognition over patterns extracted from the annotated data.

We used the machine learning approach to automatically identify NSD sentences. Using a set of

[2] A collection of documents from the Center for Nonproliferation Studies (http://cns.miis.edu) made available to participants in the AQUAINT project.

Predicate	*Description*	*Instance Example*
Non-specific Definitional	Any type of information relevant in a detailed definition of the term. NSD are a superset of the below predicates.	Costs: Pilgrims pay substantial tariffs to the occupiers of Makkah and the rulers of...
Genus	Category to which term belongs.	The hajj is a type of ritual.
Species	Describes properties other than or in addition to Genus. Species are a superset of the below predicates.	The annual hajj begins in the twelfth month of the Islamic year.
Target Partition	Divides the term into two or more conceptual or physical parts.	Qiran, Tammatu' and Ifrad are three different types of Hajj.
Cause (effect)	States explicitly that the term is the cause (effect) of something.	The pilgrimage causes the past sins of a Muslim to be forgiven.
History	Gives historical information relating to the term.	Mohammed, founder of Islam, started the tradition in 632 C.E.
Etymology	Information on the term's genesis, e.g. adaptation from another language.	In Arabic, the word Hajj means a resolve of magnificent duty.

Table 1: Definitional Predicates: Descriptions and Examples

surface features such as sentence position (relative and absolute in a document) "term concentration" (i.e. the term's frequency within a sentence and/or nearby sentences), we applied two machine learning tools: the rule-learning tool Ripper (Cohen, 1995) and the boosting-based categorization system BoosTexter (Schapire and Singer, 2000). Both algorithms performed similarly in terms of the accuracy of their predictions on test data; Ripper's rules are used in DefScriber since they were somewhat simpler to implement. Using cross-validation, accuracy of 81 percent was obtained with Ripper (76 percent using BoosTexter).

In order to identify Genus and Species predicates, we manually extracted a set of lexicosyntactic patterns to model sentences containing both Genus and Species (G-S) information, as these G-S sentences provide a strong grounding context for understanding the term. Rather than modeling the patterns at the word level, i.e. as flat templates with slots to fill, we model them as partially specified syntax trees (Figure 2). One such pattern can match a large class of syntactically similar sentences without having to model every type of possible lexical variation. This approach derives from techniques used in information extraction (Grishman, 1997), where partial subtrees for matching domain-specific concepts and named entities are used because automatic derivation of full parse trees is not always reliable. However, data-driven techniques (Section 5) offer additional protection from false or extraneous matches by lowering the importance ranking of information not corroborated elsewhere in the data.

Figure 2 illustrates the transformation from example sentence to pattern, and then shows a matching sentence. Our patterns are flexible - note that the example and matched sentences have somewhat different trees. Another point of flexibility is the verb itself; FormativeVb will match verbs in a set which our algorithm considers expressive of "belonging" to a category (e.g., "be," "represent," "exemplify").

Using our predicate-annotated data set, we have manually extracted 23 distinct patterns which match G-S sentences. Although it is difficult to reliably measure recall of the patterns without a larger set of annotated documents, precision in previous evaluations was approximately 96 perecent.

4 Data-Driven Techniques: Applying Summarization

While our set of predicates, including Genus and Species, are domain-neutral, they are not meant to model all possible important information for a given term definition. Some information types may be hard to define computationally *a priori*. Also, a given sentence may instantiate a definitional predicate but include only peripheral content. We address these issues in the data-driven stage of DefScriber's pipeline (Figure 1), applying statistical techniques adapted from multi-document summarization to the Non-specific Definitional sentences identified in the goal-driven stage.

First, a definition centroid is computed by creating a stemmed-word vector of all the NSD sentences. Then the individual sentences are sorted in order of decreasing "centrality," as approximated by IDF-weighted cosine distance from the definition centroid. This method creates a definition of length N by taking the

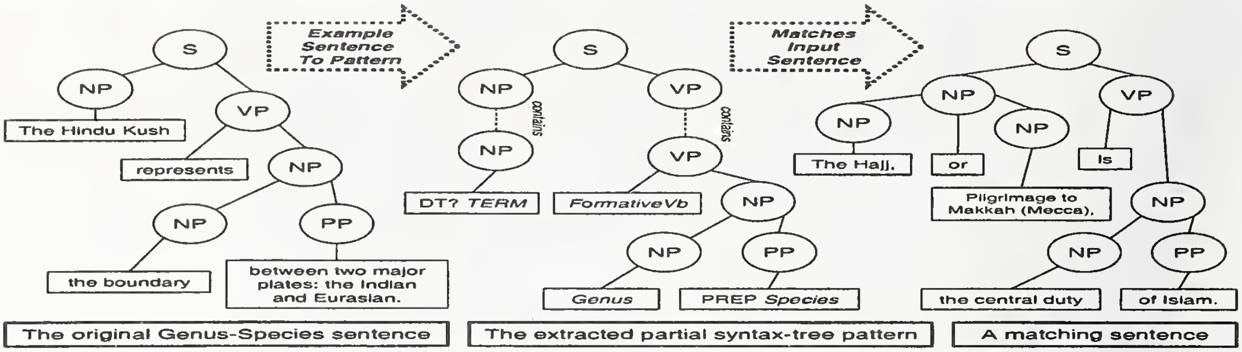


Figure 2: Pattern extraction and matching for a Genus-Species sentence from an example sentence.

first N unique sentences out of this sorted order, and serves as the TopN baseline method in our evaluation. Note that this method approximates centroid-based summarization, a competitive summarization technique (Radev et al., 2000).

After ordering sentences with TopN, we perform a non-hierarchical clustering that we use to decrease redundancy by avoiding same-cluster sentences in the answer. Since our clustering similarity measure uses IDF computed over a large collection, it can suffer from overweighting of specialized terms; to account for this, we augment the cosine distance calculation, using local IDF values calculated dynamically from the pool of NSD sentences.

The final data-driven technique improves cohesion by considering the content of the previous answer sentence when choosing a sentence to add to the answer. After choosing the first sentence as in TopN, we choose each remaining sentence as follows: we define the goodness of a cluster as an equal weighted combination of (1) its cohesion to the previous sentence and (2) its overall importance, approximated by that cluster's centroid's distance from (1) the previously chosen sentence's cluster's centroid and (2) the centroid of all NSD sentences. We also add in a penalty for clusters from which sentences have already been chosen such that no cluster gets n sentences included in the answer before all clusters have $n-1$ included sentence. Once the "best" next cluster has been chosen in this manner, we add the next sentence to the definition as the top-ranked sentence in that cluster which has not yet been included in the definition.

DefScriber's default configuration integrates all the above data-driven techniques (TopN, clustering, local IDF weighting, and cohesion ordering), combining them with the goal-driven method of G-S predicate identification. We place the top-ranking (in terms of TopN) G-S sentence first in the definition, and use the cohesion-based ordering to add the remaining sentences. We call this integrated goal- and data-driven method DefScriber.

5 Related Work

Goal-driven, or top-down, approaches are more often found in generation. Schemas (McKeown, 1985), rhetorical structure theory (Mann and Thompson, 1988; Moore and Paris, 1992; Hovy, 1993; Marcu, 1997) and plan-based approaches (Reiter and Dale, 2000) are examples of goal-driven approaches, where the schema or plan specifies the kind of information to include in a generated text. In early work, schemas were used to generate definitions (McKeown, 1985), but the information for the definitional text was found in a knowledge base. In more recent work, information extraction is used to create a top-down approach to summarization (Radev and McKeown, 1998) by searching for specific types of information which can be extracted from the input texts (e.g., perpetrator in a news article on terrorism). Here, the summary briefs the user on domain-specific information assumed a priori to be of interest.

Other long-answer QA systems are currently under development as part of the AQUAINT program (Voorhees, 2003). Some of these share attributes with DefScriber; Weischedel et al.(ARD, 2003) explore definitional and biographical questions, using a combination of methods that are largely complementary to those used in DefScriber, namely identification of key linguistic constructions and information extraction (IE) to identify specific types of semantic data.

Another important contrast between DefScriber and most of the long-answer systems developed under the AQUAINT program has to do with answer format. While these systems mostly produce answers as a ranked list of descriptive phrases or sentences, DefScriber uses summarization methods to produce a coherent, multi-sentence, encyclopedia-style definition.

6 Previous Evaluations

An evaluation of DefScriber performed previously used human judgments to measure the performance of DefScriber's definitions over a set of 24 terms from a varied set of domains. We measured five qualities of the definitions: relevance (precision), redundancy, structure, breadth of coverage, and term understanding. Overall, we found that DefScriber achieved the best scores in structure, redundancy, term understanding, and relevance, with statistically significant margins in the first two categories. In coverage, DefScriber performed below the baselines, but not at a statistically significant level. The results are reported in detail elsewhere (Blair-Goldensohn et al., 2003).

7 Modifications for TREC 2003

Although DefScriber is built specifically to answer definitional questions, several modifications and optimizations were performed before running the system for the definitional questions in the TREC QA task.

First, the data source to use needed to be fixed: usually, a user of DefScriber's web interface would specify whether to query Internet documents or local collections. For the TREC question set, we hard-coded this value so that only the TREC-11 data sets were searched.

Secondly, we needed to reconsider our metric for a "good" answer in light of the announced scoring formula. Since each answer for a definition question was to be considered on the basis of its intrinsic information content alone, the statistical cohesion measures described in Section 4 were disabled. Clustering was still used to avoid redundancy, since redundant information nuggets would receive a zero score. But answer sentences were picked from the clusters in a purely importance-based order (as approximated by our TopN ordering), without regard to cohesion.

Another point of modification was the issue of handling "Who is X?" as opposed to "What is X?" questions, since both types were included in the TREC definition question set. Although DefScriber has been designed primarily to provide definitions of objects and concepts[3], its design allows "Who" questions to be processed easily as well. In fact, a look at the predicates in Table 1 reveals that they can be applied to people, for instance we can and do identify the sentence, "John Glenn was the first astronaut." as a G-S sentence even though its subject is a person. The single difference in DefScriber's processing of "Who" as opposed to "What" questions is that sentences which include certain personal pronouns like "he" or "she" do not have their score reduced as they would for a "What" question.

Lastly, we needed to decide how many answer sentences to include for each definitional answer. Our current system takes this number as a user-specified parameter, but in this case we needed the system to try to determine an optimal value. Our approach was to use the training data provided by the AQUAINT pilot study (Voorhees, 2003), and to optimize a linear combination of a base answer length and an adjustment factor based on the number of relevant documents (i.e. containing one or more NSD sentence) found for a particular answer. We did this by using the assessor nuggets for the 25 pilot definition questions, and calculating what our score would have been if our answer length were determined as a linear function of the number of relevant documents found. We approximated this optimum as:

$$\underset{base,factor \in 1..20}{max} \underset{q \in 1..25}{avg} (F(q, base + docs(q)/factor))$$

Where $F(q, n)$ is the TREC F-measure score for DefScriber's n-sentence answer on pilot question q, and $docs(q)$ is the number of relevant documents found for question q. The optimum was found at $base = 9, factor = 16$. Therefore, the final modification of our system for the TREC task was to set it to produce $(9 + docs(q)/16)$-sentence answers for each definitional question q.

8 Performance on TREC 2003 Definition Questions

As mentioned previously, our system is designed specifically to answer definitional questions and as

[3] This is in part because a separate, complementary system with greater focus on properties specific to describing individuals, i.e. biographies, is under development at Columbia (Duboue and McKeown, 2003)

QID	*Question*	*Official Nugget*	*Matching Response?*
1901	Who is Aaron Copland?	established home for composers	Music from the Copland House made its debut Sept. 29 at Merkin Concert Hall with, appropriately, an all-Copland program.
1905	What is a golden parachute?	Agreement between companies and top executives	William M. Mercer Inc., the consulting firm, has found that 64 percent of 350 large publicly traded companies provide financial protection for one or more key executives, most often the chief executive, if the company changes control.
2274	Who is Alice Rivlin?	vital financial assistance Authority for DC	She was too busy overseeing the city government of the District of Columbia as chairman of its financial control board – and serving as vice chairman of the Federal Reserve Board in her day job – to accept the plaque.
1907	Who is Alberto Tomba?	two time world cahmpion	Italian Alberto Tomba, three-time Olympic and two-time world champion, came back to win the last World Cup slalom in Schladming, Austria, on Thursday before the world Alpine championships.

Table 2: Sentences from DefScriber's output that were judged as non-matching alongside possibly matching nuggets. These non-matches demonstrate the ambiguous nature the judgement matching proces; the first two sentences are arguable matches but require significant inference on the judge's part; the latter two seem to be clearer matches.

such was run only for the definition questions in the main QA track. Thus we will focus on our results for these questions.

Overall, our system performed well above the median for these questions, achieving an average F score of .338 compared with the median of .192. Examining the evaluation results further, we made a number of observations about the evaluation in general and our performance in particular.

An initial study of evaluation results showed that some data nuggets present in our response sets were not counted by judges, resulting in degraded recall scores. These judgements may be due in part to the need for higher level inference over response sentences and nuggets to see their connections. The issue of whether such inference is appropriate may have been a source of considerable noise in the evaluation. Table 2 gives several examples of response sentences from our output which were not scored as containing an official nugget, alongside the nugget which they might arguably have been matched with, indicating potential judgment errors. These examples are meant to show the gradient of answer-matching ambiguity from more to less ambiguous. That is, the first two answers would seem to require some level of inference on the part of the judge to be certified a match; the latter two seem more clearly to include the desired nugget.

A strongly related issue, particularly for a system like DefScriber which produces a multi-sentence answer meant to be read as a whole, is the issue of many-to-one matches. That is, should judges count a nugget as "matching" when its information is not contained in a single answer sentence, but rather in the sum of information provided by several answer sentences?

DefScriber also encountered difficulty with certain questions because of its reliance (at the time of the evaluation) the MG search engine, which lacks support for phrasal queries. Lack of phrasal search resulted in low precision, coupled with limitations on the number of documents processed resulted in low recall. This problem became more pronounced in cases where one or more words in term/person to be defined was common, resulting in a large set of documents being returned from MG, which does a boolean OR across all query words. Due to speed limitations, our system truncated such large results sets at a fixed size, and thus found only a subset of the documents which actually contained the term words in a phrase. This resulted in problems, for instance, on questions of the type "Who is X?", where X had a very common first and/or last name (e.g. "Al Sharpton", and "Andrea Bocelli"). Subsequently, updates have been made to DefScriber's IR module so that it now fully supports phrasal search capabilities on locally indexed corpora via the Lucene search engine.

For questions where this was not an issue, DefScriber's sentence selection criteria seem to have performed well, both goal- and data-driven. In some of our higher scoring answers, we see an impact from our goal-driven strategy via the identification of G-S sentences; for instance, we can see in Figure 3 that Def-

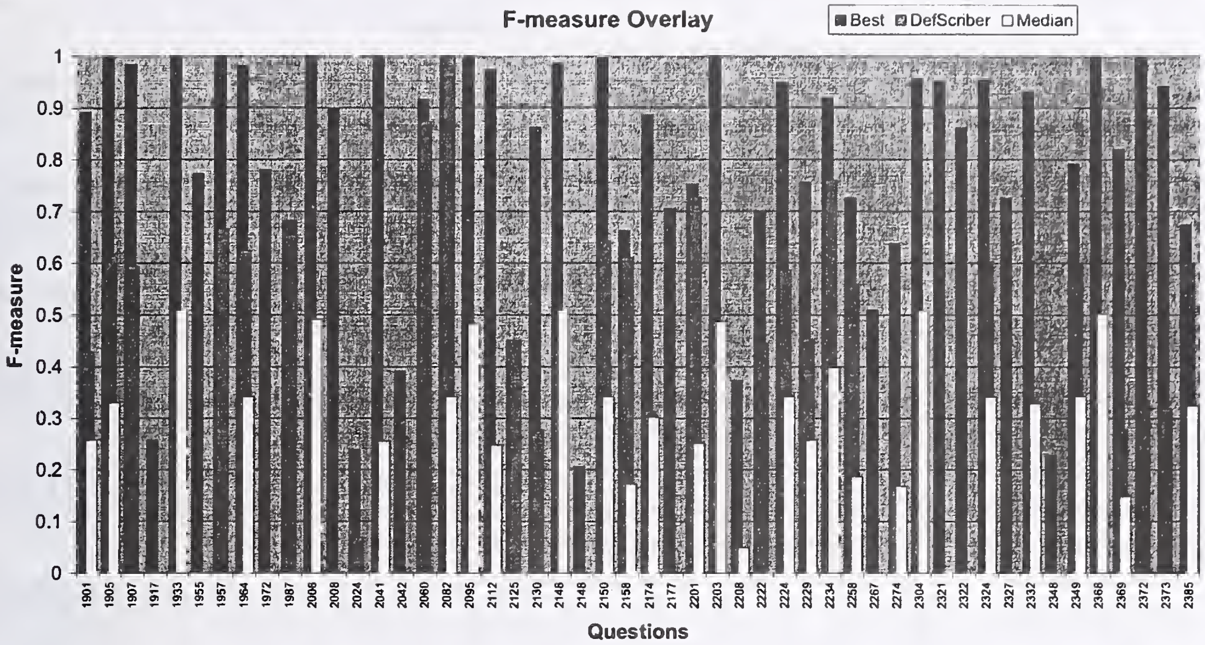


Figure 3: Overlay of evaluation-wide F-measure scores per question with those of DefScriber

Scriber achieves the highest score on question 1987, “What is ETA in Spain?”, which appears to be one of the harder questions in that its median score was zero. Our high score on this question is in part due to finding and including the G-S sentence, “ETA is an acronym for Basque Homeland and Freedom in the Basque language.”, which contains one of the “vital”information nuggets for this question, i.e. the information on what ETA stands for.

While G-S sentences are clearly helpful, we were also successful when G-S sentences were not found. In these cases, we rely on our robust data-driven methods to statistically guide answer content. These methods allowed us to select high-scoring sentences even when G-S sentences were not found. Such instances included our best and near-best scores on questions 2060 (“Who is Alberto Ghiorso?”), 2082 (“Who was Anthony Blunt?”), 2125 (“Who was Charles Lindbergh?”) and 2201 (“What is Bollywood?”).

However, even when IR returned relevant documents, we did see degradation in system performance where high numbers of response sentences were returned. We believe this is due to the precision penalties our system suffered by using the adjustable answer-length threshold explained in Section 7. Since this threshold creates a longer answer when more relevant sentences are found, our lower scores in these cases suggest that the penalties we incurred in precision did not make up for whatever additional recall nuggets we achieved by having longer answers; it would be interesting to see if the answer-length optimization described in the Section 7 would arrive at a smaller length function given the new data from this evaluation.

As suggested by the zero median F-measure of question 2024 “Who is Andrea Bocceli”, few participants in the evaluation have incorporated fuzzy search capabilities to overcome spelling errors in input questions (the singer’s name is correctly spelled “Bocelli”). From an IR prespective, this represents a very important advance that most systems should make in order to function adequately with noisy data from source materials and/or search inputs.

For future evaluations where nuggets of information are to be identified by human judges, it may be useful to perform some error analysis of adjudications made this year. Given the subjective nature of the task, attaining a “perfect” scoring is of course impossible. But an analysis of the kinds of errors or issues seen will be important as we seek to refine the design of the definition question task and the judgement process itself.

9 Future Work

Future work on DefScriber will concentrate on increasing the number of definitional predicates automatically identified by the system, as well as on improving identification performance on such predicates.

We are currently working to improve our feature-based predicate identification methods by growing our annotated data set while also extracting more and richer features to input into our machine learning methods. To improve the pattern-based methods, we are actively working with IE bootstrapping techniques developed in Snowball (Agichtein and Gravano, 2000) to automatically learn predicate patterns from manually extracted "seed" examples. Such techniques would allow us to supplement our manually-generated patterns and bring new predicates online more quickly.

10 Conclusion

We have presented an overview of DefScriber, a hybrid goal-driven and data-driven system for definitional questions. We explained how the system was modified and applied to answer definitional questions in the TREC 2003 QA track. Finally, we presented DefScriber's results on the definitional questions, which were significantly above median, achieving an average F-score of .338 compared with the median of .192. Finally, we analyzed our scores on certain individual questions, discussing areas where our system performed well and others where it could be improved, as well as noting several issues of the judgement process itself.

Acknowledgements

The authors wish to acknowledge the support of ARDA through AQUAINT contract MDA908-02-C-0008. We are also thankful for the generous and thoughtful contributions of colleagues at Columbia University and at the University of Colorado-Boulder.

References

ARDA and NIST. 2003. *AQUAINT R&D Program 18 Month Workshop*, San Diego, CA.

Sasha Blair-Goldensohn, Kathleen R. McKeown, and Andrew Hazen Schlaikjer. 2003. A hybrid approach for answering definitional questions. Technical Report CUCS-006-03.

William W. Cohen. 1995. Fast effective rule induction. In *Proc. of 12th Int'l Conf. on Machine Learning*, pages 115–123.

Pablo A. Duboue and Kathleen R. McKeown. 2003. Statistical acquisition of content selection rules for natural language generation. In *EMNLP 2003*.

Ralph Grishman. 1997. Information extraction: Techniques and challenges. In *SCIE*, pages 10–27.

Eduard H. Hovy. 1993. Automated discourse generation using discourse structure relations. *Artificial Intelligence*, 63(1-2):341–385.

William C. Mann and Sandra A. Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.

Daniel Marcu. 1997. Rhetorical parsing of natural language texts. In *In Proc. ACL–EACL 97*, pages 96–103.

Kathleen R. McKeown. 1985. *Text Generation*. Cambridge Univ. Press.

Johanna D. Moore and Cecile L. Paris. 1992. Planning text for advisory dialogues: Capturing intentional and rhetorical information. *Comp Ling*, 19(4):651–695.

Dragomir R. Radev and Kathleen R. McKeown. 1998. Generating natural language summaries from multiple on-line sources. *Comp Ling*, 24(3):469–500.

E. Reiter and R. Dale, 2000. *Building Natural Language Generation Systems*, chapter 4. Cambridge Univ. Press.

Juan C. Sager and M.C. L'Homme. 1994. A model for definition of concepts. *Terminology*, pages 351–374.

Margaret Sarner and Sandra Carberry. 1988. A new strategy for providing definitions in task oriented dialogues. In *Proc. Int'l Conf. on Computational Linguistics (COLING-88)*, volume 1.

Robert E. Schapire and Yoram Singer. 2000. BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168.

Norman Swartz. 1997. Definitions, dictionaries and meanings. Available online http://www.sfu.ca/philosophy/definitn.htm.

Ellen Voorhees. 2002. Overview of the trec 2002 question answering track. In *Text Retrieval Conference 2002*, Gaithersburg, MD. NIST.

Ellen Voorhees. 2003. Answers to definition questions. In *HLT-NAACL 2003*, pages 109–111, Edmonton, Canada.

TREC12 Web and Interactive Tracks at CSIRO

Nick Craswell[1], *David Hawking*[1] *and Trystan Upstill*[2]
nick.craswell@csiro.au, david.hawking@csiro.au and trystan.upstill@cs.anu.edu.au

Alistair McLean[1], *Ross Wilkinson*[1] *and Mingfang Wu*[1]
alistair.mclean@csiro.au, ross.wilkinson@csiro.au, mingfang.wu@csiro.au

[1]CSIRO ICT Centre, Australia

[2]Department of Computer Science, CSIT Building, ANU

Canberra, ACT 0200, Australia

1. Introduction

This year, CSIRO teams participated in all three tasks of the web track; these being: the automatic topic distillation task, the home/named page finding task and the interactive topic distillation task. This paper describes our approaches, experiments and results. The following section describes our experiments in the two automatic tasks, and Section 3 describes our experiment in the interactive task.

2. The web track

CSIRO submitted a total of 10 runs to the non-interactive portion of the 2003 Web Track - 5 runs for home/named page finding and 5 runs for Topic Distillation. The runs are labeled csiro03[TYPE][RUNID], where TYPE is ``ki" for known item runs and ``td" for topic distillation runs.

This year we focused on tuning Okapi BM25 and Web evidence parameters. Our home/named page finding submissions use tunings computed for both home and named page finding, and evaluate two run combination methods. Our topic distillation submissions are tuned for home page finding only and test whether the Web evidence evaluated is useful, and whether the use of stemming improves performance.

We did not incorporate PageRank or simple indegree this year because of previously observed poor performance for named page finding and homepage finding. Instead our query-independent Web evidence included URL length and two important sub-types of indegree (off-site and on-site).

Throughout our experiments we tuned Okapi BM25 (through the k_1 and b parameters), anchor-text weighting and other query independent Web evidence. The parameters were tuned using a hill climbing algorithm, with complete exploration of 2 parameters at a time, on a grid computer consisting of cluster of 20 dual processor Intel Xeon machines.

2.1. Home/named page finding

We submitted runs based on three tunings (for a home page task, a named page task and both at the same time), and evaluated two combination methods. We trained using last year's .GOV named page finding query/result set, and using a home page finding training set derived from a first .GOV resource listing.

We submitted runs tuned for both home page and named page finding at the same time (csiro03ki01), tuned for named page finding only (csiro03ki02) and tuned for home page finding only (csiro03ki03). We also submitted two combinations of these runs. The first was an interleaved run (csiro03ki04 -- interleaving csiro03ki02 and csiro03ki03), and the second a run that summed scores achieved in both rankings (csiro03ki05). A summary of our home/named page finding submissions, and their retrieval effectiveness is presented in Table 1.

Table 1: Home/named page submissions summary. To aid our understanding of retrieval performance we computed ARR for home pages only ``ARR (HP)" and named pages only ``ARR (NP)." We also computed a further run post-hoc (csiro03kins)

Run	*Description*	*ARR*	*S@10 (%)*	*ARR (HP)*	*ARR (NP)*
csiro03ki01	Tuned for HP and NP	0.692	83.7	0.815	0.569
csiro03ki02	Tuned for HP	0.603	77.7	0.774	0.432
csiro03ki03	Tuned for NP	0.702	84	0.755	0.649
csiro03ki04	HP and NP tunings interleaved (HP then NP) w/q.class	0.667	86.3	0.801	0.532
csiro03ki05	HP and NP tunings combined	0.699	81	0.812	0.586
csiro03kins	HP and NP tunings interleaved (NP then HP)	0.717	87	0.781	0.651

Our results show that tuning specifically for the home page finding task significantly harmed our named page retrieval effectiveness (csiro03ki02 vs. csiro03ki03). Our highest ARR was achieved using the NP-only tuning, whilst the best S@10 used interleaved lists from HP and NP tunings. The results report that an overemphasis on home page finding evidence can hinder named page searches.

The run with the highest S@10 (csiro03ki04) interleaved the csiro03ki02 and csiro03ki03 runs (i.e. top HP, top NP, second HP, second NP etc.). To improve early precision, if we encountered a keyword that strongly indicated a named page query[1] was occurring we led with the top NP, rather then the top HP result. From further post-hoc evaluations (see csiro03kins) we determined that leading with NP rather than HP would have further improved precision (achieving an ARR improvement of 0.717). In summary, interleaving HP then NP without query classification achieves an ARR of 0.646. Interleaving HP then NP with swapping if query appears to be a named page query achieves an ARR of 0.667. Finally, interleaving NP then HP without query classification achieves 0.717.

We could not find a single tuning that is equally useful for each type of query. This raises interesting query classification, or further combination of evidence questions. A superior classifier may well have taken into account other evidence, such as query length, while a better combination may have taken into account the strength of the home page evidence (and only led with a homepage result if it was sufficiently strong).

There are some limitations inherent in the training sets we used for tuning. The set of home pages was taken from a .GOV portal, which may inadvertently have favored prestigious, or larger and more popular home pages. Further, last year some of the named pages were in fact home pages, whereas this year there was a distinction made between named pages and home pages. Our named page tuning was therefore based on a mixed query set with a smaller ratio of home pages. This may have slightly biased our training towards home page queries.

2.2. Topic Distillation

Our Topic Distillation runs were based on the home page tunings built for the home/named page task. The run results are presented in Table 2.

[1] Query terms were selected from last years query set and included terms such as ``"page", "form" and "2000""

Table 2: Topic distillation submissions summary. Post-hoc we computed a run based using the name page tunings (csiro03tdns)

Run	*Description*	*Average R-Prec*
csiro03td01	Tuned for HP	0.1438
csiro03td02	Tuned for HP without query-ind. hyperlink evidence	0.1162
csiro03td03	Tuned for HP with stemming	0.1636
csiro03td04	Tuned for HP without anchor evidence	0.0988
csiro03td05	Tuned for HP with "normal" bm25 tuning (k_1=2, b=0.75)	0.1217
csiro03tdns	Tuned for NP	0.1166

Our best run (csiro03td03) used the home page tuning and incorporated stemming. When removing hyperlink evidence (i.e. csiro03td02 and csiro03td04) we observed a decrease in retrieval performance. Likewise, we observed a 2% decrease in performance when using standard tunings for Okapi BM25 (csiro03td01 vs. csiro03td05). Post-hoc we computed a new run based on the named page finding tunings used in our home/named page finding submission (csiro03tdns), this tuning reduced the Avg R-Prec to 0.1166.

The results from the topic distillation task appear to support the notion that our home page training set favored prominent resources (an advantage for Topic Distillation). Further, our results illustrate the usefulness of web evidence, and stemming, when addressing Topic Distillation.

3. The Interactive Sub-track

In this year's interactive sub-track, searchers were asked to construct a resource list that covers all major aspects of a broad topic through interaction with an information access system. Similar to that in automatic topic distillation task [1], a key resource page is defined as a main page of a website which is:

1. principally devoted to the topic,
2. providing credible information on the topic, and
3. not part of a larger site also devoted to the same topic.

Take the topic "adoption procedures" and the website <http://www.courtinfo.ca.gov/> as an example, the main page that meets the above requirements is <http://www.courtinfo.ca.gov/shelfhelp/family/adlption/>, all the pages referring to this page or referred to by this page would fail one of the above conditions.

To assess whether a web page is a key resource page, a searcher needs to make the following judgments about the page:

1) Is it relevant?

2) Does it have the right scope? (Is it too broad or too narrow compared with that of other relevant pages from the same site?)

In the interactive track, searchers were also asked to make one more judgment:

3) Does it cover a different aspect from the previous saved web pages?

The traditional ranked list provides users with a set of entry points to their corresponding websites, then users have to browse each website to decide whether the entry point is the main page, or if not, whether there is a page within the site could be the main page. The above three tasks (especially the task 2 and 3) are not explicitly supported by this kind of delivery interface.

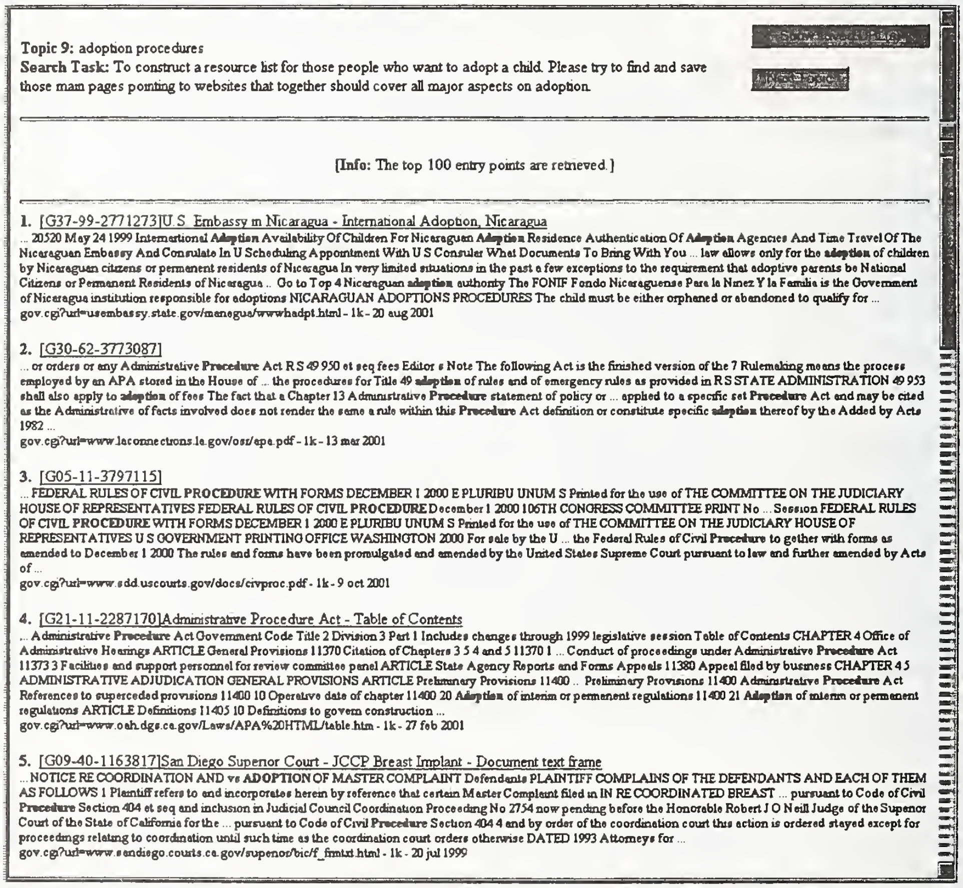
We aimed to investigate the effectiveness of a task tailored delivery method to assist searchers in making the above three judgment tasks. Our hypothesis was that searchers would have a better performance on the assigned tasks by using the interface designed to support the above judgment tasks than a generic interface (such as a ranked list).

3.1. Experimental setting

3.1.1. Delivery interfaces

The Panoptic topic distillation engine was used as the back-end search engine for both interfaces. To concentrate on the comparison of the two delivery interfaces, we decided to fix the query for all topics and for all searchers, i.e. searchers were restricted to explore the same set of retrieved documents. The queries were optimized to return shallow pages from a web site and to make sure the precision at top ten returned documents was acceptable.

The baseline interface (referred to as TDLinear for Topic Distillation with Linear interface), was the delivery interface from the Panoptic topic distillation engine. As shown in Figure 1, this interface provided searchers with a ranked list of top 100 potential relevant key resource pages in five consecutive pages, with each page showing the titles and summaries of 20 documents.



Topic 9: adoption procedures
Search Task: To construct a resource list for those people who want to adopt a child. Please try to find and save those mam pages pointing to websites that together should cover all major aspects on adoption.

[Info: The top 100 entry points are retrieved.]

1. [G37-99-2771273]U.S. Embassy in Nicaragua - International Adoption, Nicaragua
... 20520 May 24 1999 International **Adoption** Availability Of Children For Nicaraguan **Adoption** Residence Authentication Of **Adoption** Agencies And Time Travel Of The Nicaraguan Embassy And Consulate In U Scheduling Appointment With U S Consular What Documents To Bring With You ... law allows only for the **adoption** of children by Nicaraguan citizens or permenent residents of Nicaragua In very limited situations in the past a few exceptions to the requirement that adoptive parents be National Citizens or Permanent Residents of Nicaragua .. Go to Top 4 Nicaraguan **adoption** authority The FONIF Fondo Nicaraguense Para la Ninez Y la Familia is the Government of Nicaragua institution responsible for adoptions NICARAGUAN ADOPTIONS PROCEDURES The child must be either orphaned or abandoned to qualify for ...
gov.cgi?url=usembassy.state.gov/menegua/wwwhadpt.html - 1k - 20 aug 2001

2. [G30-62-3773087]
... or orders or any Administrative **Procedure** Act R S 49 950 et seq fees Editor s Note The following Act is the finished version of the 7 Rulemaking means the process employed by an APA stored in the House of ... the procedures for Title 49 **adoption** of rules and of emergency rules as provided in R S STATE ADMINISTRATION 49 953 shall also apply to **adoption** of fees The fact that a Chapter 13 Administrative **Procedure** statement of policy or ... applied to a specific set **Procedure** Act and may be cited as the Administrative of facts involved does not render the same a rule within this **Procedure** Act definition or constitute specific **adoption** thereof by the Added by Acts 1982 ...
gov.cgi?url=www.laconnections.la.gov/osr/apa.pdf - 1k - 13 mar 2001

3. [G05-11-3797115]
... FEDERAL RULES OF CIVIL **PROCEDURE** WITH FORMS DECEMBER 1 2000 E PLURIBU UNUM S Printed for the use of THE COMMITTEE ON THE JUDICIARY HOUSE OF REPRESENTATIVES FEDERAL RULES OF CIVIL **PROCEDURE** December 1 2000 106TH CONGRESS COMMITTEE PRINT No ... Session FEDERAL RULES OF CIVIL **PROCEDURE** WITH FORMS DECEMBER 1 2000 E PLURIBU UNUM S Printed for the use of THE COMMITTEE ON THE JUDICIARY HOUSE OF REPRESENTATIVES U S GOVERNMENT PRINTING OFFICE WASHINGTON 2000 For sale by the U ... the Federal Rules of Civil **Procedure** to gether with forms as amended to December 1 2000 The rules and forms have been promulgated and amended by the United States Supreme Court pursuant to law and further amended by Acts of ...
gov.cgi?url=www.sdd.uscourts.gov/docs/civproc.pdf - 1k - 9 oct 2001

4. [G21-11-2287170]Administrative Procedure Act - Table of Contents
... Administrative **Procedure** Act Government Code Title 2 Division 3 Part 1 Includes changes through 1999 legislative session Table of Contents CHAPTER 4 Office of Administrative Hearings ARTICLE General Provisions 11370 Citation of Chapters 3 5 4 and 5 11370 1 ... Conduct of proceedings under Administrative **Procedure** Act 11373 3 Facilities and support personnel for review committee panel ARTICLE State Agency Reports and Forms Appeals 11380 Appeal filed by business CHAPTER 4 5 ADMINISTRATIVE ADJUDICATION GENERAL PROVISIONS ARTICLE Preliminary Provisions 11400 .. Preliminary Provisions 11400 Administrative **Procedure** Act References to superceded provisions 11400 10 Operative date of chapter 11400 20 **Adoption** of interim or permenent regulations 11400 21 **Adoption** of interim or permenent regulations ARTICLE Definitions 11405 10 Definitions to govern construction ...
gov.cgi?url=www.oah.dgs.ca.gov/Laws/APA%20HTML/table.htm - 1k - 27 feb 2001

5. [G09-40-1163817]San Diego Superior Court - JCCP Breast Implant - Document text frame
... NOTICE RE COORDINATION AND vs **ADOPTION** OF MASTER COMPLAINT Defendants PLAINTIFF COMPLAINS OF THE DEFENDANTS AND EACH OF THEM AS FOLLOWS 1 Plaintiff refers to and incorporates herein by reference that certain Master Complaint filed in IN RE COORDINATED BREAST ... pursuant to Code of Civil **Procedure** Section 404 et seq and inclusion in Judicial Council Coordination Proceeding No 2754 now pending before the Honorable Robert J O Neill Judge of the Superior Court of the State of California for the ... pursuant to Code of Civil **Procedure** Section 404 4 and by order of the coordination court this action is ordered stayed except for proceedings relating to coordination until such time as the coordination court orders otherwise DATED 1993 Attorneys for ...
gov.cgi?url=www.sandiego.courts.ca.gov/superior/bic/f_frmtxt.html - 1k - 20 jul 1999

Figure 1. The delivery interface for the ranked list

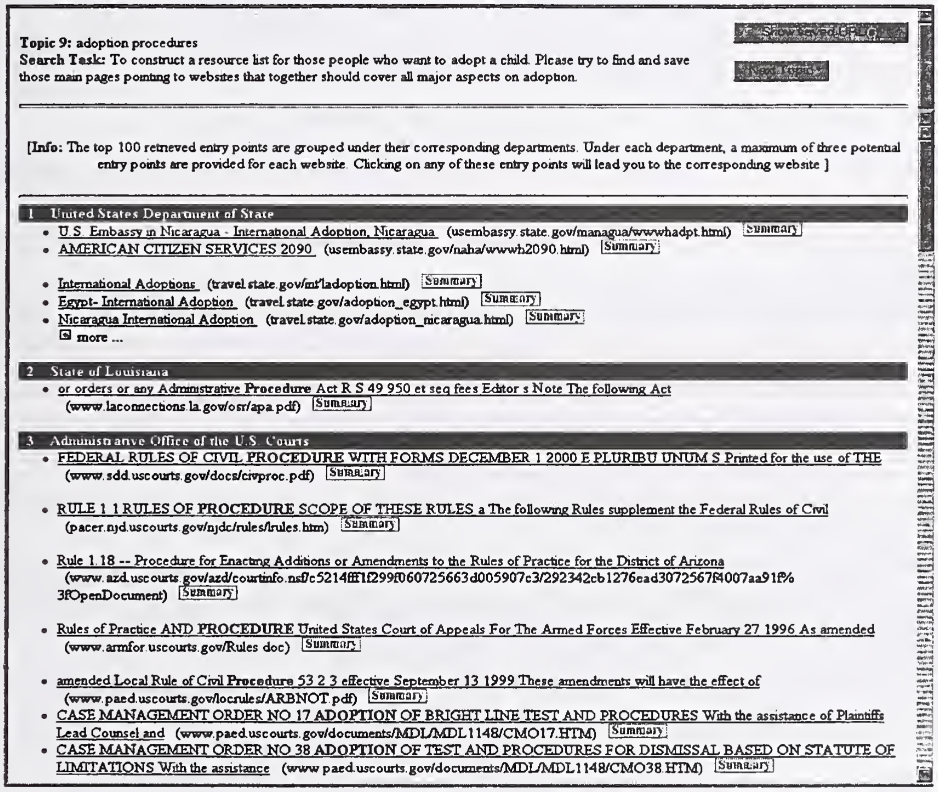
To validate our hypothesis, we designed the experimental interface (referred to as TDHierarchic for Topic Distillation with Hierarchical delivery interface) that explicitly supports searchers' assessment tasks. The experimental interface consists of two parts: the site summary and the sitemap.

1. The site summary (Figure 2): The top 100 retrieved pages (from Panoptic topic distillation engine) were firstly grouped according to their corresponding departments (organizational structure), and then further sub-divided into their secondary business units (websites). Each of the websites was summarized and represented by using the titles of the top three most relevant pages. The summary not only described the content of the site, but also provided three candidate entry points to the site. We decided not to show the summary of a document directly for two reasons: the summary of the document may not be suitable for the topic distillation task and showing the summary of a document would make the interface cluttered. Instead, we placed a "Summary" icon next to each title. If a searcher wanted to read the summary of a document, he/she could hover the mouse over the "Summary" icon, a pop-up window would appear next to the icon to show the summary. The content of this summary is the same as that for the same document in TDLinear interface.

We expected that the grouping mechanism in this interface would help searchers select a relevant website and a web page from the website as a starting point to browse from, and also support searchers with the third judgment task – the websites of different departments (or different sectors of the same department) would provide different perspectives on the searched topic.

2. The sitemap (Figure 3): After a searcher entered a web site, a hierarchical sitemap was provided to support the second judgment. The same query was used to retrieve the top 100 documents from just that site. The sitemap provided an outline view of the distribution of these retrieved pages according to the directory structure of the website. By using this sitemap, the searcher would be able to see the distribution of retrieved pages above or under the current directory, and to have an overview of the location of current page in the whole site.

Therefore, our hypothesis could be rephrased more specifically as that a searcher may perform the topic distillation task better with TDHierarchic interface than with TDLinear interface.



Topic 9: adoption procedures
Search Task: To construct a resource list for those people who want to adopt a child. Please try to find and save those main pages pointing to websites that together should cover all major aspects on adoption.

[**Info:** The top 100 retrieved entry points are grouped under their corresponding departments. Under each department, a maximum of three potential entry points are provided for each website. Clicking on any of these entry points will lead you to the corresponding website]

1 United States Department of State

- U.S. Embassy in Nicaragua - International Adoption, Nicaragua (usembassy.state.gov/managua/wwwhadpt.html) Summary
- AMERICAN CITIZEN SERVICES 2090 (usembassy.state.gov/naha/wwwh2090.html) Summary

- International Adoptions (travel.state.gov/int'ladoption.html) Summary
- Egypt- International Adoption (travel.state.gov/adoption_egypt.html) Summary
- Nicaragua International Adoption (travel.state.gov/adoption_nicaragua.html) Summary

⊞ more ...

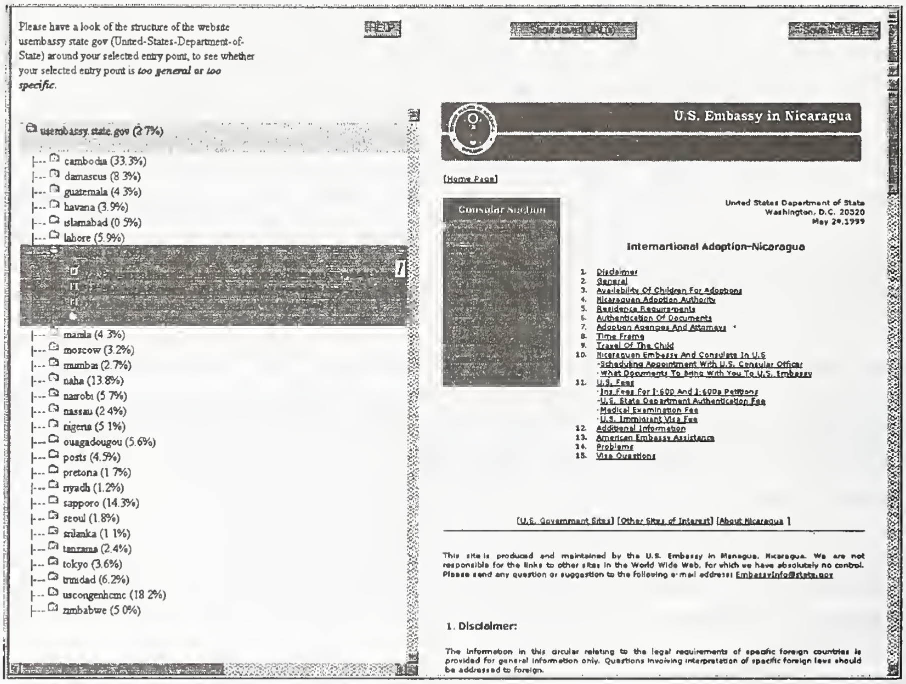
2 State of Louisiana

- or orders or any Administrative Procedure Act R S 49 950 et seq fees Editor s Note The following Act (www.laconnections.la.gov/osr/apa.pdf) Summary

3 Administrative Office of the U.S. Courts

- FEDERAL RULES OF CIVIL PROCEDURE WITH FORMS DECEMBER 1 2000 E PLURIBU UNUM S Printed for the use of THE (www.sdd.uscourts.gov/docs/civproc.pdf) Summary
- RULE 1 1 RULES OF PROCEDURE SCOPE OF THESE RULES a The following Rules supplement the Federal Rules of Civil (pacer.njd.uscourts.gov/njdc/rules/lrules.htm) Summary
- Rule 1.18 -- Procedure for Enacting Additions or Amendments to the Rules of Practice for the District of Arizona (www.azd.uscourts.gov/azd/courtinfo.nsf/c5214fff1f299f060725663d005907c3/292342cb1276ead3072567f4007aa91f%3fOpenDocument) Summary
- Rules of Practice AND PROCEDURE United States Court of Appeals For The Armed Forces Effective February 27 1996 As amended (www.armfor.uscourts.gov/Rules.doc) Summary
- amended Local Rule of Civil Procedure 53 2 3 effective September 13 1999 These amendments will have the effect of (www.paed.uscourts.gov/locrules/ARBNOT.pdf) Summary
- CASE MANAGEMENT ORDER NO 17 ADOPTION OF BRIGHT LINE TEST AND PROCEDURES With the assistance of Plaintiffs Lead Counsel and (www.paed.uscourts.gov/documents/MDL/MDL1148/CMO17.HTM) Summary
- CASE MANAGEMENT ORDER NO 38 ADOPTION OF TEST AND PROCEDURES FOR DISMISSAL BASED ON STATUTE OF LIMITATIONS With the assistance (www.paed.uscourts.gov/documents/MDL/MDL1148/CMO38.HTM) Summary

Figure 2. The site summary interface



Please have a look of the structure of the website usembassy state gov (United-States-Department-of-State) around your selected entry point, to see whether your selected entry point is ***too general*** or ***too specific***.

```
usembassy.state.gov (2.7%)
|--- cambodia (33.3%)
|--- damascus (8.3%)
|--- guatemala (4.3%)
|--- havana (3.9%)
|--- islamabad (0.5%)
|--- lahore (5.9%)
|--- manila (4.3%)
|--- moscow (3.2%)
|--- mumbai (2.7%)
|--- naha (13.8%)
|--- nairobi (5.7%)
|--- nassau (2.4%)
|--- nigeria (5.1%)
|--- ouagadougou (5.6%)
|--- posts (4.5%)
|--- pretoria (1.7%)
|--- riyadh (1.2%)
|--- sapporo (14.3%)
|--- seoul (1.8%)
|--- srilanka (1.1%)
|--- tanzania (2.4%)
|--- tokyo (3.6%)
|--- trinidad (6.2%)
|--- uscongenhcmc (18.2%)
|--- zimbabwe (5.0%)
```

U.S. Embassy in Nicaragua

[Home Page]

Consular Section

United States Department of State
Washington, D.C. 20320
May 24,1999

Internartional Adoption-Nicaragua

1. Disclaimer
2. General
3. Availability Of Children For Adoptions
4. Nicaraguan Adoption Authority
5. Residence Requirements
6. Authentication Of Documents
7. Adoption Agencies And Attorneys
8. Time Frame
9. Travel Of The Child
10. Nicaraguan Embassy And Consulate In U.S
    - Scheduling Appointment With U.S. Consular Officer
    - What Documents To Bring With You To U.S. Embassy
11. U.S. Fees
    - Ins Fees For I-600 And I-600a Petitions
    - U.S. State Department Authentication Fee
    - Medical Examination Fee
    - U.S. Immigrant Visa Fee
12. Additional Information
13. American Embassy Assistance
14. Problems
15. Visa Questions

[U.S. Government Sites] [Other Sites of Interest] [About Nicaragua]

This site is produced and maintained by the U.S. Embassy in Managua, Nicaragua. We are not responsible for the links to other sites in the World Wide Web, for which we have absolutely no control. Please send any question or suggestion to the following e-mail address: EmbassyInfo@state.gov

1. Disclaimer:

The information in this circular relating to the legal requirements of specific foreign countries is provided for general information only. Questions involving interpretation of specific foreign laws should be addressed to foreign.

Figure 3. The sitemap interface

3.1.2. Experimental procedure

We adopted the same experimental design as used by all participating groups in the interactive track. In this experimental design, subjects searching four topics on each interface, the sequence of interface and topics varied among subjects. A complete design requires a group of 16 subjects.

During the experiment, all subjects were asked to follow the following procedure:

- Subjects filled in the pre-search questionnaire about their demographic information and their search experience.
- We explained the search task to the subjects and gave subjects an example as recommended by the track guidelines.
- After acknowledged their understanding of the search task, subjects were then presented with the two experimental interfaces, and were free to ask any question.
- Subjects were randomly given a search number. The sequence of each topic and its associated interface for each search number was pre-programmed according to the experimental design. Subjects had 15 minutes for each topic, and were prompted to move to the next topic when their times run out. .
- Prior to each interface, subjects had a chance of hands-on practice with an example topic. This helps them to get familiar with the interface.
- Prior to the search of each topic, subjects were required to fill in a pre-search questionnaire about their familiarity with the topic. After the search of the topic, subjects were also required to fill in a post-search questionnaire about their experience of that particular search topic.
- Subjects filled in a post-system questionnaire after each interface (with four search topics).
- Subjects filled in an exit questionnaire at the end of the experiment.

3.1.3. Subjects

Sixteen students were recruited from local universities. They are all from computer science background. Among them, one is a PhD student; four of them are undergraduate students; and the rest eleven are all

Master students. They have an average age of 23.8. On average, they have 4.4 years of online searching experience, they regarded themselves as experienced searcher (Mean=5.44, Std=0.73); fifteen of them search the web daily. Comparatively, they have more experience with web search engines (Mean=6.19, Std=0.66) than the web site directory (Mean=5.56, Std=1.71).

3.1.4. Measurements

The saved lists from each search session (per topic, per interface) were gathered and sent to NIST for assessment. The assessment was based on four criteria: relevance, depth, coverage, and repetition. The assessors were asked to answer each of the following questions/statements on a five-point Likert scale.

Relevance: The page is relevant for the topic.

1 = Agree strongly, 2 = Agree slightly, 3 = Neutral, 4 = Disagree slightly, 5 = Disagree strongly

Depth: Is the page too broad, too narrow or at the right level of detail for the topic?

1 = Too broad, 2 = Bit broad, 3 = Right level, 4 = Bit narrow, 5 = Too narrow

Coverage: The set of saved entry points covers all the different aspects of the topic.

1 = Agree strongly, 2 = Agree slightly, 3 = Neutral, 4 = Disagree slightly, 5 = Disagree strongly

Repetition: How much repetition/overlap is there within the set of saved entry points?

1 = None, 2 = Minimal, 3 = Some, 4 = A lot of, 5 = Way too much

From the questions, we can see that the relevance and the depth judgment are document based, while the coverage and repetition are list based.

Transaction logging, questionnaire, and screen recording are the main methods used to collect data. During each search session, every significant event - such as reading a document and saving the URL - was automatically captured. Questionnaires common to all participating groups in the interactive track were adapted to our testing hypothesis. Screen recording was used to capture the search process for further detailed analysis.

3.2. Results

3.2.1. Performance with two interfaces

Tables 3 to 6 show each objective measure over all search sessions for each interface, averaged over topics and subjects. Overall, there is no significant difference between two interfaces (TDLinear vs TDHierarchic) by any measure, although there are topic variations.

As we discussed earlier, one motivation for this year's interactive track was to compare the results from the interactive topic distillation with that from automatic topic distillation. Thus, for each topic, we take a list of top N documents generated by Panoptic topic distillation engine, where N is the number nearest to the average size of all saved lists for that topic. For each of these lists, we can measure its relevance and depth, given that assessors had provided with corresponding assessments for each document. In the rare occasions when one of the top N documents was not picked up by any searcher as relevant, we would then assign it to the "highly irrelevant" category. For the lists automatically generated by Panoptic, their relevance and depth are shown in Tables 3 and 4 denoted as TDAuto (for Topic Distillation from Automatic system). From Table 3 and Table 4, we can find that, in six out of eight topics, the lists saved by searchers (TDLinear) are more relevant and closer to the right level than the lists from the automatic approach (TDAuto). Overall, these differences are significant ($p < 0.0003$[2] and $p < 0.0001$ for the relevance and depth respectively). The difference between TDHiearchic and TDAuto is not found significant in terms of relevance, but significant ($p < 0.005$) in terms of depth.

[2] All significant tests in the interactive part used paired t-test.

In the automatic topic distillation track, systems are judged according to the number of good answers they found in the top ten results. Here the "good" answers are those of high relevance and right depth. To compare the interactive system with the automatic tool using an equivalent measure, we also use the relevance and depth as the indicator of a "good" answer: if the relevance score of a saved page is 1 or 2, and the depth score of the page is between 2 and 4 inclusively, we would assume the page is a good answer. By applying this rule, the Tables 3 and 4 can be converted into the Table 7[3]. The difference between TDAuto and TDLinear is significant at 0.02 (paired t-test).

Table 3: Relevance of the saved/retrieved documents (The closer a score is to 1, the better)

	T1	T2	T3	T4	T5	T6	T7	T8	Mean
TDAuto	3.17	1.14	2.50	2.86	1.67	2.75	3.43	2.83	2.54
TDLinear	2.81	1.37	2.7	2.41	1.13	2.38	2.43	2.49	2.22
TDHierarchic	3.56	1.85	2.52	2.74	1.22	2.96	2.81	2.03	2.46

Table 4: Depth of the saved/retrieved documents (The closer a score is to 3, the better)

	T1	T2	T3	T4	T5	T6	T7	T8	Mean
TDAuto	4.00	3.71	2.50	4.14	3.33	3.13	4.14	3.67	3.58
TDLinear	3.77	3.19	2.26	3.83	2.99	3.40	3.62	3.46	3.32
TDHierarchic	4.30	2.88	2.47	3.83	2.87	3.37	3.71	3.01	3.31

Table 5: Coverage of the saved list (The closer a score is to 1, the better)

	T1	T2	T3	T4	T5	T6	T7	T8	Mean
TDLinear	1.63	2.13	3.25	4.5	1.25	1.25	1.00	4.88	2.48
TDHierarchic	2.38	2.63	2.50	4.63	2.25	1.25	1.00	3.63	2.53

Table 6: Repetition of the saved list (The closer a score is to 1, the better)

	T1	T2	T3	T4	T5	T6	T7	T8	Mean
TDLinear	2.57	2.25	1.75	2.38	1.50	3.0	3.0	2.63	2.38
TDHiearchic	2.50	1.63	2.13	3.38	2.00	3.25	3.25	1.25	2.42

Table 7: Precision of the saved/retrieved list

	T1	T2	T3	T4	T5	T6	T7	T8	Mean
TDAuto	0.33	1.00	0.67	0.29	0.83	0.50	0.43	0.50	0.57
TDLinear	0.48	0.88	0.53	0.52	0.97	0.62	0.54	0.48	0.63
TDHierarchic	0.18	0.78	0.61	0.48	0.95	0.51	0.49	0.60	0.58

[3] Compared to the automatic topic distillation task our assessment is fairly lenient and the queries have been manually adjusted to the task. Although the absolute values of the precision are high, it is the relative differences that are noteworthy.

3.2.2. Searcher effort

The numbers of unique documents (after removing duplicated occurrences and un-accessible pages) read and saved are shown in Table 8. The second row shows that subjects read significantly more documents from TDLinear interface (Mean=24.17) than that from TDHierarchic interface (Mean=17.73) ($p < 0.0002$). However, the third row did not show much difference in the number of saved documents from each interface.

Table 8: The number of read and saved documents

	TDLinear **Mean (Std)**	**TDHierarchic** **Mean (Std)**
Read-unique	24.17 (8.71)	17.73 (5.00)
Saved-unique	6.64 (3.00)	6.65 (3.79)

To understand how and where subjects put their effort, we had a closer look on how subjects divided their effort on each interface.

The TDLinear interface has two parts: the window for the ranked list (TDLinear-R) and the window to show the content of a selected document (TDLinear-C). The TDHierarchic interface has three parts: the window for the grouped ranked list (TDHierarchic-R), the frame for the tree structure of a selected web site (TDHierarchic-T), and the frame to show the content of a selected document (TDHierarchic-C).

There is not much difference between TDLinear-C and TDHierarchic-C, except their window sizes. The difference is that TDHiearchic has an extra interface panel (TDHierarchic-T), and TDHierarchic-R is probably more complex than TDLinear-R.

Table 9 shows the division of effort from the first four searchers. By examining the recorded screen actions from the these four searchers, we observed that these searchers spent an average 36% of their total search time and on average opened 15 (unique) documents to read from TDLinear-R. While in TDHierarchic-G window, searchers spent similar amount of time (37% of their total search time), but opened only 9.3 (unique) documents. We observed that searchers picked up documents to open sequentially and spent less time to read document summaries in TDLinear-R, while they spent more time to read document summaries (by hovering the mouse over the "Summary" icon) and even read summaries from a few documents before they opened a document in TDHierarchic-G.

While these four searchers spent on average 64% of their total search time and opened 7.9 documents to read from TDLinear-C, they divided their effort in two frames in TDHierarchic. These four searchers spent on average 19% of their search time on TDHierarchic-T, 44% on TDHierarchic-C, but opened a similar number of (unique) documents. This implies that the searchers used the tree structure more often to help them to browse the selected web site.

Table 9: The split of efforts in each interface

	% of total time		
TDLinear	Ranked list: 36%	Page content window: 64%	
TDHierarchic	Grouping: 37%	Tree: 19%	Page content window: 44%
	Average number of documents opened		
TDLinear	Ranked list: 15	Page content window: 7.9	
TDHierarchic	Grouping: 9.3	Tree: 4.4	Page content window: 4.2

3.2.3. Subjective measures

After each topic, subjects were required to fill in a post-search questionnaire about their experience of the search topic and their sense of the task completeness. All questions are on 7-point Likert scale with 1=strongly disagree, 4=neutral, and 7=strongly agree. Table 10 shows that subjects gave higher score to TDHierarchic interface on all seven questions.

Table 10: Post-search questionnaire

	Q1	Q2	Q3	Q4	Q5	Q6	Q7
TDLinear	5.02	5.04	4.55	5.03	5.16	4.50	4.82
TDHierarchic	5.25	5.13	4.82	5.02	5.23	5.05	4.91

Q1: The search process is easy.
Q2: The pages I just saved focuses on the topic well.
Q3: The pages I just saved are the main pages of their corresponding websites.
Q4: The pages I just saved together provide a good coverage of the topic.
Q5: The pages I just saved will be helpful for the targeted audiences.
Q6: I have enough time to do an effective search.
Q7: I believe that I have succeeded in my performance of the task.

After each system, subjects were asked to fill in a post-system questionnaire to get their opinion on the usability of each system. Table 11 shows the average score for each interface for seven questions. There are significant difference between two interfaces for question 3 and question 4, that is: searchers strongly agreed that the organization of the search results of TDHierarichic interface was clearer and more useful for them to select an entry point to start with.

Table 11: Post-system questionnaire

	Q1	Q2	Q3	Q4	Q5	Q6	Q7
TDLinear	5.50	5.38	4.25	3.81	3.94	3.81	3.81
TDHierarchic	5.63	5.88	5.81	5.50	4.94	4.50	4.88

Q1: It was easy to learn to use this system.
Q2: It was easy to use this system.
Q3: The organization of the search results is clear to me.
Q4: the organization of the search results is useful for me to select an entry point to search.
Q5: The summary of each search results helped me to decide the relevance of that website.
Q6: The summary of each search result is useful for me to select an entry point to search.
Q7: The web structure of my selected entry point is useful for me to judge whether the entry point is the main page.

Table 12 shows searchers' answer to the three questions in the exit search questionnaire. Overall, most of the searchers perceived that TDHierarchic interface is easier to use and supporting their task better, and they liked TDHierarchic interface the best overall.

Table 12. Exit questionnaire

	Q1	Q2	Q3
TDLinear	6	2	3
TDHierarchic	10	14	11
No Difference	0	0	2

Q1: Which of the two systems did you find easier to use?
Q2: Which of the two systems did you think supporting your task better?
Q3: Which of the two systems did you like the best overall?

3.3. Discussion

In this experiment, we found that our searchers preferred the experimental interface (TDHierarchic) and perceived that they fulfilled their task better by using the experimental interface than the ranked list interface (TDLinear). However, we didn't find any significant difference between the two interfaces on searchers' performance in terms of relevance, depth, coverage and repetition.

One of our hypotheses was that we could increase performance by encouraging searchers to compare items rather than make individual judgments. This was implemented on the site summary interface. By further examining searchers' behavior, we found that the interface for grouping documents into sites changed search behavior: searchers spent time selecting amongst the results from a specific site by looking at and comparing the summaries. Searchers selected fewer pages to examine and the overall results were similar to the ranked list interface indicating that users had compared and made good selection decisions. Also in the post-system questionnaire, searchers stated strongly that the grouping interface was useful for them to select an entry point to search. However, confounded by many other factors, it is not clear whether this behavior would be beneficial to the overall task.

Comparing the results from our interactive system with that of the corresponding automatic system, we found a significant improvement in terms of relevance, depth and precision. That indicates that engagement of a searcher's effort has a positive effect on the system performance, and that there is room for improvement for systems to reduce searcher effort.

4. References

[1] Web track guidelines: http://es.cmis.csiro.au/TRECWeb/guidelines_2003.html

[2] Interactive track guidelines: http://www.ted.cmis.csiro.au/TRECInt/guidelines.html

Task Descriptions: Web Track 2003

Nick Craswell and David Hawking
CSIRO ICT Centre
Canberra, Australia
nick.craswell@csiro.au and david.hawking@csiro.au

Ross Wilkinson and Mingfang Wu
CSIRO ICT Centre
Melbourne, Australia
ross.wilkinson@csiro.au and mingfang.wu@csiro.au

October 30, 2003

Part I
Non-interactive Experiments

1. Introduction

This Year's Aims

- To investigate methods for effective topic distillation: Finding a set of the best home pages, given a broad query.
- To investigate methods for effective navigational search, with a mixture of home page and named page queries: Finding a particular page desired by the user.
- To increase the available queries/judgments for the .GOV test collection.

Participants are welcome to explore other Web retrieval issues such as distributed IR, queries with misspellings, efficient indexing etc within the context of these experiments.

2. Dataset

The corpus for both tasks is the .GOV test collection, distributed by CSIRO. Documents include the information returned by the http daemon (enclosed in DOCHDR tags) as well as the page content.

The crawl is recent (start of 2002). It is the sort of crawl which might be used by a real .gov search service: breadth first, stopped after the first million html pages and including (the extracted plain text of) an additional 250,000 non-html pages (doc, pdf and ps). Text is provided for convenience. NIST assessors will view original (binary doc,pdf,ps,gif,jpg) files when judging.

3. Topic distillation task

Topic distillation involves finding a list of key resources for a particular topic. In this year's task we are concentrating solely on websites as resources. The task is to find as many different websites (represented by their entry pages) as possible within the first ten results.

For the topic, 'science', the following websites might be considered key resources:

`www.nsf.gov/`	National Science Foundation
`science.nasa.gov/`	Science @ NASA
`www.science.gov/`	Government Science Portal
`www.house.gov/science/welcome.htm`	House Committee on Science

To be judged a "key resource", the page returned should be a good entry point to a website which:

- Is principally devoted to the topic,
- Provides credible information on the topic, and
- Is not part of a larger site also principally devoted to the topic

For the 'science' topic, the page 'www.house.gov' fails the first test while the page 'www.nsf.gov/home/bio/' fails the third. Hopefully within the .gov domain, it will be hard to find sites which fail the second test! NIST will develop topics for .GOV. Example topic format:

```
<top>

<num> Number:
<title> science

<desc> Description:
Find key government websites (represented by their home page)
on the subject of 'science'.

</top>
```

The title field only should be supplied to your system as the query.

Systems will be judged according to how many good answers they find in the top ten results (the first page returned by a typical Web search system). Likely measures are precision at 10 and average precision at 10.

4. The home/named page finding task

Users sometimes search for a page by name. In such cases, an effective search system will return that page at or near rank one.

This year's task involves a mixture of tasks from two previous years: home page finding and named page finding. In both cases, there is only one target page and user queries are often the name of the page. The difference is that home page finding queries are restricted to home pages: 'Internal Revenue Service' → 'www.irs.gov', while named page finding may involve pages which are not home pages 'passport application form' → 'travel.state.gov/dsp11.pdf'. Some search/ranking metrics will be useful for both types of query, while others will only be useful for one.

NIST will devise the mixed set of queries for .GOV. A minimal amount of judging will be required to determine if the URLs of documents returned by participants are in fact equivalent to the answer originally chosen. For example, if the page is available at 2 different URLs, both would be considered correct answers.

Systems will be compared on the basis of the rank of the first correct answer. Likely measures include mean reciprocal rank of first correct answer and success rate at N (percentage of cases in which the correct answer or equivalent URL occurred in the first N documents).

No manual or interactive query modification is permitted in this task.

5. Indexing Restrictions

There are none. You may index all of each document or exclude certain fields as you wish.

6. Submissions and Judgments

All submissions are due at NIST on or before 6 August 2003.

Submission information:

Topic distillation: Submit up to 5 runs. For each query, list up to 1000 (the top 1000) results. Check your results using `check_web.pl` (available from `http://trec.nist.gov/`)

Home/named page finding: Submit up to 5 runs. For each query, list up to 50 (the top 50) results. Check your results using `check_web.pl` (available from `http://trec.nist.gov/`)

The result format is:

```
topic-id Q0 docno rank sim tag

  topic is the topic number,
  Q0    is the literal 'Q0',
  docno is the document id taken from the DOCNO field of the text,
rank  is the rank assigned to the document,
sim   is the similarity computed between the document and the topic,
tag   is the run tag.
```

It is likely that NIST will accept up to 5 official submissions for each task, but the number of fully judged runs per group will depend upon the number of submissions, the degree of overlap and the judging resources available. Hopefully it will be possible to judge two topic distillation runs and two home/named page runs per group.

All judging will be performed by NIST assessors.

Judgments will be binary. Key resource OR Not key resource. Home/named page OR Not home/named page.

Judgments will be made on the basis of the text within the document, its URL and in the case of topic distillation the pages it links to (particularly those on the same site).

Part II
Interactive Experiments

1. Motivating principles

This year the interactive track will become a sub-track of the web track. One of the web track tasks, the topic distillation, has been selected as the interactive track task. However, the interactive sub-track will focus on the human participation in topic distillation. Virtually any kind of user studies on topic distillation would be acceptable. A pre-defined protocol will be provided for the balanced studies that the past interactive tracks have used.

2. Tasks

An information searcher may often need to construct a list of resources on a topic for him/her own learning or for other people. Such a resource list could be manually constructed, such as that in Yahoo!, or automatically constructed for a user-defined topic by a topic distillation algorithm.

In the interactive sub-track, a searcher will be asked to construct such a resource list on a broad topic through interaction with an information access system. A typical task statement (or instruction) given to the searcher would look like this:

> *"Your task is to construct a learning resources for a class of 16-year-old secondary school pupils on [topic x]. Your learning resource should include the good main pages that point to websites that together cover all the major aspects of [this topic]."*

The following eight search topics are selected from the topic set as used by the web topic distillation task, and are rephrased according to the format of the above task statement.

1. **Title:** cotton industry
Search task:
You are to construct a resource list for high school students who are interested in cotton industry, from growing, harvesting cotton and turning it into cloth.

2.**Title:** folk art folk music
Search task:
Assume that you are an art teacher of a high school. You are about to introduce your students to U.S. folk art and folk music. Please prepare a list of bookmarks for your students for study materials.

3. **Title:** children's literature
Search task:
The teachers from your local primary school are spending a lot of their time on the web to search for materials on children's literature. Please help the teachers by setting up a children's literature web guide which points to useful websites for young readers/writers.

4. **Title:** wireless communications
Search task:
You are invited to give a presentation on wireless communication to university students. Please prepare a list of bookmarks as a handout to your audience. The bookmarks should cover information on existing and planned uses, research/technology, regulations and legislative interest.

5. **Title:** arctic exploration
Search task:
Assume that you are a high school student and working on an arctic exploration project. You are asked to collect some resources from the web for your project team on what kinds of exploration of the arctic are underway, especially of glaciers and ice.

6. **Title:** weather hazards and extremes
Search task:
Assume that you are a high school student and working on a project regarding the study of natural/weather hazards and extremes. You are asked to collect some resources from the web for your project team.

7. **Title:** electric automobiles

Search task:
You are going to give a seminar on the progress in producing/developing electric automobiles, and you will mention some online resources on this topic. Please prepare a list of bookmarks as a handout to your audience.

8. **Title:** Bilingual education
Search task:
You are a volunteer of your local community. You are asked to help to create a guide to all online information on bilingual education that may be of interest to your local residents.

3. Data collection and search systems

The interactive track will use the same .GOV web collection created for the TREC 2003 web track.

Panoptic at NIST

NIST will provide access to its server with the "Panoptic search engine" and the "Panoptic topic distillation engine" (both of them are accessible through `http://ir.nist.gov/`). You can also find the link to the help page there. A short description of each search algorithm is also available.

In order to be consistent with the web topic distillation task, all searches and browses are restricted within the .GOV collection. Thus, the Panoptic topic distillation engine has been configured so that all hits, and all links in hit pages, point back into the .GOV collection. (Please note this has some consequences that some links don't work, some images are missing, and this affects some pages more than others.)

Advanced usage

- You can request a page by its URL,

```
http://ir.nist.gov/search/gov.cgi?url=http://trec.nist.gov/
```

- or by its docid:

```
http://ir.nist.gov/search/gov.cgi?id=G01-01-0000000
```

XML search interface

If you have your own interface and just want to feed queries to Panoptic and get XML output, you can search using the search-xml.cgi script: `http://ir.nist.gov/search/search-xml.cgi?query=wireless+communication&collection=gov` The CGI query syntax is the same as for the standard search interface. Panoptic has a number of fancy query operators, and if you want to know how to feed them to the XML script, give them to the standards interface and look at the URL of the results page.

Participants are free to use any appropriate search engine, but need to make sure that all searches and browses are within the .GOV collection.

4. Experimental protocol

Participants are free to define an experimental protocol that suits their own experiment purpose. For those who want to compare two systems or system variants, they could adopt the following experimental protocol.

Experimental procedure

1. Entry questionnaire
2. Tutorial session
3. Before-search questionnaire per topic
4. Topic search
5. After-search questionnaire per topic
6. After-system questionnaire per system after 4 topics on a system
7. Exit questionnaire

Experimental design

The design is within subjects, and requires 16 subjects for completion. It depends upon dividing the eight topics into two blocks, varying the order of topics within each block.

If we define:

```
B1 = block 1;
B2 = block 2;

a = order 1,
b = order 2,
c = order 3,
d = order 4
```

	a	b	c	d
B1	1234	4321	3142	2413
B2	5678	8765	7586	6857

`Then the first four subjects search as follows:`

S1	System I: B1a	System II: B2a
S2	System I: B2a	System II: B1a
S3	System II: B1a	System I: B2a
S4	System II: B2a	System I: B1a

This pattern is then repeated for each of the remaining 3 topic orders (b,c,d).

Instructions to be given to searchers(during the tutorial session)

In this experiment, your task is to construct a list of key resources on a given topic. This resource list is intended to guide someone who shows interest in that topic to find more information. The goal of this experiment is to determine how well an information retrieval system can help you accomplish your task.

Each of these key resource pages should be a main page of a website which:

1. Is principally devoted to the topic,
2. Provides credible information on the topic, and
3. Is not part of a larger site also principally devoted to the same topic

Here is an example:

Topic: Adoption procedures

Search task: To construct a resource list for those people who want to adopt a child. Please try to find and save those main pages pointing to websites that together should cover all major aspects on adoption.

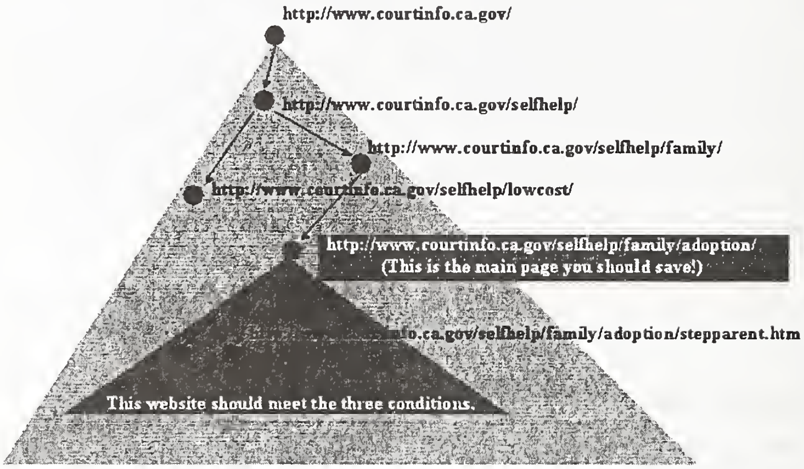
For example: the following pages would be regarded as good main pages:

G00-03-2173112 (`http://www.acf.dhhs.gov`) - Administration for Children and Families (USHHS)
G13-55-1080004 (`http://www.acf.dhhs.gov/programs/cb/dis/afcars/`) - Adoption and Foster Care Analysis and Reporting System (USHHS)
G07-03-3445073 (`http://www.courtinfo.ca.gov/selfhelp/family/adoption/`)- California Courts Self-Help Center
G00-98-2804800 (`http://www.mcdss.co.gov`) - Mesa County Dept. of Human Services

The above pages are acceptable because they are the main pages of those websites that meet all three conditions. To decide whether a current page is the main page you should save, you can try to go up or down the links in the current page, and see whether all three conditions are still meet. You may find the up link of a main page may not be principally devoted to the topic, while the down link of a main page may cover only a part of the topic. For example, the following pages would NOT be regarded as a good main page:

G00-08-0239407 (`http://www.courtinfo.ca.gov/`) - California Courts - The Judicial Branch of California - fails the first condition

G33-75-3683089 (`http://www.courtinfo.ca.gov/selfhelp/family/adoption/stepparent.htm`) - Stepparent Adoptions in California - fails the third condition


http://www.courtinfo.ca.gov/
http://www.courtinfo.ca.gov/selfhelp/
http://www.courtinfo.ca.gov/selfhelp/family/
http://www.courtinfo.ca.gov/selfhelp/lowcost/
http://www.courtinfo.ca.gov/selfhelp/family/adoption/
(This is the main page you should save!)
...info.ca.gov/selfhelp/family/adoption/stepparent.htm
This website should meet the three conditions.

Here is a hand-on practice topic:

Topic: intellectual property

Search task: You are a high school social studies teacher, and are about to introduce a module on intellectual property. Prepare a list of resources for the class that define intellectual property, and explain how creators of intellectual property are protected under US laws.

(Some example good pages:

G01-23-1524097 (`http://www.loc.gov/copyright/about.html`) - Welcome to the US Copyright Office
G00-03-2959565 (`http://patents.uspto.gov`) - US Patent and Trademark Office
G01-17-4061425 <DOCHDR> - www.cybercrime.gov/ip.html
G02-48-1320654 (`http://patents.gsfc.nasa.gov`) - NASA Office of Patent Counsel

5. Evaluation

For each topic, the following two qualitative measures will be applied and gathered.

> **Accuracy:** The extent of relevance of each page in a resource list to the corresponding search topic. The judgement will be on a seven-point Likert scale.
>
> **Coverage:** The relative coverage of relevant aspects in a resource list. (After all submitted pages per topic are aggregated and judged for the accuracy, then assessors will judge each list for the coverage.) The judgement will also be on a seven-point Likert scale.

The subjective evaluation (such as searchers' satisfaction) will be collected through the After-search questionnaire, After-system questionnaire and Exit questionnaire.

Data to be submitted to NIST

Each site should submit one ascii file only. Each line in the file represents a search topic worked on by a subject even if no DOCNO is saved. If you have 16 subjects and 8 topics, then your file should have 16X8 = 128 lines. Each line should contain the following items with intervening spaces and semicolons as indicated. Since semicolons will be used to parse the lines, they can only occur as specified in the following format:

```
SiteID; SystemID; SearcherID; TopicNum; DOCNOLIST

where:

     SiteID - unique across sites

     SystemID - unique within site to each of your IR systems

     SearcherID - unique within site to each of your subjects

     TopicNum - the topic number as in the guidelines

     DOCNOLIST - a list of TREC DOCNOs as found in the
                    documents, separated by commas.

Sites determine SiteID, SystemID, and SearcherID. They are not
allowed to contain spaces.
```

Sites are not required to submit those data from questionnaires.

6. Schedule

- 30 June 2003 - all resources (search topics and search engines) will be available, and Guidelines will also be completed.
- 31 August 2003 - All runs should be sent to Ian Soboroff for the judgment of accuracy and coverage.
- 8 September 2003 - Each participating group will submit a paragraph to Ross Wilkinson including a brief description of their studies.

On the Use of MeSH Headings to Improve Retieval Effectiveness

Stephen Blott
Cathal Gurrin
Gareth J. F.Jones
Alan F. Smeaton
Thomas Sodring

School of Computing
Dublin City University
Dublin, Ireland

{sblott,cgurrin,gjones,asmeaton,tsodring}@computing.dcu.ie

Abstract

Geneticists today spend as much time cataloguing and analysing digital data as they do in wet labs. One of the key problems they face, is that of identifying genes and publications that are relevant to their work or research. This paper describes an approach to retrieval that incorporates structural MeSH heading data into the search process. In particular, we describe a technique using a pseudo-relevance feedback process based on MeSH terms to improve retrieval effectiveness. Experimental results suggest improvements in mean average precision on the order of three percent.

1 Introduction

Molecular biologists study the biochemical function, chemical structure and evolutionary history of genes and proteins from all types of organisms, from human beings to fruit flies and yeast [8, 3]. While molecular biologists still spend much of their time in wet labs, they nowadays often spend equally as much time in front of computers. Information has become a critical research tool, and several large genomic databases have been created to facilitate the exchange of information within the community. These databases are repositories not just for genetic information, such as genes and gene sequences, but also for papers and reports relating to the sequencing and discovery of that genetic information, and the associated bibliographic data and citation indexes.

Among the larger examples of genomic databases are the nucleotide sequence database operated jointly by GenBank [4] at the National Centre for Biological Information in the US, the DNA Data Bank of Japan [1], and EMBL [2], the European Molecular Biology Laboratory. These databases have become huge. The GenBank nucleotide database, for instance, contains nucleotide sequences from more than 130,000 different organisms. As of August 2002, GenBank contained approximately 22,617,000,000 bases in 18,197,000 sequence records. Moreover, the GenBank database is growing as rapidly now as it ever has.

Life scientists spend prolonged periods of time using these databases. They may begin searching among research literature, and then search for related genes and gene sequences within GenBank. Bibliographic data associated with the related gene sequences may then lead the scientist back to other literature, and so on. Citation indexes between articles in the research literature can also be used to discover relevant related articles. Such a search session may involve navigation back and forth between genetic databases and document databases. Anybody who has searched the web is familiar with the frustrating experience of pursuing links, backtracking, and returning to the search engine to reformulate the query and begin again.

Genomic databases are not isolated collections of data. Rather, they are interrelated databases of bibliographic data, genetic data, and the links between these. The hypothesis underlying the work described here, is that the links between these databases can be used to improve retrieval effectiveness. Specifically, our approach is analogous to the way links are used on the world-wide web, as popularised in the well-known Google search engine, using graph topology information to locate potentially highly relevant nodes in the overall graph. In the case of Google, the "graph" consists of web pages and hypertext links between them. Highly-ranked web pages are those which have a high degree of similarity with the user's query, have many other web pages of relevance pointing to them, and also point themselves to many other web pages which are similar to the query.

In this paper, we explore retrieval techniques exploiting the MeSH terms in genomic bibliographics databases. MeSH, standing for "Medical Subject Headings", is a controlled vocabulary thesaurus managed by the US National Library of Medicine. It consists of sets of terms naming descriptors in a hierarchical structure that permits searching at various levels of specificity [5]. Quoting [5]:

> *MeSH descriptors are arranged in both an alphabetic and a hierarchical structure. At the most general level of the hierarchical structure MeSH terms are very broad headings such as "Anatomy" or "Mental Disorders". At more narrow levels are found more specific headings such as "Ankle" and "Conduct Disorder". There are 21,973 descriptors in MeSH. In addition to these headings, there are 132,123 headings called Supplementary Concept Records within a separate chemical thesaurus. There are also thousands of cross-references that assist in finding the most appropriate MeSH Heading, for example, "Vitamin C see Ascorbic Acid". These entries include 23,512 printed see references and 102,346 other entry points.*

MeSH terms are manually assigned to documents within genomic collections. In this paper, we investigate the hypothesis that documents discussing related topics will have similar MeSH terms associated with them, and that this similarity can be used to improve the overall retrieval effectiveness above a system treating individual documents as discrete entities.

Our experimental evaluation is in terms of the Genomic Track of the 2003 Text Retrieval Conference (TREC). In particular, our approach was to base our experimentation on runs of standard retrieval methods, and then use the MeSH categorisation of documents as the basis for a pseudo-relevance feedback process to improve upon the initial ranking.

2 Medical Subject Headings (MeSH) and Medline

This section provides the background material necessary to understand the approach described in the rest of the paper.

2.1 Medical Subject Headings (MeSH)

MeSH terms are used to categorise medical publications much as the Dewey-Decimal system is used in general libraries. MeSH is a vocabulary of terms, from the very general to the very specific, in terms of which medical literature is classified. Examples of general and specific terms include:

C01	Bacterial Infections and Mycoses
C01.252	Bacterial Infections
C01.252.400.155.569.200	Erythema Chronicum Migrans
C01.252.400.155.569.600	Lyme Neuroborreliosis

Based on the code on the left in the example, above, the terms are categorised hierarchically, with the length of the code corresponding loosely with its specificity. In addition, the same term is often repeated at different places within that hierarchy. For example:

C01.252.400.825.480	Lyme Disease
C01.252.847.193.569	Lyme Disease

Intuitively, terms whose codes share a long common prefix are specific and similar. For example, the three terms below are similar to one another:

C01.252.847.840.744.725	Syphilis, Congenital
C01.252.847.840.744.800	Syphilis, Cutaneous
C01.252.847.840.744.871	Syphilis, Latent

each being related to disease caused by members of the syphilis bacterial family, and each sharing the prefix "C01.252.847.840.744". Other terms which do not share such long common prefixes might be considered less similar, for example:

C01.252.400.210.210.250	Conjunctivitis, Inclusion

The MeSH terms used in this work was "2003 MeSH", which includes 21,837 terms organised into a hierarchy of 39,829 distinct nodes.

The basis of the work described here is to exploit the similarity of MeSH the terms used to categorise biomedical publications to improve retrieval effectiveness from biomedical bibliographic databases.

2.2 The Medline Database

Medline [6] is the US National Library of Medicine (NLM) database of indexed journal citations and abstracts now covering nearly 4,500 journals published in the United States and more than 70 other countries. Medline includes references to articles indexed from 1966 to the present, with new citations added weekly. All citations are assigned MeSH Terms and Publication Types from NLM's controlled vocabulary. MEDLINE citations and abstracts are available as the primary component of NLM's PubMed database [6], which is searchable via the Internet.

2.3 The TREC Genomics Track (TrecGen), 2003

In 2003, the Text REtrieval Conference (TREC) organised by the US National Institute for Standards and Technology (NIST) ran a track on genomic retrieval (TrecGen). Under TrecGen, participants were provided with an extract of over 500,000 records from the Medline database, and a set of sample topics. Participants were then set the task of locating documents from the Medline database that are relevant to the sample topics. This paper reports on a set of experiments carried out within the TrecGen framework designed to fulfil this task.

3 Using MeSH Terms for Retrieval

Since the focus of our work is to seek to make use of the MeSH terms to improve retrieval, we took as the starting point for the approach described the baseline set of results generated using SMART-based Okapi approach provided by Jacques Savoy of the University of Neuchatel, Switzerland. The basic approach in our work was to use a notion of similarity between documents measured using MeSH terms to generate a new score for each document. This score is then combined with the original score generated by the Okapi system. The document list is then re-ranked with the intention of creating a new ranking that places more relevant documents higher in the ranked list.

We explored two approaches to this problem, which we refer to as Method 1 and Method 2, respectively. These methods are described in detail below. However, we begin with a discussion of the issues and techniques that are common to both methods.

3.1 Basic use of MeSH Data

The Medline database consists of over 500,000 citations, each including title, authors and abstract. Given a topic, we assume an initial ranking of these Medline citations. The approach described here is then to re-rank the top N documents based on the similarity of the top N documents to the most highly ranked documents. The aim of the work was to augment the original ranked list with a document similarity score measured using MeSH terms.

Within the Medline database, each citation is annotated with several MeSH terms, with the average being around 16 terms per document. As noted in Section 2.1, MeSH terms are associated with codes, and those codes are organised in a hierarchy. In a first processing step, the mesh terms were replaced with the corresponding codes. Terms with multiple codes were replaced with all of the corresponding codes.

Each document is labelled with only a small number of MeSH terms. We might think of these as analogous to keywords assigned to other documents. Since the number of MeSH terms associated with a document is generally low and MeSH terms are often very specific, it will often be the case that there are few MeSH terms in common between documents. Because of this, standard information retrieval matching will often find very few matches between documents and the resulting match scores will bear more relation to random matches of MeSH between documents than a reliable measure of document topic similarity. Thus we need an approach to give us a quantitative measure of the "closeness" of terms that do not match exactly.

The hierarchy of MeSH codes can be considered to be a tree. The structure of the tree is used here to give us a measure of the distance between individual MeSH terms. This distance measure was computed as follows.

The collection frequency was calculated for each node in the tree; that is, the number of documents in the collection annotated with the corresponding MeSH term. These collection frequencies were used to calculate weighted collection frequencies where the weighted collection frequency of any node in the tree is the number documents that contain a MeSH term corresponding to that node or to any descendant of the node. This is motivated by the notion that the descendants are members of the class described by their parent. Intuitively, this corresponds to the idea of pushing collection frequencies up the tree. The weighted collection frequency of the root[1] is just the count of all documents in the collection, since all MeSH terms are descendants of the root.

A simplified version of this calculation is shown in Figure 1. The left-hand tree shows some assumed collection frequency values, and the right-hand tree shows the corresponding weighted collection frequencies.

In practice we adopted a slightly more complex counting procedure. In general Medline citations are annotated with several MeSH terms. In calculating the weighted collection frequency for a node, it is possible that a single document could contribute to the collection frequency of more than one descendant. In this case, the weighted collection must be calculated to reflect this. Thus, the weighted collection frequency of a node is less than the sum of its own collection frequency and its children's weighted collection frequencies if the same document appears more than once among the children.

The similarity of MeSH terms was calculated in terms of these weighted collection frequencies. In particular, the similarity between two MeSH terms was taken to be the the weighted document frequency of their lowest shared ancestor divided by the total number of documents in the collection. For example, the similarity of nodes A and B in Figure 1 is $\frac{111}{125}$, whereas that of nodes C and D is $\frac{14}{125}$ (with smaller values indicating greater similarity).

[1] The MeSH classification system is actually a forest rather than a single tree. We obtained a single tree by artificially creating a new root that is a parent of all the actual roots.

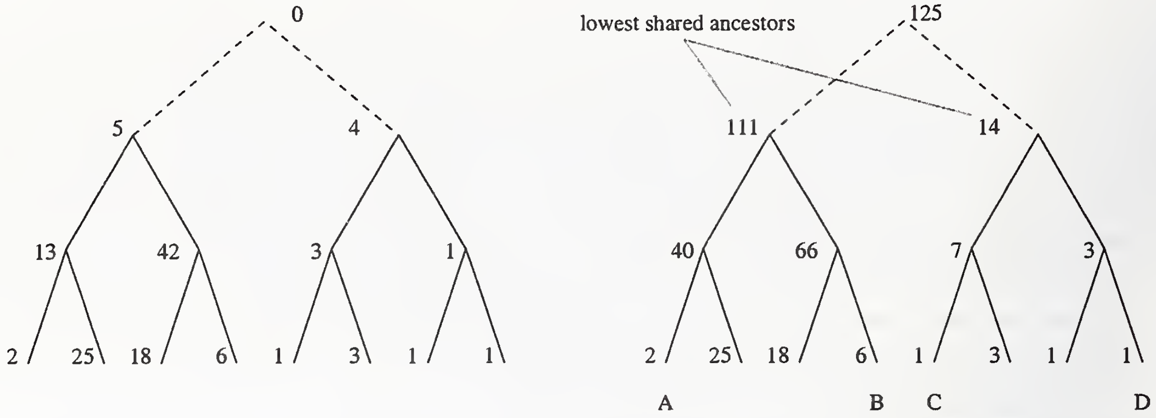


Figure 1: Calculation of weighted collection frequencies from collection frequencies.

3.2 Method 1

The TrecGen search topics do not include MeSH terms. Thus in order to exploit the terms contained in the documents themselves our first method sought to form MeSH term search queries using a pseudo-relevance feedback approach. For each of the 50 topics we took the top 100 PMIDs (document identifiers) in the lists provided by Jacques Savoy and formed a list of corresponding MeSH terms (with duplicates removed). A fixed number of these top ranked documents were then assumed to be relevant for each topic and the MeSH terms contained in these documents ranked using Robertson's Offer Weight [7]. The top ranked terms were then used as a search query which was scored against the top ranked documents.

The similarity between each MeSH term in the query and each document was computed using the distance tree. To compute the matching score between the query and each document, for each query term the highest scoring matching term was found in the document. The overall matching score between the query and document was taken as the highest scoring individual term match between the query and the document, with the highest scoring value the deciding factor. This score was then combined with the original matching score for the document computed by Savoy in a weighted sum and the document list re-ranked.

A number of runs were carried out with the training topic set to optimise the parameters of the system. After experimentation the following values were selected for the submitted test run. The top 80 ranked terms generated from assuming that the top 100 documents were relevant were used as the search query. It was only found to be beneficial re-rank the top 20 documents from Savoy's list. This effectively means that this method has no effect on precision at cutoff values below 20.

This then lead to a new score for each of the documents in the top 20 which were combined with the remaining 980 documents to produce a final ranking. The score combination was carried out by first normalising both Savoy's Okapi score and our MeSH derived matching score with respect to the highest score in each list and then multiplying Savoy's score by 9 and adding to our score.

The result from our official Method 1 submission is shown in Table 3. Table 4 shows the percentage changes for this method compared to Savoy's baseline run. It can be seen that Method 1 produces small improvements in average precision and precision at various cut off levels.

The use of the maximum term-term matching score between the query and document is not the standard approach taken in information retrieval where we would normally expect to use the sum of the matching scores for all the query terms. However, the use of the MeSH tree and the distance matching score between the terms in the query and the document is not a standard approach in information retrieval. In addition to the official submission we carried out exploratory runs using the sum of the query term matching scores. The results of these runs produced significant reductions in retrieval effectiveness and thus we did not use this approach in the final submission.

3.2.1 Further Experiments

Subsequent to our official submission we considered this issue further. That we found best performance on the development topics by assuming that the 100 ranked documents are relevant with 80 selected expansion terms for the search query is not a typical pseudo-relevance feedback result for Robertson's query expansion method. We would more typically expect to assume 10 documents relevant and form a query by selecting about the 20 top ranked terms. Given that we know that most of the assumed relevant documents in 100 will be non-relevant and that many of the 80 selected terms will not be related to relevant documents, it is clear that many of the terms in the query will be "noise" among the useful terms. In standard information retrieval where an exact match is required between the query and document terms many of these noise query would fail to find a match in the documents or would do so in small numbers and not impact significantly on the ranking of the overall query document matching scores. By computing a non-zero matching score between all term-term pairs the summation overall query terms will introduce significant noise in the overall score and lead to potentially meaningless ranking of documents. By contrast in this situation using the maximum term-term matching score as the query document matching score will in general eliminate the noise since the best match is likely to be between two strongly related (possibly identical) nodes in the MeSH tree.

In order to attempt to overcome this problem we conducted further experiments computing the query-document matching score the opposite way round. We know that each document has a (usually) small number of carefully selected highly significant MeSH terms associated with them. Thus we decided to match each document against the query. In this case we only include term-term associations once for each document term and we expect that most of the noise terms in the query will not often be the highest scoring matching term for the document terms. Thus the term-term matching scores should now on average be more meaningful.

3.3 Method 2

This section now describes, Method 2, a second, more ad hoc approach to using MeSH similarity to improve retrieval effectiveness. Under Method 2, each document was compared pairwise with each of the top 10 ranked documents to calculate This mesh score was calculated as illustrated in Figure 2. A pairwise score is calculated for document paired with each of the top 10 ranked documents. The mesh score for each document is the maximum of the pairwise scores. The pairwise score was calculated as

```
for each document i
  mesh-score[i] = 0
  for each top 10 document j, where j <> j
    score = 0
    for each MeSH term Mi in i
      for each MeSH term Mj in j
        if similarity(Mi, Mj) > score then
          score = similarity(Mi, Mj)
    if score > mesh-score then
      mesh-score[i] = score
  end for j
  mesh-score[i] = old-score[i] + CONSTANT * mesh-score[i]
end for i
```

Figure 2: Method 2 Algorithm

	Okapi run (SMART)	Method 1	Method 2
Mean Average Precision	0.1635	0.1669	0.1667
Precision at 5 docs	0.1480	0.1560	0.1680
Precision at 10 docs	0.1280	0.1360	0.1360
Precision at 15 docs	0.1133	0.1120	0.1187
Precision at 20 docs	0.1000	0.1040	0.0980
Precision at 30 docs	0.0827	0.0827	0.0058

Figure 3: Experimental Results

the maximum mesh similarity between any pair of MeSH terms between the documents.

The CONSTANT referred to on the second last line of the algorithm is simply a weighting factor (or a fudge factor). Decreasing this factor decreases the influence of the mesh scores on the resulting ranking, whereas increasing it, increases the impact of the mesh scores on the resulting ranking.

4 Experimental Evaluation

The experimental results are summarised in Figures 3 through 6. Figure 3 illustrates the mean average precision and precision at N for the baseline Okapi run, and the two methods described here. Percentage improvements are summarised in Figure 4.

Figures 5 and 6 are the equivalent results for the extended methods described in Section 3.2.1. It can be seen that there is improvement in the average precision for selection of the maximum term-term matching score between the document and the query (the same method as used in the official submission), but that the result for summing across all the terms appearing in each document is now better than the maximum matching score approach. Results for cutoff precision however still appear to be favour the maximum matching score approach.

	Okapi run (SMART)	Method 1	Method 2
Mean Average Precision	0.1635	2.08%	1.96%
Precision at 5 docs	0.1480	5.41%	13.51%
Precision at 10 docs	0.1280	6.25%	6.25%
Precision at 15 docs	0.1133	-1.15%	4.77%
Precision at 20 docs	0.1000	4.00%	-2.0%
Precision at 30 docs	0.0827	0.00%	-92.99%

Figure 4: Percentage Improvement

	Okapi run (SMART)	Max Score	Summed Scores
Mean Average Precision	0.1635	0.1680	0.1689
Precision at 5 docs	0.1480	0.1720	0.1560
Precision at 10 docs	0.1280	0.1240	0.1240
Precision at 15 docs	0.1133	0.1040	0.1013
Precision at 20 docs	0.1000	0.1010	0.0960
Precision at 30 docs	0.0827	0.0820	0.0820

Figure 5: Experimental Results

5 Conclusion

Our results for the TrecGen task demonstrate that incorporating relationships between the MeSH term labels of documents retrieved using a standard information retrieval can lead to small improvements in both average precision and cutoff precision.

The experiments described in this paper are of an exploratory nature and further work needs to be concentrated in several areas. In particular the method used to compute term-term similarity in the MeSH is only one of many possibilities and these need to explored and analysed in detail. Further than this it might be possible to use the structure of the MeSH tree more effectively than computing a term-term similarity metric.

Acknowledgements. We would like to thank Jacques Savoy of the University of Neuchatel, Switzerland, for his generosity in making available the Okapi runs that were the basis for the experiments described here. We would also like to thank Susan McDonnell and Donal O'Shea of the School of Biotechnology at DCU for their help and advice on the biological background to this work. In addition, this work was funded in part by the GenIRL (Genomic Information Retrieval using Links) Project, sponsored under Enterprise Ireland's Basic Research Grants Scheme.

References

[1] DNA Data Bank of Japan.
`http://www.ddbj.nig.ac.jp/`.

	Okapi run (SMART)	Max Score	Summed Scores
Mean Average Precision	0.1635	2.75%	3.30%
Precision at 5 docs	0.1480	16.22%	5.41%
Precision at 10 docs	0.1280	-3.13%	-3.13%
Precision at 15 docs	0.1133	-8.21%	-10.59%
Precision at 20 docs	0.1000	1.00%	-4.00%
Precision at 30 docs	0.0827	-0.85%	-0.85%

Figure 6: Percentage Improvement

[2] European Molecular Biology Laboratory.
http://www.ebi.ac.uk/Databases/.

[3] L. Hunter. *Artificial Intelligence and Molecular Biology.* Out of print, now available on the web, 1993.
http://www.aaai.org//Library/Books/Hunter/hunter.html, see particularly chapter 1.

[4] National Center for Biological Information.
http://www.ncbi.nlm.nih.gov/.

[5] N. L. of Medicine. Medical subject headings. http://www.nlm.nih.gov/mesh/.

[6] U. N. L. of Medicine. http://www.ncbi.nlm.nih.gov:80/entrez/query/static/overview.html.

[7] S. E. Robertson and K. S. Jones. Simple proven approaches to text retrieval. Technical Report Technical Report TR356, Cambridge University Computer Laboratory, 1997.

[8] J. Shrager. Just enough molecular biology for compter scientists. Notes available on web, 2002.
http://aracyc.stanford.edu/~jshrager/jeff/mbcs/.

Searching for geneRIFs: concept-based query expansion and Bayes classification

Rob Jelier, Martijn Schuemie, Christiaan van der Eijk, Marc Weeber, Erik van Mulligen, Bob Schijvenaars, Barend Mons, Jan Kors.

Rob Jelier
Department of Medical Informatics, Erasmus MC
PO Box 1738,3000 DR Rotterdam,The Netherlands
tel +31 10 4088124, fax +31 10 4089447, r.jelier@erasmusmc.nl

Primary Task

Introduction

The Biosemantics group at the Erasmus MC followed a thesaurus-based approach for the first task of the genomics track. The approach is based on a concept-based indexing engine (Collexis®), suitable for large -cale and high-speed indexing. The thesaurus used for indexing was constructed as a combination of the MESH thesaurus and the gene ontology (GO), with a species-specific gene thesaurus derived from LocusLink gene annotations. Our indexing machinery produces per indexed MEDLINE abstract a list of concepts with an accompanying weight, termed a fingerprint. Searching is done by matching a query fingerprint against fingerprints of all indexed MEDLINE abstracts. Query fingerprints are generated by combining fingerprints of four types. First, a fingerprint containing just the gene concept with all the known gene names and aliases. Second, a combination of MEDLINE fingerprints of all abstracts in which the gene concept was found without ambiguity problems. Third, a generic fingerprint with concepts typical of geneRIFs, when compared to MEDLINE in general. Fourth, a fingerprint containing the concepts of the Gene Ontology (GO) annotation

When it comes to identifying a gene name in a text the large number of synonyms and the frequent occurrence of homonymy are problematic. In our approach we attempt to deal with both. Synonymy as found in Locuslink is incorporated in our thesaurus. An attempt was made to reduce the effects of homonymy by expanding the query with fingerprints where the gene name is found unambiguously. Gene specific information, the GO annotation, is included to select for the correct gene, but also to select for abstracts with terms about basic biology. The generic fingerprint is included to select for abstracts with terms about basic biology.

System description

Producing the thesaurus

The Locuslink database is used as a basis for producing a gene thesaurus. Different thesauri were produced for the different organisms. For all the genes described in the database the following annotations were allowed as synonyms: official symbol, preferred symbol, alias symbol, official gene name, preferred gene name, alias protein, preferred product and product. A distinction was made between symbols and long forms of the gene name or product. Before indexing the lvg2002 normalizing engine (http://umlslex.nlm.nih.gov/index.html) is used to normalize the words in the text and make the system more robust. This includes removal of all capitalization. Hence indexing occurs case-insensitive. As an exception to this rule, words are not normalized when at least half of the letters are in capitals. For building our thesaurus gene and protein symbols are not normalized, though long forms are normalized. If a symbol is composed of a letter and number combination the symbol is also included in lowercase. When symbols or long forms end with a number, two forms are included in the thesaurus to better match spelling variation, one with the number directly after the last letter, the other separated with a hyphen.

To expand the thesaurus with concepts from the biomedical domain all concepts of Gene Ontology (http://www.geneontology.org/) and all MESH concepts that have an unique identifier in UMLS are added (http://www.nlm.nih.gov/mesh/meshhome.html). The structure of these thesauri is not used. All terms are normalized. From the whole of the thesaurus all words with a length of one or two letters are removed.

Preparation of the texts

For the selected MEDLINE records title, abstract and MESH headings are retrieved. One of the variables manipulated in our experiment is the use of the abbreviation expansion algorithm described by (Schwartz and Hearst, 2003) to replace abbreviations with their matching long forms. Our hypothesis is that abbreviation expansion will reduce the ambiguity of the text. Next are the removal of stop words followed by normalization of the remaining words.

Indexing

For indexing Collexis® indexing software is used (http://www.collexis.com). Identified concepts are assigned a relevance score for vector representation. This value is based on term frequency multiplied with a factor selecting against general concepts (see equation 1). The values are subsequently divided by the value of the highest ranking concept of the document, thereby normalizing to a maximum value of 1. This is the data that will be queried. The list of concepts with relevance scores will be referred to as a fingerprint.

$$F_i = \left(\frac{1}{S_i + 1} \right)^{0.3}$$

Equation 1, factor F_i is used to select against general concepts. S_i represents the number of documents a concept c_i occurs in.

Matching algorithm (MA)

To match the search queries to the document fingerprints several formulas were used. The formulas are listed below. f_c q_c represent the value of a concept in respectively the fingerprint and the query. Len(v) represents the length of a vector: Len(v)= sqrt(sum(v_c^2)).

Search queries

Queries are constructed by combining the four search fingerprints:

1. Gene Name (GN). The first search fingerprint is the gene name, including its synonyms.
2. Gene Specific Context (GSC). The second is an expansion of the search with gene specific fingerprints, creating a gene specific context. Fingerprints from the TREC set containing the name of the gene are evaluated and only added to expand the query when they meet the following demands: a. the name (or synonym) of the gene found in the text does not have a homonym in our thesaurus, and it either contains a space (i.e. it is a long-form), or it contains a number (but does not start with a number). b. the abstract or corresponding MESH headings contained the species name associated with the gene.

vector	sum(f_c * q_c) / len(f)*len(q)
collexis	sum(1/s_i), s_i represents the number of documents a concept c_i occurs in. Concept c_i is a concept which occurs in both the query and the fingerprint it is compared with.
dice	(2*sum(f_c * q_c)) / (len(f)^2 + len(q)^2)
weighted	sum(f_c* q_c) * (m_f + 1) / (l_q + 1), where m_f is the number of matched concepts of *f*, l_q is the number concepts in *q*.

3. Generic Context (GC). The third fingerprint is constructed based on a database containing all fingerprints from the documents indicated by geneRIFs in Locuslink and on a database containing all fingerprints from the TREC set. All found concepts were extracted from the database and ordered based on relative overrepresentation in the geneRIF set. Ranking was done based on equation 2. Every concept with a value larger than two is admitted in the generic fingerprint, resulting in a total of 3217 concepts.
4. Gene Ontology (GO). The fourth fingerprint contains concepts representing the GO-annotation for the gene as found in the Locuslink database.

$$C={}^{2}\log\left(\frac{\left(\frac{S_{geneRIF}}{T_{geneRIF}}\right)}{\left(\frac{S_{TREC}}{T_{TREC}}\right)}\right)$$

Equation 2, The score C calculated using this equation is used to rank concepts for the generic fingerprint, S indicates the number of documents a concept occurs in, T the total number of documents in that set.

The query is constructed by combining the search fingerprints. For combination the weights assigned to the concepts of the different fingerprints are multiplied with a scaling factor, and combined by addition to the other fingerprints. Prior to matching the concepts of the query are multiplied with the factor calculated with equation 1, followed by normalization to 1.

Experiments

Various aspects of our system were tested using the TREC training set in order to find the optimal settings to be used for our final submission. The results presented in this paper may differ from those used for our submitted runs, because of the elimination of several errors in our software.

Variations in combinations

To find the optimal combination of fingerprints, abbreviation expansion and matching algorithm, an experiment was performed evaluating a large number of possible combinations. The different variables were tested with the following discrete values:

- Abbreviation Expansion (AE): on or off
- Gene name fingerprint: weight of 0, 0.5 or 1
- Other fingerprints: weight of 0, 0.1, 0.3 or 0.5
- Matching coefficient: Vector, Dice, Weighted or Collexis

In total 1448 variations were constructed, and for each combination performance on the training was tested.

Variation in the Gene Specific Context

This experiment was aimed at evaluating the contributions of the various requirements used to select those abstracts that will be combined into the GSC fingerprint. Using the optimal combination found in the previous experiment, the system was tested using the gene specific context constructed by varying the following requirements:

- Ambiguity requirement: on or off
- Species name requirement: on or off

Evaluation

The system was used and evaluated according to the standards of the TREC genomics track. The document collection consisted of 525,938 MEDLINE records where indexing was completed between 4/1/2002 and 4/1/2003. The training set were the 47 topics distributed by the track. GeneRIFs taken from LocusLink were the documents to be retrieved for every topic. The test set are the official topics for the TREC competition. As a measure for evaluation mean average precision (MAP) was used.

Comparison of test and training set

To test whether differences existed between the composition of the test and training set we also used our optimization scheme for the test set (after submitting results). Additionally, we calculated the difference in the ratio of # geneRIFs / # retrieved documents, for test and training set.

Manual evaluation of results

An expert in molecular biology manually evaluated the first 10 retrieved documents for 10 queries of the test set. As a standard for a good result the definition as distributed by the TREC organization was used:

> *For gene X, find all MEDLINE references that focus on the basic biology of the gene or its protein products from the designated organism. Basic biology includes isolation, structure, genetics and function of genes/proteins in normal and disease states.*

Results

Variations in combinations

In table 1 the 15 highest scoring settings for our system are represented. Table 2 shows the highest score when the condition in the first column is true. This allows comparisons between different parameters. For all pairs a statistical test was performed (paired t-test, N=47) to assess whether observed differences between settings are significant at the .05 level. The best settings have a higher score than the baseline consisting of a query with only the gene name (p=0.047). Also GC=0.5, GN=0 and MA=collexis scored significantly lower than the optimal settings.

Table 1, MAP scores for 15 highest scoring settings. Abbreviations: GN, genename; GSC, gene specific context; GC, generic context fingerprint; GO, go annotation; MA, matching algorithm; AE, abbreviation expansion.

	MAP	GN	GSC	GC	GO	MC	AE
1	0.374	0.5	0.1	0	0.1	dice	false
2	0.372	1	0.3	0	0.1	dice	false
3	0.368	1	0.3	0	0.3	dice	false
4	0.367	0.5	0.1	0	0.1	vector	false
5	0.366	1	0.1	0.1	0.1	vector	false
6	0.366	1	0.1	0.1	0	vector	false
7	0.363	1	0.1	0	0.3	vector	false
8	0.361	1	0.5	0	0.1	dice	false
9	0.361	1	0.5	0.1	0	vector	false
10	0.361	1	0.3	0.1	0.1	vector	false
11	0.36	1	0.3	0	0	dice	false
12	0.359	1	0.5	0.1	0.1	vector	false
13	0.359	1	0.1	0	0.1	vector	false
14	0.359	1	0.1	0.1	0.1	vector	true
15	0.359	0.5	0.1	0	0	dice	false

Table 2, maximum average MAP scores achieved when expression in the first column is true. Abbreviations: GN, genename; GSC, gene specific context; GC, generic context fingerprint; GO, go annotation; MA, matching algorithm; AE, abbreviation expansion.

Condition	MAP	Condition	MAP	Condition	MAP
Only GN	0.336	AE=true	0.359	GO=0.1	**0.374**
Only GN & GSC	0.360	GN=0	0.096	GO=0.3	0.368
Only GN & GC	0.338	GN=0.5	**0.374**	GO=0.5	0.343
Only GN & GO	0.341	GN=1	0.372	MA=dice	**0.374**
GC=0	**0.374**	GSC=0	0.341	MA=vector	0.367
GC=0.1	0.366	GSC=0.1	**0.374**	MA=weighted	0.331
GC=0.3	0.309	GSC=0.3	**0.372**	MA=collexis	0.269
GC=0.5	0.237	GSC=1	0.361		
AE=false	**0.374**	GO=0	0.366		

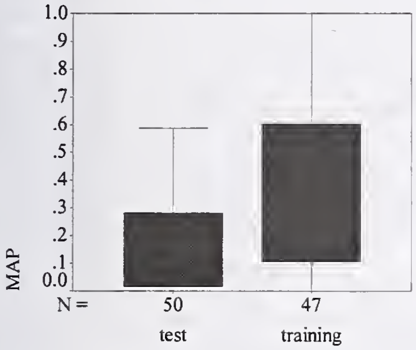
Variation in the gene specific context

Observed differences are not significant at the .05 level according to evaluation with the paired t-test.

	+Species	-Species
+Non-ambiguous indexing	0.37	0.36
- Non-ambiguous indexing	0.34	0.34

Table 3, average MAP for different ways to produce gene specific context, other parameters same as at the best configuration

Queries with the test set



	Average MAP
test	.17
training	.37

Figure 1, results on training and test set, boxplot and average MAP

Comparison of test and trainingset

Manual evaluation of results

Abstracts Analyzed	100
Relevant, geneRIF	31
Relevant, not geneRIF	53
Disputable*	14
Irrelevant	2

Table 4, summary of results of expert evaluation, first 10 results for 10 query. The annotator called an document Disputable when it contained content about the gene but less about function or also about (many) other genes

Discussion

The simplest query within our system, only the gene name, already gives a reasonable average MAP of 0.33. After optimization, the use of the gene specific fingerprint and the GO annotation fingerprint led to a better result than optimized settings for the gene name alone. Let it be noted that most of the other observed differences in table 1 are not statistically significant and that a larger set would be required to determine real effects. The results give a weak indication that the gene specific context is mostly responsible for the improvement. The GO annotation fingerprint could have a neutral or slight positive effect. The significance of the generic fingerprint is difficult to asses, though it clearly has a detrimental effect when given a large role in the query.

When the gene specific context is studied in more detail, the conditions for inclusion of fingerprints seem to play an important role. Though the differences in table 2 are small and may potentially reflect no real effects, there appears to be a trend towards better performance with more specific context.

The role of the abbreviation expansion algorithm is apparently minimal. One could expect more specific indexing (and better performance) as ambiguous acronyms are removed. On the other hand it has been observed that the long forms that are put in place have much more variations and hence are more likely not to be included in the thesaurus.

The optimal settings retrieved for the training set resulted in an average MAP score of 0.37. This appears to be reasonably good when compared to the preliminary runs reported previously (e.g. 0.35 by Prof. Jacques Savoy with the SMART system) on the TREC website. A very different score was achieved for the test set, an average MAP of 0.17. The boxplots in figure 1 clearly show that a very different distribution of MAP scores exists for the test set. A possible explanation could be that the system was over-trained on the training set and that the optimal result on the training set is far from robust. When we optimized settings for the test set the MAP score improved only modestly (MAP = 0.19), which rules out over-training. A better explanation would be that the test and training set are different in composition. After preliminary experimentation with the training set during the preparations for the TREC it was decided to exclude genes for the testset that have only one or two geneRIFs. This most likely led to the striking difference in the average number of geneRIFs per gene, 6.2 for the training set and 11.3 for the test set. Also we found a significantly lower ratio of # geneRIFs / # retrieved documents by our system for the test set. If our system can not distinguish very well between geneRIFs and other retrieved documents, the geneRIFs become more spread out in the retrieved list of documents. This would explain, at least for a part, the difference in the results between the test and training set.

Expert evaluation of retrieved documents showed some clear tendencies, which were also noted by the evaluation by the group of William Hersh (TREC genomics overview presentation, 2003). All checked geneRIFs were considered to be appointed appropriately. A large number of other retrieved documents, however, also appear to fit the description of a geneRIF. It therefore appears the collection of geneRIFs is incomplete. This makes the value of the evaluation of retrieval experiments with geneRIFs as standard difficult to assess.

Conclusion

The system of combining four different fingerprints was successful in improving performance relative to a query with the gene name. The most important contribution to this improved performance appears to be from the gene specific context fingerprint.

Results on the test set were much lower and different from those on the training set. This appears to be caused by a difference in composition of the sets.

Expert evaluation of queries showed that numerous results matched the given definition of a geneRIF but were not annotated as such. Large incompleteness in annotation and lack of difference between positives and some non-positives makes comparison of results very difficult.

Secondary Task

Introduction

With regard to the second task, our starting point was the observation that the GeneRIF annotators in approximately 42% of the cases simply use (part of) the title of a paper as annotation, in spite of the GeneRIF guideline that annotations are '*preferably more than a restatement of the title of the paper*'. Even when the title was not used, a test using the classic dice coefficient showed that the annotation usually matched (part of) a sentence from the abstract. For this reason we have reduced the problem from generating a GeneRIF to mimicking the annotator's choice for a certain sentence or title. We have approached this problem as a classification task using the naïve Bayesian classifier as proposed in (Mitchell, 1997).

Methods

The classifier constructed for this task assigns a given abstract to a class v_j that represents the choice to use one of the sentences as the annotation, where we define the title as sentence 0. To determine the number and type of the classes the set of 38.193 GeneRIFs was examined. For each GeneRIF we computed the classic dice coefficient between the annotation and each of the sentences and title of the corresponding abstract. In 16.163 (42 %) of the cases the annotation matched best with the title.

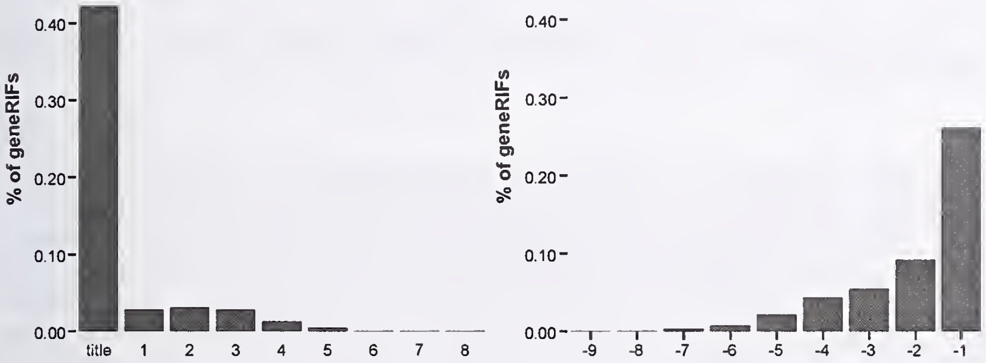


Figure 2, Distribution of positions of the annotation sentence, counting from the start and from the end of the abstract.

Figure 2 shows the distribution of the position of the non-title sentences that best matched the annotation. The first half of the figure shows the distribution of geneRIFs that were mapped to the first 8 sentences or the title of the abstract. The second half shows the same for the last sentences of the abstract. In 3.304 cases (9 %) the annotation was matched to one of the first three sentences and in 17.661 cases (46 %) the annotation matched one of the last five sentences. Because few abstracts (3%) in our test set were matched to annotation positions outside this set of 9, we did not include other positions. As a result, the classifier can assign to class v_j with j=0,..,8 which represent sentence 0 (i.e., the title) as annotation, the first sentence as annotation etc.

As features, we use the presence of normalized words in sentences. Both for training and classification, these features are determined by extracting sentences from the abstracts: nine sentences if the abstract has a length of at least nine, or less if the abstract is shorter. If the abstract is shorter than nine sentences we first extract the title, then the last five sentences, starting with the last moving backwards and then, if any sentences remain, the first three starting with the first. All words in a sentence are then normalized, using the lvg2002 normalizer. We also experimented with other features, such as presence of gene-symbols or thesaurus-based concepts in sentences, but this did not improve the results.
The prior probability of the class v_j is estimated by

$$P(v_j) = \frac{N_j}{N}$$

where N_j is the number of abstracts assigned to class v_j and N the total number of abstracts used for training.
The conditional probabilities of the occurrences $w_{k,i}$ of the normalized word k in sentence i given that the abstract is in class v_j, $P(w_{k,i} \mid v_j)$, is estimated as:

$$P(w_{k,i} \mid v_j) = \frac{n_{k,i,j} + \varepsilon}{n_{i,j} + \varepsilon \cdot d_i}$$

where $n_{k,i,j}$ is the number of occurrences of word k in sentence i in all abstracts in class v_j, and $n_{i,j}$ is the total number of distinct words in all abstracts in class v_j. The factor ε is added to ensure that $P(w_{k,i} \mid v_j)$ is never equal to zero, in which case the absence of a word in a sentence i in class v_j would cancel out all other probabilities in the next calculation. The variable d_i is the number of distinct words in sentence i. We established empirically that ε is best assigned a small value: for our experiments it was set at 10^{-6}.
An abstract a is assigned a class v_j by calculating v_{NB}:

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \cdot \prod_{i \in S} \cdot \prod_{k \in W_{a,i}} P(w_{k,i} \mid v_j)$$

where S is the set of sentence position and $W_{a,i}$ is the set of all words positions in sentence i in abstract a and V is the set of all classes.

Results

We trained our classifier on all GeneRIFs, excluding the 139 GeneRIFs that were specified as a test set for this secondary task. On the test set, the classifier correctly found the sentence from which the human annotation was derived in 65.47% of the cases. This compares favorably to the score of 43.88% for the naïve system that selects titles as annotation. However, during the TREC conference it was brought to our attention that our training set might contain GeneRIFs that duplicate with those in the test set, and indeed when we checked a number of LocusLink entries outside the test set turned out to have exactly identical GeneRIFs as those in the test set. This means that some LocusLink entries not only share PMIDs,but – rather surprisingly– annotations as well. Without these duplicates in the training set the performance of our classifier drops to that of the naïve system. The results reported in this section are our initial results obtained using all GeneRIF entries as a training set, only leaving out the test set.
Table 5 shows the results for the 139 test GeneRIFs using the TREC scoring system.

Classic Dice	54.37%
Modified unigram Dice	56.27%
Modified bigram Dice	44.58%
Modified bigram Dice phrases	46.25%

Table 5 Scores for all test GeneRIFs

The classic Dice coefficient was somewhat lower than could be expected from our calculated classification score at 54.37%.
Table 6 shows the scores of all abstracts that according to our measures were classified correctly

Classic Dice	73.26%
Modified unigram Dice	75.93%
Modified bigram Dice	67.30%
Modified bigram Dice phrases	69.32%

Table 6 Scores for correctly classified GeneRIFs

The results in Table 6 suggest that the annotators often change sentences taken from the abstract. Table 7 shows the scores for incorrectly classified abstracts.

Classic Dice	24.38%
Modified unigram Dice	25.02%
Modified bigram Dice	6.65%
Modified bigram Dice phrases	8.35%

Table 7 Scores for incorrectly classified GeneRIFs

Because some words in the suggested annotation occur in the actual annotation the classic dice score is still 24.38 %. The scores that include some measure of word order are lower, because the two sentences are very different.

Table 8 shows how a hypothetical perfect classifier would perform for the 139 GeneRIFs. This classifier always selects the sentence from the abstract that according to the classic dice coefficient most closely matches the actual annotation.

Classic Dice	70.36%
Modified unigram Dice	72.69%
Modified bigram Dice	62.51%
Modified bigram Dice phrases	64.83%

Table 8 Scores for 139 GeneRIFs using a perfect classifier

The scores in table 8 are very similar to those in table 2, albeit slightly lower, likely because the annotations that were incorrectly classified also differed most from the original sentences in the abstract.

Discussion

Our classifier is capable of selecting the title or sentences of the abstract that best matched the human annotations in 65.47 % of the cases. This is a considerable improvement when compared to the trivial method of always taking the title as the annotation, which yields a score of 43.88 %. It should be noted, however, that our high score relies on duplicate entries in the training set. Once removed, our classifier performs no better than the simple approach of always selecting the title. One could argue that for randomly selected GeneRIFs, which may have duplicate entries, our classifier performs well, but for entirely new GeneRIFs the classifier will perform no better than the baseline algorithm, which selects titles in all cases.

At least two directions for further improvement of our approach can be envisaged. First, the performance of the classifier might be improved; possibly by incorporating other features than the ones we have experimented with so far. However, even a perfect classifier will only give a moderate improvement of the TREC scoring measures. For example, the classic dice score of 54.37 % that was obtained for our current classifier will only become 70.36 % when the classifier is perfect.

A second direction for improvement could be a post-processing of the annotations suggested by the classifier to better match the human annotations. NLP techniques might be helpful in this respect. However, before embarking on further research, it would seem important to assess the quality of the GeneRIFs and to determine in how far a suggested annotation that partially differs from the actual annotations, could serve as a GeneRIF equally well.

References

- Schwartz,AS, M A Hearst, 2003, A simple algorithm for identifying abbreviation definitions in biomedical text: Pac.Symp.Biocomput., p. 451-462
- Mitchell TM. Machine learning. McGraw-Hill: New York, 1997.

Fondazione Ugo Bordoni at TREC 2003: robust and web track

Giambattista Amati, Claudio Carpineto, and Giovanni Romano

Fondazione Ugo Bordoni
Rome Italy

Abstract

Our participation in TREC 2003 aims to adapt the use of the DFR (Divergence From Randomness) models with Query Expansion (QE) to the robust track and the topic distillation task of the Web track. We focus on the robust track, where the utilization of QE improves the global performance but hurts the performance on the worst topics. In particular, we study the problem of the selective application of the query expansion. We define two information theory based functions, Info_{DFR} and InfoQ, predicting respectively the AP (Average Precision) of queries and the AP increment of queries after the application of QE. InfoQ is used to selectively apply QE. We show that the use of InfoQ achieves the same performance comparable of the unexpanded method on the set of the worst topics, but a better performance than full QE on the entire set of topics.

1 Robust Track

FUB participation in the robust track deals with the adaptation of the DFR modular probabilistic framework[2, 3, 1] together with query expansion based on distribution analysis[5, 6, 1] to this new task. In the robust track there are two new evaluation measures, the number of topics with no relevant documents in the top retrieved 10 (denoted by NrTopicsNoRel) and MAP(X), a measure related to the average precision of the worst X topics.

Experiments on the collection against the queries of TREC 6, TREC 7 and TREC 8 showed that QE deteriorates the two new robustness measures:

1. Indeed, the number of topics with no relevant retrieved documents in the top 10 ranks, NrTopicsNoRel, increases when QE is activated.
2. With a similar trend, MAP(X) always diminishes when QE was adopted.

1.1 Submitted runs: QE was adopted in all queries

At the submission time we did not have a stable and robust QE activation method to improve the performance on both the old and the new evaluation measures. Although this year the description-only queries are quite long, the automatic application of QE to all queries seemed to be the safest way to achieve a higher value of MAP. However, QE is detrimental to both MAP(X) and to NrTopicsNoRel measures. We thus submitted 4 description-only runs with full QE to maximize global performance and one description-only run with all unexpanded queries, to partially account for the worst topics.

1.2 Term-weighting models

We used 4 different DFR within-document term-weighting formulas: I(n)B2, I(n)OL2, I(n_e)OL2, I(n_e)OB2.

The models I(n)OL2, I(n_e)OL2 are variants of the model I(n)L2, while I(n_e)OB2 is a variant of I(n_e)B2.

For sake of space we just report the model I(n)OL2:

$$\begin{aligned} \text{I(n)OL2}: \quad & \frac{tfn}{tfn+1}\log_2\left(\frac{|\text{Collection}| - \text{doc\_freq}+1}{\text{doc\_freq}+0.5}\right) \qquad (1) \\ \text{where} \quad & tfn = tf \cdot \log_2\left(1 + c \cdot \frac{\text{average\_document\_length}}{\text{document\_length}}\right) \end{aligned}$$

The value of the parameter c of the within-document term-weighting DFR models was set to 3 [3, 1, 2].

1.3 Query expansion

The QE method was the same as used an TREC-10 with very good results[2] except for the parameter tuning and some additional expansion weight models.

The weight of a term of the expanded query q^* of the original query q is obtained as follows:

$$\text{weight}(\text{term} \in q^*) = tfq_n + \beta \cdot \frac{\text{Info}_{\text{DFR}}(\text{term})}{\text{MaxInfo}_{\text{DFR}}}$$

where

- tfq_n is the within-query term-frequency tfq of the term, normalized w.r.t. the maximum

$$tfq_n = \frac{tfq}{\arg\max_{t \in q} tfq} \qquad (2)$$

- Info_{DFR} is related to the probability of term-frequency computed by a DFR model:

$$\begin{aligned} \text{Info}_{\text{DFR}}(\text{term}) = & \; -\log_2 \text{Prob}(\text{Freq}(\text{term}|\text{TopDocuments})|\text{Freq}(\text{term}|\text{Collection})) \quad (3) \\ \text{MaxInfo}_{\text{DFR}} = & \; \arg\max_{\text{term} \in q^*} \text{Info}_{\text{DFR}}(\text{term}) \end{aligned}$$

Table 1: The number of selected documents on the first-pass retrieval is 10, the number of the extracted terms for the query expansion is 40.

Parameters	Runs with QE				Run without QE
	fub03InB2e3	fub03IeOLKe3	fub03InOLe3	fub03IneOLe3	fub03IneOBu3
	DFR Models				
$c = 3$	I(n)B2	I(n_e)OL2	I(n)OL2	I(n_e)OL2	I(n_e)OB2
	DFR Expansion models				
$\beta = 0.4$	Bo2	KL	Bo2	Bo2	-
	old topics				
@10:	0.3360	0.3360	0.3380	0.3300	0.3080
MAP :	C.1317	0.1315	0.1340	0.1343	0.1134
top 10 with No Rel.:	13	12	12	12	7
MAP(X)	0.0047	0.0061	0.0070	0.0057	0.0052
	new topics				
@10 :	0.5000	0.4780	0.4880	0.4660	0.4800
MAP:	0.3552	0.3692	0.3697	0.3614	0.3524
top 10 with No Rel.:	5	6	5	8	4
MAP(X)	0.0192	0.0117	0.0152	0.0098	0.0232
	all topics				
@10:	0.4180	0.4070	0.4130	0.398	0.3940
MAP:	0.2434	0.2503	0.2519	0.2479	0.2329
top 10 with No Rel.	18	18	17	20	11
MAP(X)	0.0084	0.0065	0.0077	0.0058	0.0096

In particular, the DFR models used were the normalized Kullback-Leibler measure (KL) [5, 2], and the following Bose-Einstein statistics (Bo2) [1]:

$$\begin{aligned} \mathrm{Info}_{\mathrm{Bo2}}(\mathrm{term}) = &\quad -\log_2\left(\frac{1}{1+\lambda}\right) - \mathrm{Freq}(\mathrm{term}|\mathrm{TopDocuments})\cdot\log_2\left(\frac{\lambda}{1+\lambda}\right) \quad [\mathrm{Bo2}] \\ \lambda = &\quad \mathrm{TotFreq}(\mathrm{TopDocuments})\cdot\frac{\mathrm{Freq}(\mathrm{term}|\mathrm{Collection})}{\mathrm{TotFreq}(\mathrm{Collection})} \end{aligned} \tag{4}$$

where TopDocuments denotes the pseudo-relevant set. The other parameters were set as follows:

- $\beta = 0.4$
- $|\mathrm{TopDocuments}| = 10$ and the number of terms of the expanded query is equal to 40.

1.4 Selective application of QE: new experiments

The official results confirmed the outcomes of our preliminary investigation as shown in Table 1. The unexpanded run achieved the best MAP(X) and the lowest NrTopicsNoRel, and the runs with expanded queries achieved the highest values of MAP and precision at 10.

[1] The query-term must also appear at least in 2 retrieved documents. This condition is to avoid the noise of the highly informative terms which appear only once in the set of the topmost retrieved documents.

In the following we study the problem of selectively applying QE to the set of topics. In particular in Section 1.6 we define the function InfoQ which predicts when QE can be applied.

We also establish that the sum

$$\text{Info}_{\text{DFR}}(q) = \sum_{\text{term} \in q} \text{Info}_{\text{DFR}}(\text{term})$$

of Formula 3 over the set of all terms of the query is correlated to the Average Precision (AP) of the system on that query. Therefore $\text{Info}_{\text{DFR}}(q)$ can measure the *topic-difficulty*, that is Info_{DFR} can be an indicator of a possible low outcome of AP with a topic q.

1.5 How QE affects the Robust track

Consider as an example the performance of the run fub03InOLe3 in Table 2; fub03InOLe3 uses the model I(n)OL2 (see Formula 1). With full QE, we achieved an increase of MAP equal to +7.5% with respect to the baseline run. If we had an oracle telling us when to apply QE query-by-query, the performance increase would nearly double passing from +7.5% to +13.3%.

However, without the oracle a wrong decision of omitting the QE mechanism would seriously hurt the final MAP of the run. The average gain per query is ~0.063 and the gain is much greater than the average loss (~0.039). Moreover, the number of cases with a successful application of QE (57 out 100) is larger than the number of the failure cases. Both odds are thus in favour of the application of QE.

In the robust track, the success rate of the QE application was below our expectation. Comparing the figueres of Table 2 with those relative to all the 150 queries of the past TREC data, we have observed a detriment of the success rate. The success rate was around 65% with all the 150 old queries of past TREC data. A detriment in precision at 10 was observed for only 15% of all the 150 old queries (against 19% of the TREC 2003 queries).

In addition, the increase of MAP with QE using all the old 150 queries was larger (~ +10%) than that obtained with the new queries (~ +5%).

In the next section we propose the measure InfoQ to predict successful application of QE. InfoQ is indeed correlated to the increment of Average Precision after QE activation.

1.6 Predicting the successful application of QE with InfoQ

Let Q be a sample of queries q and let

$$\text{InfoPriorQ}(q) = \sum_{\text{term} \in q} -\log_2 \frac{\text{Freq(term|Collection)}}{\text{TotFreq(Collection)}}$$

InfoPriorQ has a moderately weak negative correlation with QE, that is:

$$\rho(\text{AP}_{\text{QE}} - \text{AP},\ \text{InfoPriorQ}) = -0.27$$

where AP_{QE} is the average precision after the application of QE, and AP denotes the average precision of the system without QE.

Table 2: Run fub03InOLe3 with description-only topics. The columns with "No QE" contain the number of queries to which the QE was not applied.

Old Topics											
Baseline		run fub03InOLe3 with QE				Runs with the oracle					
MAP	P@10	MAP	%	P@10	%	MAP	%	No QE	P@10	%	No QE
0.1147	0.3100	0.1340	+14.4%	0.3380	+ 8.3%	0.1432	+19.9%	21/50	0.3640	+14.8%	10/50
New Topics											
Baseline		run fub03InOLe3 with QE				Runs with the oracle					
MAP	P@10	MAP	%	P@10	%	MAP	%	No QE	P@10	%	No QE
0.3512	0.4780	0.3697	+5.0%	0.4880	+2.1%	0.3942	+10.9%	22/50	0.5160	+7.4%	9/50
All Topics											
Baseline		run fub03InOLe3 with QE				Runs with the oracle					
MAP	P@10	MAP	%	P@10	%	MAP	%	No QE	P@10	%	No QE
0.2330	0.3940	0.2519	+7.5%	0.4130	+4.6%	0.2687	+ 13.3%	43/100	0.4400	+ 10.5%	19/100

InfoPriorQ is also linearly related to the length of the query (ρ(QueryLength, InfoPriorQ) = 0.90. Query length is thus a different indicator of the successful application of the QE. A short query in general requires QE, but QE can be easily harmful for long queries, but using only the query length as an indicator QE varies its behaviour for moderately long queries.

We now introduce InfoQ to deal with the selective application of QE. InfoQ combines InfoPriorQ and the divergence-based function Info $_{\text{DFR}}$ which we have already encountered in Section 1.3. Info $_{\text{DFR}}$ query rankings may not agree using different DFR models. A way to compare different score functions over the same set Q of queries is to normalize using their standard normal scores.

Let

$$M_q = \max\left\{ \frac{\text{InfoPriorQ}(q) - \mu_{\text{InfoPriorQ}}}{\sigma_{\text{InfoPriorQ}}}, \max_{M \in \text{DFR}} \arg \frac{\text{Info}_M(q) - \mu_{\text{Info}_M}}{\sigma_{\text{Info}_M}} \right\}$$

The function:

$$\text{InfoQ} = \frac{1}{\text{QueryLength}} \left(\frac{\text{InfoPriorQ} - \mu_{\text{InfoPriorQ}}}{\sigma_{\text{InfoPriorQ}}} + M_q \right) \tag{5}$$

where the μ_Xs and the σ_Xs stand for the mean and the standard deviation of the X values over the sample Q of queries q.

Because the correlation factor between AP increment and InfoQ is negative, we need to trigger the QE when InfoQ is below a given threshold:

$$\text{Apply QE to query } q \Leftrightarrow_{\text{def}} \text{InfoQ}(q) < \text{threshold} \tag{6}$$

A cautious way to smooth different Info $_{\text{DFR}}$ values is to compare the threshold to the maximum value among all these DFR models, InfoPriorQ included. This explain why we

Table 3: The set of queries with the highest InfoQ. The QE is not applied to such queries.

QE success	InfoQ	Query Length	Topic
y	0.482	7	604
n	0.345	8	631
n	0.335	17	320
n	0.333	13	638
n	0.329	9	621
n	0.327	14	619

first compute the maximum value among the normal standard scores of InfoPriorQ and all $\mathrm{Info}_M(q)$ where M is a DRF model.

The standard normal query–scores of Info $_{\mathrm{DFR}}$ may not agree, even in sign, using different DFR models. Since the correlation factor is negative, and since we trigger the QE when InfoQ is below a given threshold, then a cautious way to smooth different Info $_{\mathrm{DFR}}$ values is to compare the threshold to the maximum value of all these DFR standard normal scores, InfoPriorQ included.

InfoQ has a higher correlation than InfoPriorQ (see Figure 3) with QE

$$\rho(\mathrm{AP}_{\mathrm{QE}} - \mathrm{AP},\ \mathrm{InfoQ}) = -0.33$$

and a smaller correlation factor with the query length[2]

$$\rho(\mathrm{AP}_{\mathrm{QE}} - \mathrm{AP},\ \mathrm{InfoQ}) = 0.62$$

1.7 Predicting topic difficulty with Info $_{\mathrm{DFR}}$

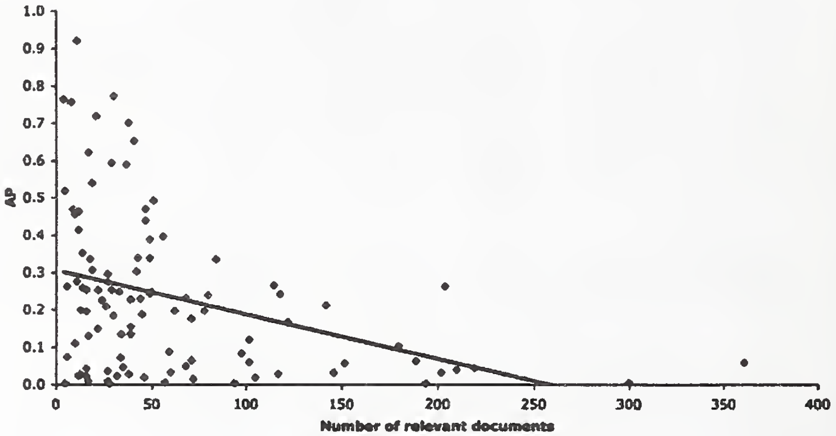
It is a well known evidence that the QE effectiveness is strictly related to the number of documents which are relevant for a given query in the set of the topmost documents in the ranking. If the early precision of the first-pass retrieval is high, then we have a good chance to extract good additional topic terms together with their relative query-weights. To start our investigation we have first computed the correlation factor beween

- the number *Rel* of relevant documents in the whole collection and the AP value over the 100 queries, and
- between *Rel* and the precision at 10(P@10).

The correlation value $-1 \leq \rho \leq 1$ indicates the degree of the linear dependence between the two pair of measurements. When $\rho = 0$ the correlation coefficient indicates that the

[2] Using $\log_2$(QueryLength) instead of QueryLength the score of Formula 5 is more correlated to the query length with $\rho(\mathrm{QueryLength},\ \mathrm{InfoQ}) = 0.74$ and $\rho(\mathrm{AP}_{\mathrm{QE}} - \mathrm{AP},\ \mathrm{InfoQ}) = -0.34$.

Figure 1: The number of relevant documents is inversely related to AP of the unexpanded query ($\rho(Rel, \text{AP}) = -0.36$). Queries with many relevant documents contribute little to MAP.



two variables are independent. When instead there is a linear correlation, the correlation coefficient is either -1 or 1 [8]. A negative correlation factor indicates that the two variables are inversely related.

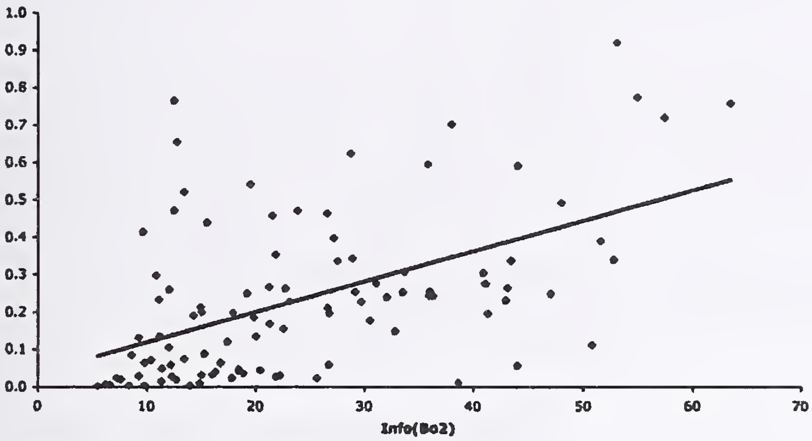
Surprisingly, both these correlation factors come out to be negative ($\rho(Rel, \text{AP}) = -0.36$ and $\rho(Rel, \text{P@10}) = -0.14$).

Although in these two cases the absolute values of the correlation coefficient are not close to -1, even small values of the correlation factor are regarded very meaningful especially in large samples [10].

Therefore, these values of the correlation factors seem to demonstrate that the greater the number *Rel* of relevant documents, the less the precision (MAP and P@10). An approximation line of the scatter line of the AP values for different numbers of relevant documents is produced in Figure 1. The fact that the correlation factor with AP is larger than that with P@10 is due to the definition of AP. The AP measure combines recall and precision by using the number *Rel* of relevant documents.

This negative correlation might appear to be counter-intuitive, since among the easiest topics there are many which possess a small number of relevant documents, and, as opposite, many difficult topics have many relevant documents. On the other hand, a possible explanation of these negative correlation factors is that a small number of relevant documents for a topic witnesses the fact that the topic is "specific" or "non-general" with respect to the content of the collection. In such a situation, common-sense says that specific queries have few relevant documents, their query-terms have few occurrences in the collection, and they thus are the easiest ones.

Figure 2: The information content Info $_{\mathrm{Bo2}}$ of the query within the topmost retrieved documents is linearly correlated to the AP of the unexpanded queries ($\rho(\text{Info}_{\mathrm{Bo2}}, \text{AP}) = 0.52$). Specific queries have a large value of Info $_{\mathrm{DFR}}$.



However, a definition of the specificity based on the number of relevant documents for the query would depend on the evaluation; we rather prefer to have a different but operational definition of the query-specificity or query-difficulty.

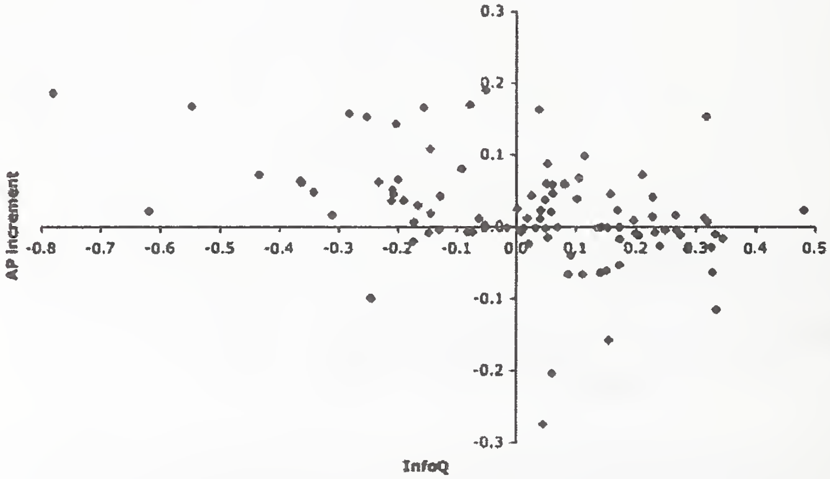
The notion of query-difficulty is given by the notion of the amount of information Info $_{\mathrm{DFR}}$ gained after a first-pass ranking. If there is a significant divergence in the query-term frequencies before and after the retrieval, then we make the hypothesis that this divergence is caused by a query which is easy-defined.

$$\text{difficulty score of } q =_{\text{def}} \text{Info}_{\mathrm{DFR}}(q) \text{ of Formula 3} \tag{7}$$

where DFR is a basic model (based on the Binomial, the Bose-Einstein statistics or the Kullback-Leibler divergence measure). We here use the probability of Bose-Einstein as defined in Formula (4). Note that the same weight was used by our expansion method in 3 runs out of the 4 expanded ones (fub03InB2e3, fub03InOLe3, fub03IneOLe3). The Kullback-Leibler divergence was adopted in the run fub03IeOLKe3 (see Table 1).

There are other information theoretic measures capturing the notion of term–specificity of the query. One possible choice, based on the language model, is the clarity score[7], but it is more difficult to implement. There is an interesting study [4] which found using the Pearson coefficient that there is no correlation between the average precision with the original query and s average precision increment by QE. Billerbeck and Zobel explored a range of query metrics to predict the QE success, but, as they report, without clear success. They assert to have included into this family the similarity score of the documents fetched in the original ranking; a measure of how distinct these documents were from the rest of

Figure 3: The information content InfoQ of the query based on the combination of the priors and Info $_{\mathrm{DFR}}$ within the topmost retrieved documents is negatively correlated to the AP increase with the QE (ρ(AP increase rate with QE, InfoQ) $= -0.33$). The first and the third quadrants contain the errors when the threshold is set to 0.



the collection; specificity of the query terms; and an approximation to query clarity.

The goodness of Info $_{\mathrm{DFR}}$ is tested with the linear correlation factor with AP of the unexpanded queries. The motivation is that easy queries usually yield high AP values. To compute the difficulty score of the query we first produced a first-pass ranking as it is done in QE. We took the set TopDocuments of the first 10 retrieved documents and we computed a score for each term occurring in the query. We considered the query-terms which appear at least twice in these pseudo-relevant documents. This score reflects the amount of information carried by the query-term within these pseudo-relevant documents. As shown in Figure 2, Info $_{\mathrm{DFR}}$ has a significant correlation with the AP of the unexpanded queries $\rho(\mathrm{Info}_{\mathrm{Bo2}}, \mathrm{AP}) = 0.52$. Similarly to the negative correlation between the number of relevant documents and the AP of the unexpanded queries, which is $\rho(Rel, \mathrm{AP}) = -0.36$, the correlation factor between the score InfoQ and Rel was negative ($\rho(Rel, \mathrm{Info}_{\mathrm{Bo2}}) = -0.23$). Again, this may be explained by the fact that specific queries possess fewer relevant documents.

We did not find a significant correlation between Info $_{\mathrm{DFR}}$ and QE; that is, Info $_{\mathrm{DFR}}$ is not able to predict a successful application of QE in a second-pass ranking. These results show that the performance of query expansion is not directly related to query difficulty, consistent with the observation [6] that although the retrieval effectiveness of QE in general increases as the query difficulty decreases, very easy queries hurt performance.

1.8 Conclusions on the selective application of QE

In Table 4 we summarize the results on the selective application of QE. The MAP(X) values are not reported since the new values are similar to those in the official runs; thus we focus on the other measures. We compare the performance of the 4 submitted runs with QE with the performance of the new runs under the same setting except for the selective application of QE.

The first remark is that the decision rule for QE activation is quite robust. The MAP of the new runs is greater than the MAP of the official runs for a large range of values of the threshold parameter ($>= 0$). In fact, InfoQ provides with a high degree of confidence the cases in which QE should be absolutely activated, which are the cases when InfoQ assumes very small negative values, as it can be seen in Figure 3. This explains why the new value of MAP keeps constantly larger than the MAP obtained with all queries expanded. This decision method is thus safe.The behavior of Precision at 10 is more variable, depending on the choice of the threshold.

The second observation is that selective QE positively affects the NrTopicsNoRel measure. The new runs have almost the same NrTopicsNoRel performance as the unexpanded runs, and this was one of the main objectives of our investigation.

2 Web track: topic distillation task

FUB participation in the topic distillation task of the Web track focused on only–content analysis.

Last year FUB didn't participate in this task, although the same baseline Information Retrieval system was employed by the Glasgow University (GU)[9].

In particular, GU analysed both the Absorbing Model and PageRank based algorithms for link analysis on top of our baseline IR system by using different content-link score combination approaches. Using the WEB corpus .GOV some utility functions combining link and text analyses were shown to moderately improve the performance over the baseline.

However, no query expansion technique was employed by GU in these TREC 11 experiments. As we successfully used query expansion for the "topic relevance task" at TREC 10, we checked whether the same strategy was also effective for the "topic distillation task" of TREC 11. The only modification we performed was to the value of the parameter c (from $c = 7$ to $c = 1$). We found that the application of the query expansion "as it was" applied in TREC 10 was still effective to the topic distillation task of TREC 11. Indeed, as shown in Table 5 we achieved better results than those reported by the best system participating in TREC 11. For topic distillation of this year we have applied exactly the same strategy as for the experiments on the TREC 11 collection.

However, official results show that the content-only term-weighting was not effective this year. The difference in performance is probably due to a change in the type of the task, with a different assessment of the notion of relevance. Judging from our results, the "topic distillation task of TREC 2002" looks very different from the "topic distillation task of TREC 2003".

Table 4: The selective application of QE.

Parameters	Runs with QE			
	fub03InB2e3	fub03IeOLKe3	fub03InOLe3	fub03IneOLe3
	DFR Models			
$c = 3$	I(n)B2	I(n_e)OL2	I(n)OL2	I(n_e)OL2
	DFR Expansion models			
$\beta = 0.4$	Bo2	KL	Bo2	Bo2
	all topics with QE			
@10:	0.4180	0.4070	0.4130	0.3980
MAP:	0.2434	0.2503	0.2519	0.2479
top 10 with No Rel.	18	18	17	20
topics with QE	100	100	100	100
InfoQ < 0.12	all topics with selective QE			
@10:	0.4230	0.3950	0.4210	0.3950
MAP:	0.2456	0.2543	0.2556	0.2524
top 10 with No Rel.	11	16	15	16
topics with QE	68	67	66	67
InfoQ < 0	all topics with selective QE			
@10:	0.4140	0.3950	0.4080	0.3950
MAP:	0.2439	0.2486	0.2527	0.2477
top 10 with No Rel.	11	16	14	16
topics with QE	41	41	37	41
	Baseline			
@10:	0.4080	0.3950	0.3940	0.3950
MAP:	0.2292	0.2282	0.2330	0.2282
top 10 with No Rel.	11	16	12	16
topics with QE	0	0	0	0

Acknowledgments

The experiments were conducted using the first version of the Terrier's Information Retrieval platform[3] initially developed by Gianni Amati during his PhD at Glasgow University.

References

[1] Giambattista Amati. *Probability Models for Information Retrieval based on Divergence from Randomness.* PhD thesis, Glasgow University, June 2003.

[2] Gianni Amati, Claudio Carpineto, and Giovanni Romano. FUB at TREC 10 web track: a probabilistic framework for topic relevance term weighting. In E.M. Voorhees and D.K. Harman, editors, *In Proceedings of the 10th Text Retrieval Conference TREC 2001*, pages 182–191, Gaithersburg, MD, 2002. NIST Special Pubblication 500-250.

[3] http://www.dcs.gla.ac.uk/ir/terrier/index.html

Table 5: Performance of the 5 submitted runs for topic distillation task on TREC 11 and TREC 12. (QE is with 3 documents and 10 terms, $c = 1$ and $\beta = 0.5$).

Topic Distillation TREC 11			
Models	QE	MAP	Pr @10
I(n)L2	Bo1	0.2166	0.3000
I(n)L2	B	0.2194	0.3020
I(n_e)B2	B	0.2251	0.2939
I(n_e)B2	BM	0.2246	0.2939
I(n)B2	BM	0.2012	0.2918

Topic Distillation TREC 12				
Run	Models	QE Models	MAP	Pr @10
fub03InLBo1t	I(n)L2	Bo1	0.0810	0.0580
fub03InLBt	I(n)L2	B	0.0778	0.0620
fub03IneBBt	I(n_e)B2	B	0.0799	0.0640
fub03IneBMt	I(n_e)B2	BM	0.0818	0.0620
fub03InBMt	I(n)B2	BM	0.0775	0.0640

[3] Gianni Amati and Cornelis Joost Van Rijsbergen. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Transactions on Information Systems (TOIS)*, 20(4):357–389, 2002.

[4] Bodo Billerbeck and Justin Zobel. When query expansion fails. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 387–388. ACM Press, 2003.

[5] C. Carpineto, R. De Mori, G. Romano, and B. Bigi. An information theoretic approach to automatic query expansion. *ACM Transactions on Information Systems*, 19(1):1–27, 2001.

[6] C. Carpineto, G. Romano, and V. Giannini. Improving retrieval feedback with multiple term-ranking function combination. *ACM Transactions on Information Systems*, 20(3):259–290, 2002.

[7] Steve Cronen-Townsend, Yun Zhou, and W. Bruce Croft. Predicting query performance. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306. ACM Press, 2002.

[8] Morris H. DeGroot. *Probability and Statistics*. Addison-Wesley, 2nd edition, 1989.

[9] V. Plachouras, I. Ounis, G. Amati, and C.J. Van Rijsbergen. University of Glasgow at the Web track of TREC 2002. In E.M. Voorhees and L. Buckland, editors, *Proceedings of the 11th Text Retrieval Conference TREC 2002*, Gaithersburg, MD, November 2003. NIST Special Publication.

[10] Robert G.D. Steel, Jamies H. Torrie, and David A. Dickey. *Principles and Procedures of Statistics. A Biometrical Approach*. MacGraw–Hill, 3rd edition, 1997.

FDUQA on TREC2003 QA task

Lide Wu, Xuanjing Huang, Yaqian Zhou, Yongping Du, Lan You
Fudan University, Shanghai, China

1 Introduction

It is the fourth time that we take part in the QA track. Our system, FDUQA, is based on our previous system (Wu et al, 2002). FDUQA includes an offline part and an online part. We make great efforts on the online part while leaving the offline part unchanged. We have tried many natural language processing techniques, and incorporated many sources of world knowledge, including Web. A novel Query formulation technique has also been put forward.

In addition, we've tried another attempt on answer extraction in this year's task. In the second section, we will describe the architecture of our QA system; and give a detailed description on the Query formulation for Web search in the third section; while in the fourth section, we will introduce our new attempt on answer extraction; and we will present our performance in the last section.

2 Architecture

FDUQA's architecture is shown in figure1. Our system can be divided in two ways. One is traditional: question analysis, retrieval, and answer extraction, as shown in figure 1 by the two horizontal lines. The other is more natural: answer type decision-making, candidate answer decision-making, and final answer decision-making, as shown in figure 1 by the two vertical lines. We'll describe the FDUQA system in the latter.

In the answer type decision-making step, FDUQA system determines the answer type of the input question based on the question's interrogative and focus words. The classifier and focus words decision algorithm are both based on heuristic rules. We adopt an eighteen-class answer type classify system, illustrated in table1. At this step our system can achieve the precision of 80%.

ABBR	NOUN_PHRASE	AGE
CAPITAL_WORDS	DESP_OF_ABBR	DATE
LOCATION	LENGTH	MEASURE
MONEY	NUMBER	ORGANIZATION
PERCENT	PREP_PHRASE	PERSON_NAME
SPEED	WRITING_NAME	NONTYPE

Table 1 Answer Type concepts

At the second step, candidate answer decision-making, our system searches the Web by Google and then tries to find the answer in the returned snippets. Finding an answer in the huge corpus is easier than in a smaller one in some sense, because system can search the corpus more easily and get more confident answer by stricter query (or query set). For example, questions such as "Where was Hans Christian Anderson born?" are very easy for Web search engine to find the answer by inputting query as "Hans Christian Anderson was born in". We'll describe the Query formulation for Web Search module in great details in the next section.

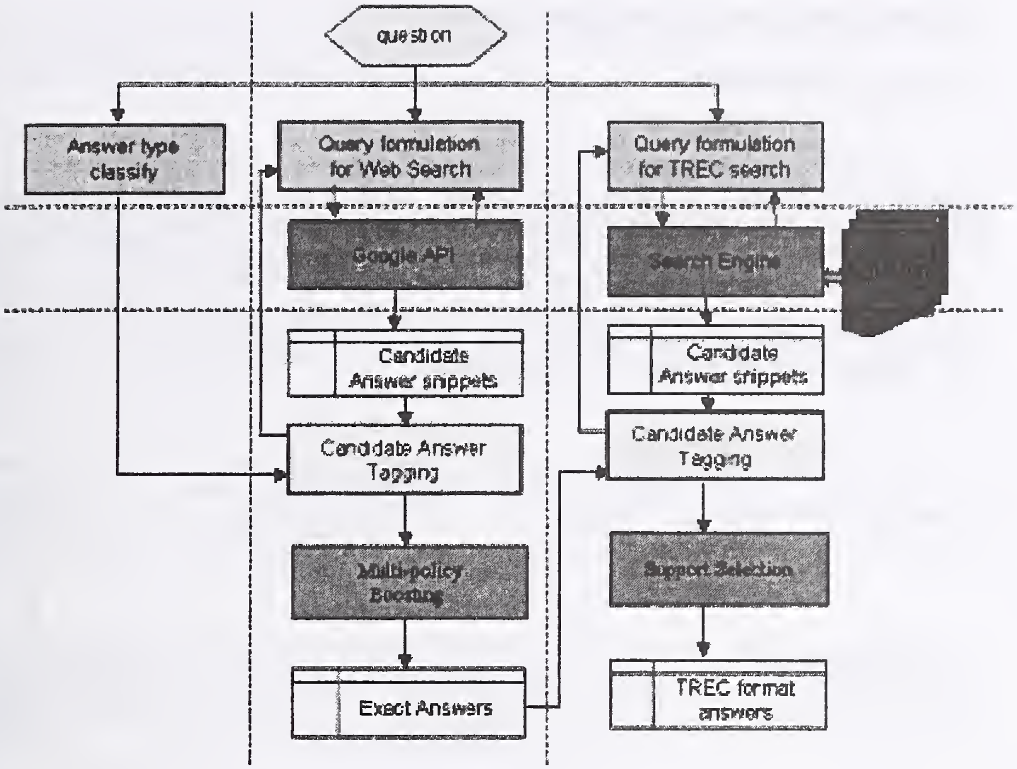


Figure 1 FDUQA system architecture

The upper two modules can be taken for question analysis and retrieval steps for the traditional QA system, while the following modules make up of the answer extraction step. Candidate answer tagging module tags candidate answers in the returned snippets based on their NE tagging and Base NP tagging results. Consider the distances of the key concepts and the distances between the candidate answers and key concepts, the multi-policy boosting module gives score to every candidate answer and snippet pair. The pairs are clustered by candidate answers, and each candidate answer set can get its score by adding up all of its elements' scores. Thus, the candidate answers can be sorted with their scores.

The third step is final answer decision-making section. At this step, the first two modules, Query generation for TREC search and search engine, are the same as last year. The following modules are almost the same as the corresponding modules in last step. The only difference between two "Candidate Answer tagging" modules is that system tagging the candidate answer in this step based on the candidate answer generated in last step, the candidate answer decision-making step. Support selection module sets the support score by adopting the same technique that used in multi-policy boosting module that give score to every candidate answer and snippet pair. The system integrates every candidate answer's support scores and their ranks in last step to sort them. FDUQA system considers the top one candidate answer as the final answer.

3 Query formulation for Web search

3.1 Query Formulation

Answer of a question may appear in a context, which is just the statement form of the question. For example, the answer of question "What book did Rachel Carson write in 1962?" appears in a context like "Rachel Carson wrote <Answer> in 1962". But mostly such a context doesn't exist in a limited corpus like AQUAINT. However, we do retrieval not only on the AQUAINT corpus, but also on Internet like some other systems (Kwok et al., 2001; Dumais et al., 2002). Because of the largeness and variety of information on Internet, this context can be retrieved now. Based on this idea, we formulate queries for Web retrieval. Figure 2 describes the process of query formulation.

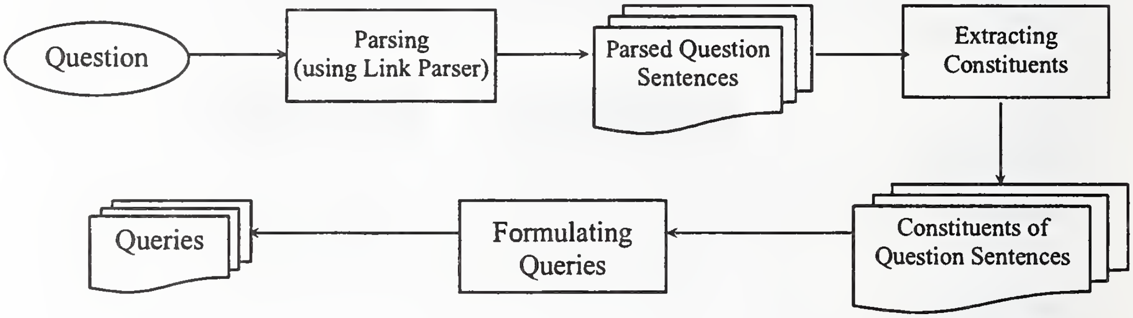

Question
Parsing (using Link Parser)
Parsed Question Sentences
Extracting Constituents
Constituents of Question Sentences
Formulating Queries
Queries

Figure 2 process of query formulation

Our system first parses questions using LinkParser (Sleator and Temperley, 1993), an English parser based on link grammar. Its precision is up to 0.9. Next we extract four constituents from the parsed question: subject, predicate, object and adverbial modifier. These constituents are then used to formulate queries for Web retrieval.

For example, we parsed the question "What book did Rachel Carson write in 1962?"

Its constituents are:

"Rachel Carson" – subject;
"wrote" – predicate;
"in 1962" – adverbial modifier.

In this question, object of "wrote" is the question focus.

The queries formulated from the above constituents are:

"Rachel Carson wrote" "in 1962"
"Rachel Carson wrote" in 1962
"Rachel Carson" wrote in 1962
Rachel Carson wrote in 1962

Words in quote marks must appear continuously in retrieved snippets, while others may appear dispersedly or even not appear. Obviously, the first query is a tight one. And the followings are relatively looser. We generate loose queries allowing for other forms of context on Web. For example, "1962 Rachel Carson wrote Silent Spring that was aimed at the general public and became the Uncle Tom's Cabin of the new environmentalism". This snippet can't be retrieved with the first query, but can be retrieved by the later three. And "...

Rachel Carson grew up on a small Pennsylvania farm, where she ... her degrees in 1932, she wrote science articles ... of the Sea, and finally Silent Spring in 1962" can only be retrieved by the last two queries.

3.2 Retrieval on Web

Among various Web search engines, we select Google because of its high performance. And the formulated queries are specified according to the requirement of Google. We submit queries to Google from tight ones to loose. Thus we can find snippets with answers for most of the questions.

We have done an experiment on questions of TREC2002 QA Track. In these 500 questions, TREC provided answers for 444 of them. So we only considered these 444 questions. And only the first 20 received results for each query are used. The result of Web retrieval is listed in table 2.

#question	#question (has answer in snippets)	#question (has answer in snippets and some snippet supports the answer)
444	367 (82.7%)	341 (76.8%)

Table 2 Web retrieval Result

We can find answers in the retrieved snippets for 82.7% of the 444 questions. And in a closer observation, the retrieved snippets support 341 answers, that is 76.8% of all these questions. Thus, most of the search results contain answers. It's important for the later processing.

4 New attempt for answer extraction

Pattern based method has been used by many other question answering systems, InsightSoft (Soubbotin and Soubbotin, 2001; Soubbotin and Soubbotin, 2001) has acquired good performance, ISI developed a method for learning patterns automatically (Ravichandran and Hovy, 2002).

We try a new pattern based method for implementing the answer extraction and give a solution to the problems that other system failed, such as only one key phrase of the question can be included within pattern. We will introduce the process of pattern learning and answer extraction with them.

The pattern for answer extraction is called context pattern, it is consisted of the following three parts: <Q_Tag>+[ConstString]+<A>. Here, <Q_Tag> stands for the key phrase in question, it includes different elements of the question, and we will introduce them later. <A> stands for the answer, any string holding the position will be extracted as the answer. "[ConstString]" is a sequence of words.

Context patterns can be learned automatically using the <Q_Tag , A> pairs as training examples. For instance, context pattern "<A>, Q_Focus of Q_NameEntity" can be used to answer the question "What is the capital of Syria?" "Q_Focus" represents the question term "the capital" and "Q_NameEntity" represents the question term "Syria".

We take the 500 questions of TREC 2002 as our training data for learning these context patterns.

4.1 Question Analysis

We define a set of notations to represent questions in advance as illustrated in table 3. They are the object or event the question asks about.

All these Q_Tag have different importance scores taking into account the possibility they appear around the answer.

The question pattern (Q_Pattern) is generated from its Q_Tag symbol set, and then the classification of questions will be built based on the Q_Pattern and the answer type. A case in point is that the question class " [DAT] When was Q_BNP_1 Q_Verb? " covered the question " When was Apollo 11 launched? ", " When was the first atomic bomb dropped? " and so on.

Q_Tag	Description	Importance_Score
Q_Quotation	the quotation part in the question	8
Q_Focus	the key word or phrase representing the object or event the question asks about (analyzed from Parser Minipar)	7
Q_NameEntity	the name entity in the question (analyzed from Name Entity tool)	6
Q_Verb	the main verb of the question (analyzed from Parser Minipar)	5
Q_BNP	the noun phrase of the question (analyzed from the BNP Chunking tool)	4

Table 3 Symbol Set of Question

4.2 Pattern Learning and Evaluation

We will explain our approach with the sample example below.

Sample question class: [LCN] What Q_Verb Q_Focus of Q_NameEntity?

Sample question: What is the capital city of New Zealand?

Where Q_Verb = "is", Q_Focus = "the capital city", Q_NameEntity = "New Zealand", and Answer = "Wellington".

The context patterns of each question class are learned by the following algorithm:

1. Constructing Query: "Q_Focus + Q_NameEntity +Answer" is constructed as the query. For example, the query of above sample question is: "the capital city"+"New Zealand"+ "Wellington".

2. Searching: the query is submitted to the search engine Google and the top 100 Web documents are downloaded.

3. Snippet Selection and Filtering: the snippets for pattern learning are extracted from the Web documents. The answer, the nearest ten words left to it, and the nearest ten words right to it are retained.

4. Context Pattern Extraction: replace the question term in each snippet by the corresponding Q_Tag, and the answer term by the tag <A>. The minimum length string containing the Q_Tag and the tag <A> is extracted as the context pattern. For example, consider the string "...the number of languages that are being spoken. Wellington the capital city of New Zealand and ...", context pattern "<A> Q_Focus of Q_NameEntity" is extracted.

5. Computing the Initial Score of Context Pattern: the score is computed as the following formula considering the importance of the Q_Tag and the distance between the different Q_Tag and the answer. (α=1,β=0.6)

$$Initial\_Score = \alpha \bullet \frac{1}{Distance} + \beta \bullet \sum_{j=1}^{n} \frac{importance\_Score(QTag_j)}{importanceAll}$$

$$Distance = \frac{\sqrt{d_1^2 + d_2^2 + \ldots + d_n^2}}{n}$$

$$importanceAll = \sum_{k=1}^{m} importance(Q\_Tag_k)$$

m is the number of Q_Tag the question class contains, n is the number of the Q_Tag the context pattern contains, d_i is the distance between the different Q_Tag and the answer.

The approach to context pattern evaluation is as follows. Query for each context pattern is formed and submitted to the Google, and the top 100 snippets are downloaded for context pattern precision calculation. The query consists of three parts: [Pre_Part]+[Post_Part]+[Q_Focus + Q_NameEntity].

[Pre_Part] stands for the word string left to tag <A> of the context pattern, and that [Post_Part] stands for the word string right to tag <A> of the context pattern. [Q_Tag] is composed of the Q_Focus and Q_NameEntity of the question. The matching score of each pattern is calculated as follows:

$$Match\_Score = \frac{Num_{Correct\_Match}}{Num_{Match}}$$

$Num_{Correct\_Match}$ denotes the number of snippets that tag <A> is matched by the correct answer; Num_{Match} denotes the number of the snippets that tag <A> is matched by any word.

At last the score of the context pattern is computed with the formula : (α=0.3,β=0.7)

$$Pattern\_Score = \alpha \bullet Initial\_Scor + \beta \bullet Match\_Score$$

4.3 Answer Extraction

The context patterns can be used to extract answer to a new unseen question as follows:

1. Determine the question class of the unseen question based on its Q_Pattern and answer type. The corresponding context patterns are also selected.

2. Replace the Q_Tag symbols in the context pattern with the corresponding word string of the question.

3. For each context pattern and each snippet search engine returned, select the words matching tag <A> as the answer.

4. Sort the answers by their context pattern's score and their frequency.

The first answer is returned to the factoid question and the top five answers are returned to the list question and definition question.

5 Conclusion

This year we only take part in the main task of QA, and submit three runs. Our results are not very satisfactory. Our first run, FDUT12QA1, is based on our main architecture; FDUT12QA2 is our new attempt; and FDUT12QA3 is the simple combination of FDUT12QA1 and FDUT12QA2. Their detail evaluation report is illustrated in table 4.

	FDUT12QA1	FDUT12QA2	FDUT12QA3

Final score	0.163	0.122	0.165
Accuracy of factoid questions	0.194	0.179	0.191
Average F of list questions	0.088	0.067	0.086
Average F of definition questions	0.176	0.065	0.192
right questions of factoid questions	80	74	79
Unsupported questions of factoid questions	28	27	27

Table 4 Evaluation report

We find in table 4, that the numbers of unsupported questions of factoid questions are very big compared with their corresponding right answered questions. That's because we can't well integrate the Web into our system.

Acknowledgements

This research was partly supported by NSF of China under contracts of 69935010 and 60103014, as well as the 863 National High-tech Promotion Project of China under contracts of 2001AA114120 and 2002AA142090. We are thankful to Xin Li, Ningyu Chen, Liyuan Han for their help in the implementation.

References

Susan Dumais, Michele Banko, Eric Brill, Jimmy Lin, and Andrew Ng. 2002. Web Question Answering: Is More Always Better? Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002), August 2002, Tampere, Finland.

Cody C. T. Kwok, Oren Etzioni and Daniel S. Weld. May 1-5, 2001. Scaling Question Answering to the Web. Tenth World Wide Web Conference. Hong Kong, China.

Deepak Ravichandran, Eduard Hovy. 2002. Learning Surface Text Patterns for a Question Answering System. Proceedings of the ACL 2002.

Daniel Sleator and Davy Temperley. 1993. Parsing English with a Link Grammar. Third International Workshop on Parsing Technologies.

Soubbotin, M.M. AND Soubbotin. 2001. Patterns of Potential Answer Expressions as Clues to the Right Answer. Proceedings of the TREC-10, Gaithersburg, Maryland, 175-182.

Martin M.Soubbotin, Sergei M.Soubbotin. 2002. Use of Patterns for Detection of Likely Answer String: A Systematic Approach. Proceedings of the TREC-11, Gaithersburg, Maryland, 134-143.

Lide Wu, Xuanjing Huang, Junyu Niu, Yingju Xia, Zhe Feng, Yaqian Zhou.2002. FDU at TREC2002: Filtering, QA, Web and Video Tasks. Proceedings of the TREC-11, Gaithersburg, Maryland.

Robust, Web and Genomic Retrieval with Hummingbird SearchServer$^{\mathrm{TM}}$ at TREC 2003

Stephen Tomlinson
Hummingbird
Ottawa, Ontario, Canada
stephen.tomlinson@hummingbird.com
http://www.hummingbird.com/

February 4, 2004

Abstract

Hummingbird participated in 4 tasks of TREC 2003: the ad hoc task of the Robust Retrieval Track (find at least one relevant document in the first 10 rows from 1.9GB of news and government data), the navigational task of the Web Track (find the home or named page in 1.2 million pages (18GB) from the .GOV domain), the topic distillation task of the Web Track (find key resources for topics in the first 10 rows from home pages of .GOV), and the primary task of the Genomics Track (find all records focusing on the named gene in 1.1GB of MEDLINE data). In the ad hoc task, SearchServer found a relevant document in the first 10 rows for 48 of the 50 new short (Title-only) topics. In the navigational task, SearchServer returned the home or named page in the first 10 rows for more than 75% of the 300 queries. In the distillation task, a SearchServer run found the most key resources in the first 10 rows of the submitted runs from 23 groups.

1 Introduction

Hummingbird SearchServer[1] is an indexing, search and retrieval engine for embedding in Windows and UNIX information applications. SearchServer, originally a product of Fulcrum Technologies, was acquired by Hummingbird in 1999. Founded in 1983 in Ottawa, Canada, Fulcrum produced the first commercial application program interface (API) for writing information retrieval applications, Fulcrum® Ful/Text$^{\mathrm{TM}}$. The SearchServer kernel is embedded in many Hummingbird products, including SearchServer, an application toolkit used for knowledge-intensive applications that require fast access to unstructured information.

SearchServer supports a variation of the Structured Query Language (SQL), SearchSQL$^{\mathrm{TM}}$, which has extensions for text retrieval. SearchServer conforms to subsets of the Open Database Connectivity (ODBC) interface for C programming language applications and the Java Database Connectivity (JDBC) interface for Java applications. Almost 200 document formats are supported, such as Word, WordPerfect, Excel, PowerPoint, PDF and HTML.

SearchServer works in Unicode internally [5] and supports most of the world's major character sets and languages. The major conferences in text retrieval evaluation (TREC [9], CLEF [1] and NTCIR [7]) have provided opportunities to objectively evaluate SearchServer's support for more than a dozen languages.

This paper looks at experimental work with SearchServer for robust retrieval (robustness of ad hoc search across topics), web navigation (find the one page the user wanted, i.e. a known-item search task), web distillation (find key resource pages for broad topics), and genomic retrieval (a domain-specific task). For the submitted runs in August 2003, an experimental post-5.x development build of SearchServer was used.

[1] Fulcrum® is a registered trademark, and SearchServer$^{\mathrm{TM}}$, SearchSQL$^{\mathrm{TM}}$, Intuitive Searching$^{\mathrm{TM}}$ and Ful/Text$^{\mathrm{TM}}$ are trademarks of Hummingbird Ltd. All other copyrights, trademarks and tradenames are the property of their respective owners.

2 Robust Retrieval

The document set of the TREC 2003 Robust Retrieval Track was a subset of the news and government data of TREC Disks 4 and 5. It consisted of 528,155 documents totaling 1,997,002,586 bytes (1.9 GB). The average document size was 3781 bytes. For more information, see the track overview paper.

For this ad hoc task, participants were asked to focus not just on mean average precision but on at least one other measure indicative of "robustness" across results, such as the number of topics for which at least one relevant was retrieved in the first 10 rows.

2.1 Indexing

The custom text reader called cTREC, described in our first TREC paper [10], already supported detailed handling of the TREC Disk collections. For example, it allowed indexing of text following particular tags (such as <HEADLINE> and <TEXT>) and disabled indexing for text surrounded by other tags (such as <PAGE>...</PAGE>) and for the tags themselves. As this year's guidelines did not restrict the fields allowed for indexing, we used the /k option of cTREC to allow indexing of text tagged as keywords (in particular, text tagged by <IN> or <SUBJECT> in the case of the disks used this year). Past experiments suggest that this detailed handling does not affect the results much.

We used the mygov.stp stopword list (99 English stopwords) first used for a web task last year [12]. The option to support inflections from lexical English stemming was enabled. We also experimented with an option to construct term vectors for result list clustering for this task.

2.2 Searching

The submitted humR03d run was a "plain" SearchServer run on the Description field of each topic. It used SearchServer's Intuitive Searching (i.e. the IS_ABOUT predicate of SearchSQL). Here is an example SearchSQL query for topic 314:

```
SELECT RELEVANCE('V2:3') AS REL, DOCNO
FROM ROBUST03
WHERE FT_TEXT IS_ABOUT 'Commercial harvesting of marine vegetation
  such as algae, seaweed and kelp for food and drug purposes.'
ORDER BY REL DESC;
```

SearchServer's relevance value calculation is the same as described last year [12]. Briefly, SearchServer dampens the term frequency and adjusts for document length in a manner similar to Okapi [8] and dampens the inverse document frequency using an approximation of the logarithm. SearchServer's relevance values are always an integer in the range 0 to 1000.

Before the queries were run, various SET statements were issued. "SET MAX_SEARCH_ROWS 1000" ensured the resulting working table would contain at most 1000 rows. Inflections from English stemming were enabled by "SET VECTOR_GENERATOR 'word!ftelp/lang=english/base/noalt | * | word!ftelp/lang=english/inflect' " (for more details on stemming for several European languages, see our CLEF paper [14]). The importance of document length to the relevance value calculation was set with "SET RELEVANCE_DLEN_IMP 750" (scale of 0 to 1000).

We automatically removed "query stop words" such as "find", "relevant" and "document" from the topics before presenting them to SearchServer, i.e. words which are not stop words in general but were commonly used in previous years' TREC and CLEF topics as general instructions (this year's topics were not reviewed). An evaluation in last year's CLEF paper [11] found this step to be of only minor impact in several European languages,including English.

The submitted humR03t run was the same as humR03d except that the Title field of the topic was used instead of the Description. For example, for topic 314, the Where clause was just "WHERE FT_TEXT IS_ABOUT 'Marine Vegetation' ". This run represented a "plain" SearchServer run for Title queries.

The submitted humR03de run used query expansion from blind feedback. The first two rows of the humR03d run were used to find additional query terms. Only terms appearing in at most 5% of the documents (based on the most common inflection of the term) were included. Mathematically, the approach is similar to Rocchio feedback with weights of one-half for the original query and one-quarter for each of the 2 expansion rows. This was the same blind feedback approach as used for Arabic experiments at TREC last year [12] except that we just used 2 rows for expansion this time instead of 5 (diagnostics on past CLEF and TREC ad hoc tasks suggested fewer rows may be more effective, perhaps because most tasks have fewer relevant documents to feed back in per topic than the Arabic task did). Blind feedback from the top retrieved documents is often effective at increasing recall later in the result list without sacrificing early precision, important components of the average precision measure. However, the large number of additional query terms negatively impacts performance. In practice, users could manually add terms to the query rather than work blindly. For this run, the measure in mind was average precision.

The submitted humR03dc run re-ordered the top 100 rows of humR03d so that the first 10 rows were from different clusters to see if that increased the chance of a relevant document in the top 10 rows. The steps were as follows:

First, the parameter settings were the same as for humR03d except that "SET MAX_SEARCH_ROWS 100" was used instead of 1000. Hence a relevant document had to appear in the top 100 for re-ordering to have a chance of moving it into the top 10.

Next, a result list clustering query was run on the 100 row working table, producing a set of up to 10 clusters. Each of the 100 rows appeared in exactly one cluster; the number of rows in each cluster could differ. Within each cluster, the rows were ordered by the original relevance value. The clusters themselves were ordered by the average relevance value of the rows of the cluster. The clustering was based on the term vectors of the documents built at index-time. Other than its impact on which 100 documents were clustered, the query had no impact on the clustering.

Finally, a round-robin of the clusters was followed, with the row of highest remaining relevance score of each cluster placed into the final result. (In the submitted file, the first 3 digits after the decimal point are the relevance value of the document, and the 4th digit is the cluster number from 0-9.)

The top 100 rows of humR03d and humR03dc should be the same except for the order. The first row for a topic in humR03d would appear somewhere in the top 10 for the topic in humR03dc. The other rows in the top 10 might differ.

The final result of humR03dc had at most 100 rows per topic. We did not bother to pad to 1000 rows. Hence for this run there was a bias against measures which consider documents retrieved past 100 rows, such as recall and average precision. For this run, the measure in mind was the 'relevant in the top 10 rows' measure.

The submitted humR03tc run was the same as humR03dc except that it was based on humR03t instead of humR03d.

2.3 Results

For this task, there were 50 "old" topics and 50 "new" topics.

The 50 old topics were selected (by the task organizers) from past years' ad hoc TREC topics 301-450 to produce a set of "tough" topics, i.e. topics on which few systems produced a high precision score when they were originally used (though there may have been a bias against topics on which all systems produced a low score; the track overview paper may elaborate more). As they were already in the public domain, the guidelines allowed groups to continue to study these topics for this year's submissions, which might also help lead to techniques for improving results on tough topics. One must be cautious however at reading too much into results on these topics (even "statistically significant" results) because of the possibility that the techniques are tuned to this data.

For the 50 new topics, (automatic) systems were not allowed to be altered based on examination of the topics, so in that sense the results may be more meaningful. But there was no reason to expect these topics to be as "tough" as the specially-selected older set (it is very hard to predict which topics will be tough in advance) so for the purpose of this track (robustness across topics) there may not be enough challenging topics to distinguish the techniques.

We separately list the results for each of these sets of topics (we do not bother to look at the combined scores). Also, for the 50 new topics, the relevance assessors distinguished "highly relevant" documents from

Table 1: Precision of Submitted Runs

Run	AvgP	P@5	P@10	P@20	Rec0	Rec30	P@R	%Rel10
(humR03te-old)	0.131	34.4%	33.0%	27.7%	0.564	0.189	19.4%	40/50
humR03t-old	0.109	30.4%	28.8%	23.9%	0.555	0.151	17.1%	42/50
humR03tc-old	0.057	20.8%	17.8%	18.1%	0.476	0.070	12.6%	38/50
humR03de-old	0.148	38.8%	35.2%	28.1%	0.656	0.187	19.6%	38/50
humR03d-old	0.127	37.2%	29.6%	24.7%	0.635	0.167	17.4%	39/50
humR03dc-old	0.071	26.0%	20.6%	17.3%	0.582	0.072	13.4%	41/50
(humR03te-new)	0.332	51.2%	44.8%	35.8%	0.684	0.482	33.6%	46/50
humR03t-new	0.280	46.8%	41.0%	32.2%	0.672	0.401	30.6%	48/50
humR03tc-new	0.147	28.4%	20.4%	19.7%	0.630	0.194	18.9%	42/50
humR03de-new	0.377	57.2%	48.4%	39.9%	0.767	0.543	37.7%	43/50
humR03d-new	0.346	57.6%	47.6%	38.6%	0.779	0.500	34.5%	46/50
humR03dc-new	0.178	33.6%	23.4%	21.1%	0.687	0.227	20.8%	44/50
(humR03te-newH)	0.253	28.8%	21.2%	16.2%	0.430	0.342	25.4%	31/43
humR03t-newH	0.225	24.6%	19.8%	15.1%	0.402	0.307	24.1%	31/43
humR03tc-newH	0.117	9.8%	8.1%	8.0%	0.333	0.140	10.0%	22/43
humR03de-newH	0.309	30.2%	24.2%	17.8%	0.551	0.424	28.2%	32/43
humR03d-newH	0.305	32.6%	23.5%	17.2%	0.579	0.413	29.8%	31/43
humR03dc-newH	0.176	18.6%	11.6%	10.2%	0.491	0.225	17.4%	27/43

Table 2: Impact of Clustering on Percentage of Topics With a Relevant in Top 10

Experiment	AvgDiff	95% Confidence	vs.	2 Largest Diffs (Topic)
D-old-Rel10	0.040	(−0.101, 0.181)	7-5-38	1.000 (330), 1.000 (401)
D-new-Rel10	−0.040	(−0.121, 0.041)	1-3-46	1.000 (605), −1.000 (608)
T-old-Rel10	−0.080	(−0.161,−0.019)	0-4-46	−1.000 (394), −1.000 (336)
D-newH-Rel10	−0.093	(−0.210, 0.001)	1-5-37	1.000 (643), −1.000 (616)
T-new-Rel10	−0.120	(−0.221,−0.039)	0-6-44	−1.000 (612), −1.000 (642)
T-newH-Rel10	−0.209	(−0.373,−0.069)	2-11-30	1.000 (631), 1.000 (620)

just "relevant" documents. 43 of the 50 topics had at least one "highly relevant" document. We list the scores averaged over those 43 topics when just considering highly relevants as relevant (tagged with "newH").

Table 1 gives an overview of several precision scores for each submitted run (also, in brackets, is an unsubmitted run (because of the 5-run submission limit) produced at the same time, humR03te, an analog of humR03de for Titles). Listed for each run are its mean average precision (AvgP), the mean precision after

Table 3: Impact of Blind Feedback on Average Precision

Experiment	AvgDiff	95% Confidence	vs.	2 Largest Diffs (Topic)
(T-new-AvgP)	0.052	(0.031, 0.075)	40-10-0	0.345 (614), 0.188 (607)
D-new-AvgP	0.031	(0.004, 0.057)	34-16-0	−0.221 (616), 0.205 (633)
(T-newH-AvgP)	0.027	(0.012, 0.043)	30-10-3	0.180 (648), 0.147 (626)
(T-old-AvgP)	0.022	(0.010, 0.035)	32-18-0	0.181 (350), 0.174 (372)
D-old-AvgP	0.021	(0.008, 0.035)	32-17-1	0.167 (350), 0.146 (320)
D-newH-AvgP	0.004	(−0.016, 0.022)	25-17-1	−0.213 (644), −0.128 (614)

Table 4: Impact of Blind Feedback on Percentage of Topics With a Relevant in Top 10

Experiment	AvgDiff	95% Confidence	vs.	2 Largest Diffs (Topic)
D-newH-Rel10	0.023	(−0.070, 0.117)	3-2-38	1.000 (601), 1.000 (633)
(T-newH-Rel10)	0.000	(−0.094, 0.094)	2-2-39	1.000 (636), 1.000 (628)
D-old-Rel10	−0.020	(−0.081, 0.041)	1-2-47	1.000 (356), −1.000 (426)
(T-old-Rel10)	−0.040	(−0.141, 0.061)	2-4-44	1.000 (356), 1.000 (439)
(T-new-Rel10)	−0.040	(−0.121, 0.041)	1-3-46	1.000 (627), −1.000 (632)
D-new-Rel10	−0.060	(−0.141, 0.001)	0-3-47	−1.000 (632), −1.000 (610)

5, 10 and 20 documents retrieved (P@5, P@10 and P@20 respectively), the mean interpolated precision at 0% and 30% recall (Rec0 and Rec30 respectively), the mean precision after R documents retrieved (P@R) where R is the number of relevant documents for the topic, and the ratio of the number of topics with at least one relevant retrieved in the top 10 vs. the total number of topics (%Rel10). (Definitions of the measures are in last year's paper [12], and they likely are also in an appendix of the conference proceedings.) It appears that for every measure listed, the score on the "new" topics is higher than the corresponding score for the "old" topics, i.e. as expected, the "old" topics were more challenging (on average).

For tables focusing on the impact of one particular difference in approach, the columns are as follows:

- "Experiment" indicates whether the Title or Description topics were used ("T" or "D" respectively) and whether the score is based on the old topics ("old"), the new topics when treating all relevants the same ("new"), or the new topics just counting highly relevants as relevant ("newH").
- "AvgDiff" is the average (mean) difference in the score.
- "95% Confidence" is an approximate 95% confidence interval for the average difference calculated using Efron's bootstrap percentile method[2] [3] (using 100,000 iterations). If zero is not in the interval, the result is "statistically significant" (at the 5% level), i.e. the feature is unlikely to be of neutral impact, though if the average difference is small (e.g. <0.020) it may still be too minor to be considered "significant" in the magnitude sense.
- "vs." is the number of topics on which the score was higher, lower and tied (respectively) with the feature enabled. These numbers should always add to the number of topics (50 or 43).
- "2 Largest Diffs (Topic)" lists the two largest differences in the score (based on the absolute value) with each followed by the corresponding topic number in brackets (the old topic numbers range from 301 to 450 and the new topic numbers from 601 to 650).

Table 2 shows the impact of the clustering-based technique on the percentage of topics with a relevant in the first 10 rows. For Description queries, this is based on subtracting the scores of humR03d from humR03dc, and for the Title queries, subtracting the scores of humR03t from humR03tc. As you can see, there was a net gain of 2 topics with a relevant in the top 10 on the old Description queries (7 gained but 5 lost), though this was not statistically significant, and the loss of 4 on the old Title queries was statistically significant. On the new queries, the finding was similar; the differences were not significant on the Description queries but were on the Title queries, both when counting all relevants or just highly relevants.

Table 3 shows the impact of the blind feedback technique on the average precision score (based on subtracting humR03d from humR03de, and humR03t from (unsubmitted run) humR03te). The increase was statistically significant for 5 of the 6 cases, the exception being for highly relevants on the new Description topics.

Table 4 shows the impact of the same blind feedback technique on the perentage of topics with a relevant in the first 10 rows (based on the same runs as Table 3). None of the impacts were statistically significant.

[2] See [11] for some comparisons of confidence intervals from the bootstrap percentile, Wilcoxon signed rank and standard error methods for both average precision and Precision@10.

Table 5: Examples of URL type and depth values

URL	Type	Depth	Depth Term
http://nasa.gov/	ROOT	1	URLDEPTHA
http://www.nasa.gov/	ROOT	1	URLDEPTHA
http://jpl.nasa.gov/	ROOT	2	URLDEPTHAB
http://fred.jpl.nasa.gov/	ROOT	3	URLDEPTHABC
http://nasa.gov/jpl/	SUBROOT	2	URLDEPTHAB
http://nasa.gov/jpl/fred/	PATH	3	URLDEPTHABC
http://nasa.gov/index.html	ROOT	1	URLDEPTHA
http://nasa.gov/fred.html	FILE	2	URLDEPTHAB

Table 6: Number of Pages of each URL Type and Depth

Type	#Pages	Depth	#Pages	Depth	#Pages
ROOT	6,906	1	635	6	269,949
SUBROOT	18,179	2	16,792	7	136,513
PATH	55,332	3	128,898	8	44,960
FILE	1,167,336	4	282,086	9	15,289
		5	344,694	10+	7,937

For the plain SearchServer runs, more topics found a relevant in the first 10 rows using the Titles than the Descriptions (42 to 39 for the old topics, 48 to 46 for the new topics (though tied at 31 when restricting to highly relevants) as per Table 1). These results might suggest that shorter queries are more "robust" (perhaps extra details throw off a system more often than missing details, even though the longer queries score higher on the other listed measures which reward recall more). However, these changes in the number of topics with a relevant in the first 10 rows did not pass a significance test.

3 Web Retrieval

Both tasks of the TREC 2003 Web Track used the same .GOV collection as last year. It consists of pages downloaded from the .gov domain of the World Wide Web in early 2002. Uncompressed, it is 19,455,030,550 bytes (18.1 GB) and a total of 1,247,753 documents. The average document size is 15,592 bytes. For more information on the .GOV collection, see [4].

3.1 Indexing

The indexing approach was the same as described in last year's paper [12] (except that a newer version of the software was used which may have contained an updated English lexicon for stemming).

Briefly: in addition to full-text indexing, the custom text reader cTREC populated particular columns such as TITLE (if any), URL, URL_TYPE and URL_DEPTH. The URL_TYPE was set to ROOT, SUBROOT, PATH or FILE, based on the convention which worked well in TREC 2001 for the Twente/TNO group [15] on the entry page finding task (also known as the home page finding task). The URL_DEPTH was set to a term indicating the depth of the page in the site. Table 5 contains URL types and depths for example URLs, and Table 6 shows the number of .GOV pages of each URL type and depth. The exact rules we used are given in last year's paper [12].

3.2 Searching

Even though the 2 web tasks are potentially quite different (the navigational task is a known-item task (one right answer), while the topic distillation task is focused on distilling broad topics to key resource pages), we used the same techniques for both tasks for each of the 5 submitted runs (and most of the techniques used were the same as last year). This allows us to compare the impact of the techniques on different tasks.

The submitted humNP03l and humTD03l runs used the same approach as the diagnostic base run described in last year's paper [12] which was just to search the content (FT_TEXT column) using the IS_ABOUT predicate (i.e. the same approach as used for the "plain" runs of the Robust task).

The submitted humNP03pl and humTD03pl runs used the same approach as last year's hum02pd run. Below is an example SearchSQL query. The queries differed from humNP03l and humTD03l in that that properties and phrases in properties were given a little extra weight. (The ALL_PROPS column contained the title, URL, first heading and some meta tags, but not most of the document content; see last year's paper for the details.) Note that the FT_TEXT column also indexed all of the properties except for the URL.

```
SELECT RELEVANCE('V2:3') AS REL, DOCNO
FROM GOV
WHERE
 (ALL_PROPS CONTAINS 'visiting pandas national zoo' WEIGHT 1) OR
 (ALL_PROPS IS_ABOUT 'visiting pandas national zoo' WEIGHT 1) OR
 (FT_TEXT IS_ABOUT 'visiting pandas national zoo' WEIGHT 10)
ORDER BY REL DESC;
```

The CONTAINS predicate does phrase searching, so the listed terms would have to occur adjacently in the specified order (except stop words). "SET PHRASE_DISTANCE 4" was previously specified so that there could be up to 4 characters between adjacent terms (plus additional whitespace). By default, the CONTAINS predicate does exact searching (i.e. no inflections from stemming), though some Unicode-based normalizations (e.g. decompositions and conversion to upper-case) are still done. The motivation for including the query as a phrase was that it seemed the query might often be in the title or other property information of the document (e.g. a query in mind was "Washington State Legislature" (which was not one of the 150 official queries last year)). The phrase searching was just given one-tenth the weight of content searching for relevance ranking purposes. Experiments on the TREC 2001 entry page finding task suggested a small weight was helpful (on average) but a strong weight had a negative impact.

The IS_ABOUT predicate uses SearchServer's Intuitive Searching. It by default matches inflections froms English stemming and just requires one of the terms to have a match. It was used with WEIGHT 1 on the ALL_PROPS column to increase the ranking of documents with query terms in the title or other property information. It was used with WEIGHT 10 on the FT_TEXT column (which represents the external document). Again, these weights were chosen based on what worked well on the TREC 2001 entry page finding task.

The submitted humNP03upl and humTD03upl runs used the same approach as last year's hum02upd run. The 'u' indicates a higher weight was given to URLs of particular type and depth. See last year's paper for an example of the SearchSQL syntax [12].

The submitted humNP03uhpl and humTDuhpl runs used the same approach as last year's hum02uhp run except for using a document length importance of 500 instead of 250 (500 was used for all submitted web runs this year). The 'h' indicates an even higher weight was given to URL_TYPE (the 3 terms of WEIGHT 10 were given WEIGHT 25). On the TREC 2001 entry page finding task, the stronger URL_TYPE weights gave similar MRR scores to the lower ones.

The submitted humNP03up and humTD03up runs were the same as humNP03upl and humTD03upl (respectively) except that linguistic expansion from English stemming was disabled (i.e. matching of inflections was disabled) by "SET VECTOR_GENERATOR '' ".

For the navigational (humNP03*) runs, the statement "SET MAX_SEARCH_ROWS 50" was previously executed so that the working table would contain at most 50 rows, whereas for the topic distillation (humTD03*) runs, the statement "SET MAX_SEARCH_ROWS 1000" was previously executed.

Table 7: Scores of Submitted Navigational Runs

		300		HP	150		NP	150	
Run	MRR	%Top10	%Fail	MRR	%Top10	%Fail	MRR	%Top10	%Fail
humNP03up	0.545	77.3%	12.3%	0.584	82.0%	8.7%	0.506	72.7%	16.0%
humNP03upl	0.535	77.7%	11.7%	0.591	82.7%	8.0%	0.480	72.7%	15.3%
humNP03pl	0.465	68.3%	17.3%	0.361	56.7%	27.3%	0.568	80.0%	7.3%
humNP03uhpl	0.386	56.7%	26.0%	0.500	70.7%	16.7%	0.271	42.7%	35.3%
humNP03l	0.321	54.3%	25.7%	0.223	41.3%	40.7%	0.420	67.3%	10.7%

Table 8: Impact of Submitted Navigational Techniques on Reciprocal Rank

Experiment	AvgDiff	95% Confidence	vs.	2 Largest Diffs (Topic)
HP u (upl - pl)	0.229	(0.168, 0.291)	87-17-46	1.000 (321), 1.000 (395)
HP p (pl - l)	0.139	(0.093, 0.187)	72-14-64	1.000 (422), 1.000 (200)
HP l (upl - up)	0.007	(−0.011, 0.026)	14-13-123	0.667 (271), 0.500 (349)
HP h (uhpl - upl)	−0.091	(−0.134,−0.050)	14-57-79	−1.000 (355), −0.977 (300)
NP u (upl - pl)	−0.088	(−0.134,−0.042)	13-66-71	−1.000 (359), 0.950 (154)
NP p (pl - l)	0.147	(0.092, 0.204)	74-17-59	−1.000 (416), 0.975 (151)
NP l (upl - up)	−0.026	(−0.055, 0.002)	14-32-104	0.889 (215), −0.875 (306)
NP h (uhpl - upl)	−0.209	(−0.257,−0.162)	1-92-57	−1.000 (249), −1.000 (154)

For the web queries, no query terms were discarded (e.g. there was no expectation that discarding the words "find", "relevant" and "document" would be beneficial, unlike for the Robust task). Of course, the index omitted a few stop words (e.g. "the", "by") as previously mentioned.

SearchServer's relevance value calculation is the same as described for the Robust task. Additionally, when multiple predicates are combined, as was done for some of the web approaches, SearchServer currently does not normalize by query length. For example, the URL_TYPE clauses would have a lot less relative impact if the topic query contained 5 words instead of 1.

3.3 Results

The evaluation measures are likely explained in an appendix of this volume. Briefly, for the navigational task, "Reciprocal Rank" for a topic is one divided by the rank in which the home or named page was found (using the smallest rank if there were duplicates of the page), or zero if the page was not found. "Mean Reciprocal Rank" (MRR) is the average of the reciprocal ranks over all the topics. "%Top10" is the percentage of topics for which the home or named page was found in the first 10 rows. "%Fail" is the percentage of topics for which the home or named page was not found in the first 50 rows. The topic distillation measures are the same as described previously in the Robust section.

Table 7 shows the scores of the submitted navigational runs in descending order by mean reciprocal rank over all 300 queries. The HP columns show the scores just for the 150 home page queries. The NP columns show the scores just for the 150 named page queries. (The topics did not state whether they were of HP or NP type; that information was provided by the organizers after the submission date for use in analysis.)

Table 8 shows the impact when isolating each technique distinguishing the submitted navigational runs:

- The 'u' factor (extra weight for URL type and depth) increased MRR dramatically on the home pages (23 points) but (like last year) was detrimental on the named pages (9 points). More diagnostics are below.

- The 'p' factor (extra weight for HTML properties and phrases in properties) increased MRR 14 points on both home and named pages. More diagnostics are below.

Table 9: Diagnostics of Extra Weight on Document Structure (Navigational Task, Reciprocal Rank)

Experiment	AvgDiff	95% Confidence	vs.	2 Largest Diffs (Topic)
HP v (vl-l)	0.079	(0.045, 0.115)	60-19-71	0.933 (425), 0.917 (333)
HP q (ql-l)	0.056	(0.027, 0.088)	44-15-91	0.976 (385), 0.909 (307)
HP qv (vql-l)	0.115	(0.073, 0.159)	62-19-69	1.000 (334), 1.000 (389)
HP v (vql-ql)	0.058	(0.025, 0.095)	44-27-79	0.978 (389), 0.917 (425)
HP q (vql-vl)	0.036	(0.009, 0.066)	27-18-105	1.000 (389), 0.800 (322)
HP other (pl-vql)	0.024	(−0.015, 0.064)	37-29-84	1.000 (266), −0.857 (334)
NP v (vl-l)	0.120	(0.073, 0.169)	74-16-60	0.975 (151), 0.969 (286)
NP q (ql-l)	0.050	(0.014, 0.089)	41-20-89	−1.000 (416), 0.975 (151)
NP qv (vql-l)	0.128	(0.078, 0.180)	71-18-61	−1.000 (416), 0.975 (151)
NP v (vql-ql)	0.078	(0.041, 0.117)	55-17-78	0.963 (178), −0.909 (304)
NP q (vql-vl)	0.008	(−0.017, 0.032)	21-18-111	−0.857 (248), −0.750 (259)
NP other (pl-vql)	0.019	(−0.013, 0.051)	30-27-93	−0.938 (196), −0.800 (449)

Table 10: Diagnostics of Extra Weight on URL Structure (Navigational Task, Reciprocal Rank)

Experiment	AvgDiff	95% Confidence	vs.	2 Largest Diffs (Topic)
HP d5 (d5pl-pl)	0.158	(0.110, 0.207)	73-7-70	1.000 (384), 1.000 (395)
HP d10 (d10pl-pl)	0.205	(0.151, 0.261)	85-10-55	1.000 (244), 1.000 (384)
HP d15 (d15pl-pl)	0.213	(0.152, 0.275)	84-16-50	1.000 (184), 1.000 (244)
HP d20 (d20pl-pl)	0.173	(0.109, 0.238)	79-22-49	1.000 (392), 1.000 (395)
HP r5 (r5pl-pl)	0.168	(0.121, 0.217)	76-8-66	0.941 (201), 0.941 (328)
HP r10 (r10pl-pl)	0.213	(0.157, 0.269)	81-13-56	1.000 (395), 1.000 (321)
HP r5d5 (r5d5pl-pl)	0.231	(0.176, 0.288)	87-11-52	1.000 (184), 1.000 (321)
NP d5 (d5pl-pl)	−0.001	(−0.024, 0.024)	23-29-98	0.950 (154), 0.750 (264)
NP d10 (d10pl-pl)	−0.027	(−0.062, 0.008)	20-45-85	−0.952 (359), 0.950 (154)
NP d15 (d15pl-pl)	−0.084	(−0.127,−0.043)	16-61-73	−1.000 (359), −0.875 (189)
NP d20 (d20pl-pl)	−0.166	(−0.215,−0.119)	10-79-61	−1.000 (359), −1.000 (249)
NP r5 (r5pl-pl)	−0.013	(−0.039, 0.013)	11-34-105	0.833 (383), 0.750 (264)
NP r10 (r10pl-pl)	−0.069	(−0.109,−0.030)	10-61-79	−0.941 (359), −0.900 (216)
NP r5d5 (r5d5pl-pl)	−0.040	(−0.075,−0.005)	19-48-83	0.950 (154), −0.929 (359)

- The 'l' factor (linguistic expansion (inflections) from lexical English stemming) made little difference (on average).
- The 'h' factor (even more extra weight for URL type) was detrimental even on the home page queries, even though it had a neutral impact on the TREC 2001 entry page task.

Table 9 isolates the components of the 'p' factor. 'v' denotes that the run included TITLE IS_ABOUT (i.e. vector) matching with weight 1, and 'q' denotes that the run included TITLE CONTAINS (i.e. phrase) matching with weight 1. Adding the 'v' factor (to a full content search with weight 10) increased MRR significantly for both home pages and named pages (8 and 12 points respectively as per the "v (vl-l)" rows). The 'q' factor had significant, though smaller, increases (6 and 5 points respectively as per the "q (ql-l)" rows). If one of these was already done, adding the other still led to a significant increase except in the case of adding phrase matching to a vector match for named pages (as per the "NP q (vql-vl)" row). Using the ALL_PROPS column instead of the TITLE column did not lead to a further significant increase as per "other (pl-vql)" rows. So for the 'p' factor, like last year, most of the benefit appears to have come from the TITLE weighting, but unlike last year, both vector and phrase matching helped significantly, not just vector

Table 11: Scores of Submitted Topic Distillation Runs

Run	AvgP	P@5	P@10	P@20	Rec0	Rec30	P@R	Topics
humTD03upl	0.139	14.4%	12.8%	9.2%	0.382	0.166	14.9%	50
humTD03up	0.120	14.8%	12.4%	8.9%	0.362	0.147	13.3%	50
humTD03uhpl	0.098	13.2%	10.2%	6.9%	0.357	0.105	10.7%	50
humTD03pl	0.100	6.8%	5.6%	5.2%	0.247	0.118	9.0%	50
humTD03l	0.051	4.8%	4.4%	3.1%	0.152	0.077	3.6%	50

Table 12: Impact of Topic Distillation Techniques on Precision@10

Experiment	AvgDiff	95% Confidence	vs.	2 Largest Diffs (Topic)
u (upl - pl)	0.072	(0.035, 0.113)	25-5-20	0.600 (7), 0.400 (32)
p (pl - l)	0.012	(−0.009, 0.035)	8-6-36	0.300 (48), 0.200 (15)
l (upl - up)	0.004	(−0.019, 0.027)	7-4-39	0.300 (43), −0.300 (31)
h (uhpl - upl)	−0.026	(−0.049,−0.003)	5-13-32	−0.300 (7), −0.200 (9)

matching (perhaps the topics this year happened to be part of the title of the desired page more often than last year).

Table 10 isolates the components of the 'u' factor. 'r' denotes the weight assigned to the URL_TYPE values (ROOT, SUBROOT, PATH) and 'd' denotes the weight assigned to the URL_DEPTH values ('u' was 'r10d5' and is in Table 8). A small weight on either the url depth or type increased the home page score substantially without a significant drop in the named page score (as per the 'd5' and 'r5' rows). So it may be reasonable to include a small weight on url structure in a general web page search system, regardless of the expected frequency ratio of home page and named page queries. Higher weights may be reasonable if home page queries are expected to be a lot more common.

Table 11 shows the scores of the submitted topic distillation runs in descending order by Precision@10. The humTD03upl run had the highest Precision@10 score of any submitted run from the 23 groups, even though its score means it found on average just more than 1 key resource page in the first 10 rows (the judgements contained 8 key resource pages per topic on average). The topics were broad (e.g. "science" was an example in the task guidelines) and the top retrieved rows may have been filled with many more pages that were "relevant" to the topic even though they were not judged "key resources" by the assessors.

Tables 12, 13 and 14 show the impact of the submitted topic distillation techniques on Precision@10, average precision and Precision@R respectively:

- The 'u' factor (extra weight for URL type and depth) increased Precision@10 by 7 points and produced a statistically significant increase for all 3 examined measures. This is not surprising because the key resources were required to be home pages this year.
- The 'p' factor (extra weight for HTML properties and phrases in properties) did not have a significant impact on Precision@10, but Tables 13 and 14 show it led to a significant increase in the average

Table 13: Impact of Topic Distillation Techniques on Average Precision

Experiment	AvgDiff	95% Confidence	vs.	2 Largest Diffs (Topic)
p (pl - l)	0.049	(0.002, 0.111)	39-9-2	0.956 (24), 0.944 (17)
u (upl - pl)	0.038	(0.009, 0.069)	35-12-3	0.343 (49), 0.337 (18)
l (upl - up)	0.019	(−0.011, 0.066)	16-22-12	1.000 (17), −0.152 (15)
h (uhpl - upl)	−0.041	(−0.079,−0.012)	13-35-2	−0.750 (17), −0.187 (7)

Table 14: Impact of Topic Distillation Techniques on R-Precision

Experiment	AvgDiff	95% Confidence	vs.	2 Largest Diffs (Topic)
u (upl - pl)	0.059	(0.021, 0.100)	20-3-27	0.667 (49), 0.462 (7)
p (pl - l)	0.054	(0.004, 0.119)	11-3-36	1.000 (24), 1.000 (17)
l (upl - up)	0.016	(−0.018, 0.065)	6-3-41	1.000 (17), −0.250 (15)
h (uhpl - upl)	−0.042	(−0.095,−0.001)	7-15-28	−1.000 (17), −0.333 (49)

precision and Precision@R measures.

- The 'l' factor (linguistic expansion (inflections) from lexical English stemming) made little difference for most topics except for some measures for topic 17 ("Polygraphs").
- The 'h' factor (even more extra weight for URL type) had a significant negative impact on the examined measures.

Overall, the impacts on the distillation scores were much more like the impacts on the home page finding scores than the named page finding scores.

4 Genomic Retrieval

For the primary task of the Genomics Track (find all records focusing on the named gene), the MEDLINE data consisted of 525,938 documents (records), all in one file ("trec-medline") of 1,158,771,473 bytes (1.1 GB) uncompressed. The average record length was 2203 bytes. More information should be in the track overview paper.

4.1 Indexing

The cTREC text reader (described in the Robust section) was enhanced to include a /M option for identifying the MEDLINE records (documents) during table expansion from the "trec-medline" file. For the individual records, the /p option of cTREC was used, i.e. we just passed through all of the text for indexing, including identifiers such as "UI", "PMID", "MH" etc. (Maybe next year we will enhance the text reader to populate columns from particular fields, such as the Title, allowing experiments with the record structure like we did for the web data.)

A different stopfile, mynum.stp, was used for this task. It contained just one instruction, AL = "0-9", which means to treat the digits 0 to 9 as alphabet characters. For example, this would cause the symbol "CDKN1A" to be indexed as 1 term instead of 3. Experiments on the training queries found the scores were a little higher with this indexing change. The mynum.stp stopfile did not contain any stop words as the training queries did not seem to use much natural language.

Punctuation characters, including hyphens and parentheses, were still treated as term separators.

4.2 Searching

The submitted runs used IS_ABOUT queries based on combining just 5 of the 8 query fields: the 2 name fields (OFFICIAL_GENE_NAME, PREFERRED_GENE_NAME) and the 3 symbol fields (OFFICIAL_SYMBOL, ALIAS_SYMBOL, PREFERRED_SYMBOL). The other 3 fields were omitted (PREFERRED_PRODUCT, ALIAS_PROT, PRODUCT) because they were found to be harmful on the training topics. Also, the species information was ignored for the submitted runs.

The submitted humG03ns run gave equal weight to the five fields. An example SearchSQL query is below (from run humG03ns test topic 1). The parentheses between query fields were just added for readability and did not affect the IS_ABOUT search:

Table 15: Precision of Genomics Runs

Run	AvgP	P@5	P@10	P@20	Rec0	Rec30	P@R	Topics
humG03ns	0.175	16.4%	14.8%	11.7%	0.394	0.239	15.3%	50
humG03ns5	0.185	18.0%	15.8%	12.3%	0.399	0.257	16.7%	50
base (diag.)	0.312	27.2%	23.6%	18.0%	0.599	0.420	28.9%	50
+5x phrases	0.356	33.2%	24.8%	20.0%	0.613	0.463	31.6%	50

```
SELECT RELEVANCE('V2:3') AS REL, DOCNO
FROM med03n
WHERE FT_TEXT IS_ABOUT 'activating transcription factor 2 ()
  ATF2 () HB16 () CREB2 () TREB7 () CRE-BP1'
ORDER BY REL DESC
```

The submitted humG03ns5 run gave 5 times the weight to the three symbol fields (by repeating them 5 times in the query, rather than using the WEIGHT clause), which was modestly helpful on the training topics (though not significantly so).

Inflections from stemming were disabled for both runs. The document length importance was set to 0 for both runs. More details of the relevance ranking are in the Robust and Web sections.

4.3 Results

Table 15 shows various scores of the submitted runs, humG03ns and humG03ns5 (the column headings are explained in the Robust section).

At the conference, some groups found that filtering by species (organism) was helpful, presumably because of the artificial way the relevance assessments were created for this first Genomics task. The NLM paper [6] described how to convert the species name in the topic statement to the MH field in the MEDLINE records (map 'Homo sapiens' to 'Human', map 'Mus musculus' to 'Mice', map 'Rattus norvegicus' to 'Rats', map 'Drosophila melanogaster' to 'Drosophila'). The NRC reported that just 10 of the official right answers were discarded if restricting to fields of the same organism [2].

The diagnostic "base run" is the same as humG03ns except that it adds a phrase-match restriction to just include documents of the specified species, e.g. "AND (FT_TEXT CONTAINS 'MH - Human' WEIGHT 0)" was added to the query if the species was 'Homo sapiens'. It was assigned "WEIGHT 0" so that it would not affect the relevance calculation. Table 15 shows that the base run scored a 0.312 mean average precision, an increase of more than 13 points over humG03ns.

The "+5x phrases" diagnostic run of Table 15 additionally boosted the scores of records which contained any of the query fields as complete phrases, by use of the CONTAINS predicate. In the CONTAINS predicate, hyphenated terms match not just terms separated with different punctuation or white space, but also concatenations of the terms (e.g. a CONTAINS search for 'CRE-BP1' would additionally match not just 'CRE(BP1)', 'CRE BP1', etc., but also 'CREBP1'). The WHERE clause for topic 1 was

```
WHERE ( FT_TEXT IS_ABOUT 'activating transcription factor 2 ()
               ATF2 () HB16 () CREB2 () TREB7 () CRE-BP1'
   OR (FT_TEXT CONTAINS 'activating transcription factor 2' WEIGHT 5)
   OR (FT_TEXT CONTAINS 'ATF2' WEIGHT 5)
   OR (FT_TEXT CONTAINS 'HB16' WEIGHT 5)
   OR (FT_TEXT CONTAINS 'CREB2' WEIGHT 5)
   OR (FT_TEXT CONTAINS 'TREB7' WEIGHT 5)
   OR (FT_TEXT CONTAINS 'CRE-BP1' WEIGHT 5) )
  AND (FT_TEXT CONTAINS 'MH - Human' WEIGHT 0)
```

Table 16 compares a number of diagnostic runs to the base run (always subtracting the base run's scores in average precision from the listed run). For example, the first row shows that the "+5x phrases" run

Table 16: Impact of Genomics Techniques on Average Precision

Experiment	AvgDiff	95% Confidence	vs.	2 Largest Diffs (Topic)
+5x phrases	0.044	(0.015, 0.076)	28-18-4	0.441 (24), 0.266 (23)
+2x phrases	0.043	(0.016, 0.073)	29-17-4	0.441 (24), 0.316 (23)
+1x phrases	0.037	(0.011, 0.066)	32-14-4	0.441 (24), 0.314 (23)
2x sym	0.027	(0.007, 0.052)	28-19-3	0.441 (24), 0.205 (4)
idf squared (V2:4)	0.020	(−0.006, 0.048)	24-23-3	0.441 (24), −0.231 (18)
5x sym	0.019	(−0.010, 0.051)	22-23-5	0.441 (24), 0.280 (15)
phrases only	0.002	(−0.058, 0.058)	28-19-3	−0.733 (7), −0.631 (20)
DLEN 500	−0.006	(−0.014, 0.001)	17-29-4	−0.119 (16), −0.074 (18)
stemming on	−0.012	(−0.036, 0.005)	11-27-12	−0.463 (27), −0.167 (16)
omit names	−0.030	(−0.079, 0.016)	16-32-2	−0.733 (7), 0.441 (24)
number parsing	−0.038	(−0.071,−0.004)	16-30-4	0.334 (11), −0.331 (9)
all fields	−0.055	(−0.083,−0.031)	9-35-6	−0.382 (31), −0.312 (36)
vector species	−0.069	(−0.102,−0.034)	7-41-2	−0.333 (27), −0.329 (28)
omit symbols	−0.113	(−0.152,−0.075)	8-40-2	−0.489 (31), −0.429 (19)
terms count (2:2)	−0.136	(−0.182,−0.094)	5-44-1	−0.717 (27), −0.507 (29)
omit species	−0.137	(−0.177,−0.099)	2-46-2	−0.489 (20), −0.489 (27)
hits count (2:1)	−0.199	(−0.246,−0.154)	3-47-0	−0.619 (27), −0.543 (47)

scored on average 4 points higher than the base run (0.356 minus 0.312 is 0.044), and this difference was statisticially significant (see the Robust section for a detailed explanation of the column headings of Table 16).

Phrasing: The "+2x phrases" and "+1x phrases" runs used 'WEIGHT 2' and 'WEIGHT 1' for the phrases instead of 'WEIGHT 5', and they also produced significant 4 point gains. The "phrases only" run just used the phrases (dropping the IS_ABOUT predicate) and scored about the same as the base run on average, though with a lot of variance. The "number parsing" run used a table with the default parsing of alphanumerics (e.g. "CDKN1A" would be treated as 3 terms (CDKN, 1, A) instead of 1), in a sense removing the natural phrasing of symbols, and the 4 point drop in score passed the significance test. (Note that if the symbols matched as phrases when names did not, the effect would be the same as just increasing the symbol weight (described below), which may be why topic 24 shows a similar increase in Table 16 for both phrases and symbol weighting.) There is probably room for improvement in this term matching area (e.g. a search for 'CDKN1A' will not match 'CDKN 1A' with the parsing rules used for this task).

Query fields: The "2x sym" and "5x sym" runs were the same as the diagnostic base run except that the symbols were each listed twice and five times (respectively) to boost their impact on the score. Table 16 shows the 3 point gain of "2x sym" was statistically significant, while the 2 point gain of "5x sym" was not. Just using the symbols (omitting the name fields) scored 3 points lower on average (as per the "omit names" run), but with a lot of variance. Just using the names and not the symbols scored a significant 11 points lower (as per the "omit symbols" run). Adding in the 3 other fields (preferred product, product, alias prot) scored a significant 5 points lower (as per the "all fields" run). Overall, it appears the symbols are the most useful of the query fields for this task, though perhaps we're not making as effective use of the names as we could (as the phrase experiments suggested).

Relevance ranking: Squaring the importance of inverse document frequency to the relevance calculation (by using SearchServer relevance method 'V2:4' instead of 'V2:3') scored 2 points higher, but did not quite pass the significance test, as per the listed "idf squared (V2:4)" run. Enabling document length normalization or matching of inflections from English stemming made little difference for this task as per the listed "DLEN 500" and "stemming on" runs. Simpler ranking techniques, such as just counting the number of query terms matched (relevance method '2:2') or simply counting all the matches in a record (relevance method '2:1') scored dramatically lower (14 and 20 points respectively, as per the listed "terms count (2:2)" and "hits count (2:1)" runs) indicating that a combination of term frequency dampening and inverse document frequency is still valuable for this task (though we have not separated the impact of these techniques).

As previously mentioned, not restricting to the species given in the topic scored more than 13 points lower as per the listed "omit species" run (the difference of the humG03ns run and the base run). Adding the species to the IS_ABOUT vector instead of using a strict CONTAINS match scored 7 points lower (as per the listed "vector species" run). At the conference it was stated that in a real task it can be useful to find the gene in different species, so these species results apparently are examples of misleading conclusions from the artificial nature of the judgements used this year.

References

[1] Cross-Language Evaluation Forum web site. http://www.clef-campaign.org/

[2] Berry de Bruijn and Joel Martin. Finding Gene Function using LitMiner. Institute for Information Technology, National Research Council of Canada. Notebook paper in draft TREC 2003 Conference Proceedings.

[3] Bradley Efron and Robert J. Tibshirani. An Introduction to the Bootstrap. 1993. Chapman & Hall/CRC.

[4] The .GOV Test Collection. http://www.ted.cmis.csiro.au/TRECWeb/govinfo.html

[5] Andrew Hodgson. Converting the Fulcrum Search Engine to Unicode. In Sixteenth International Unicode Conference, Amsterdam, The Netherlands, March 2000.

[6] Mehmet Kayaalp et al. Methods for accurate retieval of MEDLINE citations in functional genomics. National Library of Medicine. Notebook paper in draft TREC 2003 Conference Proceedings.

[7] NTCIR (NII-NACSIS Test Collection for IR Systems) Home Page. http://research.nii.ac.jp/~ntcadm/index-en.html

[8] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, M. Gatford. (City University.) Okapi at TREC-3. In D. K. Harman, editor, Overview of the Third Text REtrieval Conference (TREC-3). NIST Special Publication 500-226. http://trec.nist.gov/pubs/trec3/t3_proceedings.html

[9] Text REtrieval Conference (TREC) Home Page. http://trec.nist.gov/

[10] Stephen Tomlinson and Tom Blackwell. Hummingbird's Fulcrum SearchServer at TREC-9. In E. M. Voorhees and D. K. Harman, editors, Proceedings of the Ninth Text REtrieval Conference (TREC-9). NIST Special Publication 500-249. http://trec.nist.gov/pubs/trec9/t9_proceedings.html

[11] Stephen Tomlinson. Experiments in 8 European Languages with Hummingbird SearchServer^{TM} at CLEF 2002. In Carol Peters, editor, Working Notes for the CLEF 2002 Workshop. http://clef.iei.pi.cnr.it:2002/workshop2002/WN/26.pdf

[12] Stephen Tomlinson. Experiments in Named Page Finding and Arabic Retrieval with Hummingbird SearchServer^{TM} at TREC 2002. In E. M. Voorhees and Lori P. Buckland, editors, Proceedings of the Eleventh Text REtrieval Conference (TREC 2002). NIST Special Publication 500-251. http://trec.nist.gov/pubs/trec11/t11_proceedings.html

[13] Stephen Tomlinson. Hummingbird SearchServer^{TM} at TREC 2001. In E. M. Voorhees and D. K. Harman, editors, Proceedings of the Tenth Text REtrieval Conference (TREC 2001). NIST Special Publication 500-250. http://trec.nist.gov/pubs/trec10/t10_proceedings.html

[14] Stephen Tomlinson. Lexical and Algorithmic Stemming Compared for 9 European Languages with Hummingbird SearchServer^{TM} at CLEF 2003. In Carol Peters, editor, Working Notes for the CLEF 2003 Workshop. http://clef.iei.pi.cnr.it/2003/WN_web/19.pdf

[15] Thijs Westerveld, Wessel Kraaij and Djoerd Hiemstra. Retrieving Web Pages using Content, Links, URLs and Anchors. In E. M. Voorhees and D. K. Harman, editors, Proceedings of the Tenth Text REtrieval Conference (TREC 2001). NIST Special Publication 500-250. http://trec.nist.gov/pubs/trec10/t10_proceedings.html

IBM Research and the University of Colorado TREC 2003 Genomics Track

Eric W. Brown*, Andrew Dolbey†, Lawrence Hunter†

*IBM TJ Watson Research Center
PO Box 704
Yorktown Heights, NY 10598
ewb@us.ibm.com

†School of Medicine
University of Colorado
Denver, CO 80262
{Andrew.Dolbey, Larry.Hunter}@uchsc.edu

Introduction

IBM Research and the University of Colorado collaborated on their submission to the inaugural Genomics track at TREC 2003. IBM Research has extensive experience in natural language processing, text analysis, and large-scale systems [9, 13, 3, 5, 16, 10]. IBM also has numerous research and business activities in the broad areas of bioinformatics and bio-medical information processing [14, 8]. IBM Research is currently developing *BioTeKS*, a middleware system for text analysis, mining, and information retrieval in the bio-medical domain. The University of Colorado (CU) has been working in the area of bioinformatics and text analysis in the bio-medical domain for a number of years and has made substantial contributions to the field [7, 11, 15, 12]. CU contributed their domain expertise to enhance the BioTeKS system and jointly we designed and evaluated experiments while preparing our track submissions.

The basic premise of BioTeKS is that the best way to enable effective exploitation of vast text resources is to associate meaningful semantics with the tokens and phrases in the text. With a better understanding of the semantic content of text as a foundation, we can build information extraction, summarization, and indexing systems that address specific information needs in complex domains. For example, bio-medical researchers often need to find documents that contain specific entities (e.g., genes, proteins, cellular components) interacting in certain ways. To satisfy such requests, we must first be able to identify the entities (which may be named using a variety of aliases or synonyms) and then recognize textual constructs that describe these entities interacting with the desired relationships.

BioTeKS is built on the IBM Unstructured Information Management Architecture (UIMA) [6], which is a framework for building unstructured information analysis applications. UIMA provides a number of standard facilities for managing the flow of data through the system, scheduling and orchestrating low-level analysis tasks, and assembling, analyzing, and storing results. BioTeKS uses the UIMA framework to assemble text analysis engines that provide tokenization, named entity recognition, part of speech tagging, shallow and deep parsing, relationship extraction, and semantic indexing.

For our Genomics track submissions, we focused on developing named entity recognizers for genes, proteins, and functions. Our basic recognizer was dictionary based, where each dictionary entry contained all known synonyms for the corresponding entity, and matching of synonyms against the text involved normalization heuristics appropriate for the entity type. For example, authors are often lax in their use of capitalization, spaces, hyphens, and slashes when writing gene symbols [4]. Our matching heuristics consider this behavior, and much of our pre-submission work involved experiments to determine which heuristics provide the best balance of precision and recall for the Genomics track tasks. Our dictionary of gene and protein names was derived from the full LocusLink database, and our function dictionary was derived based on a statistical analysis of verbs and related nominalizations that frequently co-occur with the gene of interest in the Genomics track training data.

For the primary task (ad-hoc retrieval), we analyzed the test corpus with our named entity recognizers and created annotations in the text for recognized entities. An annotation spans the original text in the document and contains meta-data about the annotation, such as a canonical form and the semantic role of the entity. We then used the JuruXML search engine [1, 2] to index the full text and annotations for each document in the corpus. The JuruXML query language supports both free-text queries as well as queries over annotations, annotation attributes, and the text spanned by annotations. We automatically generated the JuruXML queries from the test topics, with the final query generation algorithm selected based on experiments with the training data. Our final average precision over 11 points of recall for the 50 test queries was 0.28.

For the secondary task (information extraction/summarization), we applied the same set of named entity recognizers to annotate the text documents. We then scored each sentence using a weighted combination of features, including annotations, location in the document, and structural role of the sentence (e.g., title). The weights were determined empirically, and the best scoring sentence was returned as the summary.

In the remaining sections we describe our overall architecture, present our approach in more detail, briefly analyze our results, and close with conclusions.

Architecture

Our BioTeKS system is built on the IBM Unstructured Information Management Architecture (UIMA) [6], which is summarized in Figure 1. The UIMA provides a framework for implementing and deploying an unstructured information processing and analysis system. Unstructured information (text in this scenario) is fed to an Application Logic layer, which represents a specific instantiation of the architecture for the current domain and provides an application level interface to the framework. The Application Logic layer passes documents to the Collection Analysis Engine, which calls on the Collection Processing Manager to orchestrate the text analysis processing steps. The actual text analysis operations are performed by pluggable Text Analysis Engines.

A Text Analysis Engine, or TAE, performs a specific text analysis task, such as tokenization, lemmatization, part of speech tagging, parsing, named entity recognition, relationship extraction, etc. TAEs may operate directly on document content, or they may process the output of previously run TAEs. The UIMA framework defines a standard API for building a TAE and describing its functionality, inputs, and outputs. For TAEs that require access to other information resources during analysis, the Structured Knowledge Access module provides mechanisms for accessing various knowledge resources, such as dictionaries, lexicons, or ontologies. These resources may be provided locally, or they may be standard, external resources (e.g., MeSH, UMLS, GO) with appropriate Knowledge Source Adapters that allow access to the resource through the framework.

When the text analysis processing steps for a given document are complete, the Collection Processing Manager submits the results for indexing by the Semantic Search Engine (JuruXML), stores selected analysis results and document meta-data in the Store, and returns the results to the Collection Analysis Engine. The Collection Analysis Engine accumulates results over all of the documents in the collection and performs any collection-wide analyses specified by the application logic layer, saving those results in the Store.

The UIMA framework exploits standard middleware software to implement various components of the framework as appropriate (e.g., a relational database management system, such as IBM DB2™, for the Store, or an application server, such as IBM WebSphere™, for deploying Text Analysis Engines as web services). UIMA in turn provides support for deploying Collection Processing and Text Analysis steps in a variety of local, distributed, and parallel configurations, depending on the underlying computing infrastructure.

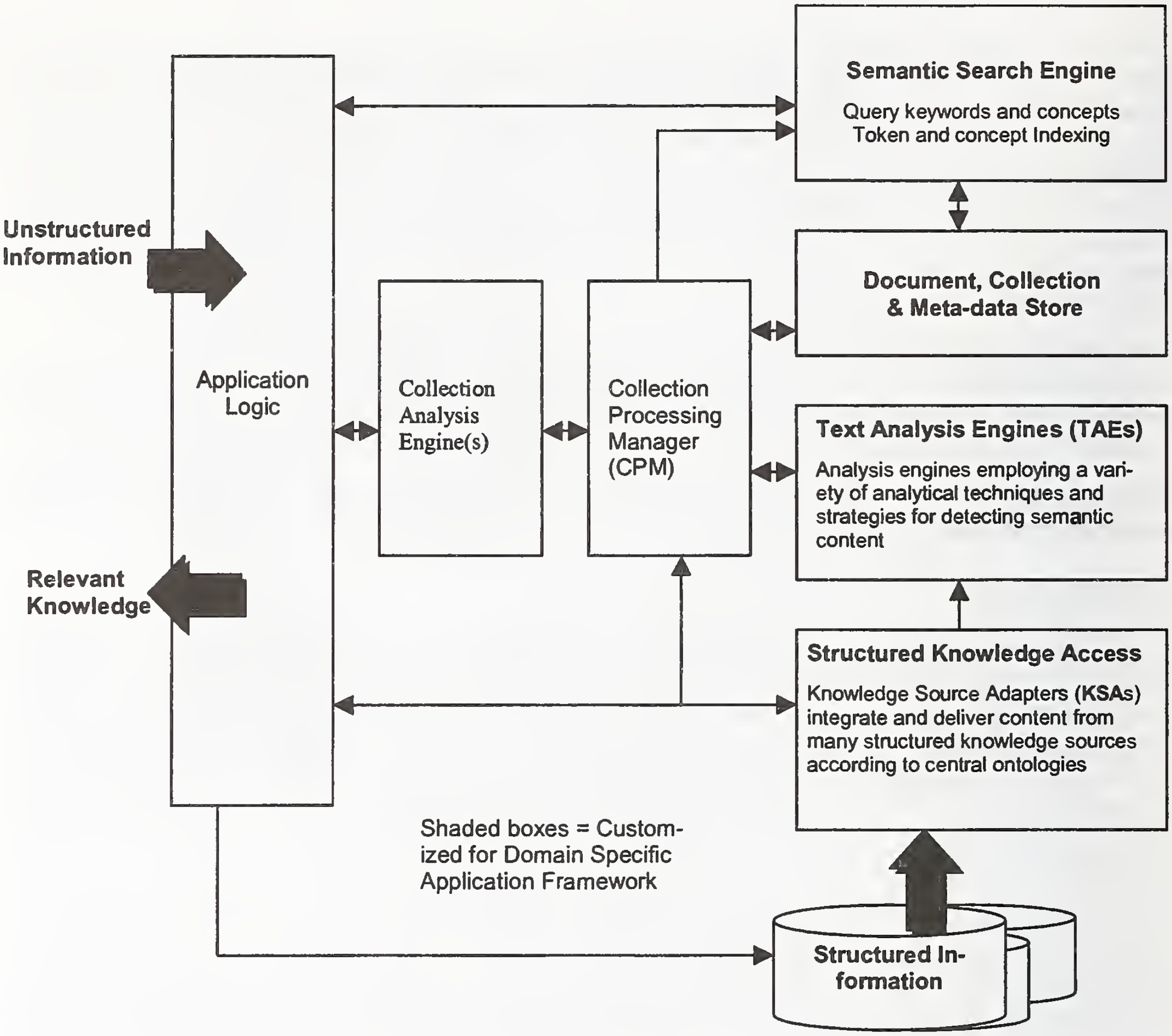

Semantic Search Engine
Query keywords and concepts
Token and concept Indexing
Unstructured Information
Document, Collection & Meta-data Store
Application Logic
Collection Analysis Engine(s)
Collection Processing Manager (CPM)
Text Analysis Engines (TAEs)
Analysis engines employing a variety of analytical techniques and strategies for detecting semantic content
Relevant Knowledge
Structured Knowledge Access
Knowledge Source Adapters (KSAs) integrate and deliver content from many structured knowledge sources according to central ontologies
Shaded boxes = Customized for Domain Specific Application Framework
Structured Information

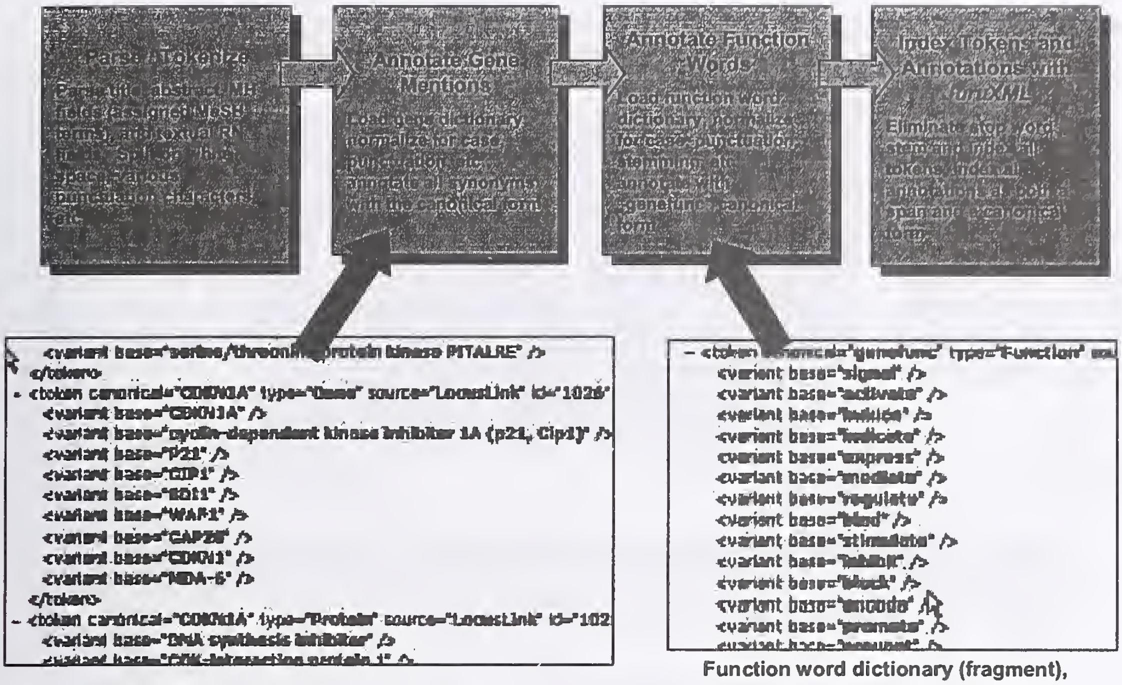
Figure 1 UIMA High-Level Architecture

When all of the documents have been processed, the Application Logic layer provides access to the processing results either via the Semantic Search Engine or through custom access functions to the Store. The Semantic Search Engine is a key component of the system, providing the ability to express complex and sophisticated queries over both the raw text in the documents and the results of the text analysis processing.

Approach

Task 1

Our overall approach to solving the Task 1 problem (retrieve documents that describe the function of a given gene) was to parse and tokenize the MEDLINE abstracts, recognize gene mentions and annotate them with a canonical form, recognize "function words" that are indicative of gene function and annotate them with a canonical form, index the full text plus annotations with JuruXML, and automatically generate JuruXML queries. The text analysis flow is summarized in Figure 2.


Parse / Tokenize
Parse title, abstract, MH fields (assigned MeSH terms), and textual RN fields. Split on white space, various punctuation characters, etc.
Annotate Gene Mentions
Load gene dictionary, normalize for case, punctuation, etc.; annotate all synonyms with the canonical form
Annotate Function Words
Load function word dictionary, normalize for case, punctuation, stemming, etc.; annotate with genefunc canonical form
Index Tokens and Annotations with JuruXML
Eliminate stop word, stem and index all tokens; index all annotations as both span and canonical form.

```
    <variant base="serine/threonine protein kinase PITALRE" />
  </token>
- <token canonical="CDKN1A" type="Gene" source="LocusLink" id="1026"
    <variant base="CDKN1A" />
    <variant base="cyclin-dependent kinase inhibitor 1A (p21, Cip1)" />
    <variant base="P21" />
    <variant base="CIP1" />
    <variant base="SDI1" />
    <variant base="WAF1" />
    <variant base="CAP20" />
    <variant base="CDKN1" />
    <variant base="MDA-6" />
  </token>
- <token canonical="CDKN1A" type="Protein" source="LocusLink" id="102
    <variant base="DNA synthesis inhibitor" />
    <variant base="CDK-interacting protein 1" />
```

Gene dictionary (fragment), derived from LocusLink

```
- <token canonical="genefunc" type="Function"
    <variant base="signal" />
    <variant base="activate" />
    <variant base="induce" />
    <variant base="indicate" />
    <variant base="express" />
    <variant base="mediate" />
    <variant base="regulate" />
    <variant base="bind" />
    <variant base="stimulate" />
    <variant base="inhibit" />
    <variant base="block" />
    <variant base="encode" />
    <variant base="promote" />
```

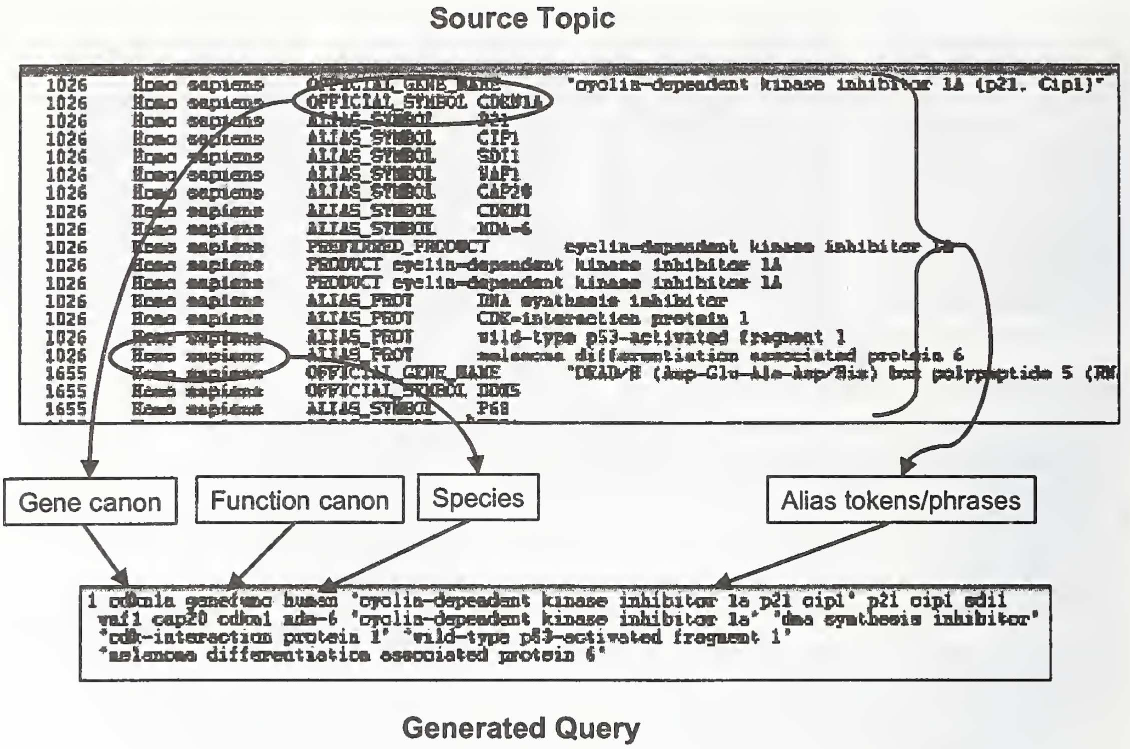
Function word dictionary (fragment), derived by analyzing statistical co-occurrence of verbs with genes in training data, plus manual curation

Figure 2 BioTeKS processing steps and dictionaries

To recognize gene mentions and function words, we used a dictionary-based named entity recognizer implemented as a UIMA Text Analysis Engine. The dictionary contains an entry for each named entity, which in turn includes all known synonyms for the entity, a "canonical" or preferred name for the entity, and additional optional meta-data associated with the entity. A synonym may be a single token or a multi-token phrase. The entity recognizer TAE scans the input text and at each token searches for the longest matching synonym in the dictionary. When a matching synonym is found an annotation is created in the text that spans the matching tokens. The annotation includes the canonical form for the entity and all other meta-data specified in the dictionary entry for the entity. The TAE will optionally perform stemming and case folding when attempting to match the text against the dictionary of synonyms, and the set of characters used to separate tokens is configurable.

The dictionary for finding gene mentions was automatically derived from the full LocusLink database, and included 156,533 genes with a total of 387,850 synonyms. The preferred gene symbol was used for the canonical form and the synonyms were extracted from the LocusLink entry fields that contain the known gene or protein aliases used for the gene. During dictionary matching we did not use stemming, but we did case fold all tokens that contained at least one numeric character, and the set of characters used to separate tokens included white space, punctuation characters, and in particular hyphen, forward and backward slash, and parentheses. Tokenizing on these characters and eliminating them from the tokens improved the recall of our gene identifier and addressed some of the variability found in gene names associated with inconsistent use of space, hyphen, slashes, and parentheses.

The dictionary of function words was derived in a semi-automatic fashion. Using the training data for Task 1, we identified verbs that frequently occur in sentences with genes as the subject. We sorted this list based on a scoring function of the significance of this co-occurrence, and then manually curated the list to select important gene function words, yielding a rather small list of 28 function words. We used



```
1026    Homo sapiens    OFFICIAL_GENE_NAME      "cyclin-dependent kinase inhibitor 1A (p21, Cip1)"
1026    Homo sapiens    OFFICIAL_SYMBOL CDKN1A
1026    Homo sapiens    ALIAS_SYMBOL    P21
1026    Homo sapiens    ALIAS_SYMBOL    CIP1
1026    Homo sapiens    ALIAS_SYMBOL    SDI1
1026    Homo sapiens    ALIAS_SYMBOL    WAF1
1026    Homo sapiens    ALIAS_SYMBOL    CAP20
1026    Homo sapiens    ALIAS_SYMBOL    CDKN1
1026    Homo sapiens    ALIAS_SYMBOL    MDA-6
1026    Homo sapiens    PREFERRED_PRODUCT       cyclin-dependent kinase inhibitor 1A
1026    Homo sapiens    PRODUCT cyclin-dependent kinase inhibitor 1A
1026    Homo sapiens    PRODUCT cyclin-dependent kinase inhibitor 1A
1026    Homo sapiens    ALIAS_PROT      DNA synthesis inhibitor
1026    Homo sapiens    ALIAS_PROT      CDK-interaction protein 1
1026    Homo sapiens    ALIAS_PROT      wild-type p53-activated fragment 1
1026    Homo sapiens    ALIAS_PROT      melanoma differentiation associated protein 6
1655    Homo sapiens    OFFICIAL_GENE_NAME      "DEAD/H (Asp-Glu-Ala-Asp/His) box polypeptide 5 (RN
1655    Homo sapiens    OFFICIAL_SYMBOL DDX5
1655    Homo sapiens    ALIAS_SYMBOL    P68
```

```
1 cdkn1a genefunc human "cyclin-dependent kinase inhibitor 1a p21 cip1" p21 cip1 sdi1
 waf1 cap20 cdkn1 mda-6 "cyclin-dependent kinase inhibitor 1a" "dna synthesis inhibitor"
 "cdk-interaction protein 1" "wild-type p53-activated fragment 1"
 "melanoma differentiation associated protein 6"
```

Figure 3 Task 1 automatic query query generation.

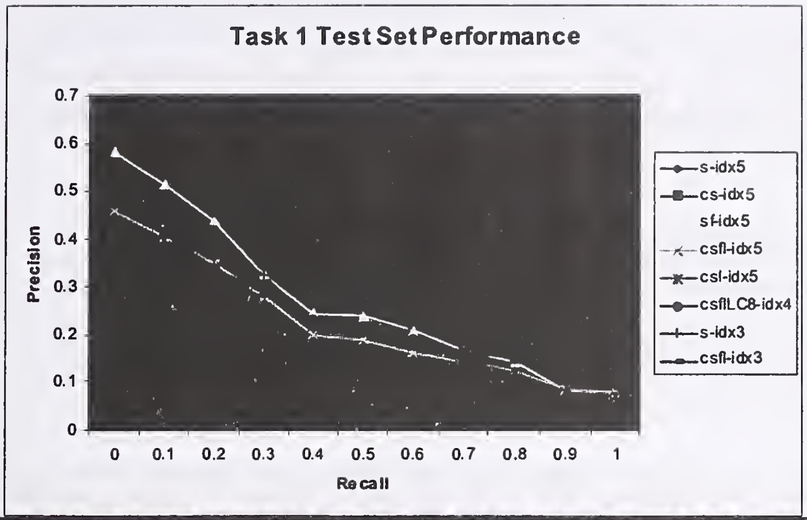
stemming and case folding during matching, allowing the function word finder to match noun forms as well as verb forms. Samples of our dictionaries are shown in Figure 2.

For the Task 1 queries, we explored using the gene canon, the species name (*human*, *mice* / *mouse*, *rat*, or *Drosophila*), the function word canon, and all given gene aliases as tokens or phrases. We automatically generated queries for JuruXML based on the provided query source topics. Our query generation process is summarized in Figure 3.

Task 2

For Task 2 (automatically extract and summarize a gene's function given a document known to describe the gene's function), we decided to chunk the document into whole sentences, score each sentence, and return the best scoring sentence as our answer. A sentence's score is based on whether or not it contains the target gene, how many gene function words it contains, what structural role the sentence plays (i.e., is it the *title*), and where in the document the sentence occurs. To identify genes and function words in the documents we applied the text analysis processing steps shown in Figure 2, excluding the final step of indexing with JuruXML.

Given that we had committed to extracting the single best scoring sentence as the summary, we performed a simple analysis to determine if it was worthwhile to analyze the full article versus just the MEDLINE abstract. For each document we scored every sentence in the document against the gold standard (the GeneRIF for the given gene and article) using the classic Dice coefficient (as implemented in the scoring code provided by the track) and identified the best scoring sentence. We then returned this sentence as our answer and calculated the average performance over the set of documents. This essentially produces an upper bound the best possible score that could be obtained assuming a strategy of returning the single best sentence.



Key	Query Generation Option (Query always includes canonical gene name)
s	Species name included in query
c	For gene name "X" ambiguous w/ common words, "X" goes to "X gene" and "X protein"
f	Gene function annotation term used in query
l	Long form: all topic fields used, multi-word fields in quotes (phrases), all tokens & phrases unique
lL	Same as long form, but multi-word fields are not in quotes
Cn	Canonical gene name (official or preferred name) repeated n times
	Text Analysis / Indexing Option
idx4,5	Gene matching done with case-folding of names with 1+ digits, hyphens stripped
idx3	Heuristic expansion of hyphen/space combinations

Figure 4 Task 1 test set performance.

For the set of full-length articles, the optimal classic Dice score is 70.61%. For the set of MEDLINE abstracts, the optimal classic Dice score is 71.09%. This result is somewhat surprising given that the abstract should be a proper subset of the full-length article. This anomaly is due to the following: most GeneRIFs are extracted directly from the MEDLINE abstracts. The full-length article contains SGML entities that must be translated to ASCII for the MEDLINE abstract, e.g., '&agr;' -> 'alpha'. This translation is not done consistently, such that a fragment extracted from the full text may not exactly match the GeneRIF using the Dice measures. Given this result, we chose to use the abstracts rather than the full-length articles for our actual Task 2 run.

Results

Task 1

Using the training queries for Task 1 we explored a variety of query generation options and measured the performance of the system. On the training queries we were able to obtain an 11pt average precision of 0.4259. Based on these results, we submitted two runs on the test queries. Run **IBMbt1** was generated using queries that comprise only the gene canonical form and the species name. This run produced an 11pt average precision of 0.2823, but found only 456 of 566 possible relevant documents. Run **IBMbt2** was generated using queries that comprise the gene canonical form, species name, function keyword, and all alias forms from the source query topic. This run produced an 11pt average precision of 0.2259 while returning 534 of 566 possible relevant documents. Adding more terms to the query improved recall but resulted in poorer overall ranking.

With the relevance judgments for the test queries we performed a more detailed analysis of various query generation options on the test data. These results are shown in Figure 4. From the plot we see that the relatively simple query **s-idx5** (species name and gene canonical form) produces better precision at low recall, while the queries with additional terms produce worse precision at low recall but better precision at higher recall levels.

Task 2

Although we explored a variety of parameter settings for our sentence scoring function, we were not able to obtain a scoring function that performed better than simply returning the title. We are currently exploring a number of ways to improve our scoring function, such as incorporating a shallow parse in the analysis to more accurately connect the target gene with the function words in the sentence.

Conclusion

Based on our results, we conclude that using a comprehensive gene dictionary with appropriate normalization during matching is an effective way to annotate gene mentions in biomedical text. The normalization and matching heuristics are very important, however, given the considerable variability found in gene names, especially in the use of capitalization, spaces, hyphens, slashes, and parenthesis. Unfortunately, identifying gene "function words" is not necessarily useful in a bag-of-words query context. We suspect, however, that they might be more effective when identified in syntactic relationships with genes. This will require the addition of parsing (e.g., shallow parsing) to the analysis phase.

The significant difference in our training and test results for Task 1 (a phenomenon observed by many of the Task 1 participants) suggests that the overall test set is not stable. There may be too few relevant documents for some of the test queries, or the relevance judgments may be too incomplete. This latter issue is particularly important and was raised by a number of the track participants.

Given the exploratory nature of Task 2 and the relatively late decision by the track to collect official runs for the task, we did not invest as much time in this Task. In the process of developing our sentence scoring function, we observed a number of cases where our extracted sentence appeared to convey the same meaning as the gold standard, but due to the wording the sentence scored poorly using the various Dice measures. Based on our experience and the experience of others on this task, we are not convinced that this particular evaluation accurately measures a system's ability to perform what is arguably an important real world task.

Given the overall constraints under which this inaugural Genomics track was run, we feel the track was very successful and accomplished its goals for the first year. In particular, it brought this important area of research to the attention of the Information Retrieval community and made a positive step in the direction of building useful test sets in this domain, which currently suffers from a severe lack of well constructed test sets. We look forward to next year's track and the development of more realistic tasks supported by more thorough evaluation.

References

[1] D. Carmel, E. Amitay, M. Herscovici, Y. S. Maarek, Y. Petruschka and A. Soffer, Juru at TREC 10--Experiments with Index Pruning, *Proceedings of The Tenth Text REtrieval Conference (TREC 2001)*, 2002,

[2] D. Carmel, Y. S. Maarek, M. Mandelbrod, Y. Mass and A. Soffer, Searching XML Documents via XML Fragments, *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, Toronto, Canada, 2003, 151-158.

[3] J. Chu-Carroll, J. Prager, C. Welty, K. Czuba and D. Ferrucci, A Multi-Strategy and Multi-Source Approach to Question Answering, *Proceedings of The Eleventh Text REtrieval Conference (TREC 2002)*, Gaithersburg, MD, 2003,

[4] K. B. Cohen, A. Dolbey, G. Acquaah-Mensah and L. Hunter, Contrast and Variability in Gene Names, *Proceedings of the ACL Workshop on Natural Language Processing in the Biomedical Domain*, Philadelphia, PA, 2002, 14-20.

[5] J. W. Cooper and R. J. Byrd, Lexical Navigation: Visually Prompted Query Expansion and Refinement, *Proceedings of the ACM International Conference on Digital Libraries*, Philadelphia, PA, 1997, 237-246.

[6] D. Ferrucci and A. Lally, Accelerating Corporate Research in the Development, Application and Deployment of Human Language Technologies, *Proceedings of the HLT-NAACL 2003 Workshop on Software Engineering and Architectures for Language Technology Systems*, Edmonton, Canada, 2003, 68-75.

[7] L. Hunter, R. C. Taylor, S. M. Leach and R. Simon, GEST: a gene expression search tool based on a novel Bayesian similarity metric, Bioinformatics, 17 (2001), pp. S115-S122.

[8] IBM, Life Sciences Solutions, http://www-3.ibm.com/solutions/lifesciences/.

[9] A. Ittycheriah and S. Roukos, IBM's Statistical Question Answering System-TREC 11, *Proceedings of The Eleventh Text REtrieval Conference (TREC 2002)*, Gaithersburg, MD, 2003,

[10] D. E. Johnson, F. J. Oles, T. Zhang and T. Goetz, A decision-tree-based symbolic rule induction system for text categorization, IBM Systems Journal, 41 (2002), pp. 428-437.

[11] P. V. Ogren, K. B. Cohen, G. Acquaah-Mensah, J. Eberlein and L. Hunter, The Compositional Structure of Gene Ontology Terms, *Proceedings of the Pacific Symposium on Biocomputing*, 2004, 9:214-225.

[12] T. Phang, M. Rudolph, M. Neville and L. Hunter, Trajectory Clustering: A Non-Parametric Method for Grouping Gene Expression Time Courses, With Applications to Mammary Development, *Proceedings of Pacific Symposium on Biocomputing*, 2003, 8:351-362.

[13] J. Prager, E. Brown, A. Coden and D. Raden, Question-Answering by Predictive Annotation, *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Athens, Greece, 2000, 184-191.

[14] I. Rigoutsos, T. Huynh, A. Floratos, L. Parida and D. Platt, Dictionary-driven Protein Annotation, Nucleic Acids Research, 30 (2002).

[15] R. Shenkar, J. P. Elliott, K. Diener, J. Gault, L. J. Hu, R. J. Cohrs, T. Phang, L. Hunter, R. E. Breeze and I. A. Awad, Differential gene expression in human cerebrovascular malformations, Neurosurgery, 52 (2003), pp. 465-477.

[16] T. Zhang, F. Damerau and D. Johnson, Text Chunking based on a Generalization of Winnow, Journal of Machine Learning Research, 2 (2002), pp. 615-637.

Juru at TREC 2003 - Topic Distillation using Query-Sensitive Tuning and Cohesiveness Filtering

Einat Amitay, David Carmel, Adam Darlow, Michael Herscovici,
Ronny Lempel, Aya Soffer
IBM Haifa Research Lab
Reiner Kraft, Jason Zien
IBM Almaden Research Center

1 Introduction

This is the third year that our group participates in TREC's Web track, the second year in the topic distillation task. Our experiments last year, as well as those of other participants, indicated that sophisticated link-based measures did not significantly improve search results in comparison to standard text-based relevance scoring. We thus focused our experiments this year on improving the ranking algorithms of our core search engine, Juru, and on developing measures that are good indicators of topical pages.

In particular, realizing that one ranking flavor does not fit all queries [3][6], we developed a method, which fine tunes the parameters governing the ranking formula based on the nature of the query. This novel ranking method, called the *QUEry Sensitive Tuner* (or QUEST), tunes the ranking parameters according to the query type. QUEST classifies queries into "informational" vs. "navigational" by considering both the query's length and the expected number of documents containing all query terms (*edf*). For queries with a few expected results, each document's score is primarily determined according to the document's textual score, i.e. its similarity to the query. On the other hand, for queries with many expected results, document scores are determined by considering additional factors such as anchor-text data, number of in-links, etc.

In addition, we continued experimenting with some of the topic distillation filters we introduced last year [2], as well as with a new cohesiveness filter. The cohesiveness filter tries to identify pages that focus on the desired topic in contrast to pages than just mention it in passing, or which mention it in the context of a broader topic. This is achieved by identifying pages in which the query terms are uniformly distributed over the entire page.

The rest of this paper is organized as follows: Section 2 describes the QUEST algorithm and the query parameters tuned by the algorithm according to the query type. In section 3 we describe the cohesiveness filter that tries to distill pages that focus on the desired topic. Section 4 describes the results of the official runs submitted to TREC. Section 5 concludes.

2 QUEry Sensitive Tuner (QUEST)

The QUEST algorithm tunes the query parameters according to the query's characteristics, which in turn imply its type. QUEST classifies queries into "informational" vs. "navigational" by considering both the query's length and the expected number of documents containing all query terms (*edf*). The main rationale is that for short queries with many expected results (large *edf*), standard IR techniques based on textual scores cannot discriminate between topical and non-topical pages, therefore more factors, especially static scores and anchor-text scores associated with the documents, should be used in order to distill the best results. On the other hand, for long queries with few expected results, document's static scores, which are independent of the query, should only take a secondary role, as standard IR techniques are expected to return satisfactory results.

After query classification, the query parameters are tuned according to the query type. For "navigational" queries, parameters are set such that the number of in-links per page has stronger effect on the page's final score than for "informational" queries for which the textual score is dominant in determining the final score. Similarly, the anchor text associated with these in-links is weighted more heavily in navigational queries compared to informational ones. QUEST does not, however, assume just the two extremes; rather it tunes the parameters on a sliding scale ranging from purely navigational to purely informational.

We now describe in more details the QUEST algorithm. QUEST treats separately queries containing one, two and three+ terms. For each query length, it maintains a threshold on the query *edf*. In addition, it also maintains two sets of values for several ranking parameters, one set for informational queries and one set for navigational queries. A query with an *edf* lower than the threshold is classified as "informational" and its parameters are set using the informational set of parameters. A query with an *edf* higher then the threshold is considered "navigational" and its parameters are set using the navigational set of parameters. See Section 2.2 for details on the calculation of the *edf*.

2.1 Query Parameters tuned by QUEST

QUEST tunes three sets of parameters as described below:

I. ***Boosts for different token types***. The tokens of a document are classified into several types, and the significance of a token and its contribution to the document's textual score is determined by the boost associated with its type. Thus, the occurrence of tokens with a high boost in the document's content significantly affects its textual score, while tokens with a low boost contribute much less. The token types include:

   a. **Textual tokens**: tokens extracted from the document's raw text which are differentiated into:
      i. Title tokens – extracted from the document's title.
      ii. Strong tokens – extracted from the document's headers.
      iii. Mid tokens – extracted from the document's emphasized text (colored, bold, etc.).
      iv. Regular tokens – all the rest.

b. **Anchor tokens**: tokens extracted from the anchor text of the document's in-links. These tokens are differentiated according to the relation between the source and target of the link:
   i. Different site anchor: anchor tokens where the source site differs from the target site
   ii. Same site anchor: anchor tokens where the anchor and the target pages are from the same site but in different directories.
   iii. Same dir anchor: anchor tokens where the source and the target pages reside in the same directory.

c. **URL tokens**: tokens extracted from the document's URL.

d. **Snippet tokens**: Tokens extracted from the document's snippet. We compute for each document a snippet based on its anchors, using the method described in [1].

For informational queries, textual tokens are given the highest boosts while for navigational queries anchor tokens, URL tokens, and snippet tokens receive higher boosts.

II. ***Lexical Affinity weight (LA-Weight).*** Our ranking algorithm takes into account lexical affinities common to the query and the document, in addition to simple query terms. Lexical affinities are pairs of closely related terms frequently found in proximity to each other [7]. Each query term, a simple keyword or a lexical affinity, contributes to the textual score of the document according to its term frequency and to its inverse document frequency (following the *tf-idf* formula). The LA-weight determines the relative contribution of lexical-affinities to the document's score compared to simple keywords. Experiments have shown that the LA-weight should be smaller for longer queries [4]. In accordance, QUEST assigns a lower LA-weight for informational queries as compared to navigational queries.

III. ***Static Score coefficient:*** The final score of a document is computed by linearly combining its textual score with a static score. The static score is based on the number of its in-links. The Static Score coefficient determines the relative weight of the static score with respect to the weight of the textual score of the document. QUEST assigns a higher value to the static score coefficients for navigational queries.

2.2 Approximating the expected document frequency (*edf*) per query

The main feature used by QUEST for query classification is the expected document frequency *edf*. For one-term queries, the document frequency (*df*) can be precisely determined since the *df* of each term is stored within the index. For multi-term queries, the *edf* must be approximated since the only way to derive the precise *edf* is to process the query.

Given a query with k terms $q = q_1..q_k$. The *edf* of the query is approximated based on the *df* values of the individual query terms. Assuming independence between query terms, the number of documents containing all of the query terms can be estimated by multiplying the occurrence probability of all query terms. The occurrence probability of a query term q_i can be approximated by $Pr(q_i) = df(q_i)/|D|$, where $df(q_i)$ is the document frequency of term q_i and $|D|$ is the total number of documents in the collection. Thus, the *edf* of a query q with k independent terms is:

$$edf(q) = \frac{\prod_{i=1}^{k} df(q_i)}{|D|^k} |D|$$

Since query terms are usually not independent, but are rather expected to co-appear in documents, we heuristically multiply the above by the number of query terms k:

$$edf(q) = \frac{k\prod_{i=1}^{k} df(q_i)}{|D|^{k-1}}$$

3. The Cohesiveness Filter

The relevance score computed above finds good individual candidates for topical pages. However, given that the goal of the topic distillation task is to find a *set* of topical pages, we apply some additional filters that influence the final ranking. The goal of these topic distillation filters is to identify pages that exhibit features of a good topical page, and to boost their query relevance score. We applied the following sequence of filters to the initial search results: 1) duplicate-elimination filter, 2) site-compression filter, and 3) the new cohesiveness filter. The first two filters were already reported last year in [2]. The new cohesiveness filter tries to identify pages that focus on the desired topic in contrast to pages that just mention it in passing, or which mention it in the context of a broader topic. This is achieved by identifying pages in which the query terms are uniformly distributed over the entire page.

More specifically, for each document in the result set we measure the uniformity of the query terms along the document's content. This is done by measuring the entropy of the occurrence distribution of the query terms within the document. The entropy is maximal when the term occurrences are uniformly distributed over the document's content. The entropy is minimal when all term occurrences are close to each other. We conjecture that the larger the entropy of the term distribution, the higher its uniformity.

Given a query term t with a list of positions $o_1, o_2,...,o_k$ within document d of length $|d|$, the entropy of the term occurrence distribution within d is measured by:

$$entropy(t,d) = -o_1 \log o_1 - \sum_{i=2}^{k}(o_i - o_{i-1})\log(o_i - o_{i-1}) - (|d| - o_k)\log(|d| - o_k)$$

The cohesiveness of the document d for query q is defined by the weighted average entropy of q's query terms within d:

$$cohesiveness(d,q) = \sum_{t \in q} idf(t) * entropy(t,d)$$

The cohesiveness filter computes for each document in the result set a new score based on its previous score and its cohesiveness. It then re-ranks the search results based on the new score. The new score is a linear combination of the previous score and the cohesiveness score. The cohesiveness filter weight determines the relative weight between the two scores. This weight is set by QUEST according to the query type. For purely informational queries this weight is low, while for purely navigational queries the weight is high.

The cohesiveness filter is especially useful for queries with high frequency terms. In such cases, the cohesiveness filter will prefer pages where the query terms occur throughout the entire document over pages where query terms appear only in part of the document.

4. Results

We used the Juru search engine [5] to index and search the pages in the ".gov" domain. Each page was indexed based on its content as well as its anchor descriptions, its URL, and its snippet (see Section 2). Each page is scored by a linear combination of its textual score and its link topology score (a static score). The static score of page *p* is based on the number of links *n* pointing to *p*:

$$St(p) = \begin{cases} 1.0 & n \geq N \\ \sqrt{n/N} & otherwise \end{cases}$$

The constant *N* determines an upper bound on a page's in-link number; each page with more than *N* in-links receives the maximum static score of 1. The *N* parameter is also set by QUEST according to the query type, low value for informational queries and high value for navigational queries.

The combined scores are used to rank the set of pages. The top 200 pages are re-ranked using the sequence of filters described above designed to guarantee a mixture of good sources in the top-10 list returned by the system. The top 100 pages were submitted to TREC.

We submitted 5 runs for the topic distillation task. The JuruFull run scored pages based on both a textual and a topological score. The query parameters were tuned separately for each query using the QUEST algorithm as described above, and all filters were invoked on the search results. In the JuruNoAnchor run we zeroed the boosts of all anchor tokens, thus, textual ranking is based only on the document content. In the JuruNoCohes run the cohesive filter was ignored by zeroing the cohesiveness filter weight. In JuruNoQueryDiff the QUEST algorithm was ignored by fixing the values of the ranking parameters for all the queries. In JuruNoSS the document static scores were ignored.

Table 1 shows the average P@10 and average R-precision of our runs and the average-best and median P@10 of all participants. While the results of all our runs are much higher than the median, the results are somewhat disappointing. For 16 topics

JuruFull could not find pages marked relevant by the assesors in its top 10 results, among them 3 topics for which all participants completely failed. On the other hand, for 7 topics JuruFull achieved the best result among all participants. Both the JuruNoAnchor and the JuruNoSS runs achieved significantly lower results than the other runs, indicating the significance of link analysis, contradicting our findings from previous year about the relatively insignificance of link analysis for the topic distillation task. There was however no difference between the runs applying QUEST and the cohesiveness filter, and thus the experiments we hoped to achieve by participating in this task are inconclusive.

	Best	Median	JuruFull	JuruNoAnchor	JuruNoCohes	NoQueryDiff	JuruNoSS
P@10	0.28	0.064	0.122	0.088	0.122	0.122	0.086
R-Precision			0.110	0.100	0.106	0.117	0.099

Table 1 -- Average P@10 and R-precision of our runs and the average-best and median P@10 of all participants.

Figure 1 shows the difference between P@10 of our runs and the median P@10 of all participants. For almost all topics (except 3 for the JuruFull run) our runs achieved a better result than the median.

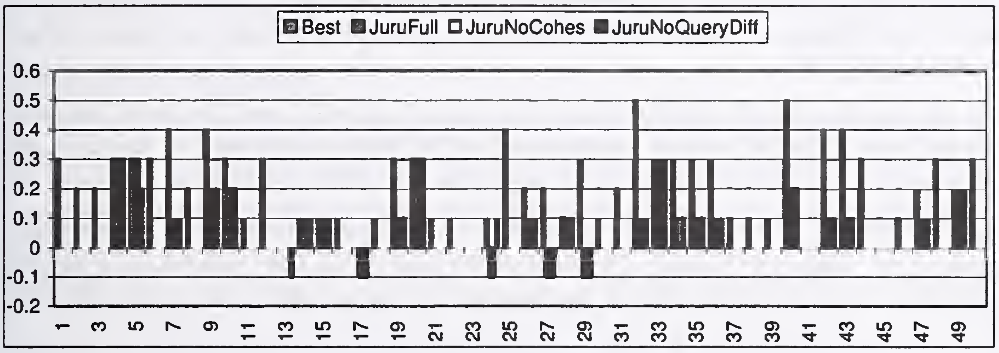


Figure 1 -- The difference between P@10 of the best result, some of our runs, and the median P@10 of all participants

5. Summary

Our experiments this year focused on improving the ranking algorithm of our core search engine, and on developing measures that are good indicators of topical pages. We experimented with the QUEST algorithm that tunes the query parameters according to the query's characteristics. We also experimented with the cohesiveness filter that tries to find topical pages by identifying those in which the query terms are uniformly distributed over the entire page. Our results demonstrate that link analysis and anchor-text data slightly improved the results this year, in contrast to last year. However, our results do not indicate any advantage for QUEST or the cohesiveness filter. One reason for this is the apparent disparity between our understanding and the assessors understanding of the notion of a 'topical page". The topic distillation task, in our opinion, is still not well defined. Consequently, our system in several cases returned many good pages (according to our judgment) that were rejected by the assessors as non-relevant. We believe that QUEST and cohesiveness can indeed make a difference - more exhaustive experiments are needed to study their effectiveness.

References

[1] Amitay E., Paris C. (2000). *Automatically Summarizing Web Sites - Is There A Way Around It?* CIKM 2000, pp. 173-179.

[2] Amitay E., Carmel D., Darlow A., Lempel R., Soffer A. (2002). *Topic Distillation with Knowledge Agents.* In Proceedings of the Eleventh Text REtrieval Conference (TREC 2002), National Institute of Standards and Technology (NIST).

[3] Broder A. (2002). *A taxonomy of web search.* ACM SIGIR Forum 36 (2), pages 3--10, 2002.

[4] Brown E.W., Chong H.A. (1998). *The GURU system in TREC-6.* The Sixth Text Retrieval Conference (TREC-6), pages 535–540, 1998. National Institute of Standards and Technology (NIST).

[5] Carmel D., Amitay E., Herscovici M., Maarek Y., Petruschka Y., Soffer A. (2001). *Juru at TREC 10 - Experiments with Index Pruning.* In Proceedings of the Tenth Text REtrieval Conference (TREC 2001), National Institute of Standards and Technology (NIST).

[6] Kang I.H., Kim G. (2003). *Query type classification for web document retrieval.* Proceedings of the 26th annual international ACM SIGIR conference on Research and Development in Information Retrieval, pages 64—71, 2003. ACM Press.

[7] Maarek Y., Smadja F. (1989). *Full text indexing based on lexical relations: An application: Software libraries.* In Proceedings of the 12th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 198--206, 1989.

IBM's PIQUANT in TREC2003

John Prager, Jennifer Chu-Carroll, Krzysztof Czuba, Christopher Welty, Abraham Ittycheriah, Ruchi Mahindru
{jprager,jencc,kczuba,welty,abei,rkalra}@us.ibm.com
IBM T.J. Watson Research Center
Yorktown Heights, NY 10598

Introduction

For the most part, the system we used for TREC2003 was a smooth evolution of the one we ran in TREC2002 [Chu-Carroll et al, 2003b]. We continued to use our multi-source and multi-agent architecture. For Factoid questions we used all of our previous answering agents with an additional pattern-based agent, an enhanced answer resolution algorithm, and increased coverage of the Cyc sanity checker. We will devote a portion of this paper to performing a post-mortem of our experiences with Cyc this year. For List questions, which we did not attempt previously, we ran our Factoid system with different parameters. For Definition questions we took an entirely new approach, which we call QA-by-Dossier, and which will be the other focus of this paper. While we think that our system performed reasonably well in this subtask, the NIST evaluation results do not reflect this, raising some questions about the Definition subtask specification and evaluation.

The PIQUANT System

We will briefly describe the PIQUANT system, which is depicted in Figure 1. A fuller description can be found in [Chu-Carroll et al., 2003a] and [Prager et al., 2003]. The processing begins with Question Analysis, which involves deep parsing, named-entity recognition and feature-structure unification. Question Analysis produces a QFrame that contains the required answer type, the question type, query keywords and a simple semantic form. The QFrame is passed to the QPlan generator. A QPlan is in principle a general program which directs the subsequent processing, but is currently little more than a list of one or more of the available agents which are to be run on the QFrame, with results passed to Answer Resolution. Answer Resolution combines the candidate answers from multiple agents and using a voting mechanism and pre-learned weights, generates a final list of answers, with confidences.

PIQUANT Agents

The agents will now be briefly described. The Linguistic Query Agent (LQA) uses our Predictive Annotation techniques ([Prager et al., 2000]) to generate a query which includes the desired answer type as a query term, and searches an index made from a pre-annotated corpus. Our GuruQA search engine returns passages of size 1-3 sentences. The top 10 such hits (an empirically and theoretically determined optimum, see [Prager, 2002]) are then passed to an Answer Selection module, which determines the best answer candidates based primarily on syntactic features. This agent is a general-purpose agent, in the sense that it is designed to find answers for any of the approximately 100 answer types that the QFrame might propose and that our named-entity recognizer can detect.

The Description Agent (DSA) is used primarily for "Who is" and "What is" questions, and looks for syntactic constructions such as appositions and relative clauses that are likely loci of descriptions of people and things. Because there is no specific answer type to prime the query with, this agent tends to have lower recall, but it can find constructions that the LQA cannot.

The Pattern-based Agent (PBA) is similarly motivated to the Description Agent, but uses a much more sophisticated matching algorithm. It is somewhat similar to the pattern-based approach of [Ravichandran and Hovy, 2002], with parse trees being the level of representation at which patterns are matched. It is a high precision agent that can be used for many question types, but its coverage is currently very low so its recall is also low.

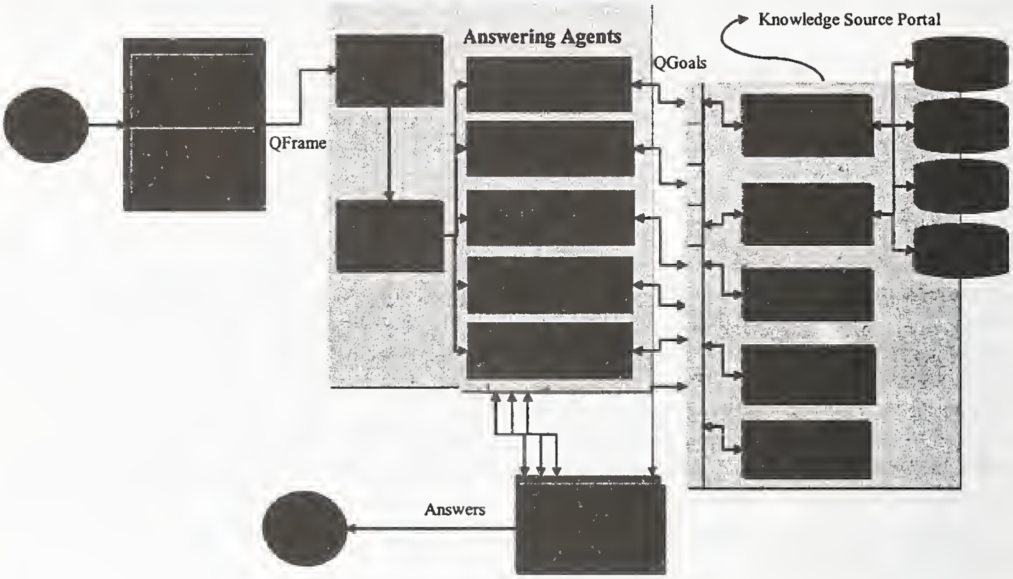


Figure 1 PIQUANT Architecture

The Definition Agent (DFA) employs the Virtual Annotation technique ([Prager et al, 2001] to answer "What is" questions. The focus of the question is looked up in WordNet to find all hypernyms, and the ones that are most likely to co-occur with the question focus in the reference collection, penalized by WordNet path length, are returned. Passages are then retrieved that contain both the question focus and the selected hypernym(s).

The Structured Knowledge Agent (SKA) works in a similar fashion to the Definition Agent. When the QFrame contains a logical form predicate fully expressing the question, the predicate is used as a query through the Knowledge Source Portal, behind which are collections of facts obtained from the Web and elsewhere. When an answer is returned, it is then located in the corpus. Of particular interest and use for Definition questions was its access to data from the Who2 site (http:// www.who2.com) which provided us with biographical information including text snippets that described what a person was *famous-as* and *known-for*.

The Statistical Query Agent (SQA) is another general-purpose agent. We have previously shown ([Chu-Carroll et al., 2003a]) that use of this agent substantially increases our performance, so we used it again this year, for all questions. For TREC2003, the SQA was largely unchanged from last year [Ittycheriah and Roukos, 2002] with the following exceptions:

1. Web pages were retrieved from a popular search engine. Exact answers were extracted from the resulting search results page and the top two web pages as indicated by the search engine. Answers that exceeded a rejection threshold were added to the query for retrieval from the AQUAINT corpus. This improved the precision of this agent: in separate testing the Document MRR of the retrieved set using this agent alone moved from 0.4 to 0.421 by adding these expansion terms while the recall rate remained at 94.75% at Top-1000 documents.

2. The answer selection used a maximum entropy model for chunk selection trained from true sentences of previous evaluations, followed by a maximum entropy chunk ranking model trained on

our system output for 5K questions. This substantially reduced the number of inexact answers compared with our 2002 implementation.

The GuruQA Agent (GQA) is a new agent built specially for our *QA-by-Dossier* methodology, described later. Its function is similar to that of the Linguistic Query Agent, except with no identified answer type, so it can be thought of as a traditional Information Retrieval engine. It is used in conjunction with other facilities, such as the Structured Knowledge Agent, or WordNet, whose glosses contain descriptive phrases or sentences about objects and people. The GuruQA Agent is used to locate instances of these descriptions in a corpus.

Sanity Checking

The integration with the Cyc knowledge base was expanded in a number of ways from our TREC 2002 system. The primary directions we pursued were to expand the coverage of our semantic mapping to Cyc predicates, and to resolve issues with the previous interface regarding the treatment of known answers.

The mapping to Cyc predicates is an important element of our interface to Cyc. Rather than simply running Cyc on any question, we severely restrict the connection to predicates for which we can reliably detect the predicate and focus in questions, and reliably extract answers from passages. For these questions we rely on Cyc's knowledge and reasoning to produce a result within 10 seconds (per candidate answer). These requirements are satisfied by experimentation, thus adding a semantic mapping is not a simple matter of adding terms to a mapping table, and takes some time. In 2002, we were limited to five predicates. For TREC 2003, this was expanded to 35 predicates. The most frequently used predicate in the sanity checker expansion was the *located-in* predicate, which exploits Cyc's excellent coverage of geography for sanity checking "Where" questions. As of TREC-2003, this predicate mapped only to Cyc's geographical containment relation (*inRegion*) and was only used to validate that an answer was correct (i.e. correct answers were given a confidence boost and no answers were thrown away).

The sanity checking API in 2002 had an important problem dealing with the difference between known *ranges* and known *answers* for questions. In our initial formulation of sanity checking, we envisioned using Cyc's knowledge to simply do range checking on candidate answers, and rejecting answers that were outside the range that Cyc knew for that predicate and focus type (e.g., rejecting "200 miles" as a candidate answer for the "height of Mt. Everest", since Cyc knows mountains – *the focus type* – are between 1,000 and 30,000 ft. high). In our post-TREC analysis, we found that in half of the 2002 questions for which the sanity checker was invoked, Cyc actually *knew the answer* to the question (i.e. it knew that the height of Mt. Everest – *the focus* – is 29,200 ft.). In response to this, we expanded the API to include several possible results for each predicate, focus, and candidate answer triple: answer is known to be correct, answer is known to be incorrect, answer is in range, answer is out of range, and unknown.

In our analysis of TREC-2003 performance, the Cyc post-processor had no impact on our factoid QA results. In detailed analysis, we found that it fired correctly on 4% of the questions. In slightly over two-thirds of those cases (10 questions) the correct answer was already being returned, thus the post-processing would only have impacted the confidence in those answers, which was not part of the TREC-2003 evaluation. For the other third (four questions), the sanity checker was throwing out all the answers returned by the Linguistic Query Agent, and thus that agent produced no results and the final answer came from another agent. This latter problem can be fixed by moving sanity checking to post-processing of all agents.

In detailed failure analysis, we found that the sanity checker could have been used on 13% of the questions. The 9% difference was mainly due to the inability of our question analyzer to generate an appropriate semantic form. For example, at the time of TREC-2003, our question analyzer did not recognize “In what city…?” as a located-in question (that is, it did not emit the *located-in* semantic form). Clearly this is a problem that will decrease with time or resources, as question analysis generates a semantic form using a rule-based approach. Of the questions for which the sanity checker could have fired, Cyc had an answer or range for more than half, and of those our system was returning the wrong answer in roughly half the cases. In actual numbers, it is reasonable to project that with a perfect interface, Cyc’s existing coverage of common-sense knowledge could have improved our TREC-2003 score by 2% absolute - an addition 9 questions correct - and improved our confidence (not relevant for TREC-2003) for 30 questions in total.

Our Cyc post-processor remains a proof-of-concept, however, and with significantly more resources placed on generating a semantic form in question analysis and improving Cyc's general coverage, our evaluation indicates this approach would work for as much as 40% of TREC-style factoid questions. However, a critical factor to consider is that our system without Cyc post-processing already performs relatively well on that subset – returning a correct answer for nearly 70% of those questions. This indicates that the significant effort required to get broad coverage would only net at best a 12% absolute improvement in number of correct answers over the system without Cyc.

That limitation is based on our current approach to question analysis, which is intentionally simple – we do not generate a deep semantic analysis of the question. At the moment, for example, the semantic form of a question is only generated for questions that ask for a property of an object, e.g., *"How big is Mars?"* Therefore questions like *"How far away is the moon?"* would not generate a semantic form since the correct form requires a binary predicate: distance(Earth, Moon). More significantly for analyst usage, any kind of temporal qualification would require such an expanded semantic form. It has always been our plan to start with simple semantic analysis and deepen it as necessary. We are yet in the early stages of evaluating how these kinds of improvements in semantic analysis would impact overall performance.

A critical early goal for our use of Cyc was exploitation of its reasoning capabilities. Within the knowledge representation world, Cyc takes a particular approach to representation and reasoning that ignores tractability and computational problems and uses a language (CycL) and a representation style with maximum expressiveness. This proved to be a significant barrier in practice to utilizing Cyc in our system. Early experiments would run for several days on a single question.

Cyc's ready answer to problems like this are micro-theories – basically modules of knowledge within which a reasoning task can be bounded to a significantly smaller amount of information. We performed some simple experiments to test whether this was the case and could be used by our system. We began with the question, *"How large is the everglades?"* Cyc does not know the answer to this question, however it knows that the Everglades is in Florida, it knows the size of Florida, and it has a common-sense rule expressing the constraint that a spatial region cannot exceed the size of any spatial region it is contained in (e.g. the Everglades cannot be larger than Florida). This should give us knowledge to throw away any proposed answer that is larger than the size of Florida.

Without using micro-theories, a single query for size(Everglades, *n*) took on the order of two days to produce a true/false result. Given that our system performs a query for each candidate answer, and receives on the order of 10 candidate answers from each of five answering agents, that performance is not even close to being acceptable. Limiting the query to the US Geography micro-theory, however, allows answers to come back in 2-5 seconds (two if the answer is provably true or false, and five if not).

This seems like a good result, however we were not able to find any general way to map from questions to the appropriate micro-theory; there is no "meta knowledge" in a sense that tells us, for a question we have not yet seen, what the best micro-theory is. What information from the question, and what information from a micro-theory, could impact micro-theory selection? After several days trying to make sense of the micro-theory structure of Cyc, we found it to be either inscrutable or completely unprincipled and arbitrary. We found micro-theories that appeared to contain things that were true during a particular year, things that were true in some fictional world, information relevant to a particular project, information culled from a particular source, information in some domain of interest, information that didn't seem to belong anywhere else yet, etc.

In the actually TREC-2003 system, therefore, the only reasoning used was "inheritance" of range constraints down the generalization hierarchy.

NIL Processing

To address No-Answer questions, we used a two-pronged approach. We had developed in training a confidence threshold of 0.26; questions whose confidence fell below this were classified as NIL. In addition to the threshold, for those questions for which the SKA returned an internal candidate answer, if the answer found in the corpus itself was not the same, then our system classified the question as NIL. Our performance was 7/85 precision and 7/30 recall.

We parlayed our NIL threshold logic over to List questions. There, our system generated as many candidate answers as it could of the sought type in the top 40 documents, and all those below a threshold of 0.3 were rejected.

Performance

We submitted three runs which differ primarily in the use of the SQA agent. The *IBM2003a* run used the SQA agent for Factoid questions only; *IBM2003b* used SQA for all question types, but only as support for List (it could not propose answers); and *IBM2003c* included SQA as a primary agent for all question types. In addition, other internal parameters were adjusted for *IBM2003c* to favor recall. Our scores were as in the following table. The difference in scores between runs *b* and *c* for the Definition task is entirely due to inconsistent judging since the submissions were identical but were assessed differently for 6 questions. Since we did best in IBM2003c, we will use that run as the basis for subsequent discussions.

Run	Factoid	List	Definition	Overall
IBM2003a	.298	.070	.124	.197
IBM2003b	.298	.065	.177	.210
IBM2003c	.298	.077	.175	.212

Definition Questions

The Definition task required "Who is X?" and "What is X?" questions to be answered by a list of facts or *nuggets* describing X. The guidelines did not specify what kind of facts should be included in these lists, nor how "atomic" each fact needed to be. This effectively provided a framework, but not a precise specification. To guide our effort to come up with an implementation we interpreted the framework as follows. We determined that obituaries and short encyclopedia articles were in effect answers to "Who is X?" questions, and it seemed to us that the Definition task therefore could be viewed as the gathering of the raw information that would go into such articles. Organizations and objects could be similarly described.

QA-by-Dossier

QA-by-Dossier (QbD) is a new technique which we used for the first time for Definition questions in TREC2003. It was developed last year under the ARDA AQUAINT program. The essence of the approach is that the original question is not necessarily asked directly but instead a number of *auxiliary questions* are asked instead. In doing so, the entire PIQUANT system is called recursively. The answers to the auxiliary questions are assembled into a *dossier* and returned to the user. QbD was co-developed with a more sophisticated counterpart, QA-by-Dossier-with-Constraints (QDC), in which answers to the auxiliary questions (plus possibly some others asked just for this purpose) are checked for consistency with each other. QDC is described in [Prager et al., forthcoming], but was not ready to be used for TREC2003.

The fact-gathering by QbD seemed to us to align very nicely with the requirements of the Definition task. Specific factoid questions could be formulated to find the information that is "always" present in definitional articles, and more open-ended techniques, such as our Description Agent uses, could attempt to find unanticipated facts.

Our Approach

The QA-by-Dossier approach has been adapted for TREC2003 to answer three types of definition questions where the question focus is a PERSON, an ORGANIZATION, or a THING. This classification is performed by our question analysis module, and a different set of auxiliary questions is issued for each question type. Our auxiliary question set may in principle contain multiple rounds of follow-up questions in which different subsequent questions may be issued based on answers to earlier questions. The most obvious application of this idea is to ask profession-dependent questions after the occupation of a person is discovered. For example, if the person is a composer, then one might ask what music he has written; if he is a scientist or engineer, then what he has discovered or invented; if he is a politician, then with whom has he had an affair. To support this functionality of automatically determining focus-dependent follow-up questions, we needed a mapping from occupation name to characteristic activity (i.e. verb), but we did not have this ready either for TREC2003.

The auxiliary question sets we developed therefore consisted of two kinds of questions. The first kind is the general life-cycle type of question that should be applicable to all subjects. For the second, because we did not have the ability to automatically determine follow-up questions, we decided to also ask what would be reasonably general follow-up questions, on the assumption that if the question was wholly inapplicable, no high-confidence answers would be returned and our rejection threshold would take care of eliminating any weak answers that were found.

The task instructions gave no guidance as to what kind of information would be required and/or acceptable for Definition questions. It was stated that a list of facts or properties were desired, unlike earlier year's QA-tracks which sought short noun phrases, but did not attempt to classify these facts. The Definition Question Pilot run under the ARDA/AQUAINT program was not informative for this task. Therefore, to answer PERSON-type Definition questions, we made an informal survey of obituaries and encyclopedia articles in an attempt to determine what information was considered important to give in these pieces, which were in effect implicit answers to "Who was X?" questions. We found that a common element was the set of major events in a person's life-cycle (birth, college, marriage, death), which we could clearly seek, plus some specialized facts that we hoped our general methods would locate. These articles did not typically mention single incidents in the people's lives, unless they had historical significance. Based on this analysis, we manually derived auxiliary question sets for each question type.

For PERSON-type Definition questions of type "Who is/was X?", we asked the following:

No.	Question	Agents	Answers	Threshold
P1	When was X born?	LQA	2	.3
P2	Where was X born?	LQA	1	.3
P3	When did X die?	LQA	2	.3
P4	How did X die?	PBA	variable	
P5	Who was X married to?	LQA	1	.3
P6	What occupation did X have?	LQA	1	.3
P7	What did X do?	PBA	variable	
P8	What did X invent?	LQA	1	.24
P9	What did X discover?	LQA	1	.24
P10	What did X win?	LQA	1	.3
P11	Who is X?	LQA & DSA	5	.3
P12	What compositions did X have?	LQA	1	.3
P13	X „ <famous-activity>	SKA then GQA	variable	.3
P14	X „ <known-for>	SKA then GQA	variable	.3

With multiple agents firing on multiple questions, we needed some criteria for deciding what to return as a final answer to the question. We generally used thresholds established in training and returned the best answer to each sub-question, as long as it beat the threshold. We noted in training that for questions #P1 and #P3 we often got the wrong answer in first place but correct in second place, so for these questions we returned the top two answers. This seemed to be a useful strategy knowing that the precision calculation gave an average allowance of 100 bytes per answer, and it took only about a dozen bytes to return each answer to #P1 and #P3.

The answers column indicates how many answers were returned from the other agents, assuming their confidence passed the indicated threshold. The PBA had its own internal threshold and returned a variable number of answers (including zero), all of which were accepted. When the SKA was used, it would find a variable number of text snippets; all of these that the GQA could locate in the TREC corpus with a confidence above the given threshold were accepted. The SQA was used for all questions for all types, and it had its own threshold (0.1).

We omitted questions about a person's education because our training found our system to be unreliable at such tasks. The last three P-questions need some explanation. The *compositions* question #P12 triggers our named-entity type COMPOS that is used for all kinds of titled works – books, films, poems, music, even physical artifacts such as Betsy Ross's "Stars and Stripes" or Lindbergh's "Spirit of St. Louis". Our named-entity recognizer has rules to detect compositions by phrases that are in apposition to "the film ..." or the "the book ..." etc., but by default captures any short phrase in quotes beginning with a capital letter. The particular phrasing we used in question #P12 does not commit us to a particular creative verb.

The final questions #P13-P14 are ultimately plain IR queries. The system uses the Structured Knowledge Agent's Who2 data, which has short descriptions of famous people, to find brief descriptive phrases that are then combined with the question subject into a bag-of-words which is used as a query against the TREC corpus, using a window-size of one sentence.

For THING questions, we asked the following:

No.	Question	Agents	Answers	Threshold
T1	What is X?	DSA, DFA	1	.15
T2	What is another name for X?	DFA	1	.15
T3	X ,, <WordNet gloss entry>	SKA then GQA	variable	.15

And for ORGANIZATION questions

No.	Question	Agents	Answers	Threshold
01	What does X manufacture?	LQA	1	.15
02	Where are the headquarters of X?	LQA	1	.15
03	Who is the CEO of X?	LQA	1	.15
04	What did X invent?	LQA	1	.15
05	What did X discover?	LQA	1	.15
06	What does X do?	PBA	variable	.15
07	What is X?	DSA	variable	.3

Answer Format

The required answer format was left undefined in the task guidelines. It was not clear whether, if the question was "Who was Leonardo da Vinci?", for example, "the Mona Lisa" alone would be an acceptable answer or if the fact that Leonardo painted it would have to be indicated. Arguing for being more inclusive was the sense that incompleteness might hurt, especially as we had no idea what the assessors would consider an atomic fact. Arguing against was the extra chance of including some incorrect material, along with a potential length penalty. Since generation was allowed, we decided that whenever we knew the relationship between the subject and the answer we would provide it, but as briefly as possible. It would be in the form "relationship: answer", for example, "death: 1066". For compositions we used the non-committal "work". This approach is in line with the "exact answer" requirement for Factoid questions, but was not required here. In fact we think this approach hurt us, since an informal analysis of different system's answer formats and scores, along with our experience with the ARDA AQUAINT *Definition Pilot* exercise, leads us now to think that longer, clause- or sentence-length answers are psychologically more appealing to assessors, even if the information content is the same.

Definition Task Performance

We now present our answers to some of the Definition questions to illustrate the foregoing discussion. We show the answers to #1907 "Who is Alberto Tomba, #1933 "Who was Vlad the Impaler?", #1957 "What are fractals?" and #2201 "What is Bollywood?". We tag each answer with the subquestion/agent that was responsible for it. These charts are only an approximation since in some cases multiple agents proposed an answer.

Returned nugget	Agents	Sub-Question
known as `` La Bomba ," (the bomb) for his explosive skiing style	DSA	P11
work: La Bomba	LQA	P12
most successful and popular Italian skier ever	DSA	P11
personal coach	SQA	
star	SQA	
the most famous ski racer of all time	DSA	P11
vaulting him from seventh to fourth with 1:41.48	DSA	P11
born: Alberto Tomba , Italy	LQA	P2
lawyer	SQA	
job: champion	LQA	P6
work: Slalom for Peace	LQA	P12
born: Italy	LQA	P2
work: the Bomba	LQA	P12
some World Cup skiers	LQA	P11

Results for #1907 "Who is Alberto Tomba?"

Returned nugget	Agents	Sub-Question
job: prince	LQA	P6
Dracula	SQA	
Bram Stoker's main character	LQA	P11
his victims on spikes	LQA	P11
main character was inspired by	DSA	P11
Bram Stoker	SQA	
Some historians	LQA	P11
the Romanian prince	LQA	P11
Ivan the Terrible	SQA/DSA	

Results for #1933 "Who was Vlad the Impaler?"

Returned nugget	Agents	Sub-Question
Fractal geometry is a field of mathematics founded in 1975 by Dr. Benoit Mandelbrot .	SKA then GQA	T3
Endlessly repeated fractal patterns	DFA	T1
is: patterns	DFA	T1

Results for #1957 "What are Fractals?"

Returned nugget	Agents	Sub-Question
based film industry , known as	DSA	O7
churns out nearly 200 feature films in Hindi and other Indian languages every	DSA	O7
HQ: Bombay	LQA	O2
derived	SQA	
the nickname given to Bombay	DSA	O7
movie industry	DSA	O7
more	DSA	O7
makes: capital	LQA	O1
discover: backseat	LQA	O5
invent: backseat	LQA	O4
ceo: Benjamin Gaon	LQA	O3

Results for #2201 "What is Bollywood?"

Discussion of Definition Task Results

Despite the number of obviously incomplete and completely wrong answers, there was certainly useful information returned in the examples shown above. As a rough approximation, we estimated about 2-3 nuggets per example shown – in some cases possibly more depending on the fact granularity assumed by the assessors. As it happens, we scored a total of *zero* for the shown examples. Over all the 50 Definition questions, we scored an average recall of .174, an average precision of .325, and an average F of .175.

While our answers were aligned with our interpretation of the NIST evaluation framework, our scores indicate they were not aligned with NIST's interpretation. Upon a subjective consideration of our results, however, we believe, it is obvious that the nuggets PIQUANT produced, often provided suitable information in response to the definition questions (e.g., Alberto Tomba certainly was a famous ski racer). Based on the evaluation framework provided by NIST, we argue that it is possible to come up with actual evaluation guidelines that conform to this framework but produce drastically different outcomes.[1] Although our analysis shows that our own divergence from NIST in interpretation played as big a role in the final score for this subtask as our system's errors, we believe less potential for such variability in the interpretation of the evaluation framework for definition questions would better serve the TREC QA track.

Summary

Our analysis of the contribution of Cyc this year showed that the major limiting factor is still in the area of coverage. Major manual effort is required both to generate appropriate semantic forms and to map to Cyc's predicates, and also to add instance information into Cyc. With the current state of the system, Cyc helps more to improve our answer confidences (not a part of the evaluation this year) than to get answers right.

The major novelty in our system this year was the implementation of QA-by-Dossier to answer Definition questions. Here, a collection of predetermined factoid questions are asked about the subject in order to gather facts that seem to be typically mentioned in definitional articles in newspapers and reference works. An advantage of this method over others which locate definitional syntactic constructs is that our system "knows" the nature of the relationship of the retrieved item to the subject. In the evaluation, we felt that our system had performed relatively well according to our expectations of what was required, but we were very disappointed to find that the NIST assessors had different opinions regarding acceptable answers.

[1] To illustrate this point, we developed a set of evaluation guidelines based on our interpretation of the framework put forth by NIST. These guidelines (admittedly) coincide with the principles we used in developing the QA-by-Dossier agent used in answering definition questions. Precision was calculated using the 100-byte-per-nugget allowance, following the TREC 2003 formula. Recall was approximated by pooling the NIST assessors' nuggets with additional facts found by our system. The self-assessed averages were .385 for recall, .583 for precision and .387 for F, significantly different from the NIST-assessed scores.

The task guidelines provided a framework for answering Definition questions and their subsequent evaluation, but both were left open to some interpretation. We made the assumptions that any correct fact found about the subject of the question would be at least "okay", and that all facts that are typically reported in obituaries and short encyclopedia articles would, by our definition, be "vital". The NIST assessors came up with a different instantiation of the framework for their evaluation task – and to this day we do not understand exactly what that was – and our results have shown that these different instantiations give rise to significantly different scores for the same answer set.

In this paper, we have provided our own detailed evaluation criteria for three types of Definition questions, which we hope will generate useful discussions in outlining more specific evaluation guidelines for the next TREC QA track. Similarly to the Factoid subtask, we believe the perceived output quality of a question answering system on the Definition subtask strongly depends on the user model and expectations. This has proved difficult to capture in evaluation, and this year the difficulty was compounded by the introduction of the concept of "vital" nuggets. Given our own interpretation presented above that was consistent with the initial guidelines, we believe that the concept is not well-defined and simply dropping it in favor of less restrictive judging to allow for more "okay" nuggets (based on pooling) would have made the evaluation more realistic and less controversial.

Acknowledgments

This work was supported in part by the Advanced Research and Development Activity (ARDA)'s Advanced Question Answering for Intelligence (AQUAINT) Program under contract number MDA904-01-C-0988.

References

J. Chu-Carroll, K. Czuba, J. Prager, A. Ittycheriah. 2003a. In Question Answering, Two Heads are Better than One. *Proceedings of the Human Language Technologies Conference.* Pages 24-31.

J. Chu-Carroll, J. Prager, C. Welty, K. Czuba, D. Ferrucci. 2003b. A Multi-Strategy and Multi-Source Approach to Question Answering. *Proceedings of TREC2002*. Pages 281-288.

A. Ittycheriah and S. Roukos, "IBM's Statistical Question Answering System - TREC-11", The Eleventh Text REtrieval Conference Proceedings, Trec 2002, pp. 273-280, 2002.

J.M. Prager, E.W. Brown, A. Coden, and D. Radev. "Question-Answering by Predictive Annotation". Proceedings of SIGIR 2000, pp. 184-191, Athens, Greece.

J. Prager, D. Radev, K. Czuba. 2001. "Answering what-is questions by virtual annotation." *Proceedings of Human Language Technologies Conference.* Pages 26-30.

J.M. Prager. "In Question-Answering, Hit-List Size Matters", IBM T.J. Watson Research Center Research Report #RC22297, Jan 2002.

J. Prager, J. Chu-Carroll, E. Brown, K. Czuba. 2003. Question answering using predictive annotation. In *Advances in Question Answering*, forthcoming.

J.M. Prager, J. Chu-Carroll and K. Czuba, "A Multi-Agent Approach to using Redundancy and Reinforcement in Question Answering" in *New Directions in Question-Answering*, Maybury, M. (Ed.), AAAI Press, forthcoming.

D. Ravichandran, E. Hovy. 2002. Learning surface text patterns for a question answering system. *Proceedings of the 40th Annual Meeting of the ACL.* Pages 41-47.

From TREC to DUC to TREC Again

John M. Conroy[*] Daniel M. Dunlavy[†] Dianne P. O'Leary[‡]

February 3, 2004

1 Introduction

The Document Understanding Conference (DUC) uses TREC data as a test bed for algorithms for single and multiple document summarization. For the 2003 DUC task of choosing relevant and novel sentences, we tested a system based on a Hidden Markov Model (HMM). In this work, we use variations of this system on the tasks of the TREC Novelty Track for finding relevant and new sentences.

Our complete information retrieval system couples a query handler, a document clusterer, and a summary generator with a convenient user interface. For the TREC tasks, we use only the summarization part of the system, based on an HMM, to find relevant sentences in a document and we use linear algebra techniques to determine the new sentences among these.

For the tasks in the 2003 TREC Novelty Track we used a simple preprocessing of the data which consisted of term tokenization and SGML DTD processing. Details of each of these methods are presented in Section 2.

The algorithms for choosing relevant sentences were tuned versions of those presented by members of our group in the past DUC evaluations (see [5, 8, 15] for more details). The enhancements to the previous system are detailed in Section 3.

Several methods were explored to find a subset of the relevant sentences that had good coverage but low redundancy. In our multi-document summarization system, we used the QR algorithm on term-sentence matrices. For this work, we explored the use of the singular value decomposition as well as two variants of the QR algorithm. These methods are defined in Section 4. The evaluation of these methods is discussed in Section 5.

[*] IDA/Center for Computing Sciences, conroy@super.org

[†] University of Maryland, ddunlavy@cs.umd.edu

[‡] University of Maryland, oleary@cs.umd.edu

2 Preprocessing

2.1 Tokenization

The tokenization was quite simple. First the text was converted to lower case. All contiguous strings of characters taken from the set {a,b,...,z} were terms except for those matched on a short list of stop words.

2.2 Parsing Files using DTDs

Using the SGML document type definition (DTD) for a document allowed us to determine the set of all possible SGML tags that exist in documents of that type. Using these tag sets, we distinguished which sentences 1) were candidates for relevant sentences, 2) were not candidates for relevant sentences but which contained key terms or phrases that would aid in identifying relevant sentences, and 3) contained no useful information for the task of extracting relevant sentences. We created a new attribute, *stype*, for the SGML tag denoting a sentence boundary, <s>, in order to denote each of these three types of sentences. The possible values for this new attribute are 1, 0, and -1, respectively. Table 1 presents the values of *stype* used for sentences embedded into the SGML tags encountered in the several types of documents used in the evaluation.

Choosing to embed information into the document itself instead of creating a processing module in our algorithm allowed us flexibility in using the information throughout the various stages of our system. Furthermore, it will allow us to expand the types of sentence classification without changing the code.

3 Finding Relevant Sentences

An HMM, in contrast to a naive Bayesian approach ([1], [12]), has fewer assumptions of independence. In particular, it does not assume that the probability that sentence i is relevant is independent of whether sentence $i-1$ is relevant. In the HMM developed for this evaluation, we used a joint distribution for the features set which varied based upon the position in the document.

All of the features used by the HMM were based upon the terms (as defined in Section 2.1) found in a sentence. The features for the HMM were as follows:

- number of signature terms, n_{sig}, in a sentence—value is $o_1(i) = \log(n_{sig} + 1)$.
- number of subject tokens, n_{subj}, in a sentence—value is $o_2(i) = \log(n_{subj} + 1)$.
- position of the sentence in the document—built into the state-structure of the HMM.

The signature terms are the terms that are more likely to occur in the document (or document set) than in the corpus at large. To identify these terms, we used the log-likelihood statistic suggested by Dunning [9] and used first in summarization by Lin and Hovy [13]. The statistic is equivalent to a mutual information statistic and is based on a 2-by-2 contingency table of counts for each term.

The subject terms are a special subset of the signature terms. These are the signature terms that occur in sentences with *stype* $= 0$, for example, headline and subject heading sentences.

File	DTD	Filename	SGML Tag	*stype*
APW* NYT* XIE*	ACQUAINT	acquaint.dtd	<TEXT> <HEADLINE>	1 0 0
FBIS*	FBIS	fbis.dtd	<TEXT> <TI> <H1>, . . . , <H8>	1 0 0
FR*	Federal Register	fr.dtd	<TEXT> <SUMMARY> <SUPPLEM> <FOOTNOTE> <DOCTITLE>	1 1 1 1 0
FT*	Financial Times	ft.dtd	<TEXT> <HEADLINE>	1 0
LA*	LA Times	latimes.dtd	<TEXT> <HEADLINE> <SUBJECT> <GRAPHIC>	1 0 0 0

Table 1: Mapping SGML tags to *stype* values. All tags not shown but allowed by each DTD are assigned *stype* $= -1$.

The features were normalized component-wise to have mean zero and variance one. In addition, the features for sentences with *stype* 0 and -1 were coerced to be -1, which forced these sentences to have an extremely low probability of being selected as relevant sentences.

An HMM handles the positional dependence, dependence of features, and Markovity. (For more details about HMMs, see [2] and [14].) The model we proposed has $2s + 1$ states, with s relevance states and $s + 1$ non-relevance states. A picture of the Markov chain is given in Figure 1. Note that we allowed hesitation only in non-relevance states and skipping of states only from relevance states. This chain was designed to model the extraction of up to $s - 1$ lead relevant sentences and an arbitrary number of supporting relevant sentences. Using training data, we obtained a maximum-likelihood estimate for each transition probability and this formed an estimate, M, for the transition matrix for our Markov chain, where element (i, j) of M is the estimated probability of transitioning from state i to state j.

Associated with each state i is an output function, $b_i(O) = Pr(O|state\ i)$, where O is an observed vector of features. We made the simplifying assumption that the features were multivariate normal. The output function for each state was estimated by using the training data to compute the maximum-likelihood estimate of its mean and covariance matrix. We estimated $2s + 1$ means, but assumed that all of the output functions shared a common covariance matrix.

Training for the HMM was straightforward given marked data. Since the states of the HMM were known in the training data, creating the model simply amounted to computing the maximum likelihood statistics given the counts.

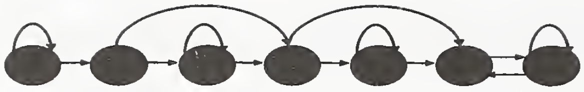


Figure 1: Markov Chain to Extract 2 Lead Sentences and Supporting Sentences

In particular, the training data helped determine the number of states for the HMM. The upshot was that a state space consisting of thirteen states (six relevance states and seven non-relevance states) was optimal given TREC 2002 data. (For Tasks 3 and 4, when some of the TREC 2003 data is allowed for training, the optimal number of states was three— one relevance state and two non-relevance states.)

With this model we computed $\gamma_j(i)$, the probability that sentence j corresponded to state i. We computed the probability that a sentence was a relevant sentence by summing $\gamma_j(i)$ over all even values of i, values corresponding to relevance states. This posterior probability, which we define as g_j, was used to select the most likely relevant sentences. We refer the reader to [4] for details.

This posterior probability was used to select which sentences were likely to be relevant. The selection algorithm attempted to choose the number of sentences so that the expected F_1 score was maximized. The approximate F_1 score was computed based on the expected precision, $E(P)$, and expected recall, $E(R)$, as follows:

$$\widehat{F_1} = \frac{2E(P)E(R)}{E(P)+E(R)}$$

where

$$E(P) = \frac{\sum_{t \epsilon S} g_t}{|S|}$$

where $|S|$ is the cardinality of the set S of sentences selected, and

$$E(R) = \frac{\sum_{t \epsilon S} g_t}{\sum_t g_t}.$$

The set S was chosen by selectively choosing the sentences in decreasing order of their probability of being a relevant sentence. The score $\widehat{F_1}$ was then computed and the set S increased as long as $\widehat{F_1}$ increased.

Another feature that was considered previously (during the DUC evaluation) for our system was based on the query terms derived from the topic descriptions. We attempted to use this information in two ways. The first was to simply add an additional feature to the HMM. This approach actually decreased the precision of the system. The second method we considered used the derived query terms in conjunction with an information retrieval (IR) system to rank each document. The hope was to use a combination of these IR scores and the HMM sentence scores to generate the relevant sentences. Unfortunately, the IR scores did not correlate strongly with the likelihood that a document's sentence would be chosen as relevant. We hypothesize that since the document collection only contains documents relevant to the query, the topic description terms do not add any additional information. Clearly, more analysis is required to determine why the topic descriptions did not help in the generation of relevant sentences.

4 Finding New Sentences

To choose a subset of the candidate relevant sentences to produce new sentences we experimented with three algorithms: a QR decomposition, a pivoted QR decomposition, and the singular value decomposition (SVD). These methods all work on the term–sentence matrix, A, where A_{ij} is 1 if term i occurs in relevant sentence j. Before applying the sentence selection algorithms, the columns of A were normalized; the Euclidean length of a column was set equal to the probability that the corresponding sentence was indeed relevant. For Tasks 2 and 4 these probabilities were 1 since the relevant sentences were given, while for Task 3, the probability was equal to the score produced by the HMM for that sentence, g_j.

The SVD was used as an optional preprocessing to the matrix A before applying the QR or pivoted QR. The SVD is a matrix factorization method that returns the best low rank approximation for a matrix. The idea of using such preprocessing was borrowed from information retrieval, where the SVD is the basis for Latent Semantic Indexing (LSI) [6]. LSI has been shown to be quite useful in uncovering latent relationships between columns of term–document matrices, thus allowing for more conceptual rather than exact term matching for query-based document retrieval (see [3, 7]). The goal was to use the SVD to help uncover latent redundancy amongst the relevant sentences. Given $A \in \mathbb{R}^{m \times n}$, let $\bar{k} = \min(m, n)$. Then the SVD of A [10] is defined by orthogonal vectors u_i of length m, and orthogonal vectors v_i of length n, $i = 1, \ldots, \bar{k}$, and nonnegative numbers $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\bar{k}} \geq 0$, such that

$$A = \sum_{i=1}^{\bar{k}} \sigma_i u_i v_i^T.$$

As described in [11], the rank-k matrix ($k \leq \bar{k}$) that gives the optimal approximation to a given matrix A (as measured in the 2-norm or Frobenius norm) is

$$\tilde{A}_{SVD} = \sum_{i=1}^{k} \sigma_i u_i v_i^T.$$

The rank k was determined empirically for this application and corresponds to a preassigned small error.

A QR decomposition, with or without pivoting, can be applied either to the weighted term-sentence matrix $A_w = A$ or the lower rank approximation of $A_w = \tilde{A}_{SVD}$. The QR decomposition was used to determine whether a sentence should be considered new or redundant. In the QR factorization a sentence was considered redundant if the vector corresponding to it was of small weight, say less than τ, a predefined threshold. Specifically, we developed the following algorithms for selecting new, or novel, sentences.

Algorithm 4.1 (Thresholded QR Decomposition) *Suppose A_w has m rows and n columns: i.e., the document has m unique terms and n sentences. The following iteration constructs a matrix Q with columns q_i, a matrix R with nonzero elements r_{ji}, and an ordering for the columns in an array Index.*

For $i = 1, 2, \ldots, \min(m, n)$,

> *Among the remaining columns of A_w, choose the first column with 2-norm greater than τ. Denote this column by a_ℓ, where ℓ is its index in the original matrix.*
>
> *Set $Index_i = \ell$.*
>
> *Set $q_i = a_\ell / \|a_\ell\|$.*
>
> *Update the remaining columns of A_w to make them orthogonal to the chosen column: for each unchosen column a_j, set $r_{ji} = a_j^T q_i$ and set $a_j = a_j - r_{ji} q_i$.*

The set of "new" sentences of size k contains sentences $Index_1, \ldots, Index_k$.

The standard implementation of the pivoted QR decomposition is a "Gram-Schmidt" process and was used to select new sentences as follows.

Algorithm 4.2 (Pivoted QR Decomposition) *Suppose A_w has m rows and n columns: i.e., the document has m unique terms and n sentences. The following iteration constructs a matrix Q with columns q_i, a matrix R with nonzero elements r_{ji}, and an ordering for the columns in an array Index.*

For $i = 1, 2, \ldots, \min(m, n)$,

> *Among the remaining columns of A_w, choose the column with maximal norm. Denote this column by a_ℓ, where ℓ is its index in the original matrix.*
>
> *Set $Index_i = \ell$.*
>
> *Set $q_i = a_\ell / \|a_\ell\|$.*
>
> *Update the other columns of A_w to make them orthogonal to the chosen column: for each unchosen column a_j, set $r_{ji} = a_j^T q_i$ and set $a_j = a_j - r_{ji} q_i$.*

The set of "new" sentences of size k contains sentences $Index_1, \ldots, Index_k$.

5 Results

For Task 1 the HMM used for TREC was trained using the marked relevant and new sentences in the Novelty data from TREC 2002. Specifically, for Task 1 three models were built. The first focused on only the novel sentences. To strengthen the model further a subset of the novel sentences were chosen by hand for 24 of the document sets. This process removed many sentences that did not convey relevant information when taken out of their original context. These data were then used to build an HMM to score the sentences and determine which features should be included. This was the model that our group used in DUC 2003 and in the entries labeled *ccsummeoqr* and *ccsummeosvd* for Task 1.

A second model used the subset of the data from the LA Times articles only. It was hoped that this subset was more representative of the TREC 2003 data than the complete collection from TREC 2002. One entry for Task 1 used this model and was labeled *ccsumlaqr.*

The third model was based on all of the relevant sentences from TREC 2002. For Task 1 the given relevant sentences were used to build the HMM and the entries were labeled *ccsumrelqr* and *ccsumrelsvd.*

Note that for Task 1 the suffixes "svd" and "qr" denote the results using a truncated SVD followed by a pivoted QR and those using just a pivoted QR, respectively.

All three models for extracting relevant sentences performed comparably and unfortunately, generated fewer sentences than the human judges did in 2003, since they were predicting relevant sentences based upon the smaller number of sentences selected by the judges in 2002.

For the task of selecting the new sentences given a list of putative relevant sentences and only TREC 2002 data, it appears that the preprocessing by using a truncated SVD was not worthwhile. The two SVD methods gave median F_1 scores below those given by the pivoted QR method. The results of extracting relevant and new sentences for Task 1 are presented in Tables 2 and 3, respectively.

Run	*Median Precision*	*Median Recall*	*Median F_1*	*Median Rank*
ccsumlaqr	64	22	32	34
ccsummeoqr	69	19	29	36
ccsummeosvd	69	19	29	36
ccsumrelqr	66	21	31	34
ccsumrelsvd	66	21	31	34

Table 2: Performance of CCSUM on Task 1: Relevant Sentences; 55 Total Entries

Run	*Median Precision*	*Median Recall*	*Median F_1*	*Median Rank*
ccsumlaqr	48	20	27	27
ccsummeoqr	46	19	24	30
ccsummeosvd	48	17	23	31
ccsumrelqr	48	21	26	27
ccsumrelsvd	47	18	25	29

Table 3: Performance of CCSUM on Task 1: New Sentences; 55 Total Entries

Run	*Median Precision*	*Median Recall*	*Median F_1*	*Median Rank*
ccsum2svdpqr	70	90	78	19
ccsumt2svdqr	69	92	80	15
ccsumt2pqr	70	95	80	9
ccsumt2qr	69	92	80	15

Table 4: Performance of CCSUM on Task 2: New Sentences; 45 Total Entries

In Task 2 we were given the relevant sentences and had to determine the new sentences. We submitted 4 approaches: an SVD followed by a pivoted QR (*ccsum2svdpqr*), an SVD followed by a thresholded QR (*ccsumt2svdqr*), a pivoted QR (*ccsumt2pqr*), and a thresholded QR (*ccsumt2qr*). The thresholded QR was added for this task, since all the relevant sentences were known and a thresholded QR was thought to more closely simulate how a human would perform the task by scanning the sentences in order and deleting those that were redundant. All of these methods performed comparably and tended to give relatively high recall (see Table 4). It is interesting to note that the pivoted QR appears to have a considerably higher median rank for F_1 relative to the peer systems, despite its median precision, recall and F_1 being comparable with the other 3 entries. Overall, our system performed comparably with the best systems, which also generated F_1 scores around 0.80.

In Tasks 3 and 4 we were given the relevant and new sentences for the first 5 documents of each of the document sets. We realized after submitting our results that we should not have included any sentences from these first 5 documents, even if they were correct, since the scoring script was keying on only documents from the last 20 in each document set. As a result of our submission error our precision numbers were penalized. Therefore, for Tasks 3 and 4, we present here tables giving the corrected results (Tables 6 and 8) as well as the results of those submitted (Tables 5 and 7). The former are a true reflection of the performance of our submitted methods, while the latter is a "monument" reminding us to read submission rules carefully!

Run	*Median Precision*	*Median Recall*	*Median F_1*	*Median Rank*
ccsum3pqr	41	93	56	24
ccsum3qr	41	93	56	24
ccsum3svdpqr	41	93	56	24

Table 5: Performance of CCSUM on Task 3:Relevant Sentences; 38 Total Entries

Run	*Median Precision*	*Median Recall*	*Median F_1*	*Median Rank*
ccsum3pqr	51	93	66	11
ccsum3qr	51	93	66	11
ccsum3svdpqr	51	93	66	11

Table 6: Corrected Performance of CCSUM on Task 3:Relevant Sentences; 38 Total Entries

For Task 3 we were given the relevant and new sentences for the first 5 documents of each document set. We built a single HMM based on these relevant sentences for our methods using a pivoted QR (*ccsum3pqr*), a thresholded QR (*ccsum3qr*), and an SVD followed by a pivoted QR (*ccsum3svdpqr*). Consequently, the precision, recall, and rank for our three entries were the same (see Table 6). Our method for estimating the length based upon expected F1 score appeared to want to "go long,... very long," thus, giving a median recall of 93. In contrast, the models used in Task 1 generated too few sentences. Still the overall F_1 score was 66 (for the corrected submissions), which was comparable to the best scoring systems as given in the overview of the results of the Novelty Track evaluation (see Figure 8 in [16]).

For the second part of Task 3, selecting the new sentences based on the predicted relevant sentences, the method of pivoted QR nudged out the thresholded QR and the SVD followed by a pivoted QR (see Table 8).

Run	*Median Precision*	*Median Recall*	*Median F_1*	*Median Rank*
ccsum3pqr	26	91	41	21
ccsum3qr	25	94	38	24
ccsum3svdpqr	26	89	41	22

Table 7: Performance of CCSUM on Task 3:New Sentences; 38 Total Entries

Run	*Median Precision*	*Median Recall*	*Median F_1*	*Median Rank*
ccsum3pqr	33	91	48	11
ccsum3qr	30	94	44	15
ccsum3svdpqr	32	89	47	12

Table 8: Corrected Performance of CCSUM on Task 3:New Sentences; 38 Total Entries

For Task 4 we were given all relevant sentences and the new sentences from the first 5 documents in each set. Here, we attempted to optimize the thresholds for both the truncated SVD and the pivoted QR based on the given new sentences. In Table 9, *ccsum4spq001* refers to the entry with a threshold set to 0.001 while *ccsumt4sqr01* refers the the entry using a threshold of 0.01. The smaller threshold resulted in fewer new sentences, although it did not increase the median precision and did reduce the recall, which resulted in a lower F_1 score. Of the group of entries, the pivoted QR did the best. Its shining virtue was that it did not miss a single new sentence; however it did generate nearly twice the number that the judges did. Also, the precision of these methods is generally lower than in Task 2, which indicates that the tuning of the model based on the new sentences from the first 5 documents did not help.

Run	*Median Precision*	*Median Recall*	*Median F_1*	*Median Rank*
ccsum4spq001	67	98	80	9
ccsum4svdpqr	68	92	79	17
ccsumt4pqr	67	100	80	7
ccsumt4qr	72	92	82	9
ccsumt4sqr01	67	92	78	15

Table 9: Corrected Performance of CCSUM on Task 4:New Sentences; 41 Total Entries

References

[1] C. Aone, M. Okurowski, J. Gorlinsky, and B. Larsen. “A Scalable Summarization System Using Robust NLP”. In *Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization*, pages 66–73, 1997.

[2] L. Baum, T. Petrie, G. Soules, and N. Weiss. “A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains”. *Ann. Math. Stat.*, 41:164–171, 1970.

[3] M. W. Berry, S. T. Dumais, and G. W. O'Brien. Using Linear Algebra for Intelligent Information Retrieval. *SIAM Rev.*, 37(4):573–595, 1995.

[4] J. Conroy and D. O'Leary. "Text Summarization via Hidden Markov Models and Pivoted QR Matrix Decomposition". Technical report, University of Maryland, College Park, Maryland, March, 2001.

[5] J. Conroy, J. Schlesinger, D. O'Leary, and M. Okurowski. "Using HMM and Logistic Regression to Generate Extract Summaries for DUC". In *DUC 01 Conference Proceedings*, 2001.

[6] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[7] S. T. Dumais. Improving the Retrieval of Information from External Sources. *Behavior Research Methods, Instruments, and Computers*, 23(6):229–326, 1991.

[8] D. Dunlavy, J. Conroy, J. Schlesinger, S. Goodman, M. Okurowski, D. O'Leary, and H. van Halteren. "Performance of a Three-Stage System for Multi-Document Summarization". In *DUC 03 Conference Proceedings*, 2003.

[9] T. Dunning. "Accurate Methods for Statistics of Surprise and Coincidence". *Computational Linguistics*, 19:61–74, 1993.

[10] G. Golub, V. Klema, and G. Stewart. "Rank Degeneracy and Least Squares Problems". Technical Report No. TR-456, University of Maryland, College Park, Maryland, 1976.

[11] G. Golub and C. V. Loan. *Matrix Computations.* The Johns Hopkins University Press, Baltimore, MD, 3rd edition, 1996.

[12] J. Kupiec, J. Pedersen, and F. Chen. "A Trainable Document Summarizer". In *Proceedings of the 18th Annual International SIGIR Conference on Research and Development in Information Retrieval*, pages 68–73, 1995.

[13] C. Lin and E. Hovy. "The Automatic Acquisition of Topic Signatures for Text Summarization". In *Proceedings of the 18th International Conference on Computational Liguistics (COLLING 2000)*, 2000.

[14] L. Rabiner. "A Tutorial on Hidden Markov Models and Selected Applications". *Proceedings of the IEEE*, 77:257–285, 1989.

[15] J. Schlesinger, M. Okurowski, J. Conroy, D. O'Leary, A.Taylor, J. Hobbs, and H. Wilson. "Understanding Machine Performance in the Context of Human Performance for Multi-document Summarization". In *DUC 02 Conference Proceedings*, 2002.

[16] I. Soboro and D. Harman. "Overview of the TREC 2003 Novelty Track". In *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*, to appear.

Generic Text Summarization using Wordnet for Novelty and Hard

Ganesh Ramakrishnan, Kedar Bellare, Chirag Shah, Deepa Paranjpe
{hare,kedar,chirag,adeepa}@cse.iitb.ac.in
Dept. of Computer Science and Engg.,
Indian Institute of Technology, Mumbai, India

Abstract

This paper presents a Random Walk approach to text summarization using the Wordnet for text representation. For the HARD track, the specified corpus is indexed using a standard indexing engine - lucene and the initial passage set is retrieved by querying the index. The collection of passages is considered to be a document. In Novelty, the documents are as directly supplied by NIST. In either case, the document is used to extract a "relevant" sub-graph from the wordnet graph. Weights are assigned to each node of this sub-graph using a strategy similar to the Google Page-ranking algorithm. A matrix of sentences against the nodes of the sub-graph is created and principal component analysis is performed to extract the sentences for the summary. Our approach is not specific to any particular genre of documents, such as news articles. We use the semantics in the document rather than using the more common statistical measures like term frequency and inverse-document frequency.

Keywords : text summarization, wordnet, PCA, synsetranking

1 Introduction

Text summarization is "the process of distilling the most important information from a source (or sources) to produce an abridged version for a particular user (or users) and task (or tasks)". Text summarization finds varied applications in today's world. Some notable ones are: search engine hit summarization (summarizing the information in a hit list retrieved by some search engine); physicians' aids (to summarize and compare the recommended treatments for a patient); generating the blurb of a book ; and so on. Building automated summarizers can be very helpful in many such applications and saves a lot of manual work. An extract is a summary containing only material from the text and involves no natural language generation. Given a piece of text, our aim is to select the most "representative" sentences which will form the summary.

A good summary should ideally have the following features

1. Relevance to the text
2. Informativeness
3. Conciseness

1.1 Our approach

For the HARD track, we index the document collection using the lucene indexer. The query is fired to the search component of lucene and a set of relevant passages are extracted after lucene querying. These passages are combined to form a document which is further subject to summarization. In Novelty, the documents are as directly supplied by NIST.

We use wordnet to understand the links between different parts of the document; Subsequently, we extract the portion of the wordnet graph which is most relevant and contains the main ideas present in the document. We do this by starting from the words in the document as the leaves and traverse the wordnet links upward towards synsets that are more general at each level. The idea of first getting a global view of the whole document, even before beginning to rank sentences is what differentiates our approach from the rest. After obtaining the relevant sub-graph we rank its nodes (synsets of wordnet) with random walks on wordnet and use them to compute the importance of individual sentences.

This idea of getting an overall picture of the document before picking sentences makes our approach generic (not necessarily tailored to give good results only on a specific class of documents).

The last, and the most crucial phase of our approach is to actually pick out sentences that will form the summary. It is important that we do not pick two sentences with a similar meaning and only pick those sentences which represent the text to the maximum extent possible. For this purpose, we use the method of **Principal Component Analysis** and dump sentences most higly correlated with each principal component as the summary sentences. This is where, we believe, our approach captures human thinking. It is natural for a human to identify very similar sentences from the text and pick only one of them.

2 Conclusion

There is an ever-increasing need for better automatic text summarization systems with the explosion in the amount of information available the user. Most existing text summarization systems analyze a text statistically and linguistically, determine important sentences, and generate an extract for it. The linguistic features used are generally specific for a particular kind of document which make existing systems very specialized. We propose an algorithm for a generic text summarizer which selects sentences on the basis of their semantic content and its relevance to the main ideas contained in the text. We use Wordnet to abstract the ideas contained in the text so that sentences are selected on the basis of their meaning, and not on the presence of some keywords or frequently-occuring term.

Passage Scoring for Question answering via Bayesian inference on lexical relations

Deepa Paranjpe, Ganesh Ramakrishnan, Sumana Srinivasan
{adeepa,hare}@cse.iitb.ac.in, sumana@it.iitb.ac.in
Dept. of Computer Science and Engg.,
Indian Institute of Technology, Mumbai, India

Abstract

Many researchers have used lexical networks and ontologies to mitigate synonymy and polysemy problems in Question Answering (QA), systems coupled with taggers, query classifiers, and answer extractors in complex and ad-hoc ways. We seek to make QA systems reproducible with shared and modest human effort, carefully separating knowledge from algorithms. To this end, we propose an aesthetically "clean" Bayesian inference scheme for exploiting lexical relations for passage-scoring for QA . The factors which contribute to the efficacy of Bayesian Inferencing on lexical relations are *soft word sense disambiguation*, *parameter smoothing* which ameliorates the data sparsity problem and *estimation of joint probability over words* which overcomes the deficiency of naive-bayes-like approaches.

1 Introduction

This paper describes an approach to probabilistic inference using lexical relations, such as expressed by a WordNet, an ontology, or a combination, with applications to passage-scoring for open-domain question answering (QA).

The use of lexical resources in Information Retrieval (IR) is not new; for almost a decade, the IR community has considered the use of natural language processing tcchniques (Lewis and Jones, 1996) to circumvent synonymy, polysemy, and other barriers to purely string-matching search engines. In particular, a number of researchers have attempted to use the English WordNet to "bridge the gap" between query and response. Interestingly, the results have mostly been inconclusive or negative (Fellbaum, 1998a). A number of explanations have been offered for this lack of success, some of which are

- presence of unnecessary links and absence of necessary links in the WordNet (Fellbaum, 1998b),
- hurdle of Word Sense Disambiguation (WSD) (Sanderson, 1994)
- ad-hocness in the distance and scoring functions (Abe et al., 1996).

2 Proposed approach

2.1 An inferencing approach to QA

Given a question and a passage that contains the answer, how do we correlate the two ? Take for example, the following question

> What type of animal is Winnie the Pooh?

and the answer passage is

> A Canadian town that claims to be the birthplace of Winnie the Pooh wants to erect a giant statue of the famous bear; but Walt Disney Studios will not permit it.

It is clear that there is a linkage between the question word *animal* and the answer word *bear*. That the word *bear* occurred in the answer, in the context of Winnie, means that there was a hidden "cause" for the occurrence of *bear*, and that was the concept of { animal}.

In general, there could be multiple words in the question and answer that are connected by many hidden causes. The causes themselves may have hidden causes associated with them. These causal relationships are represented in ontologies and WordNets. The familiar English WordNet, in particular, encodes relations between words and concepts. For instance WordNet gives the *hypernymy* relation between the concepts { animal} and { bear}.

2.2 WordNet

WordNet (Fellbaum, 1998b) is an online lexical reference system in which English nouns, verbs, adjectives and adverbs are organized into synonym sets or *synsets*, each representing one underlying lexical concept. Noun synsets are related to each other through *hypernymy* (generalization), *hyponymy* (specialization), *holonymy* (whole of) and *meronymy* (part of) relations. Of these, (*hypernymy*, *hyponymy*) and (*meronymy*,*holonymy*) are complementary pairs.

The verb and adjective synsets are very sparsely connected with each other. No relation is available between noun and verb synsets. However, 4500 adjective synsets are related to noun synsets with *pertainyms* (pertaining to) and *attra* (attributed with) relations.

For example, the synset { dog, domestic_dog, canis_familiaris} has a hyponymy link to { corgi, welshcorgi} and meronymy link to { flag} ("a conspicuously marked or shaped tail"). While the hyponymy link helps us answer the question (TREC#371) "A corgi is a kind of what?", the meronymy connection here is perhaps more confusing than useful: this sense of *flag* is rare.

2.3 Inferencing on lexical relations

It is surprisingly difficult to make the simple idea of bridging passage to query through lexical networks perform well in practice. Continuing the example of Winnie the bear (section §2.1), the English WordNet has five synsets on the path from *bear* to *animal*: {carnivore...}, {placental_mammal...}, {mammal...}, {vertebrate..}, {chordate...}.

Some of these intervening synsets would be extremely unlikely to be associated with a corpus that is not about zoology; a common person would more naturally think of a bear as a kind of animal, skipping through the intervening nodes.

It is, however, dangerous to design an algorithm which is generally eager to skip across links in a lexical network. E.g., few QA applications are expected to need an expansion of "bottle" beyond "vessel" and "container" to "instrumentality" and beyond. Another example would be the shallow verb hierarchy in the English WordNet, with completely dissimilar verbs within very few links of each other. There is also the problem of missing links.

Another important issue is *which 'hidden causes'* (synsets) should be inferred to have caused words in the text. This is a classical problem called word sense disambiguation (WSD). For instance, the word *dog* belongs to 6 noun synsets in WordNet. Which of the 6 synsets should be treated as the 'hidden cause' that generated the word *dog* in the passage could be inferred from the fact that *collie* is related to *dog* only through one of the latter's senses - it's sense as {dog, domestic dog, Canis_familiaris}. But this problem of finding the 'appropriate' hidden causes, in general, in non-trivial. Given that state-of-the-art WSD systems perform not better than 74% (Sanderson, 1994) (Lewis and Jones, 1996) (Fellbaum, 1998b), in this paper, we use a probabilistic approach to WSD - called 'soft WSD' (Pushpak,) ; hidden nodes are considered to have probabilistically 'caused' words in the question and answer or in other words, causes are probabilistically 'switched on'.

Clearly, any scoring algorithm that seeks to utilize WordNet link information must also *discriminate* between them based (at least) on usage statistics of the connected synsets. Also required is an estimate of the likelihood of instantiating a synset into a token because it was "activated" by a closely related synset. We find a Bayesian belief network (BBN) a natural structure to encode such combined knowledge from WordNet and corpus.

2.4 Bayesian Belief Network

A Bayesian Network (Heckerman, 1995) for a set of random variables $X = \{X_1, X_2, \ldots, X_n\}$ consists of a directed acyclic graph (DAG) that encodes a set of conditional independence assertions about variables in X and a set of local probability distributions associated with each variable. Let $\mathbf{Pa}_i$ denote the set of immediate parents of X_i in the DAG, and $\mathbf{pa}_i$ a specific instantiation of these random variables.

The BBN encodes the joint distribution $\Pr(x_1, x_2, \ldots, x_n)$ as

$$\Pr(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} \Pr(x_i|\mathbf{pa}_i) \quad (1)$$

Each node in the DAG encodes $\Pr(x_i|\mathbf{pa}_i)$ as a "conditional probability table" (CPT).

The idea of constructing BBN from WordNet has been proposed by (Rebecca, 1998). But that idea is centered around doing hard-sense disambiguation - to find the 'correct' sense each word in the text.

In this paper, we particularly explore the idea of doing soft sense disambiguation *i.e.* synsets are probabilistically considered to be causes of their constituent words. Moreover, WSD is not an end in itself. The goal is to connect the words within question and answer passage and also across the question and answer passage. WSD is only a by-product.

Our goal is to build a QA system which implements a clear division of labor between the knowledge base and the scoring algorithm, codifies the knowledge base in a uniform manner, and thereby enables a generic algorithm and a shared, extensible knowledge base. Based on the discussion above, our knowledge representation must be probabilistic, and our system must combine and be robust to multiple, noisy sources of information from query and answer terms.

Moreover, we would like to be able to *learn* important properties of our knowledge base from continual *training* of our system with corpus samples as well as samples of successful and unsuccessful (question, answer) pairs. In essence, we would like to automate as far as possible, the customization of lexical networks to QA tasks. Given the English WordNet, it should be possible to reconstruct our algorithm completely from this paper.

Toward these ends, we describe how to induce a Bayesian Belief Network (BBN) from a lexical network of relations. Specifically, we propose a semi-supervised learning mechanism which simultaneously trains the BBN and associates text tokens ,which are words, to synsets in the WordNet in a probabilistic manner ("soft WSD"). Finally, we use the trained BBN to score passages in response to a question.

2.5 Building a BBN from WordNet

Our model of the BBN is that each synset from WordNet is a boolean *event* associated with a question, a passage, or both. Textual tokens are also events. Each event is a node in the BBN. Events can *cause* other events to happen in a probabilistic manner, which is encoded in CPTs. The specific form of CPT we use is the well-known **noisy-OR** of Pearl (Pearl, 1988).

We introduce a node in the BBN for each noun, verb, and adjective synset in WordNet. We also introduce a node for each (non-stop-word) token in the corpus and all questions. Hyponymy, meronymy, and attribute links are introduced from WordNet. *Sense links* are used to attach tokens to potentially matching synsets. E.g., the string "flag" may be attached to synset nodes {sag, droop, swag, flag} and {a conspicuously marked or shaped tail}. (The purpose of probabilistic disambiguation is to estimate the probability that the string "flag" was *caused* by each connected synset node.)

This process creates a hierarchy in which the parent-child relationship is defined by the semantic relations in WordNet. A is a parent of B iff A is the *hypernym* or *holonym* or *attribute-of* or A is a synset containing the word B. The process by which the Bayesian Network is built from the WordNet hypergraph of synsets and and from the mapping between words and synsets is depicted in figure 1. We define *going-up* the hierarchy as the traversal from child to parent.

Ideally, we should update the entire BBN and its CPTs while scanning over the training corpus. In practice, BBN training and inference are CPU- and memory-intensive processes.

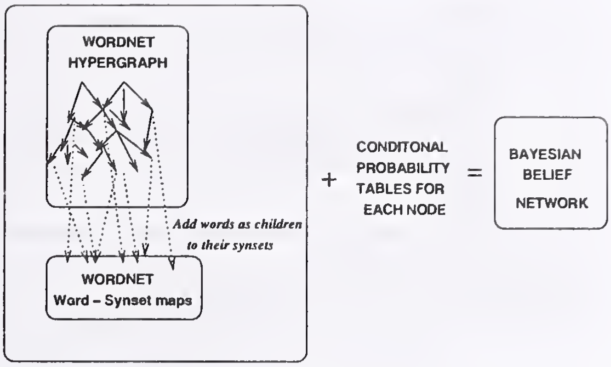

WORDNET HYPERGRAPH
Add words as children to their synsets
WORDNET Word – Synset maps
+
CONDITONAL PROBABILITY TABLES FOR EACH NODE
=
BAYESIAN BELIEF NETWORK

Figure 1: Building a BBN from WordNet and associated text tokens.

We compromise by first attaching the token nodes to their synsets and then walking up the WordNet hierarchy up to a maximum height decided purely by CPU and memory limitations. We believe that the probabilistic influence from distant nodes is too feeble and unreliable to warrant modeling.

3 Our QA system

The overall question answering system that we propose is depicted in figure 2.

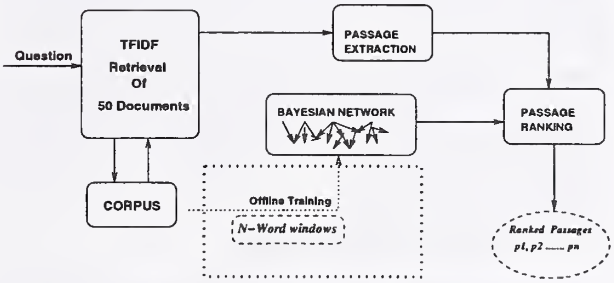


Figure 2: The overall QA system.

The question triggers the TFIDF retrieval module to pick up 50 most relevant documents. These documents are subjected to a sliding window to produce K passages of length N each. The Bayesian belief network described in section 2.5 ranks these passages. The first ranked passage is supposed to contain the answer. The belief network parameters are the CPTs, which are initialized as noisy-or CPTs. The Bayesian belief network is trained offline using the Expectation Maximization algorithm (Dempster, 1977) on windows sliding over the whole corpus.

```
while CPTs do not converge do
    for each window of $M$ words in the text do
        Clamp the word nodes in the Bayesian Network to a state of ‘present’
        for each node in Bayesian network do
            find its joint probabilities with all configurations of its parent nodes (E Step)
        end for
    end for
    Update the conditional probability tables for all random variables (M Step)
end while
```

Figure 3: Training the Bayesian Network for a corpus

3.1 Training the belief network

The figure 3 describes the algorithm for training the BBN obtained from the WordNet. We initialize the CPTs as *noisy-or*. The instances we use for training are windows of length M each from the corpus. Since the corpus is normally not tagged with WordNet senses, all variables, other than the words observed in the window (i.e. the synset nodes in the BBN) are hidden or unobserved. Hence we use the Expectation Maximization algorithm (Dempster, 1977) for parameter learning. For each instance, we find the expected values of the hidden variables, given the present state of each of the observed variables. These expected values are used after each pass through the corpus to update the CPT for each node. The iterations through the corpus are done till the sum of the squares of Kullback-Liebler divergences between CPTs in successive iterations do not differ more than a threshold, or in other words, till the convergence criterion is met. Figure §3 outlines the algorithm for training the Bayesian Network over a corpus. We basically customize the Bayesian Network CPTs to a particular corpus by learning the local CPTs.

3.2 Ranking answer passages

Given a question, we rank the passages with the joint probability of the question words, given the candidate answer. Every question or answer can be looked upon as an event in which the its word nodes are switched to the state ‘present’. Therefore, if $p_1, p_2 .... p_n$ are passages and q is the question, the answer is that passage p_i which maximizes $P(q|p_i)$ over all passages p_i deemed as candidate answers. $\Pr(q|p_i)$ is the joint probability of the words of q, each being in state ‘present’ in the Bayesian network, given that all the word nodes for p_i are clamped to the state ‘present’ in the belief network. Figure §4 outlines the actual passage ranking algorithm.

```
Load the Bayesian Network parameters
for each question q do
    for each candidate passage p do
        clamp the variables (nodes) corresponding to the passage words in network to a state of ‘present’
        Find the joint probability of all question words being in state ‘present’ i.e., $\Pr(q|p)$
    end for
end for
Report the passages in decreasing order of $\Pr(q|p)$
```

Figure 4: Ranking answer passages for given question

The reason for choosing $\Pr(q|p_i)$ over $\Pr(p_i|q)$ is that (a) q typically contains very few words. $\Pr(p_i|q)$, therefore, may not help in bridging the relation between answer words. (b) The passage will be penalized if contains many words which are not present in the question and are also not closely related to the question words through the WordNet. This could happen despite the fact that the passage contains a few words which are all present in the question and/or are semantically closely related to the question, in addtion to containing the answer to the question. Also, (c) if passages p_i's are of varying lengths, $\Pr(q|p_i)$'s are brought to the same scale—that of question words which are fixed across passages/snippets, whereas, $\Pr(p_i|q)$ can be affected and penalized by long snippets.

In fact, our apprehensions about using $\Pr(p_i|q)$ will be justified in the experimental section - the QA performace obtained using $\Pr(p_i|q)$ is drastically poorer - in fact it is worse than the baseline QA algorithm.

Dealing with non-WordNet words: Suppose, there is a word w in the question which is not there in the WordNet. Like the answer passages, we could have ignored such words. But, the question may be seeking an answer to precisely such a word. Also, the number of words being very small in the question, no word in the question should be ignored. We deal with this situation in the following way. We call a word, a *connecting word* if it the key word that links the passage to the question. Note that for WordNet words, the connecting nodes were WordNet concepts. In the case of non-WordNet words, we don't have any hidden, connecting nodes. So we consider the words themselves to be possible connections.

Let *connectw* be a random variable which takes the state ‘present’ if w is a connecting word between the question and the answer. It's state is ‘absent’ if it is not a connecting word. Let wq, wp be random variables that are ‘present’ if w occurs in the question or answer respectively, else they are ‘absent’.

By Bayes rule, we get the following probability that the word w occurs in the question, given that it occurs in the answer (1=Present, 0=absent).

$$\Pr(wq = 1|wp = 1) \approx$$

$$\Pr(wq = 1|connectw = 1) \times \Pr(wp = 1|connectw = 1) \times$$

$$\Pr(connectw = 1) +$$

$$\Pr(wq = 0|connectw = 0) \times \Pr(wp = 0|connectw = 0) \times$$

$$\Pr(connectw = 0)$$

where $Pr(connectw = 1)$, $Pr(wq = 1|connectw = 1)$, $Pr(wp = 1|connectw = 1)$, and $Pr(connectw = 1)$ and their complements are estimated from question answer pairs. Moreover, the occurrence of non WordNet words is assumed to be independent of each other and also of the occurrence of WordNet words.

4 Use of Regular Expressions for passage Filtering

A study of the available question-answer pairs from the earlier TREC releases, helped us to identify patterns for filtering passages corresponding to every question type. The question type is identified for a group of question cue phrases. For every group, a regular expression is identified. A question cue phrase can belong to more than one group of cue phrases.

For example, the group of cue phrases that belong to the class of DURATION such as how_long how_often, how_short, how_frequently, how_far how_fast, how_swift, how_old and how_new make it manditory for the answer to contain regular expressions such as $[CD]+$. The regular expressions that were used were quite involved. A balance between general versus specific regular expression needs to be achieved since very general regular expressions do not serve any purpose in the answer phrase filtering while very specific regular expressions give a very low recall.

5 BBN simplification

Following are some of our observations regarding the approach of Bayesian Inferencing for identifying the answer passages.

5.1 Observation on variety of dependency arcs in BBN

In the preliminary experiments with Bayesian Inferencing, we initialized all CPTs as noisy−or. Noisy−or CPTs make sense for nodes which could be caused exclusively by one of the parents *i.e.* when the occurrence of one parent event precludes the occurrence of other parent events. This is true for word nodes, whose parents are factually its different senses and occurrence of one sense of a word, precludes occurrence of its other senses.

But this is not true for nodes that are synsets − the parents of such nodes are compositional in nature − the child is simultaneously the hyponym of some of its parents and the meronym of its other parents. Hence, we need to revamp our approach of using a noisy−or model for the entire network, from the leaf nodes corresponding to the words upto the root nodes. We propose to initialize the words nodes with noisy−or CPTs and synset nodes with noisy−and CPTs.

5.2 Observation on senses of a word

Our observation is that word senses as given by WordNet, or for that matter word senses given by any lexicon, are not completely orthogonal or unrelated. In fact, to different extents, word senses overlap and form soft−clusters. We feel that any algorithm that attempts to exploit relations between word senses must explicitly take care of this fact. In doing bayesian inferencing with the whole network, we tried to capture this phenomenon implicitly through the idea of *soft sense disambiguation*. But this was at the cost of computationally and memory intensive algorithms. We are working in the direction of simplifying the network before−hand. We present some observations and a simple algorithm in pursuit of the goal.

The observation is that word−senses with similar ancestral lineage have more overlap in their meanings than those with completely distinct ancestral lineages.

6 Discussion and future work

We have described a passage-scoring algorithm for QA via Bayesian inference on lexical relations. By separating the inference algorithm from the design of the knowledge base, we made our system extensible and trainable from a corpus.

Our work hinges upon the existence of lexical relations in the WordNet. We would like to point out here that no special efforts were made in the construction of the Bayesian Network from WordNet nor did we attempt to fill in the desirable 'missing links' between words or synsets in WordNet or remove spurious links in WordNet. Thus, we are able to find probabilities based on semantic relations to the extent given by links in WordNet and we are able to uncorrelated words from each other to the extent they are disconnected in WordNet. To some extent, we attempt to learn the Bayesian Network parameters and this does result in improvement in Question Answering performance. But it will be interesting to see if training the network with bigger corpora improves the performance further. Another experiment that remains to be tried is training the Bayesian Network with samples of successful and unsuccessful (question, answer) pairs.

One thing to note is that if all the question words are contained in the passage, the passage will get a high rank because it will induce a joint probability score of 1 on the question. This can happen even if the answer is not contained in the passage.

Another limitation is the computational and memory cost. On an average it took 0.03 seconds for Bayesian inferencing on a passage. The memory requirement goes upto 30MB. One future work will comprise of reducing the online memory and computational requirements by simplifying the network structure and/or making certain computations offline.

We would also like to find better initial values to speed up learning and avoid local optima. We would like to re-introduce the notion of lexical proximity into our inference process, so as to further improve the accuracy of WSD. We also wish to explore how continual feedback and retraining of the BBN can improve the accuracy of our system.

References

Abe, Naoki, and Hang Li. 1996. Learning word association norms using tree cut pair models. In *Proceedings of the 13th International Conference on Machine Learning*.

C. Buckley. 1985. Implementation of the smart information retrieval system. Technical report, Technical Report TR85-686, Department of Computer Science, Cornell University.

C. L. A. Clarke, Gordon V. Cormack, and Thomas R. Lynam. 2001. Exploiting redundancy in question answering. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 358–365. ACM Press.

C. Fellbaum, 1998. *WordNet: An Electronic Lexical Database*, chapter Using WordNet for Text Retrieval, pages 285–303. The MIT Press: Cambridge, MA.

Christiane Fellbaum. 1998b. *WordNet: An Electronic Lexical Database*. The MIT Press.

Sanda Harabagiu, Dan Moldovan, Marius Pasca, Rada Mihalcea, Mihai Surdeanu, Razvan Bunescu, Roxana Girju, Vasile Rus, and Paul Morarescu. 2000. Falcon: Boosting knowledge for answer engines. In *Proceedings of the ninth text retrieval conference (TREC-9)*, November.

David Heckerman. 1995. A Tutorial on Learning Bayesian Networks. Technical Report MSR-TR-95-06, March.

Boris Katz. 1997. From sentence processing to information access on the world wide web. *AAAI Spring Symposium on Natural Language Processing for the World Wide Web, Stanford University, Stanford CA.*

Cody C. T. Kwok, Oren Etzioni, and Daniel S. Weld. 2001. Scaling question answering to the web. In *Proceedings of the Tenth International World Wide Web Conference*, pages 150–161.

David D. Lewis and Karen Sparck Jones. 1996. Natural language processing for information retrieval. *Communications of the ACM*, 39(1):92–101.

J. Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc.

Adwait Ratnaparkhi. 1996. A maximum entropy part-of-speech tagger. In *Proceedings of the Empirical Methods in Natural Language Processing Conference, May 17-18, 1996. University of Pennsylvania.*

Mark Sanderson. 1994. Word sense disambiguation and information retrieval. In *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pages 49–57, Dublin, IE.

Ellen Vorhees. 2000. Overview of TREC-9 question answering track. *Text REtreival Conference 9.*

Wiebe, Janyce, O'Hara, Tom, Rebecca Bruce. 1998. Constructing Bayesian networks from WordNet for word sense disambiguation: representation and processing issues. In *Proc. COLING-ACL '98 Workshop on the Usage of WordNet in Natural Language Processing Systems.*

.P. Dempster, N.M. Laird and D.B. Rubin. 1977. Maximum Likelihood from Incomplete Data via The EM Algorithm. In *Journal of Royal Statistical Society*, Vol. 39, pp. 1-38, 1977.

Ganesh Ramakrishnan and Pushpak Bhattacharyya. 2003. Text Representation with WordNet Synsets: A Soft Sense Disambiguation Approach. To appear in *Proceedings of the* 8^{th} *International Conference on Natural Language in Information Systems*, Springer Verlag.

IIT at TREC-2003

Task Classification & Document Structure for Known-Item Search

Steve Beitzel, Eric Jensen,
Rebecca Cathey, Ling Ma
David Grossman, Ophir Frieder
{steve, ej, cathey, maling, grossman, ophir}@ir.iit.edu

Information Retrieval Lab
Department of Computer Science
Illinois Institute of Technology
Chicago, Illinois

Abdur Chowdhury, Greg Pass,
Herman Vandermolen
{cabdur, gregpass1}@aol.com,
herman@aol.net

Search Technologies Group
America Online, Inc.
Dulles, VA

Abstract:

This year's TREC 2003 web task incorporated two retrieval tasks into a single set of experiments for Known-Item retrieval. We hypothesized that not all retrieval tasks should use the same retrieval approach when a single search entry point is used. We applied task classifiers on top of traditional web retrieval approaches. Our traditional retrieval is based on fusion of result sets generated by query runs over independent parts of the document structure. Our task classifiers combine query term analysis with known information resources and URL depth. This approach to task classification shows promise: our classified runs improved overall MRR effectiveness over our traditional retrieval results by ~10%; provided an MRR of .665; ranked 87% of relevant results in the top 10; correctly ranked the #1result 56% of the time. 67% of the queries performed above the average, and 49% above the median.

Keywords: Known-item search, document structure retrieval, query task classification

Introduction

Many years of research have been devoted to examining the question of what is the best retrieval strategy for retrieving information. This year we explore a variation on the task in which a specific home/named page or known-item is sought after given a query or topic. Our research this year builds on prior known-item and homepage retrieval techniques by examining the question of whether these two tasks should be treated differently.

Basic retrieval work has focused on ranking strategies: for example, some of the most studied algorithms include PDLN (Pivoted Document Length Normalization) [1], Okapi BM25 [2], Self-Relevance [3], and Language Models [18]. All these ranking strategies try and find better ways to estimate relevance, as do many of the newer language models. In our tests, BM25 has consistently outperformed the other strategies, so we use it in our experiments.

Web retrieval extends basic full-text retrieval by using link and document structures to provide various document representations [4]. This multi-document representation approach was shown to be effective in the top web track systems at the 2002 TREC conference. The basic hypothesis is that content developers use HTML elements/tags to improve the readability of their documents, thus using that information during the ranking process via multiple document representations will improve effectiveness. Examples of these representations could be title, section headers, anchor text, bold, underlines, comments, referring page anchor text, etc. We initially focus on title, anchor text, and referring anchor text.

Given multiple document representations, the most fitting method of using and combining those representations for a given query becomes a research question. In recent years, the category of work known as data fusion, or multiple-evidence, describes a range of techniques in information retrieval whereby multiple pieces of information are combined to achieve improvements in retrieval effectiveness. These pieces of information can take many forms including different query representations, different document representations, and different retrieval strategies used to obtain a measure of relationship between a query and a document.

Several researchers have used combinations of different retrieval strategies to varying degrees of success in their systems [5, 6]. Belkin et al. examined the effects of combining several different query representations to achieve improvements in effectiveness [7, 8]. Lee examined the effect of using different weighting schemes to retrieve different sets of documents using a single query and document representation, and a single retrieval strategy [9]. Fox and Shaw examined combination algorithms that increase the score of a document based on repeated evidence of its relevance in [5].

One of the algorithms designed by Fox and Shaw, CombMNZ, has proven to be a simple, effective method for combining result sets. It was used by Lee in his fusion experiments, and has become the standard by which newly developed result combination algorithms are judged. More recent research in the area of meta-search engines has led to the proposal of several new result combination algorithms [10, 11, 12]. Although these algorithms were shown to be comparable, and on occasion superior, to CombMNZ, we use the widely-used CombMNZ for this work, leaving other approaches as a topic of further research.

Our traditional web search approach fuses the results from different document structure indices to produce a single ranked list for the known-item task. The results were fused using linear combinations based on estimated MRR values in order to maximize mutual evidence [13].

In the next section we describe our basic search approach in more detail. In the task classification section we present our approach to using task information to improving task and overall system effectiveness. Lastly, we present our experimental results and conclude with future possible research directions.

1 Traditional Search Approach

To conduct our research we use the IIT retrieval system AIRE (http://ir.iit.edu/projects/AIRE.html) [14]. This system builds a traditional inverted index based on a given document structure(s). For stemming, our system uses conflation classes [15] instead of a more commonly used stemmer such as Porter [16]. Those classes have been modified over the years as problem term variants have.been encountered. Additionally, AIRE uses a generated statistical phrase list, where the statistical phrases were generated with a news collection and IDF filtering to reduce the final phrase list size. Phrases are generated via a bi-gram sliding window algorithm and weighted with 25% importance in relation to keyword weighting for retrieval. Basic term weighting uses the Okapi BM25, Equation 1.

$$\sum \log\left(\frac{(N-n)+.5}{(n+.5)}\right) * \left(\frac{(k1+1)*tf}{(K+tf)} * \frac{(k3+1)*qtf}{(k3+qtf)}\right)$$

$$K = k1*((1-b)+b*dl/avdl)$$

Equation 1: Okapi BM25

Where:

- tf = frequency of occurrences of the term in the document
- qtf = frequency of occurrences of the term in the query

- dl = document length
- avdl = average document length
- N = is the number of documents in the collection
- n = is the number of documents containing the word
- k1 = 1.2
- b = 0.75 or 0.25 (we use .75 for full text and .25 for shorter representations, see appendix)
- k3 = 7, set to 7 or 1000, controls the effect of the query term frequency on the weight.

1.1 Parsing

We indexed the 18GB .GOV collection producing a full-text index, an HTML title term index, and an anchor text index. The anchor text index differed from the other indexes, in that an additional mapping stage was required so referencing anchor text data can be linked to the referenced TREC document name. For our experimental layout we first produced a baseline run using BM25, conflation classes, phrases, and full-text indexing (referred to as the "Full text" run in the results summarized in Table 1).

1.2 Fusion

Our linear combination consists of the following steps. First, for each document representation retrieved, the scores are normalized using exponential z-score normalization, as in Equation 3. The advantage of this method is that it preserves all relationships of the values exactly; it does not introduce any potential bias into the data. The final scores are calculated using CombMNZ, as in Equation 2, where each individual score is biased via weights assigned to the document structure by prior MRR estimates.

CombMNZ = SUM(Individual Similarities) * Number of Nonzero Similarities

Equation 2: CombMNZ

norm_score(d,x) = ((e^(orig_score(d)) – mean(x)) / stddev(x))

Equation 3: Exponential Z-Score Normalization for document d and document representation x

2 Task Classification

To explore our hypothesis, we identify home pages via two techniques. The first technique uses known information resources and seeks to match those resources to queries. The second approach classifies queries based on keywords like "homepage", and then uses probability distributions of URL length to improve the classification.

2.1 Known-Resource Matching

As many of the homepages in the .GOV domain are government agencies, we hypothesized that simply pairing queries with homepages by matching names and acronyms of agencies would be effective. We searched the web for lists of government agencies and their associated acronyms and homepages, choosing http://www.ulib.iupui.edu/subjectareas/gov/docs_abbrev.html because it provided all three pieces of information, was reasonably large, and was easy to parse with a simple regular expression.

We matched queries to this parsed list of agency name, acronym, and URL tuples using the matching algorithm below. Our system matched 26 of the 300 queries, found the correct homepage for 24 of them, improving our results for 11 queries over our traditional web approach combined with URL normalization. We combined these matching homepages with the final result sets by simply inserting them at rank one. Of the matching queries, 13 already had the matched result at rank one in our final fused, URL length-weighted result set and 2 had not previously been found in the top 1000 results. The other 11 queries matched the relevant homepage, so inserting that homepage at the first result instead of its previous lower position in the result set offered an improvement. In total, MRR was improved by 0.05. Our complete known-resource matching algorithm is shown in Figure 1.

Known-Resource Matching Algorithm:

Step 1. Strip "home", "homepage", and "page" from the query. Strip "the" if it appears as the first word.

Step 2. If the remaining query is an acronym (any sequence of capital letters and spaces), look it up in the list of acronyms by case-insensitive exact string matching. Else, remove any acronyms that might be present alongside other terms from the query, normalize the spacing in the query, and look it up in the list of agency names by case-insensitive exact string matching.

Step 3. If we found a matching acronym or agency name, convert its URL to a canonical form by stripping "http://", "www", trailing slashes, etc. and look it up in a list of all the URLs in the .GOV by case-insensitive exact string matching.

Step 4. If we found a matching acronym, but could not find its corresponding URL in the .GOV, look for its corresponding URL with the last path element stripped off and just the matching acronym as "http://www.ACRONYM.gov" in the .GOV

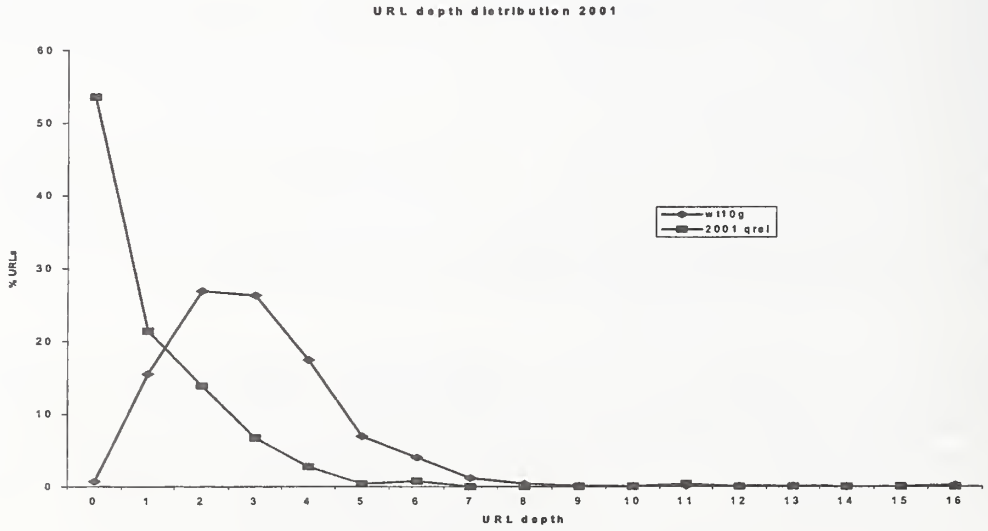
Figure 1: Known-Resource Matching Algorithm

2.2 Task Classification

Kraaij, Westerveld, and Hiemstra [17] previously examined differences in the distributions of URL depth (the length of the path in the URL) between known home pages (from TREC-2001 answers) and the WT10g test collection. They showed that these distributions were very different, and that this could be used to improve the ranking of the results for home page queries. Thus it appeared that if we would be able to successfully classify queries as either home page queries or something else (named page queries in this case), we should be able to improve the results for the homepage queries.

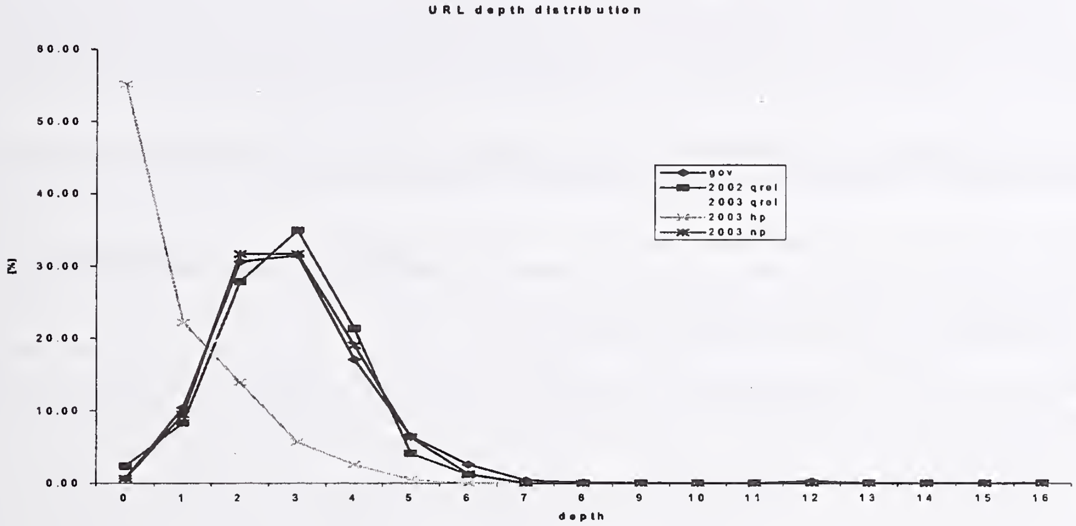
We used a definition of URL depth that was slightly different from the one used by Kraaij et al. but confirmed the differences in distributions. We removed from the URL the leading parts, including host, domain, port, etc., up to the path. We then removed trailing occurrences of "index.htm" and "index.html", and counted the number of path elements remaining to determine the URL depth. The graph below shows the URL depth distribution for the WT10g collection and the correct answers for the TREC-2001 homepage task.

Table 1: WT10g collection URL distributions



For TREC-2003 we ran the same analysis against the .GOV collection and the now known correct answers (qrels) for both the homepage queries and the named-page queries. The analysis shows that for homepage queries, the same distribution differences can be seen between the correct answers and the collection as a whole that were observed in TREC-2001. In addition, it shows that the URL depth distributions for the named page query results are virtually identical to that of the collection as a whole, and thus no advantage can be gained for named page queries.

Table 2: GOV collection URL Distribution



To determine if there were other variables we could utilize, we examined last-modified-date, and in-domain and out-domain link information, and found no significant difference in distributions for the correct answers versus the .GOV collection as a whole.

To be able to take advantage of the URL depth information for home page queries without disturbing the rankings for the named page queries, we attempted to classify the queries into one of these two groups. We created a list of 32 keywords that we believed were good indicators of a home page query. This list includes words like “home”, “homepage”, “administration”, “agency”, etc. Some of these terms were generic, but many would likely be specific to the GOV collection. We parsed the queries looking for these words. Our algorithm categorized 108 (36%) queries (out of 300 combined) as home page queries. Of those 108, 15 were false positives, and 93 were correctly classified. Of the 150 home page queries in the query set, 57 did not match any of the criteria in the classifier and were not marked as home page queries (false negatives).

We took the results from the fusion run and modified the scores of the documents for those queries that our classifier marked as home page queries. The algorithm for the score boosting is shown in Equation 4:

$$\boxed{S_i^* = S_i + \alpha P(d_i)}$$

Equation 4: Score Boosting Formula

where S_i^* is the newly assigned score of document i, S_i is the original score of document i (after fusion), α is a constant, d_i is the URL depth of document i, and $P(d_i)$ is the probability that a document would have

URL depth d_i if it was given that it was a home page. After some experimentation we set the value of α to 12.

3 Results

Our approach is to build on prior approaches to known-item retrieval. To that end, we first examined the effectiveness of a full text approach based solely on BM25 ranking. In the following table we see that our estimated (.52) and actual (.42) effectiveness of this approach can be improved.

We next followed the structured approach that others have shown to be effective by exploiting HTML structure. In the second set of experiments we fused the title, anchor, and full text indices with the CombMNZ algorithm with linear weighting based on estimated MRR values. The Appendix displays the results of those experiments; while we did not have the final qrels, our estimated qrels provided equivalent results to real probabilities. The overall improvement of using document structure over full text retrieval by using CombMNZ with MRR linear combinations improved our effectiveness by 42%.

We next examined our use of known resource information to our traditional web based search approach. By using our known resource information that is based on that task, our MRR is .65.

Next, we examined our classification approach over prior web techniques. With the usage of prior probability factoring with our task classifier, we improved the effectiveness of the system by 3% estimated and 4% actually. We then examined this effectiveness assuming a perfect classifier, and found that our MRR increased to .663, or an additional 4% improvement.

Finally, we examined the improvements of combining both our known resource and URL factoring on the overall effectiveness: we found that by combining those approaches our MMR was raised to .665 and with a perfect classifier .685, an improvement of 9% over our fused results and 55% over our full text results.

Features	Run Tag	Description	Training MRR	Actual MRR	W/ Perfect Classifier & P Dist
Full text	iit03wp75	Full text using statistical phrases weighted at 0.25, BM25 with b=0.75	.52	.43	n/a
Fusion	Iit03wtaez	CombMNZ(fulltext b=0.75, title b=0.25, anchor b=0.25) Using Z-Score and Exponential Normalization	.62	.61	n/a
Fusion, *Known-Resources*	iit03sa	Same fusion, insert known resources with matching names or acronyms at first position	.6889	.65	.65
Fusion, *URL Length Weighting*	iit03su	Same fusion, re-weight results using prior probabilities of relevance given URL lengths calculated by maximum likelihood of training qrels	.6430	.636	.663
Fusion, *URL Length Weighting,* *Known-Resources*	iit03sau	Same fusion, same re-weighting based on URL length priors, and same known-resource insertion	.6945	.665	.685

Table 1: Submitted Runs

	iit03wp75	iit03wtaez	iit03sa	iit03su	iit03sau
MRR	.443	.611	.651	.636	.665
In Top 10	.67	.84	.867	.857	.87
Not Found	.197	.087	.073	.08	.07
=>Median	.253	.44	.47	.47	.49
>= Mean	.407	.613	.653	.647	.67

Table 2: Submitted Runs Official Evaluation

Our final iit03sau approach for the known-item task was 49% of the time equal or above the median and 67% above the mean score of submitted runs. Additionally, our approach produced the item in the top 10 results 87% of the time and only missed 7% of the results.

These results provide validation of the robustness of our task algorithm; more research needs to be conducted to find other task specific information to determine how that information should be incorporated into the ranking strategy.

4 Preliminary Failure Analysis

Examining Table 2 gives some indication of how each approach performs. Notice for the iit03sa and iit03sau runs that relevant documents are not found for approximately 7% of the queries, and are found in the top ten for 87% of the queries, leaving 6% for which the relevant document is found, but is poorly ranked. For failure analysis we focus on these runs, as they achieve the highest MRR.

We examined the queries where the relevant document was not found or poorly ranked (found, but not in the top ten). For each of these queries from the iit03sa and iit03sau runs, we examined the relevant documents and noted where in the each document the query text was present. Our hope was that this analysis would give us an idea of which of our document representations was most likely to contain query text for queries that performed poorly, and we could use that knowledge to improve our parsers. This analysis is shown in Table 3, where the left side represents queries where the relevant document was poorly ranked, and the right side represents queries for which the relevant document was missing.

From Table 3 we see that for queries where the relevant document is missing or poorly ranked, the query text appears within the relevant document approximately 95% of the time. Since the title is most likely to contain the query text (72-82%), it is reasonable to conclude that our title parser might be failing often on these documents. Query text is also likely to appear in the body of the document (56-62%), and particularly in the anchor text for queries where the relevant document is missing (50-52%). From these results we conclude that our title and body parsers may be at fault and worthy of further examination.

To examine this further, we attempted to determine where our parsers were failing, paying particular attention to the anchor text and title parsers. We noticed that although the title parser extracts most of the titles, there are several instances where it fails for various idiosyncratic reasons. Likewise, the anchor text parser also worked fairly well, extracting all the data from within the <a> tags. However, it did not extract any data where the "href" parameter of the <a> tag was pointing to itself. For example, when the "#" sign is used with hyperlinks it is usually followed by redirection within the page such as "top" or "back". However, in some cases, there are hyperlinks with "#" sign followed by meaningful text. In the case of "<a href = "#3214">u.s.s. monitor </a>", our anchor text parser would ignore "u.s.s. monitor". By improving the efficiency of both the title and anchor text parsers, we believe the accuracy of each individual result file can be improved, thus improving the accuracy of the final results file.

We also did some analysis to try and determine whether our fusion process was causing relevant documents to be pushed down in the result set to poor ranks. To test this, we examined the three result files used in the fusion process: iit03wp75, iit03t_np, and iit03a_np. Once again we examined two cases: poorly ranked documents and missing documents. In the final result file there are 18 queries that perform poorly and 21 queries that don't return relevant documents. For each of these queries, we calculated the distribution of relevant documents over each individual result file. In addition, we computed the percentage of relevant documents whose ranks occurred in the top ten, in between ten and fifty, and over fifty. The results of this analysis are given in Table 4.

For queries with poor performance, relevant documents are most likely to occur in the title (45%) and word (66%) result files. Overall, 50% of the relevant documents occurred without overlap in the top ten of the individual result files, and the fusion process increases these ranks, thereby damaging overall MRR.

Similarly, for queries whose relevant documents are missing, the word and title result files contain a high percentage of relevant documents. Unfortunately in this case the majority of the relevant documents occurring in the word and title result files have ranks over fifty prior to fusion, therefore, performance on these queries was initially bad and the fusion process is not likely to have had a significantly negative impact in this case. From the analysis we can conclude that a more finely tuned fusion method may have the potential to help cases where the relevant documents end up poorly ranked in the final result set.

	Poorly Ranked		**Missing**	
	iit03sa	**Iit03sau**	**Iit03sa**	**iit03sau**
Title	.72	.72	.82	.81
Word	.56	.56	.59	.62
ImgAlt	.44	.39	.45	.48
Anchortext	.39	.39	.50	.52
Meta	.28	.28	.14	.10
Not Found	.06	.06	.05	.05

Table 3: Presence of Query Terms in Documents

Rank in	**Poorly Ranked**			**Missing**		
resultfile	**iit03wp75**	**iit03t_np**	**Iit03a_np**	**iit03wp75**	**iit03t_np**	**iit03a_np**
top 10	.22	.17	.11	.05	.0	.0
10-50	.44	.28	.22	.10	.10	.0
over 50	.28	.28	.0	.57	.48	.29
total	.94	.73	.33	.72	.58	.29

Table 4: Presence of Relevant Documents in Individual Result files

5 Conclusion

This year we participated in the homepage and known-item web retrieval task. We explored the concept of multiple tasks being issued via the same interface. To that end we explored using a task classification approach where we could use task specific information to improve those queries. This approached showed promise in that by using task specific information our results improved ~10% over our baseline traditional web retrieval approach and would have improved by 12% given an optimal classifier. Given the simplicity of our classifier this approach seems to help the overall system effectiveness. Our future work will continue examining other features that can help in the other tasks.

Appendix

Table 5: B-value = .25

Index	Run Description	Run Name	Est. MRR	Actual MRR
gov.anchor	.GOV anchor terms only, good HTML parser, no phrases	iit03a_np.dat	0.24	0.29
gov.anchor	.GOV anchor terms only, good HTML parser, with phrases	iit03a_p.dat	0.24	0.30
gov.title	.GOV title terms only, good HTML parser, no phrases	iit03t_np.dat	0.34	0.33
gov.title	.GOV title terms only, good HTML parser, with phrases	iit03t_p.dat	0.35	0.34
gov.bigtext	.GOV bigtext terms only, good HTML parser, no phrases	iit03b_np.dat	0.18	0.15
gov.bigtext	.GOV bigtext terms only, good HTML parser, with phrases	iit03b_p.dat	0.18	0.15
gov.word	.GOV conglomerate, Words only, good HTML parser, no phrases	iit03w_np.dat	0.34	0.29

gov.word	.GOV conglomerate, Words only, good HTML parser, with phrases	iit03w_p.dat	0.34	0.29
gov.meta	.GOV meta, Words only, good HTML parser, no phrases	iit03m_np.dat	0.17	0.14
gov.meta	.GOV meta, Words only, good HTML parser, with phrases	iit03m_p.dat	0.17	0.14
gov.imgalt	.GOV img/alt, Words only, good HTML parser, no phrases	iit03i_np.dat	0.16	0.16
gov.imgalt	.GOV img/alt, Words only, good HTML parser, with phrases	iit03i_p.dat	0.15	0.16

Table 6: B-value = .75

Index	Run Description	Run Name	est. MRR	actual MRR
gov.anchor	.GOV anchor terms only, good HTML parser, no phrases	iit03a_np75.dat	0.29	0.33
gov.anchor	.GOV anchor terms only, good HTML parser, with phrases	iit03a_p75.dat	0.29	0.33
gov.title	.GOV title terms only, good HTML parser, no phrases	iit03t_np75.dat	0.30	0.26
gov.title	.GOV title terms only, good HTML parser, with phrases	iit03t_p75.dat	0.30	0.26
gov.bigtext	.GOV bigtext terms only, good HTML parser, no phrases	iit03b_np75.dat	0.18	0.16
gov.bigtext	.GOV bigtext terms only, good HTML parser, with phrases	iit03b_p75.dat	0.18	0.16
gov.word	.GOV conglomerate, Words only, good HTML parser, no phrases	iit03w_np75.dat	0.49	0.42
gov.word	.GOV conglomerate, Words only, good HTML parser, with phrases	iit03w_p75.dat	0.52	0.43
gov.meta	.GOV meta, Words only, good HTML parser, no phrases	iit03m_np75.dat	0.12	0.09
gov.meta	.GOV meta, Words only, good HTML parser, with phrases	iit03m_p75.dat	0.11	0.09
gov.imgalt	.GOV img/alt, Words only, good HTML parser, no phrases	iit03i_np75.dat	0.12	0.11
gov.imgalt	.GOV img/alt, Words only, good HTML parser, with phrases	iit03i_p75.dat	0.12	0.11

References

[1] A. Singhal, et al., "Pivoted document length normalization", ACM-SIGIR, 1996.

[2] S. Robertson, et al., "Okapi at TREC-4", Proceedings of the 4th annual Text Retrieval Conference (TREC-4), NIST, November 1995.

[3] K. Kwok, et al., "TREC-7 Ad-Hoc, High precision and filtering experiments using PIRCS", Proceedings of the 7th annual Text Retrieval Conference (TREC-7), NIST, November 1998.

[4] S. Brin, L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine", WWW7 / Computer Networks 30(1-7): 107-117 (1998).

[5] E.A. Fox and J.A. Shaw, “Combination of Multiple Searches,” Proceedings of the 2nd Text Retrieval Conference (TREC-2), NIST Special Publication 500-215, pp. 243-252, 1994.

[6] B.T. Bartell, G.W. Cottrell, and R.K. Belew, “Automatic Combination of multiple ranked retrieval systems,” Proceedings of the 17th Annual ACM-SIGIR, pp. 173-181, 1994.

[7] N.J. Belkin, C. Cool, W.B. Croft and J.P. Callan, “The effect of multiple query representations on information retrieval performance,” Proceedings of the 16th Annual ACM-SIGIR, pp. 339-346, 1993.

[8] N.J. Belkin, P. Kantor, E.A. Fox, and J.A. Shaw, “Combining evidence of multiple query representation for information retrieval,” Information Processing & Management, Vol. 31, No. 3, pp. 431-448, 1995.

[9] J.H. Lee, “Combining Multiple Evidence from Different Properties of Weighting Schemes,” Proceedings of the 18th Annual ACM-SIGIR, pp. 180-188, 1995.

[10] J. Aslam and M. Montague, et al., “Models for Metasearch”, Proceedings of the 24th Annual ACM Conference on Research and Development in Information Retrieval (SIGIR), September 2001.

[11] M. Montague, et al., “Relevance Score Normalization for Metasearch”, Proceedings of the 10th Annual ACM Conference for Information and Knowledge Management (CIKM), 2001.

[12] M. Montague, et al., “Condorcet Fusion for Improved Retrieval”, Proceedings of ACM-CIKM, November 2002.

[13] P. Ogilvie, J. Callan, "Combining Document Representations for Known-Item Search," Proceedings of the 26th Annual ACM Conference on Research and Development in Information Retrieval (SIGIR), 2003.

[14] A. Chowdhury, et al., "Improved query precision using a unified fusion model", Proceedings of the 9th Text Retrieval Conference (TREC-9), November 2000.

[15] J. Xu, B. Croft, “Corpus-based stemming using co-occurrence of word variants”. ACM Transactions on Information Systems, January, 1998.

[16] Porter, “An algorithm for suffix stripping”. Program, 14(3):130—137, 1980.

[17] W. Kraaij, T. Westerveld, and D. Hiemstra. The importance of prior probabilities for entry page search. In Proc. of the 25th annual international ACM SIGIR conference on research and development in information retrieval, pages 27{34. Association for Computing Machinery, 2002.

[18] J. Ponte and W. B. Croft. A language modeling approach to information retrieval. In 21st ACM Conference on Research and Development in Information Retrieval (SIGIR'98) 275-281, 1998.

Identifying Gene Function Descriptions by Probability-based Sentence Selection

Kazuhiro Seki, Nihar Sheth, and Javed Mostafa

Laboratory for Applied Informatics Research, Indiana University
1320 East Tenth Street, LI 011, Bloomington, Indiana 47405-3907, USA

{kseki,nisheth,jm}@indiana.edu

Abstract

This paper proposes an approach to the secondary task in the TREC Genomics Track. We regard the task as identification of the sentences describing gene functions (i.e., GeneRIFs) and propose a method considering two factors: topicality and relevance. The former refers to the topicality of a sentence and is measured based on location information and word frequencies in the article. The latter refers to the relevance as a GeneRIF based on the vocabulary used in the article. We formalize a probabilistic model combining these features. Our method is evaluated on the test set of 139 MEDLINE abstracts, and the results demonstrate that (a) function words in input could help to identify gene function descriptions and that (b) there is a vocabulary peculiar to GeneRIFs and that (c) location information shows the highest predictive power for this particular task despite its simplicity. Additionally, we examine some alternative methods in comparison with our method.

1 Introduction

The volume of publications in the biomedical domain has been rapidly growing, making it difficult for individual researchers to keep themselves updated. This resulted in a strong demand for information retrieval (IR) and information extraction (IE) techniques which could help us manage the information overload.

To foster the IR and IE research in the area of biomedicine, the Genomics Track was launched at the Text REtrival Conference (TREC) 2003 (Hersh, 2002). TREC is one of the major conferences targeting IR and has been contributing to the advance in IR research and related areas (e.g., question answering and filtering) since it first started in 1992.

The Genomics Track is aiming at IR and IE, reflecting the increasing interest in the practical applications of those techniques to the biomedical literature. This year, the Genomics Track offers two independent tasks for IR and IE, namely, the primary and secondary tasks. In short, the primary task aims at finding MEDLINE articles stating the functions associated with given gene names, and the secondary task aims at automatically generating concise descriptions of gene functions stated in given research articles. We are particularly interested in the great potential of IE in this field and therefore targeted the secondary task.

The rest of this paper is structured as follows: Section 2 overviews the secondary task. Section 3 summarizes the past research related to the task. Section 4 describes our proposed method for identifying gene function descriptions. Section 5 reports experiments carried out to evaluate our method. Section 6 compares our method with alternative approaches. Lastly, Section 7 concludes this paper with a brief summary and possible directions for future research.

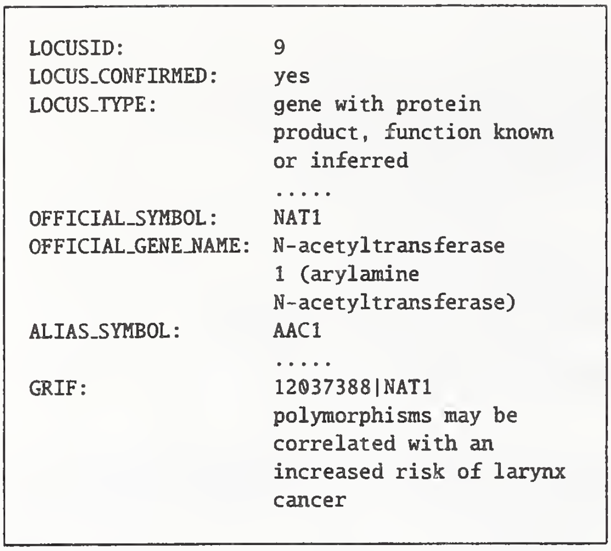
2 The Secondary Task

The secondary task targets information extraction (IE) from the biomedical literature. Specifically, it aims at generating descriptions related to gene functions in an automated way. For this year, the Track Steering Committee decided to experimentally use GeneRIF (Gene References into Function) entries as the gold standard, which are described in the LocusLink database (Pruitt and Maglott, 2001) maintained by National Center for Biotechnology Information (NCBI).

GeneRIFs are functional annotations of genes and, according to the NCBI web page[1], is defined as "*a concise phrase describing a function or functions (less than 255 characters in length, preferably more than a restatement of the title of the paper).*" They have been mainly annotated by experts in the life sciences at National Library of Medicine (NLM). Figure 1 shows an example, where `GRIF` provides PubMed identifier (PMID) and the associated GeneRIF, separated by a vertical line (i.e., PMID is `12037388` and GeneRIF is "`NAT1 polymorphisms may be ...`").

Our goal is to automatically generate a GeneRIF, given an abstract or a full text associated with the correspond-

[1] `http://www.ncbi.nlm.nih.gov/LocusLink/GeneRIFhelp.html`



```
LOCUSID:              9
LOCUS_CONFIRMED:      yes
LOCUS_TYPE:           gene with protein
                      product, function known
                      or inferred
                      .....
OFFICIAL_SYMBOL:      NAT1
OFFICIAL_GENE_NAME:   N-acetyltransferase
                      1 (arylamine
                      N-acetyltransferase)
ALIAS_SYMBOL:         AAC1
                      .....
GRIF:                 12037388|NAT1
                      polymorphisms may be
                      correlated with an
                      increased risk of larynx
                      cancer
```

Figure 1: A fragment of a LocusLink record. GRIF gives a PMID and a gene function description (i.e., GeneRIF).

ing PMID as input. In addition, we may use official gene names and aliases provided by LocusLink (e.g., OFFICIAL_GENE_NAME) in generating a GeneRIF. (However, as described later, we use only abstracts and basically do not use gene names in this study.)

The generated GeneRIF candidates are to be evaluated using the Dice coefficient and its variants, which measure the extent of word overlap between the generated GeneRIF candidate and the actual GeneRIF. Section 5 will formally describe the evaluation measures.

3 Related Works

Given that 95% of actual GeneRIFs are reported to contain some text from titles or abstracts (Hersh, 2003), we view the secondary task as sentence selection, that is, we simplify the task to identifying those sentences which are likely to describe gene functions. Sentence selection (or passage retrieval) is one of core components of automatic summarization and question answering (QA) systems and has been widely explored.

Text summaries are typically generated by extracting text segments based on several features, such as existence of title words, locations of text segments, and similarities between the text segments and the entire text (Gong and Liu, 2001; Chuang and Yang, 2000; McDonald and Chen, 2002). By using these features, each text segment is given a score indicating the extent to which the segment would be included in a summary. We will utilize some of these features to identify topical descriptions.

QA aims at providing the information that directly answers users' questions, as opposed to a ranked list of documents usually returned by conventional IR systems. Most of current QA systems are composed of four basic modules (Tellex et al., 2003): question analysis, document retrieval, passage retrieval, and answer extraction. Here, let us focus on passage retrieval, which breaks down documents retrieved in the preceding module into smaller units, such as sentences, and returns only passages potentially relevant to the query. Passage retrieval is often treated on an analogy to IR, where each passage is regarded as a document and relevant passages are retrieved based on their similarities to a query. In the secondary task, the query is "gene functions," which will be too general to find relevant passages. Instead, we use vocabulary related to gene functions for measuring relevancy of passages, which will be described next.

4 Our Method

4.1 Probabilistic Sentence Selection

The secondary task can be performed by identifying those sentences which describe gene functions, assuming that such sentences exist in an input article. We propose a probabilistic model incorporating two measures: relevance and topicality.

For the relevance as GeneRIFs, we make use of word frequencies in GeneRIFs; higher scores are given to those sentences which contain more words frequently appearing in GeneRIFs. Our assumption is that there is a typical vocabulary used for gene functions frequently. For example, "activate" or "bind" may be often used in describing gene functions and then a sentence containing those words could receive higher scores. For sentence s composed of a sequence of words $w_1 \, w_2 \cdots w_n$, the probability of being GeneRIF can be formalized as a product of the relative frequencies of the words composing the sentence as in Equation (1), where $F_G(w_j)$ and N_G denote the frequency of word w_j and the total number of words in GeneRIFs, respectively.

$$P_G(s) = \prod_{j=1}^{n} P_G(w_j) = \prod_{j=1}^{n} \frac{F_G(w_j)}{N_G} \tag{1}$$

Incidentally, this can be regarded as a unigram language model; that is, it models GeneRIF descriptions by word unigrams.

For the topicality of sentences, we use word frequencies in a given article; higher scores are given to those sentences which contain more words frequent in the article itself. Note that word frequencies here are based on the *input*, as opposed to P_G based on word frequencies in GeneRIFs. The rationale behind it is that the topic of the article is likely to be repeatedly stated. Given this assumption, the probability of being a topical sentence can be expressed as in Equation (2), where N_T denotes the total number of word tokens in the given article and $F_T(w_j)$ is a frequency of word w_j in the given article.

$$P_T(s) = \prod_{j=1}^{n} P_T(w_j) = \prod_{j=1}^{n} \frac{F_T(w_j)}{N_T} \tag{2}$$

As with P_G, P_T can be regarded as a unigram language model; that is, it models an input article by word unigrams.

Additionally, we use location information as an indicator of topicality, given the fact that there is a conventional structure of where topics of articles appear at some typical locations. We define $P_L(L(s))$, which is a probability that sentence s is a topic sentence, based on its location $L(s)$. The function $L(s)$ returns a location of s and its possible values are *title*, *abstract_last*, *abstract_body*, defined based on our preliminary study having suggested that actual GeneRIFs occur in many cases in titles or the end of abstracts. It returns *title* if s is a (part of) title; *abstract_last* if s is the last sentence of abstracts; and *abstract_body* otherwise.

We combine these probabilities introduced above, i.e., P_G, P_L, and P_T, assuming their mutual independency, as in Equation (3).

$$P(s) = P_T(s) \cdot P_L(L(s)) \cdot P_G(s) \tag{3}$$

Since $P(s)$ is influenced by sentence lengths (longer sentences are generally result in smaller values), we normalize it by the number of words n in sentence s and take a logarithm for computational efficiency, forming a score indicating the extent to which s is likely to be a GeneRIF.

$$\begin{aligned} S_{GRIF}(s) &= \log P(s)^{\frac{1}{n}} \\ &= \log\left(P_T(s) \cdot P_L(L(s)) \cdot P_G(s)\right)^{\frac{1}{n}} \\ &= \frac{1}{n}\left(\log P_T(s) + \log P_L(L(s)) + \log P_G(s)\right) \end{aligned} \tag{4}$$

We select the sentence which maximizes $S(\cdot)$ as output (GeneRIF), that is:

$$\hat{s} = \arg\max_{s_i} S_{GRIF}(s_i) \tag{5}$$

4.1.1 Probability estimation

To compute S_{GRIF}, we estimate the probabilities, P_G, P_T, and P_L, as follows.

Firstly, as defined in Equation (1), $P_G(s)$ is a product of relative frequencies of words composing sentence s in GeneRIFs. This can be estimated based on word frequencies in a training set of GeneRIFs. However there are two things to take into account, that is, function words (e.g., *the* and *to*) and word inflection (e.g., *activate* and *activation*). We use a stop word list[2] containing 571 words so as to exclude function words, and use the Porter stemmer (1980) so as to eliminate inflectional variations. To observe their effect on this task, we create four different models of P_G with/without applying the stop word list and the stemmer. In addition, we employ a discounting (smoothing) method, since a significant number of words in input texts would never appear in GeneRIFs and thus we will encounter unknown words in estimating their probabilities, i.e., the zero frequency problem. The absolute discounting method (Ney et al., 1994) is experimentally used to remedy the problem. Absolute discounting takes out a constant proportion from the probability mass and uniformly distributes it to unknown words.

Secondly, $P_T(s)$ can be calculated by simply counting word frequencies in the input. As with P_G, again we make use of the stop word list and the Porter stemmer in order to deal with function words and word inflection, and create four different models of P_T by using/not using the stop word list and the stemmer. Notice that estimating probabilities P_T does not require to train the model in advance, as opposed to P_G.

Lastly, P_L can be estimated by counting where GeneRIFs appear in their corresponding articles, given pairs of articles and GeneRIFs. For example, if most GeneRIFs are taken from titles, $P_L(title)$ would have a high probability. However, this cannot be automatically done because GeneRIFs are not marked in articles and they are not even guaranteed to literally appear in articles since they are generated by human; words, phrases, or word orders may be changed in abstracting GeneRIFs from articles. Thus, it is ideal to use human in order to accurately identify where GeneRIFs (or similar sentences) appear, which is however costly. Instead, we use *bigram phrase* Dice coefficient (see Section 5.1) to measure how similar each sentence is to the corresponding GeneRIF, and consider the computed similarity as the number of occurrences of the GeneRIF. To put it differently, given an article (sentences) and GeneRIF, we compute a similarity score between each sentence and the GeneRIF, and the similarity scores are summed up within each category of location (i.e., *title*, *abstract_last*, and *abstract_body*), which is regarded as a frequency of the GeneRIF occurrences in the location.

5 Evaluation

5.1 Methodology

We evaluate our proposed method on the 139 MEDLINE abstracts provided for the secondary task, where each of the abstracts is associated with an actual GeneRIF. Full-text articles are also available for this task, but we use only abstracts (and titles) given that 95% of actual GeneRIFs contained some text from titles and abstracts (Hersh, 2003). In addition, gene names associated with each GeneRIF can be utilized, but our framework does not incorporate them because, in the actual GeneRIF annotation, indexers do not have specific gene names in mind in advance.

[2] ftp://ftp.cs.cornell.edu/pub/smart/english.stop

As evaluation metrics, the secondary task uses the Dice coefficient to measure similarities between actual GeneRIFs and generated GeneRIF candidates. Given two strings s_x and s_y, the Dice coefficient σ between s_x and s_y is defined as in Equation (6), where N_x, N_y, and N_{xy} are the numbers of words in s_x, in s_y, and in both s_x and s_y, respectively.

$$\sigma(s_x, s_y) = \frac{2 \cdot N_{xy}}{N_x + N_y} \tag{6}$$

However, the Dice coefficient has several limitations as an evaluation metric for this task. Most of them result from the fact that it treats strings as *bags of words* and treats words just as symbols. To compensate for the problems to some extent, the secondary task uses four variants of the Dice coefficient below.

- Classic Dice (CD):
  Uses the Dice coefficient after removing stop words and stemming suffixes. This enables an evaluation based on normalized contents words.
- Modified Unigram Dice (MD):
  Similar to CD but considers word frequencies to give additional scores to words appearing multiple times in both strings compared.
- Bigram Dice (BD):
  Regards two adjacent words (bigrams) as a unit for comparison and applies the Dice coefficient. This allows us to take word order into account to some extent.
- Bigram Phrases (BP):
  Same as BD but excludes bigrams containing stop words. This metric has more focus on noun phrases.

5.2 Results

5.2.1 Exploring Word Frequencies in Input

We examined the effects of stemming word suffixes and removing stop words on $P_T(s)$, which indicates the topicality of sentence s based on word frequencies in an input text. We applied the model with/without stemming and removing stop words, and output the predicted GeneRIFs with the highest probabilities. Table 1 shows the result, where bold figures indicate the highest similarities for each evaluation metric.

Somewhat unexpectedly, the result indicates that the stemmer and the stop word list did *not* contribute to predicting actual GeneRIFs. Especially, when stop words were removed, Dice coefficients radically dropped by more than 10 points, irrespective of evaluation criteria. In IR and related areas, stop words are commonly thought to be less (or not at all) informative and removed, but for this particular task, stop words appear to play a certain role to characterize GeneRIFs.

In the remainder, we do not use the stemmer nor exclude stop words for P_T estimation.

Table 1: Effects of stemming and removing stop words in estimating $P_T(s)$. CD, MD, BD, and BP denote classic Dice, modified unigram Dice, bigram Dice, and bigram phrase, respectively.

		Stop words			
		remained		excluded	
Stemmer	off	CD	**38.37**	CD	26.36
		MD	**39.04**	MD	26.37
		BD	**21.45**	BD	11.38
		BP	**24.55**	BP	13.59
	on	CD	36.69	CD	25.86
		MD	37.38	MD	25.24
		BD	20.27	BD	10.27
		BP	23.42	BP	12.43

5.2.2 Exploring Word Frequencies in GeneRIF

As with $P_T(s)$ above, we examined the effects of stemming word suffixes and removing stop words on $P_G(s)$, which is a probability that a given sentence s is relevant to gene functions based on the vocabulary used in GeneRIFs.

Estimating P_G requires training data. However, since there are no training data besides the test data of 139 GeneRIFs, we trained the model by a leave-one-out cross-validation using the test data, where each GeneRIF was predicted based on the model trained on the other 138 GeneRIFs; that is, training data and test data are always mutually exclusive. We created four different models with/without stemming and removing stop words, and evaluated their effectiveness. The result is shown in Table 2, where bold figures indicate the highest similarities for each metric.

Table 2: Effects of stemming and removing stop words in estimating $P_G(s)$.

		Stop words			
		remained		excluded	
Stemmer	off	CD	32.68	CD	35.55
		MD	31.30	MD	36.83
		BD	15.94	BD	20.80
		BP	18.75	BP	23.41
	on	CD	31.72	CD	**36.68**
		MD	29.25	MD	**37.55**
		BD	14.59	BD	**22.02**
		BP	16.77	BP	**24.88**

As opposed to the case with P_T, stemming and removing stop words resulted in the best result. Especially, removing stop words improved the similarity scores by 3–8 points (9–50%). It proves that there exists a vocabulary particularly used for describing GeneRIFs (or gene functions). Incidentally, it was found that when only the stemmer was applied without removing stop words, it decreased the similarities.

As an illustration, Table 3 shows the 12 stems, excluding stop words, which most frequently appeared in the test data set of 139 GeneRIFs, where one can find a number of stems related to gene functions, such as “activ”, “regul”, and “role”.

Table 3: The most frequent 20 stems in the test data of 139 GeneRIFs. The figures on their right show the logarithms of their relative frequencies.

Rank	Stem	$\log P_G(w)$	Rank	Stem	$\log P_G(w)$
1	activ	−1.4838	7	express	−1.7137
2	cell	−1.4838	8	gene	−1.8031
3	regul	−1.5342	9	induc	−1.8031
4	protein	−1.6396	10	signal	−1.8031
5	role	−1.6569	11	mediat	−1.8286
6	1	−1.7137	12	receptor	−1.8286

In the remainder, we use the stemmer and remove stop words for P_G estimation.

5.2.3 Exploring Optimal Combinations of Models

As defined in Equation 3, our final model combines three independent estimations: $P_T(\cdot)$, $P_G(\cdot)$, and $P_L(\cdot)$. To demonstrate the contribution of each model and to explore their optimal combination, we evaluated each model and every combination on the test data set. Table 4 summarizes the results of the different models, where the top row indicates the combinations of models applied. The right most column ($P_T \cdot P_G \cdot P_L$) shows our submitted official run.

From the results in Table 4, it is apparent that, despite its simplicity, location information (P_L) dominantly contributed to the result and the other two models hardly had effect on the outcome when combined with P_L. This is mainly because the actual GeneRIFs are more or less taken from titles in many cases.

6 Discussions

The evaluation in Section 5 revealed that location information impacts the most in identifying GeneRIFs. However, it does not mean that we can ignore the contents of input sentences because whether each sentence describes gene functions depends on its contents. We explore an alternative method making use of contents (word frequencies) from a viewpoint of classification.

The secondary task can be seen as classification, assigning a class (c_{GRIF} or $c_{nonGRIF}$) to each sentence and selects the one which is most likely to be a GeneRIF. There is a number of methods that can be applied, e.g., naive Bayes classifiers, decision trees, support vector machines. These methods have been compared for their effectiveness and, to our knowledge, there is no clear evidence about which performs best; it depends on tasks applied to, training data size, the number of classes, and so on (Chuang and Yang, 2000; Yang and Liu, 1999). For comparison, we experimentally implemented a naive Bayes classifier, which has been widely used in past research.

The naive Bayes classifier predicts class $\hat{c}$ for input s, where $\hat{c}$ maximizes the probability $P(c_k|s)$ and c_k can be either c_{GRIF} or $c_{nonGRIF}$.

$$\begin{aligned} \hat{c} &= \arg\max_{c_k} P(c_k|s) \\ &= \arg\max_{c_k} P(s|c_k)P(c_k) \end{aligned} \tag{7}$$

For each sentence s_i, we compute a likelihood ratio of a probability associated with class c_{GRIF} to one associated with class $c_{nonGRIF}$, and select a sentence as a GeneRIF which produces the highest ratio.

$$\begin{aligned} \hat{s} &= \arg\max_{s_i} \frac{P(s_i|c_{GRIF})P(c_{GRIF})}{P(s_i|c_{nonGRIF})P(c_{nonGRIF})} \\ &\approx \arg\max_{s_i=w_1 \dots w_n} \prod_{w_j} \frac{P(w_j|c_{GRIF})}{P(w_j|c_{nonGRIF})} \end{aligned} \tag{8}$$

We used the GeneRIFs in the test set to train the classifier for class c_{GRIF} (i.e., the numerator) and used the abstracts to train it for class $c_{nonGRIF}$ (i.e., the denominator). Although most abstracts would include GeneRIFs, it should not be harmful as long as there are more non-GeneRIFs than GeneRIFs in the training data. We evaluated the method on the test set; Table 5 compares the results produced by our model ($P_T \cdot P_G$) and the naive Bayes classifier.

Table 5: Comparison between our model based on word frequencies ($P_T \cdot P_G$) and the naive Bayes classifier for identifying GeneRIFs.

	$P_T \cdot P_G$	Bayes
CD	39.11	34.70 (−11.2%)
MD	40.62	34.66 (−14.7%)
BD	22.42	19.64 (−12.4%)
BP	25.78	22.18 (−13.4%)

Our method outperformed the naive Bayes classifier in all evaluation criteria. The result demonstrates the effectiveness of our method but, at the same time, it implies the limitation of the methods based solely on word distributions, as location information alone results in much higher similarity scores.

In order to combine multiple information sources, our model multiplies the resulting probability estimates (i.e., P_T, P_G, and P_L). This can be regarded as probability voting. On the other hand, one of voting algorithms often used is *majority voting* where each information source gives a vote to its best candidate and the one which received the majority of votes wins. We implemented a (modified) majority voting method for comparison. The voting scheme considers every candidate and cast $1/n$ votes for n-th ranked candidate, so as

Table 4: Results for different combinations of models. Bold characters indicate the best score for each row across the combinations. The right most column ($P_T{\cdot}P_G{\cdot}P_L$) is our submitted official run.

	P_T	P_G	P_L	$P_T \cdot P_G$	$P_T \cdot P_L$	$P_G \cdot P_L$	$P_T{\cdot}P_G{\cdot}P_L$
CD	38.37	36.68	**50.47**	39.11	50.25	50.44	50.40
MD	39.04	37.55	**52.60**	40.62	52.36	52.52	52.56
BD	21.45	22.02	34.82	22.42	34.66	**34.93**	34.83
BP	24.55	24.88	37.91	25.78	37.92	**38.05**	37.97

to avoid the case where no candidate receives the majority. Equation (9) shows the formula.

$$\hat{s} = \arg\max_{s_i} \sum_{P \in \{P_T, P_G, P_L\}} \frac{1}{rank(P(s_i))} \qquad (9)$$

where, $rank(P(s_i))$ is a rank of candidate (sentence) s_i based on probability $P(\cdot)$. Table 6 compares two voting schemata, i.e., probability voting (our model) and majority voting.

Table 6: Results for different voting schemata: probability voting (our model) vs. majority voting.

	$P_T \cdot P_G$		$P_T \cdot P_G \cdot P_L$	
	prob	majority	prob	majority
CD	39.11	39.67 (+1.4%)	50.40	42.43 (−15.6%)
MD	40.62	41.06 (+1.1%)	52.56	44.20 (−15.9%)
BD	22.42	24.06 (+7.3%)	34.83	26.75 (−23.2%)
BP	25.78	27.74 (+7.6%)	37.97	30.59 (−19.4%)

When used for combining two probabilities ($P_T{\cdot}P_G$), majority voting improved the result, especially for bigram-based evaluation criteria (BD and BP). On the other hand, when applied to combine P_T, P_G, and P_L, it significantly decreased the similarity scores by 15%–23%. This is presumably because the probability P_L has much more predictive power than the others. Weighted voting, which gives certain weights to each source, could work better for this model.

Lastly, we report the result when gene names are used as a filter. Each MEDLINE article in the test set is associated with specific gene names, thus it is very likely that gene function descriptions (GeneRIFs) would contain those gene names in them. Based on this assumption, we restricted the system output to those containing the associated gene names. In cases where no gene name appeared in input sentences, the highest ranked sentence was outputted. The experimental result is presented in Table 7.

The filter using gene names did not raise the result. This implies that (exact) gene names do not necessarily appear in GeneRIFs.

7 Conclusions and Future Directions

This paper presented a method for identifying gene function descriptions (GeneRIFs) in biomedical articles. We regarded the task as sentence selection assuming that input articles do contain actual GeneRIFs. Our method exploits location information and word frequencies both in input and GeneRIFs and, given an input text, identifies a sentence which is most likely a GeneRIF using a probabilistic model. We evaluated our method on the test set of 139 MEDLINE abstracts, and the results indicated that (a) function words in input can be used for identifying GeneRIFs, that (b) there exists a vocabulary peculiar to gene function descriptions, and that (c) location information has the most impact in identifying GeneRIFs.

Table 7: Results when gene names are used/not used as a filter. The columns labeled "not used" are the results of our proposed method.

	$P_T \cdot P_G$		$P_T \cdot P_G \cdot P_L$	
	not used	used	not used	used
CD	39.11	36.18 (−7.5%)	50.40	48.41 (−3.9%)
MD	40.62	36.64 (−9.8%)	52.56	50.28 (−4.3%)
BD	22.42	21.51 (−4.1%)	34.83	32.98 (−5.3%)
BP	25.78	24.44 (−5.2%)	37.97	36.32 (−4.3%)

Future directions would include the use of a larger training set and wider contexts for modeling and probability estimation, rather than independent word occurrences. In addition, the effect of using full text articles in estimating P_T (which is based on word frequencies in input) should be investigated.

Acknowledgment

This project is partially supported by the NSF ITR grant #9817572.

References

Chuang, W. T. and Yang, J. (2000). Extracting sentence segments for text summarization: a machine learning approach. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 152–159.

Gong, Y. and Liu, X. (2001). Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international ACM SIGIR* *conference on Research and development in information retrieval*, pages 19–25.

Hersh, W. (2002). Text retrieval conference (TREC) genomics pre-track workshop. In *Proceedings of the second ACM/IEEE-CS joint conference on Digital libraries*, pages 428–428.

Hersh, W. (2003). TREC genomics track overview. In *The Twelfth Text REtrieval Conference (TREC 2003) Notebook.* National Institute of Standards and Technology. A/ailable at http://medir.ohsu.edu/~genomics/overview.pdf.

McDonald, D. and Chen, H. (2002). Using sentence-selection heuristics to rank text segments in TXTRACTOR. In *Proceedings of the second ACM/IEEE-CS joint conference on Digital libraries*, pages 28–35.

Ney, H., Essen, U., and Kneser, R. (1994). On structuring probabilistic dependences in stochastic language modeling. *Computer Speech and Language*, 8(1):1–38.

Porter, M. (1980). An algorithm for suffix stripping. *Program*, 14(3):130–137.

Pruitt, K. D. and Maglott, D. R. (2001). RefSeq and LocusLink: NCBI gene-centered resources. *Nucleic Acids Research*, 29(1):137–140.

Tellex, S., Katz, B., Lin, J., Fernandes, A., and Marton, G. (2003). Quantitative evaluation of passage retrieval algorithms for question answering. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 41–47.

Yang, Y. and Liu, X. (1999). A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42–49.

WIDIT in TREC-2003 Web Track

Kiduk Yang
School of Library and Information Science
Indiana University
Bloomington, Indiana 47405, U.S.A.
kiyang@indiana.edu

Dan Albertson
School of Library and Information Science
Indiana University
Bloomington, Indiana 47405, U.S.A.
daalbert@indiana.edu

1. Introduction

The Web IR experiment of TREC, otherwise known as the Web track, investigated in its initial stages the strategies for the same ad-hoc retrieval task as was done previously with plain text documents. Although many TREC participants explored methods of leveraging non-textual sources of information such as hyperlinks and document structure, the general consensus among the early Web track participants was that link analysis and other non-textual methods did not perform as well as the content-based retrieval methods fine-tuned over the years (Hawking et al., 1999; Hawking et al., 2000; Gurrin & Smeaton, 2001; Savoy & Rasolofo, 2001).

There have been many speculations as to why link analysis, which showed much promise in previous research and has been so readily embraced by commercial Web search engines, did not prove useful in Web track experiments. Most such speculations point to potential problems with Web track's earlier test collections, from the inadequate link structure of truncated Web data (Savoy & Picard, 1998; Singhal & Kazkiel, 2001), and relevance judgments that penalize the link analysis by not counting the hub pages as relevant (Voorhees & Harman, 2000) and boost the content analysis by counting multiple relevant pages from the same site as relevant (Singhal & Kazkiel, 2001), to unrealistic queries that are too detailed and specific to be representative of real world Web searches (Singhal & Kaszkiel, 2001).

In an effort to address the criticism and problems associated with the early Web track experiments, TREC abandoned the ad-hoc Web retrieval task in 2002 in favor of topic distillation and named page finding task and replaced its earlier Web test collection of randomly selected Web pages with a larger and potentially higher quality domain-specific collection[1]. The topic distillation task in TREC-2002 is described as finding a short, comprehensive list of pages that are good information resources, and the named page finding tasks is described as finding a specific page whose name is described by the query (Hawking & Craswell, 2002; Craswell & Hawking, 2003). Adjustment of the Web track environment brought forth renewed interest in retrieval approaches that leverage Web-specific sources of evidences such as link structure and document structure.

For the home page finding task, where the objective is to find the entry page of a specific site described by the query, Web page's URL characteristics, such as its type and length, as well as the anchor text of Web page's inlinks proved to be useful sources of information to be leveraged (Hawking & Craswell, 2002). In the named page finding task, which is similar to home page finding task except that the target page described by the query is not necessarily the entry point of a Web site but any specific page on the Web, the use of anchor text still proved to be an effective strategy but the use of URL characteristics did not work well as it did in the home page finding task (Craswell & Hawking, 2003). In the topic distillation task, anchor text still seemed to be a useful resource, especially as a mean to boost the performance of content-based methods via fusion (i.e. result merging), although the level of its usefulness fell much below that achieved in named page finding tasks (Hawking & Craswell, 2002; Craswell & Hawking, 2003). Various site compression strategies, which attempt to select the "best" pages of a given site, was another common theme in the topic distillation task, once again demonstrating the importance of fine-tuning the retrieval system according to the task at hand (Amitay et al., 2003; Zhang et al., 2003). It is interesting to note that link analysis (e.g. PageRank, HITS variations) has not yet proven itself to be an effective strategy and the content-based method seems to be still the most dominant factor in the Web track. In fact, the two best results in TREC-2002 topic distillation task were achieved by the baseline systems that used only the content-based methods (MacFarlane, 2003; Zhang et al., 2003).

[1] Current test collection of the Web track (i.e. .GOV) consists of 1.25 million Web pages (19 gigabytes) from .gov domain, which is larger, less diverse and likely to be of higher quality than the previous collection (i.e. WT10g), which was a 10 gigabyte subset of the Web crawl from Internet Archive.

In our earlier studies (Yang, 2002a; Yang, 2002b), where we investigated various fusion approaches for ad-hoc retrieval using the WT10g collection, we found that simplistic approach that combine the results of content- and link-based retrieval results did not enhance retrieval performance in general. TREC participants in recent Web track environment, however, found that use of non-textual information such as hyperlinks, document structure, and URL could be beneficial for specific tasks such as topic distillation and named/home page finding tasks. We believe that this is not only due to the change in the retrieval environment (i.e. test collection, retrieval task) but also the result of more dynamic approach to combining multiple sources of evidence than straightforward result merging. Thus, our focus in TREC-2003 Web track was in exploring fusion strategies that utilize various information sources in a dynamic manner to optimize retrieval for specific search environment. For our experiment, we used the experimental fusion retrieval system called WIDIT[2] to combine content and link information, and then reranked the combined result based on heuristics arrived at from dynamic system tuning process.

2. WIDIT

Basic approach of WIDIT in the Web track consisted of four main phases: indexing, searching, result merging, and reranking. Indexing phase involved indexing various sources of evidence to generate multiple indexes, which was followed by the searching phase that produced multiple result sets from using different query formulations against multiple indexes. The result sets were combined using weighted sum formula, after which a reranking heuristics were applied to optimize the ranking of the merged results. The overview of WIDIT system architecture is displayed in Figure 1.

2.1 Indexing Module

WIDIT preprocessed documents by removing HTML tags and stopwords and applying the simple plural remover (Frakes & Baeza-Yates, 1992)[3]. The stopwords consisted of non-meaningful words such as words in a standard stopword list, non-alphabetical words, words consisting of more than 25 or less than 3 characters, and words that contain 3 or more repeated characters. Hyphenated words were split into parts before applying the stopword exclusion, and acronyms and abbreviations were kept as index terms[4].

In addition to extracting body text terms (i.e. terms between <body> and </body> tags), WIDIT extracted terms from document title, meta keywords and descriptions, and "emphasized" text (e.g. text with <b>, <em>, <font>, <u>, <h1> tags) as well as extracting terms from the anchor texts of incoming links. Thus, WIDIT created three sets of term indexes: first based on document content (i.e. body index), second based on document structure (header index), and third based on link structure (anchor index).

In order to enable incremental indexing as well as to scale up to larger collections, each of the indexes consisted of smaller subcolllections, which were created and searched in parallel. The whole collection term statistics were derived after the creation of the subcollections.

2.2 Retrieval Module

The retrieval component of WIDIT was based on a Vector Space Model (VSM) using the SMART length-normalized term weights as was implemented in IRIS (Yang & Maglaughlin, 2000). Documents were ranked in decreasing order of the inner product of document and query vectors,

$$\mathbf{q}^{\mathrm{T}}\mathbf{d}_i = \sum_{k=1}^{t} q_k d_{ik}, \qquad (1)$$

[2] WIDIT (Web Information Discovery Integrated Tool; http://widit.slis.indiana.edu/), which extends IRIS research (http://ils.unc.edu/iris/) at the University of North Carolina, is an experimental IR system with a suite of modular retrieval tools designed for easy integration of multiple Web IR approaches. WIDIT is currently being developed in the School of Library and Information Science at Indiana University.

[3] The simple plural remover was chosen to speed up indexing time and to minimize the overstemming effect of more aggressive stemmers.

[4] Acronym and abbreviation identification was based on simple pattern matching of punctuations and capitalizations.

where q_k is the weight of term k in the query, d_{ik} is the weight of term k in document i, and t is the number of terms in the index. SMART *Lnu* weights with the slope of 0.3 were used for document terms (Buckley et al., 1997), and SMART *ltc* weights (Buckley et al., 1995) were used for query terms. *Lnu* weights attempt to match the probability of retrieval given a document length with the probability of relevance given that length (Singhal et al., 1996).

Two sets of queries, one resulting from simple stop and stemming, and another with phrases, acronyms and abbreviations extracted, were applied against three sets of document indexes[5] to produce six sets of retrieval results[6].

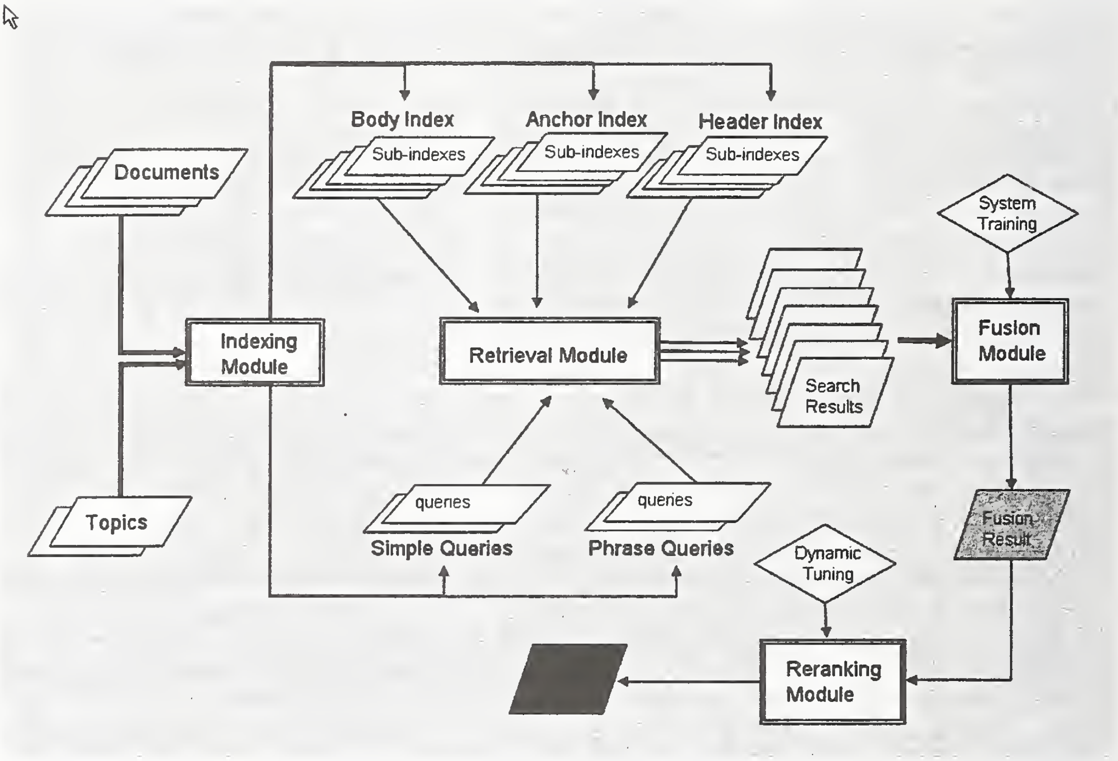


Figure 1. WIDIT System Architecture

2.3 Fusion Module

In post-retrieval fusion, where multiple sets of search results are combined after retrieval time, two of the most common fusion formulas are *Similarity Merge* (Fox & Shaw, 1995; Lee, 1997) and *Weighted Sum* (Bartell et al., 1994; Thompson, 1990). The similarity merge formula multiplies the sum of fusion component scores for a document by the number of fusion components that retrieved the document (i.e. overlap), based on the assumption that documents with higher overlap are more likely to be relevant. Instead of relying on overlap, the weighted sum formula sums fusion component scores weighted with the relative contributions of the fusion components that retrieved them, which is typically estimated based on training data. Both formulas compute the fusion score of a document by a linear combination of fusion component scores.

[5] Body text index consisted of title and body text terms. Anchor text index consisted of title and inlink anchor text terms. Header text index consisted of title, meta keywords and descriptions, and emphasized text terms.

[6] In practice, retrieval for each document index consisted of parallel searches of 46 subcollections using the whole collection term weights, whose results were merged and sorted by document score.

In our earlier study (Yang, 2002b), similarity merge approach proved ineffective when combining content- and link-based results, so this time we tried three variations of the weighted sum fusion formula, which were shown to be more effective in combining fusion components that are dissimilar (Yang, 2002a). Equation (2) describes the simple *Weight Sum* (WS) formula, which sums the normalized system scores multiplied by system contribution weights. Equation (3) describes the *Overlap Weight Sum* (OWS) formula, which multiplies the WS score by overlap. Equation (4) describes the *Weighted Overlap Weighted Sum* (WOWS) formula, which multiplies the WS score by overlap weighted by system contributions:

$$FS_{WS} = \Sigma(w_i * NS_i), \qquad (2)$$

$$FS_{OWS} = \Sigma(w_i * NS_i * olp), \qquad (3)$$

$$FS_{WOWS} = \Sigma(w_i * NS_i * w_i * olp), \qquad (4)$$

where: FS = fusion score of a document,
w_i = weight of system i,
NS_i = normalized score of a document by system i,
$= (S_i - S_{min}) / (S_{max} - S_{min})$
olp = number of systems that retrieved a given document.

The normalized document score, NS_i, is computed by Lee's min-max formula (1996, 1997), where S_i is the retrieval score of a given document and S_{max} and S_{min} are the maximum and minimum document scores by method i.

One of the main challenges in using the weighted fusion formula lies in determination of the optimum weights for each system (w_i). We assessed various weight combinations[7] (e.g. 0.9 for body text, 0.08 for header text, 0.01 for anchor text) with the training data of past Web track results to tune our fusion module.

2.4 Reranking Module

In order to optimize retrieval performance in top ranks, fusion results were reranked based on combinations of site compression technique and content-link evidence ranking heuristic. The site compression involved clustering results by sites, sorting sites by the highest document score of each site, and floating the top *n* documents from top *m* sites to the top ranks. The content-link evidence ranking heuristic consisted of a set of ranking and document score boosting rules arrived at by dynamic tuning process involving interactive retrieval and manual system tuning in real time. The dynamic tuning process was applied to the best single and best fusion systems to "tune" the ranking heuristic.

The dynamic tuning component of WIDIT produces retrieval results that display individual scores for each source of evidence such as inter/intrasite in/outdegree, phrase/proximity match counts in body/header/anchor texts, and query term matches in URL as well as ranking and retrieval scores before/after fusion and site compression. We performed a series of dynamic tuning sessions using training data, which involved repeated cycles of retrieval and tuning the reranking heuristic based on real time evaluation of retrieval results. In contrast to the static tuning of fusion formulas, dynamic tuning process, though ad-hoc, allows tuning of systems with numerous parameters by leveraging human intelligence. The main components of content-link evidence ranking heuristic we used were inter/intrasite in/outdegree (e.g. boost score if large outdegree for topic distillation, boost score if large indegree for home/named page finding), phrase/proximity match (e.g. boost ranking if phrase match in title or anchor text), and query term match in URL (e.g. boost to top 10 rank if acronym match in URL). The reranking heuristics for home/named page finding task also involved the query classification component, which assigned different emphasis on evidence sources according to the query type. The queries were classified into either named page or home page based on term occurrence patterns observed in training queries.

[7] Eight weight combinations for each fusion formula (24 systems per task) were examined.

2.5 Topic Distillation vs. Home/Named Page Finding Tasks

WIDT systems for topic distillation and home/named page finding runs shared the indexing and searching modules, but used different fusion formulas arrived at from different training data. The reranking module of topic distillation systems employed both site compression and content-link evidence reranking heurisitic, but home/named page finding systems used only the reranking heuristics that included the query classification component and emphasized the acronym matching.

The training data for topic distillation systems were 2002 topic distillation topics and relevance judgments, and home/named page finding system used 2002 named page finding topics and relevance judgments as training data. Unfortunately, both training data were problematic. The topic distillation task in TREC-2003 was about finding relevant "home pages" given a broad query, which introduced bias towards home page finding approaches not present in the training data. As the name indicates, home/named page finding task in TREC-2003 combined home page and named page finding tasks, only half of which were present in the training data used. In addition to suboptimal training data, the topic distillation systems were trained based on precision at rank 10 (P@10), whereas the main evaluation metric for topic distillation in TREC-2003 was changed to R-precision.

3. Results

In the topic distillation test runs using training data, the best single system was *widittdb1* (P@10 = 0.168), which used simple query and body text index. The best fusion runs, which used fusion weights of 0.9, 0.01, 0.09 for body, header, and anchor text respectively, improved the baseline (*widittdb1*) results by 8% (P@10 = 0.182). The reranking of fusion results improved the baseline results by 17% (P@10 = 0.196). In the named page finding test runs, same baseline system resulted in mean reciprocal rank (MRR) of 0.471, which fusion with weights 0.8, 0.19, and 0.01 improved by 10% (MRR = 0.520) but reranking failed to improve (MRR = 0.471).

The official run results were comparable to test run results in that fusion improved retrieval performance in both tasks and reranking further enhanced performance in topic distillation runs according to P@10. The official home/named page finding runs, which consisted of the baseline and the best fusion run were comparable to test runs in that the fusion run (MRR = 0.400) improved the baseline result by 10% (MRR = 0.362). The official topic distillation runs showed 21% improvement by fusion (P@10 = 0.092) and 29% improvement by reranking (P@10 = 0.098) over baseline (P@10 = 0.076).

3.1 TREC System Rankings

By official TREC system rankings, which ranked all TREC topic distillation runs by R-precision, WIDIT appeared to perform rather poorly when compared with other TREC systems (Table 1). By R-precision, the best WIDIT run, which was the baseline run, was ranked 71 of 107 systems (18 of 23 groups). When we reranked the systems using P@10, however, WIDIT runs were ranked much higher (Table 2). In fact, the best WIDIT run, which was the reranked fusion run, ranked 28 of 107 systems (12 of 23 groups). When ranked by mean average precision (Table 3), WIDIT ranked 38 of 107 systems (11 of 23 groups). Table 4 and 5, which shows system rankings for home/named page finding runs, reflects the poor performance level of WIDIT[8], which we tried to compensate for in post-submission runs described in the next section.

Table 1. Topic Distillation ranking by Mean R-Precision (MRP)

	MRP	MAP	avgP@10
Best TREC system	**0.1636**	0.1543	0.1240
Median TREC system	**0.0699**	0.0896	0.0700
Best WIDIT system	**0.0736**	0.1016	0.0760
Worst TREC system[9]	**0.0181**	0.0250	0.0160

[8] WIDIT fusion run ranked 48 of 75 systems (13 of 19 groups) by MRR, and 41 of 75 (13 of 19 groups) systems by mean success rate at rank 10.

[9] Outlier was excluded to keep the system performance comparisons in perspective.

Table 2. Topic Distillation ranking by Mean Precision at rank 10 (avgP@10)

	MRP	MAP	**avgP@10**
Best TREC system	0.1485	0.1387	**0.1280**
Best WIDIT system	0.0626	0.0787	**0.0980**
Median TREC system	0.0871	0.1057	**0.0807**
Worst TREC system	0.0181	0.0250	**0.0160**

Table 3. Topic Distillation ranking by Mean Average Precision (MAP)

	MRP	**MAP**	avgP@10
Best TREC system	0.1636	**0.1543**	0.1240
Best WIDIT system	0.0736	**0.1016**	0.0760
Median TREC system	0.0699	**0.0896**	0.0700
Worst TREC system	0.0230	**0.0222**	0.0200

Table 4. Home/Name Page ranking by Mean Reciprocal Rank (MRR)

	MRR	avgS@10
Best TREC system	**0.727**	89.3
Median TREC system	**0.496**	64.3
Best WIDIT system	**0.400**	66.3
Worst TREC system	**0.065**	8.7

Table 5. Home/Name Page ranking by Mean Success Rate at rank 10 (avgS@10)

	MRR	**avgS@10**
Best TREC system	0.727	**89.3**
Median TREC system	0.465	**68.3**
Best WIDIT system	0.400	**66.3**
Worst TREC system	0.065	**8.7**

3.2 Post-submission Runs

After receiving the results of the official TREC runs, we conducted post-submission analysis to discover and address the shortcomings of the submitted systems. In retrospect, it was easy to see the negative effects of improper system tuning that resulted from biased training data as well as system overtuning that penalized system rankings based on the different evaluation measure than one used in training. To compensate for these shortcomings, we conducted another cycle of dynamic tuning iterations that involved the adjustments in reranking and query classification heuristics and the implementation of home page identification method. The WIDIT dynamic tuning component was modified to produce ranking by both P@10 and R-precision, so that system tuning could be optimized for both evaluation measures.

Home page identification was based on URL typing (Tomlinson, 2003; Kraaij et al., 2002), where Web pages are classified into categories of root, subroot, path, and file. The root page was defined as URL with zero slash counts or URL that ends with home page file name (e.g. index.htm, default.htm) and 1 slash count. Subroot page was defined as home page ending with 2 slash count, and path page was defined as home page ending with 3 or more slash count. The file page was defined as URL that meets none of these conditions. Based on the observations from post-submission analysis, which suggested the strong performance of fusion results in top ranks as well as the importance of home page finding approaches, following rank boosting rules were added to the reranking heuristic:

- Keep top 5 ranks static
- Boost the rank of potential home pages (root, subroot, path)
- Boost the rank of file type page with two or more query terms in URL
- Stop if top 20 ranks are filled

The query classification heuristics for home/named page finding task was also improved by adding additional rules such as classifying queries that ends in all capitalized words as home page queries.

3.3 Post-submission Results

The post-submission topic distillation run results that introduced home page finding bias and tuned the system for both P@10 and R-precision improved system ranking by MRP from 71 to 23 (Table 6), system ranking by avgP@10 from 28 to 25 (Table 7), and system ranking by MAP from 38 to 10 (Table 8). The post-submission home/named page finding run results, however, were little different from official results.

Table 6. Post-submission Topic Distillation ranking by Mean R-Precision (MRP)

	MRP	MAP	avgP@10
Best TREC system	**0.1636**	0.1543	0.1240
Best WIDIT system	**0.1139**	0.1281	0.0980
Median TREC system	**0.0699**	0.0896	0.0700
Worst TREC system	**0.0181**	0.0250	0.0160

Table 7. Post-submission Topic Distillation ranking by Mean Precision at rank 10 (avgP@10)

	MRP	MAP	**avgP@10**
Best TREC system	0.1485	0.1387	**0.1280**
Best WIDIT system	0.0626	0.1216	**0.0981**
Median TREC system	0.0871	0.1057	**0.0807**
Worst TREC system	0.0181	0.0250	**0.0160**

Table 8. Post-submission Topic Distillation ranking by Mean Average Precision (MAP)

	MRP	**MAP**	avgP@10
Best TREC system	0.1636	**0.1543**	0.1240
Best WIDIT system	0.1139	**0.1281**	0.0980
Median TREC system	0.0699	**0.0896**	0.0700
Worst TREC system	0.0230	**0.0222**	0.0200

3. Discussion

In this year's topic distillation task, there were on the average 10.32 relevant documents per topic (5 topics with 1 relevant document, 3 topics with 2 relevant documents, 20 topics with 5 or fewer relevant documents, 33 topics with 10 or fewer relevant documents). The best WIDIT topic distillation run results had 17 topics with zero P@10, 23 topics with zero R-Precision, and only 4 topics with relevant documents at top 20 ranks. The key question for WIDIT topic distillation runs, therefore, is why they performed so poorly with topics with few relevant documents. Whether this is due to some WIDIT specific factors or it is a TREC system-wide phenomenon remains to be seen.

In the home/named page finding task, WIDIT exhibited suboptimal performance, which we attribute largely to incomplete training data (e.g. omission of home page finding training data) based on the following observation: the best WIDIT home/named page run had 24 home page topics with first correct answer beyond rank 100, compared to 6 named page topics with first correct answer beyond rank 100.

Overall, we believe fusion is a promising area of investigation for Web IR. Our results show that exploiting the richness of Web search environment by combining multiple sources of evidence via result merging and dynamic system tuning can enhance retrieval performance in the topic distillation task. As for the home/named page finding task, we suspect our approach was hampered by incomplete training data, which will be investigated in a follow-up study.

References

Amitay, E., Carmel, D., Darlow, A., Lempel, R., & Soffer, A. (2003). Topic Distillation with Knowledge Agents. *Proceedings of the11th Text Retrieval Conference (TREC 2002)*, 263-272.

Bartell, B. T., Cottrell, G. W., & Belew, R. K. (1994). Automatic combination of multiple ranked retrieval systems. *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval.*

Buckley, C., Salton, G., & Allan, J., & Singhal, A. (1995). Automatic query expansion using SMART: TREC 3. *Proceeding of the 3rd Text Rerieval Conference (TREC-3)*, 1-19.

Buckley, C., Singhal, A., & Mitra, M. (1997). Using query zoning and correlation within SMART: TREC 5. *Proceeding of the 5th Text REtrieval Conference (TREC-5)*, 105-118.

Craswell,N., & Hawking, D. (2003). Overview of the TREC-2002 web track. *Proceedings of the 11th Text Retrieval Conference (TREC 2002)*, 86-95.

Frakes, W. B., & Baeza-Yates, R. (Eds.). (1992). *Information retrieval: Data structures & algorithms.* Englewood Cliffs, NJ: Prentice Hall.

Fox, E. A., & Shaw, J. A. (1995). Combination of multiple searches. *Proceeding of the3rd Text Rerieval Conference (TREC-3)*, 105-108.

Gurrin, C., & Smeaton, A.F. (2001). Dublin City University experiments in connectivity analysis for TREC-9. *Proceedings of the 9th Text Retrieval Conference (TREC-9)*, 179-188.

Hawking, D., & Craswell, N. (2002). Overview of the TREC-2001 web track. *Proceedings of the 10th Text Retrieval Conference (TREC 2001)*, 25-31

Hawking, D., & Craswell, N., Thistlewaite, P., & Harman, D. (1999). Results and challenges in web search evaluation. *Proceedings of the 8th WWW Conference*, 243-252.

Hawking, D, Voorhees, E, Craswell, N., & Bailey, P. (2000). Overview of the TREC-8 web track. *Proceedings of the 8th Text Retrieval Conference (TREC-8)*, 131-148.

Kraaij, W., Westerveld, T., & Hiemstra, D. (2002). The Importance of Prior Probabilities for Entry Page Search. *Proceedings of the 25th ACM SIGIR Conference on Research and Development in Information Retrieval*, 27-34

Lee, J. H. (1997). Analyses of multiple evidence combination. *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, 267-276.

Savoy, J., & Picard, J. (1998). Report on the TREC-8 Experiment: Searching on the Web and in Distributed Collections. *Proceedings of the 8th Text Retrieval Conference (TREC-8)*, 229-240.

Savoy, J., & Rasolofo, Y. (2001). Report on the TREC-9 experiment: Link-based retrieval and distributed collections. *Proceedings of the 9th Text Retrieval Conference (TREC-9)*, 579-516.

Singhal, A., Buckley, C., & Mitra, M. (1996). Pivoted document length normalization. *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, 21-29.

Singhal, A., & Kaszkiel, M. (2001). A case study in Web search using TREC algorithms. *Proceedings of the 11th International WWW Conference*, 708-716.

Thompson. P. (1990). A combination of expert opinion approach to probabilistic information retrieval, part 1: The conceptual model. *Information Processing & Management*, 26(3), 371-382.

Tomlinson, S. (2003). Robust, Web and Genomic Retrieval with Hummingbird SearchServer at TREC 2003. *The 12th Text REtrieval Conference (TREC 2003) Notebook*, 372-385.

Voorhees, E., & Harman, D. (2000). Overview of the Eighth Text Retrieval Conference. *Proceedings of the 8th Text Retrieval Conference (TREC-8)*, 1-24.

Yang, K. (2002a). Combining Text-, Link-, and Classification-based Retrieval Methods to Enhance Information Discovery on the Web. (*Doctoral Dissertation*. University of North Carolina).

Yang, K. (2002b). Combining Text- and Link-based Retrieval Methods for Web IR. *Proceedings of the 10th Text Rerieval Conference (TREC2001)*, 609-618.

Yang, K. & Maglaughlin, K. (2000). *IRIS at TREC*-8. *Proceedings of the 8th Text Rerieval Conference (TREC-8)*, 645-656.

Zhang, M., Song, R., Lin, C., Ma, S., Jiang, Z., Jin, Y., Liu, Y., & Zhao, L. (2003). THU TREC 2002: Web Track Experiments. *Proceedings of the 11th Text Retrieval Conference (TREC 2002)*, 591-594.

TREC NOVELTY TRACK AT IRIT – SIG

Taoufiq Dkaki[(1,2)], Josiane Mothe[(1,3)]

(1) Institut de Recherche en Informatique de Toulouse, 118 Rte de Narbonne, 31062 Toulouse CEDEX
(2) Université Toulouse le Mirail, ISYCOM/GRIMM,
(3) Institut Universitaire de Formation des Maîtres Midi-Pyrénées

Abstract

In TREC 2003, IRIT improved the strategy that was introduced in TREC 2002. A sentence is considered as relevant if it matches the topic with a certain level of coverage. This coverage depends on the category of terms used in the texts. Different types of terms have been defined: highly relevant, scarcely relevant, non-relevant and highly non-relevant. With regard to the *novelty part*, a sentence is considered as novel if its similarity with previously processed sentences and with the n-best-matching sentences does not exceed certain thresholds.

1 Introduction

«The TREC novelty track is designed to investigate systems' abilities to locate relevant and new information within the ranked set of documents retrieved in answer to a TREC topic » [trec.nist.gov].

Retrieving relevant texts is traditionally based on computing a similarity between the representations of the information need (or topic) and the texts. This general statement has been applied to full documents as well as chunks of texts (passage retrieval). Intuitively, the same idea can be applied when sentences retrieval is involved. In TREC 2002 IRIT developed a new strategy in order to detect the relevant sentences. This approach has not been used in a more general context of document retrieval but we did used it previously and partially in document categorization (Mothe, 2002). In our approach a sentence is considered as *relevant* if it matches the topic with a certain level of coverage. This level of coverage depends on the category of the terms used in the texts. Three types of terms were defined for TREC 2002: highly relevant, scarcely relevant and non-relevant. In TREC 2003 we introduced a new class of terms: highly non-relevant terms. Terms from this category are extracted from the narrative parts of the queries that describe what is a non-relevant document. A negative weight can be assigned to these words. With regard to the *novelty part*, a sentence is considered as novel if its similarity with each previously processed and -selected as novel- sentences does not exceed a certain threshold. In addition, this sentence should not be too similar to a virtual sentence made of the n-best-matching sentences.

The results we obtained in TREC 2002 were quite good regarding the 'relevant' subtask. Indeed, for 36 topics (73%), the R*P was higher or equal to the average of the 42 runs which were submitted. In TREC 2003, we improved these results as we obtained 46 topics (92%) for which the F-measure (2*R*P/(R+P)) was equal or higher to the average of the 55 runs submitted. With regard to the 'novelty' part, when considering the retrieved sentences, we also obtained 46 topics (92%) for which the F-measure is higher or equal to the average of the 55 runs. However, an interesting result is that our method is better when there is some noise in the sentence set. Indeed the results are better when considering the retrieved sentences than when considering the relevance sentences only, relatively to other participants' methods (i.e. our system ranks better over the submitted runs). We obtained 41 topics (82%) for which the F-measure is higher or equal to the average of the 55 runs.

This paper is organized as follows: in section 2 we describe the method we used, including the way documents and topics are represented and the strategies we developed for the two sub-tasks (relevant part and novelty part either considering only relevant sentences or all retrieved sentences). In section 3 we present the results and comment them. We also present results we obtained on runs that were not submitted.

2 Description of the method

2.1 Document and topic representation

In our method, topics and sentences are considered as chunks of text. Each chunk is pre-processed the same way in order to extract representative terms. Then, terms extracted from a given topic are categorized into different groups: highly relevant terms (HT), scarcely relevant terms (LT) and highly non-relevant terms (IT). Notice that non-relevant terms (iT) correspond to stop words. Each text is finally represented by these sets of terms, weights being associated with each term.

2.1.1 Text processing

Texts are processed using the following method:

1. Stop words are removed,
2. The remaining words are normalized using a dictionary that provides a common root for different words. This dictionary contains 21291 entries.
3. Alternatively phrases are extracted. Phrases correspond to frequent sequences of words or frequent sequences of word roots.

2.1.2 Topic processing

A topic is pre-processed in order to mark-up the sentences that describe the information relevancy and the sentences that describe the non-relevancy (see Figure 1: NarrativeRel and NarrativeNonRel tags).

Topic: 35
Title: NATO, Poland, Czech Republic, Hungary
Type: event
Descriptive: Accession of new NATO members: Poland, Czech Republic, Hungary, in
1999.
NarrativeRel: Identity of current and newly-invited members, statements of support for and opposition to NATO enlargement and steps in the accession process and related special events are relevant. Impact on the new members, i.e., requirements they must satisfy, and their expectations regarding the implications for them are relevant. Progress in the ratification process is relevant.
NarrativeNonRel: Future plans for NATO expansion, identification of nations admitted on previous occasions, and comments on future NATO structure or strategy are not relevant.

Figure 1: topic 35 (TREC 2003)

Then it is analyzed in order to extract the representative terms (words or phrases) as explained in the previous section. Each term is then weighted and categorized into 3 groups:

- Highly relevant terms are terms that get a weight greater than τ_H ,
- Scarcely relevant terms are terms that get a weight equal to τ_L ,
- Highly non-relevant terms are terms that are associated with non-relevancy in the narrative part of the documents.

More precisely, the formula used to compute the term weights is defined as follows:

Given Q_k a topic and t_i a term, $T_k = \{t_i \in Q_k / t_i \textit{ is not a stop word}\}$

$T_k = TT_k \ U \ TD_k \ U \ TNR_k \ U \ TNN_k$ where TT_k corresponds to the set of terms extracted from the Title of the topic, TD_k from the Descriptive, TNR_k from the NarrativeRel and TNN_k is the NarrativeNonRel topic part.

$tf_{i,k,P}$ is the frequency of t_i in the TP_k part, $P \in \{T, D, NP, NN\}$

The term weight regarding a topic is computed as follows:

$$\omega_{1,i,k} = \sum_{P \in \{T,D,NP\}} \mu_P \cdot tf_{i,k,P}$$

$$\omega_{2,i,k} = \mu_{NN} \cdot tf_{i,k,NN}$$

$$\begin{aligned} \omega_{i,k} = \omega_{1,i,k} + f(\omega_1, \mu_{NN}) \cdot \omega_{2,i,k} \qquad \textit{where} \quad f(\omega_1, \mu_{NN}) &= 0 \quad \textit{if} \quad \omega_{1,i,k} > 0 \ \textit{and} \ \mu_{NN} < 0 \\ &= 1 \quad \textit{otherwise} \end{aligned}$$

$$\begin{aligned} weight(t_i, Q_k) &= \omega_{i,k} \quad \textit{if} \quad \omega_{i,k} \geq \tau_H \\ &= \omega_{2,i,k} \quad \textit{if} \quad \omega_{1,i,k} = 0 \\ &= \tau_L \quad \textit{if} \quad 0 < \omega_{i,k} < \tau_H \\ &= 0 \qquad \textit{otherwise} \end{aligned}$$

τ_L and τ_H are used in order to obtain a significant difference -in terms of importance- between highly relevant terms and scarcely relevant terms. The weights associated to scarcely relevant terms are set to τ_L (1 in the experiments submitted to TREC). τ_H is set to 3 in the TREC runs. This formula is also used in order to take into account highly non-relevant terms.

The term weight is used to categorized a term into one of the groups defined as follows:

$$HT_k = \{t_i / t_i \in \bigcup\{TT_k, TD_k, TNR_k\} \ and \ weight(t_i, T_k) > \tau_L\}$$

$$LT_k = \{t_i / t_i \in (TNN_k - \bigcup\{TT_k, TD_k, TNR_k\}) \ and \ weight(t_i, T_k) = \tau_L\}$$

$$iT_k = \{t_i / \ weight(t_i, TP_k) = 0 \quad \forall P \in \{T, D, NR, NN\}\}$$

$$IT_k = \{t_i / t_i \in TNN_k \ and \ weight(t_i, T_k) < 0\}$$

2.1.3 Document processing

Each sentence of a document is considered as a text and the representative terms are extracted as explained in the section 2.1.1. To each term is associated a weight defined as follows:

Given S_j a sentence, t_i a term and $tf_{i,j}$ is the frequency of t_i in S_j. $\qquad weight(t_i, S_j) = tf_{i,j}$

2.2 Relevant sentences

In order to decide if a sentence is relevant, we associate three components to each sentence:

- a score that reflect the sentence – topic matching :

Given a topic Q_k and a sentence S_j

$$Score\,(S_j, Q_k) = \sum \left(weight(t_i, S_j) \cdot weight(t_i, Q_k) \right)$$

- and two groups of terms:

$$HS_j = \{t_i / t_i \in (Sj \cap HT_k)\}$$

$$LS_j = \{t_i / t_i \in (Sj \cap LT_k)\}$$

HS_j corresponds to the highly relevant terms from the topic that also occurs in the sentence,

LS_j corresponds to the scarcely relevant terms from the topic that also occurs in the sentence.

Note that IT_k and iT_k sets are only used to calculate the term weight and it is not used in the sentence selection process.

A given sentence S_j is then considered as relevant iff :

$$Score\,(S_j, Q_k) > f\left(\frac{|LS_j|}{|LS_j| + |HS_j|}\right) \cdot |HT_k| + g\left(\frac{|HS_j|}{|LS_j| + |HS_j|}\right) \cdot |LT_k|$$

where $|X|$ is the number of elements of X

In the experiments that correspond to the runs sent to TREC, the function $f(\,)$ and $g(\,)$ have been set to:

$$f(x) = 2 - 1.5\,x \qquad and \qquad g(x) = 0.85 - 0.5\,x$$

2.3 Novel sentences

To decide if a sentence p is to be considered as novel, we compute the similarity between the sentence p and the previous successfully processed sentences p_i (novel) and the similarity between the sentence p and a sentence P' automatically built from the union of the set of p_i:

Given

- $\Pi = \{p_1, p_2, \ldots, p_n\}$ a set of sentences labeled as novel and $P' = \bigcup_{i \in \{1,\ldots,n\}} p_i$, P' is a sentence made of the set of sentences from Π,
- $Sim(x, y\,)$ a function that compute a similarity between x and y and
- p a sentence for which the system has to decide if it brings new information.

We first compute the following similarities:

$Sim(p, P') = \alpha_p$ and for $i \in \{1,\ldots,n\}$ $Sim(p, p_i) = \omega_{p,i}$

We then consider the q best previous sentences:

for $i \in \{1,\ldots,n\}$ $\quad P_{p,i}$ is the series of sentences obtained by ordering Π in decreasing order of $\omega_{p,i}$.

$\beta_p = \sum_{i \in \{1,\ldots,q\}} Sim(p, P_{p,i})$ where $q \in \{4,5\}$ in the runs sent to TREC.

p is considered as redundant (not novel) iff:

$$\alpha_p \geq \tau_1 \text{ and } \beta_p \geq \tau_2$$

where $\tau_1 = 1$ and $\tau_2 = 0.6$ for the runs sent to TREC.

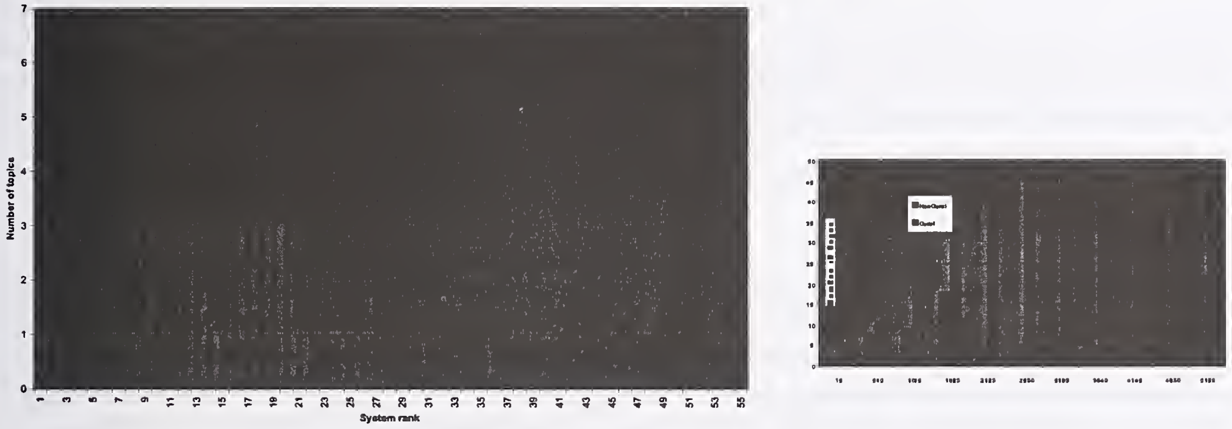
3 Results

This section presents the results we obtained with the method we developed and using the parameters as described in section 2. When comparing the results with the other runs, we can notice that our system is better in finding relevant sentences than in detecting novelty in the sentences. The difficulties of our system to detect novelty can be linked to the fact that the system does not take into account the order of the sentences in the documents.

3.1 Relevant sentences

Figure 1 indicates the number of topics for which our best system (or run) has been ranked at the Xth position among the 55 runs according to the F-Measure. For example, our method obtains the best results for 0 topic, the second position for 1 topics, the third for 1 topics, etc. and has a rank higher than 36th for only one topic (see figure 1.a). Figure 1.b provides a graph that summarizes figure 1.a by grouping together the results obtained for ranges of ranks. Additionally, the cumulative number of topics per range of system position is provided on the same graph. For example, we obtained a rank between 1 to 5 for 3 topics. The system obtains a rank equal or higher than 20 for 38 topics.

This clearly shows that our method is better than the average of the results. To be more precise, over the 50 topics, we obtained 46 topics (92%) for which the F-measure is higher or equal to the average of the 55 runs. And if we consider the run ranks, we obtained a rank higher or equal to the median (27) for 42 topics (84%).



a) Number of topics per run rank : detailed results

b) Number of topics per run rank : summarized results

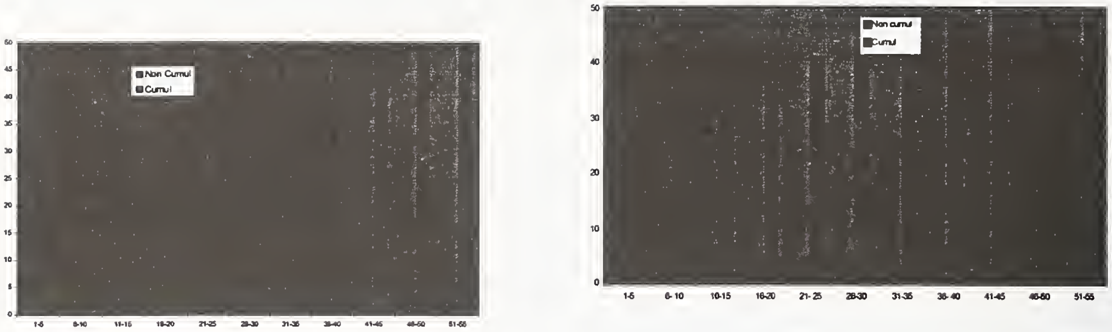
Figure 1: Number of topics per run rank – relevant sentences

3.2 New sentences

We present the results obtained in the second subtask the same way (see Figure 2). We distinguish the results when novel sentences are extracted from the retrieved sentences (TREC task 1 ; figure 2.1) and when they are extracted from the set of known relevant sentences (TREC task 2, figure 2.1).

Regarding the first case, over the 50 topics, we obtained 46 topics (92%) for which the F-measure is higher or equal to the average of the 55 runs. And if we consider the run ranks, we obtained a rank higher than the median (27) for 41 topics (82%).

However, when considering the relevant sentences, over the 50 topics, we obtained 41 topics (82%) for which the F-measure is higher or equal to the average of the 55 runs. And if we consider the run ranks, we obtained a rank higher than the median (27) for 30 topics (60%).



a) Novelty from retrieved sentences

b) Novelty from relevant sentences

Figure 2: Number of topics per run rank – summarized results

3.3 Other results

We modified the term weighting function in order to take better into account the query part in which the term occurs. The best results for the *relevant* subtask we obtained are the following: over the 50 topics, we obtained 46 topics (92%) for which the F-measure is higher or equal to the average of the 55 runs. And if we consider the run ranks, we obtained a rank higher or equal to the median (27) for 45 topics (90%). For one topic we obtained the best rank.

4 Conclusion

The approach we developed leads to relevant results for the first part of the task (relevant sentences). Over the 50 topics, we obtained 46 topics (92%) for which the F-Measure is higher or equal to the average of the 55 runs. And if we consider the run ranks, we obtained a rank higher than the middle (26) for 42 topics. Our best-submitted run obtains the following results: Average precision 0.64, Average recall 0.58 and Average F 0.526. With regard to the second sub-task (novelty), the submitted results over the 50 topics, we obtained 41 topics (82%) for which the F-measure is higher or equal to the average of the 55 runs.

5 References

[trec.nist.gov] TREC web site.

(Dkaki et al., 2002) T. Dkaki, J. Mothe, J. Augé, Novelty track at IRIT-SIG, Text Retrieval Conference TREC 2002, pp 332-336, 2003.

(Luhn, 1960) Luhn, H., Keyword in Context Index for Technical Literature, American Documentation XI (4), 1960, 288-295.

(Mothe et al., 2002) J. Mothe., C. Chrisment, B. Dousset, J. Alaux, DocCube : multi-dimensional visualisation and exploration of large document sets, Journal of the American Society for Information Science and Technology, JASIST, Special topic section: web retrieval and mining, Guest Editor: Hsinchun Chen, 54 (7), pp. 650-659, March 2003.

(Corral, 1995) M-L. Corral, J. Mothe, How to retrieve and display long structured documents, *Basque International Worshop on Information Technology*, pp 10-19, 1995

Mercure at TREC'2003
Web track – Topic Distillation task

MOHAND BOUGHANEM, KAREN SAUVAGNAT, CECILE LAFFAIRE
IRIT- SIG,
118 Rte de Narbonne,
F- 31062 Toulouse CEDEX 4
Email : {bougha, sauvagna, laffaire}@irit.fr

1 – Summary

The tests performed for TREC'2003 web track were focused on the topic distillation part. The aim of our participation is to validate the results we obtained last year and to test the use of term proximity on Mercure model.
As last year, ad-hoc methodologies were used to answer the topic distillation task.
4 runs were submitted to NIST this year.

2 – Mercure model

Mercure is an information retrieval system based on a connexionist approach and modeled by a multi-layer network. The network is composed of a query layer (set of query terms), of a term layer (representing the indexing terms) and of a document layer [Bougha 99].
Mercure includes the implementation of retrieval process based on spreading activation forward and backward through the weighted links. Queries and documents can be used either as inputs or outputs. The links between layers are symmetric and their weights are based on the *tf-idf* measure inspired by OKAPI [Robertson 00] and SMART term weighting.

The query-term links are weighted as follows :

$$q_{ui} = \begin{cases} \dfrac{nq_u * qtf_{ui}}{nq_u - qtf_{ui}} \text{ if } (\mathrm{nq}_u > qtf_{ui}) \\ qtf_{ui} \text{ otherwise} \end{cases}$$

Where:
- q_{ui} : the weight of the term t_i in the query u
- qtf_{ui}: the query term frequency of t_i in the query u
- nq_u: the number of terms in the query u

The term-document link weights are expressed by :

$$d_{ij} = \frac{tf_{ij} * (h_1 + h_2 * \log(\frac{N}{n_i}))}{h_3 + h_4 * \frac{dl_j}{\Delta_l} + h_5 * tf_{ij}}$$

Where:

- d_{ij} : term-document weight of term t_i and document d_j
- tf_{ij}: term frequency of t_i in the document d_j
- N: total number of documents
- n_i: number of documents containing term t_i
- h_1, h_2, h_3, h_4 and h_5: constant parameters
- Δ_l : average document length

2.1. Query evaluation

The query evaluation is based on spreading evaluation. Each node computes an input and spreads an output signal. A query is evaluated as follows:

1 – The query u is the input of the network. Each node from the term layer computes an input value from this initial query: $In(t_i) = q_{ui}$ and then an activation value: $Out(t_i) = g(In(t_i))$, where g is the term layer activation function.

2 - Each term node propagates then this activation value to the documents nodes through the term-document links. Each document node computes an input value: $In(d_j) = \sum_i Out(t_i) * d_{ij}$ and an activation value: $Out(d_j) = g(In(d_j))$, where g is the document layer activation function.

The set of retrieved documents, $Output_u\ (Out(d_1), Out(d_2), \ldots, Out(d_N))$ is then ranked in a decreasing order of the activation value.

2.1. Term proximity

The ranking function (activation) was modified to take into account term proximity in a document [Kean 91]. Thus, documents having query terms close to each others compute a new input value:

$$In(d_j) = \sum_i (Out(t_i) * d_{ij}) * \sum_i \frac{\alpha}{prox_{i,i-1}}$$

Where:

- α is a constant parameter such as $\frac{\alpha}{prox_{i,i-1}} \geq 1$. α is set to 4 for the TREC'2003 experiments.
- $prox_{i,i-1}$ is the number of terms separating the query terms t_i and t_{i-1} in a window of α terms in the document. The query terms are ranked according to their position in the query text.

In other words, documents having close query terms (i.e. no more than α words separate query term t_i and query term t_{i-1} in the document content) increase their input value.

3 – Topic distillation task Experiments

3.1. Indexing methodology

The GOV collection was indexed with scripts allowing to take into account term positions in the documents. Terms are stemmed with Porter algorithm and a stop-word list is used in order to remove non-significant terms from the index.

The queries used for the runs were built from the title and the description fields of the topics.

3.2. Web methodology

4 runs were performed and submitted to NIST. All runs are ad-hoc retrieval, and have been performed with no relevance feedback and no query expansion.
The first two runs are based on a simple search without term proximity. The queries are built using the title field only for the run named *Merc1ti*, and the title and description for the second run named *Merc1td*.
The last two runs, named *Merc2tp* and *Merc2tm*, use term proximity, and only the title field of topics was taking into account.
Merc2tm was performed as follows: if the title field of a topic contains phrases (about 25% of the total number of queries), term proximity is chosen to perform the query, otherwise, simple search without term proximity is used. This choice was made manually.

4 – Results

4.1. Analysis

Table 1 describes the results obtained at TREC'2003 Topic Distillation Task for official runs.

Precision	**Merc1ti**	**Merc1td**	**Merc2tp**	**Merc2tm**
At 5 documents	0.0720	0.0400	0.0800	0.0960
At 10 documents	**0.0720**	**0.0400**	**0.0680**	**0.0760**
At 15 documents	0.0653	0.0360	0.0573	0.0667
At 20 documents	0.0560	0.0330	0.0520	0.0660
At 30 documents	0.0473	0.0313	0.0487	0.0593
At 100 documents	0.0290	0.0216	0.0290	0.0312
At 200 documents	0.0205	0.0145	0.0205	0.0225
At 500 documents	0.0110	0.0088	0.0108	0.0130
At 1000 documents	0.0070	0.0056	0.0069	0.0077
Exact	0.0784	0.0433	0.0669	0.0783

Table 1: Precision at n documents and exact precision for TREC'2003 official runs

Our best performing run for precision at 10 documents is *Merc2tm*, performed using simple search *and* term proximity, depending on the presence of phrases in the query. The R-precision for runs *Merc2tm* and *Merc1ti* is comparable.
The use of term proximity does not affect the results in a significant way.

Using title field only is more suitable than using both title and description fields (run *Merc1td*), as observed last year.

However, it can be noticed that the results we obtained this year are much worse than those obtained in TREC'2002 for the topic distillation task.

Indeed, in TREC'2002 our run *Mercah*, which was an ad-hoc run performed with the Mercure system, was ranked 6th for the precision at 10 documents. The algorithm used for the run *Mercah* is the same as the one used for the run *Merc1ti* performed this year.

The following table compares the precisions obtained in TREC'2002 and those obtained in TREC'2003 for comparable runs.

Precision	**Mercah 2002**	**Merc1ti 2003**	**Merc2tp[1] 2002**	**Merc2tp 2003**	**Merc2tm[1] 2002**	**Merc2tm 2003**
At 5 docs	0.2449	0.0720	0.2122	0.0800	0.2735	0.0960
At 10 docs	0.2163	0.0720	0.1816	0.0680	0.2224	0.0760
At 15 docs	0.2041	0.0653	0.1673	0.0573	0.1878	0.0667
At 20 docs	0.1765	0.0560	0.1378	0.0520	0.1582	0.0660
At 30 docs	0.1463	0.0473	0.1143	0.0487	0.1381	0.0593
At 100 docs	0.0898	0.0290	0.0810	0.0290	0.0882	0.0312
At 200 docs	0.0661	0.0205	/	0.0205	/	0.0225
At 500 docs	0.0356	0.0110	/	0.0108	/	0.0130
At 1000 docs	0.0203	0.0070	0.0185	0.0069	0.0203	0.0077
Exact	**0.1984**	**0.0784**	**0.1542**	**0.0669**	**0.2023**	**0.0783**

Table 2: Comparison of TREC'2002 and TREC'2003 runs

TREC' 2003 precisions are much more lower than those obtained in TREC'2002.

Nevertheless, the analysis of precisions should not be the only criterion of judgement. Table 3 compares the runs *Mercah* (TREC'2002) and *Merc1ti* (TREC'2003) against the published median runs in TREC'2002 and TREC'2003.

	Worst	**< Median**	**Median**	**> Median**	**Best**
2002 (Mercah)	5	2	16	20	6
2003(Merc1ti)	6	0	32	6	6

Table 3: Comparative results at precision at 10

In TREC'2002 and TREC'2003, 6 topics obtain the best results and almost the same number of topics obtain the worst results (5 in TREC'2002 and 6 in TREC'2003).

On the other hand, the distribution between "median" and ">median" is different. Indeed, in TREC'2002 more topics were above the median (20) compared to this year (6). In TREC' 2003, much more topics are exactly on the median.

4.2. Discussion

The first explanation we could give to argue such results comes from this year definition of relevant judgements. Table 4 compares the number of documents considered to be relevant by the assessors for all topics in TREC'2002 and TREC'2003.

[1] These runs were not submitted as official runs in the TREC'2002 web track.

	Number of relevant documents	Average number of relevant documents per topic
2002	1574	32,12
2003	516	10,32

Table 4: Comparison of the number of relevant documents per topic in TREC'2002 and TREC'2003.

In TREC'2003, the average number of documents per topic is 10.32 and has strongly fell down.

For topics with less than 10 documents relevant, the maximum possible scores is less than 1.0. So, it is almost impossible to obtain highest precision this year and it explains our results in term of precision in 2003. So precision at 10 documents is not an appropriate measure.

Let us consider another measure, the R-precision, which has became the main measure in TREC'2003. If we analyze each topic's scores for the best run performed with the original Mercure (*Merc1ti*), we note that for 20% of the topics, the R-precision is relatively high (i.e. higher than 0.2), but for about 60% of the topics, the R-precision is 0. In this last case, most of the topics were those for which the number of relevant documents is smaller than 5. In fact, if there are for example 2 relevant documents, and if they are retrieved at position 3 and 4 by the retrieval system, the R-Precision is 0, even if the relevant documents are in the top 5.

The second explanation of our results could come from the TREC'2003 topic distillation task definition.

Indeed, in [Craswell 02], authors maintain that" *for a few topics, the number of such resources is very much higher than expected. While hand-listed pages in Web directories tended to have short URL' S and high indegree, key resources from this year's track did not show such tendencies as strongly "and in conclusion"[...] the topic distillation task proved difficult to explain to both participants and assessors and there was considerable disparity between the interpretations of these two groups... the task is worth repeating in 2003 but more explanatory effort is needed*".

In TREC' 2002, ad hoc runs could obtain good results on the topic distillation task because the assessors judged more resources relevant than needed and these resources were not always home Page of site or short URL. In TREC'2003, the topic distillation task was better specified [Trec Guidelines 03]: "*We are concentrating solely on websites as resources. The task is to find as many different websites (represented by to their entry pages) as possible within the first ten results.* ".

Thus, considering this definition of the task, ad-hoc runs are no more suitable.

5– Conclusion

The goal of our participation was to test a ranking function based on term proximity on the Mercure model and to validate the results we obtained in TREC'2002 with ad-hoc strategies on the topic distillation task.

The use of term proximity does not improve the results in a significant way.

Moreover, for the 4 ad-hoc runs we submitted this year, performances are not as high as last year. In fact in TREC'2003, the topic distillation task has evolved. Indeed, the spirit of the topic distillation task is to take a large set of relevant results and distill it down to a few key home pages. An overview of the TREC 2003 web track for all participants lets appear that referring anchor text was important and that URL information and link structure was very useful in several cases. Thus, the top 5 groups at R-precision have used at least document structure, anchor text or link structure. Ad hoc runs show their performances regressed for this topic distillation task.

References

[Bougha 99] : M. BOUGHANEM, C. CHRISMENT, C. SOULE-DUPUY : *Query modification based on relevance back-propagation in ad-hoc environment.* Information Processing and Management, 35 (1999), pages 121-139, 1999.

[Craswell 02]: N. CRASWELL and D. HAWKING : *Overview of the TREC-2002 Web Track*, in Proceedings of TREC'2002. http://trec.nist.gov/pubs/trec11/papers/WEB.OVER.pdf

[Kean 91] : E. Michael KEEN*: The use of term position devices in ranked output experiments*, Journal of Documentation, v.47 n.1, p.1-22, March 1991 .

[Robertson 00]: S.E. ROBERTSON, S. WALKER: *Okapi/Keenbow at TREC-8*. In Proceedings of the TREC-8 Conference, National Institute of Standards and Technology, pages 151-161, 2000.

[Trec guidelines 03]: *Guidelines of the TREC'2003 Topic distillation task.* http://es.cmis.csiro.au/TRECWeb/

ITC-irst at TREC-2003:
the DIOGENE QA system

Milen Kouylekov, Bernardo Magnini, Matteo Negri, and **Hristo Tanev**
ITC-Irst, Centro per la Ricerca Scientifica e Tecnologica
Via Sommarive, 38050 Povo (TN), Italy
{kouylekov, magnini,negri,tanev}@itc.it

Abstract

This paper describes a new version of the DIOGENE Question Answering (QA) system developed at ITC-Irst. The recent updates here presented are targeted to the participation to TREC-2003 and meet the specific requirements of this year's QA main task. In particular, extending the backbone already developed for our participation to the last two editions of the QA track, special attention was paid to deal with the principal novelty factors of the new challenge, namely the introduction of the so-called *definition* and *list* questions. Moreover, we experimented with a first attempt to integrate parsing as a deeper linguistic analysis technique to find similarities between the syntactic structure of the input questions and the retrieved text passages. The outcome of such experiments, as well as the variations of the system's architecture and the results achieved at TREC-2003 will be presented in the following sections.

1 Introduction

The new version of DIOGENE described in this paper results from recent improvements to the well-tested backbone built in the framework of our participation to the last two editions of the TREC QA main task (see Magnini et al., 2001 and Magnini et al., 2002a) and to the first edition of the multiple language QA track at CLEF 2003 (Negri et al., 2003). This year, due to the availability of a relatively stable and reliable version of the system, most of the work concentrated on handling the new question classes introduced to complicate the TREC QA main task, namely the *definition* and *list* questions. To this aim, a specific module for definition questions (*e.g.* "*Who is Aaron Copland?*", "*What are fractals?*") has been created, which relies both on a set of specific hand-crafted answer patterns, and on the evaluation of the answer candidates through Web-based statistical techniques. Furthermore, as for list questions (*e.g.* "*Which past and present NFL players have the last name of Johnson?*"), the system was tuned by considering as correct answers all the candidates ranked over an experimentally determined threshold by the statistical answer validation component already described in (Magnini et al., 2002a).

Besides the *ad hoc* improvements specifically targeted to the TREC-2003 QA main task, some preliminary experiments were also carried out with the long-term goal of integrating parsing as a core technique to improve system's performance. More specifically, such integration is intended to improve part of the DIOGENE's basic components that usually fail when dealing with particular kinds of questions. For instance, *Answer-Type Identification* will benefit from the capability of finding more precisely the head of the (sometimes rather complex) NPs that follow the WH-word, as in "*What Boston Red Sox* ***infielder*** *was given his father's first name, with the letters reversed?*", or "*What country singer's first* ***album*** *was titled "Storms of Life"?*". Moreover, the introduction of parsing in the QA loop is a crucial step to refine the whole *Answer Extraction* process. With regard to this issue, while in the last year's version of DIOGENE this process was carried out only by considering the presence in a paragraph of named entities matching the answer type category, in the new version of the system we tried to consider the syntactic similarity between the input questions and the retrieved text passages as a further clue for candidate answers' selection. Being the extraction of answer candidates a critical issue, and one of the weakest modules in last year's version of DIOGENE, our experiments on parsing were mainly focused on this direction. The expected result was not only the improvement of system's performance over the types of questions it was already capable to deal with, but also some improvement over types of questions that the previous version of the system could not handle at all. As an example, this is the case of the very frequent (and apparently simple) questions whose answer is not a named entity (such as "*What instrument did Louis Armstrong play?*", and "*What color hair did Thomas Jefferson have before gray?*") and the "*HOW-DID*" questions (such as "*How did Jimi Hendrix die?*"), which represent a challenging direction for future developments.

Starting from these general premises, this paper will mainly describe the novelties and the experiments carried out to develop this year's version of DIOGENE. In particular, after a short overview of the system's architecture (Section 2), we will focus on the new module developed to handle definition questions (Section 3), and on the experiments carried out to use parsing as a technique to refine the answer extraction process (Section 4). Finally, Section 5 will conclude the paper presenting the results achieved by DIOGENE at TREC-2003, as well as some final remarks about strengths and weaknesses of our system.

2 DIOGENE's Architecture

The overall system's architecture (depicted in Figure 1) relies on the rather standard three-components backbone used for the participation to the last two editions of the TREC QA main task. Such a backbone relies on a *question processing* component, which is in charge of the linguistic analysis of input questions, a *search* component, which performs the query composition and the document retrieval, and an *answer extraction* component, which extracts the final answer from the retrieved text passages (see (Magnini et al., 2002a) for further details).

Within this rather conservative framework, the automatic answer validation technique developed last year still plays a crucial role. The algorithm, fully described in (Magnini et al., 2002a), relies on discovering relations between a question and the answer candidates by mining the Web or a large text corpus for their co-occurrence tendency. Summarizing, the answer validation process is carried out through the following steps:

1. Compute the set of representative keywords *Kq* and *Ka* both from the question and the answer.
2. From the extracted keywords compute a set of *validation patterns* (*i.e.* textual expressions where the question and the answer keywords co-occur closely).
3. Submit the validation patterns to the Web.
4. Estimate an *answer relevance score (ARS)* considering the number of retrieved documents.

The *ARS* is calculated on the basis of the number of *hits* (*i.e.* retrieved pages) by means of a statistical co-occurrence metric called *corrected conditional probability* (Magnini et al., 2002b). The formula we used is the following:

$$ARS(a) = \frac{P(Ka \mid Kq)}{P(Ka)^{2/3}} = \frac{hits(pattern(Ka, Kq))}{hits(Kq) * hits(Ka)^{2/3}} * |EnglishPages|$$

Such a general formula had to be specified to deal with the new kinds of questions presented in this year's edition of the TREC QA main task. In particular, while *factoid* questions were handled with the original *ARS* calculation formula, *list* questions required the experimental definition of a relative threshold to select a larger number of answers, and *definition* questions required the development of *ad-hoc* validation patterns. While the experimental setting of a threshold to capture relevant answers to an input list question is a relatively easy task, let's focus on the more interesting and challenging issue of answering definition questions.

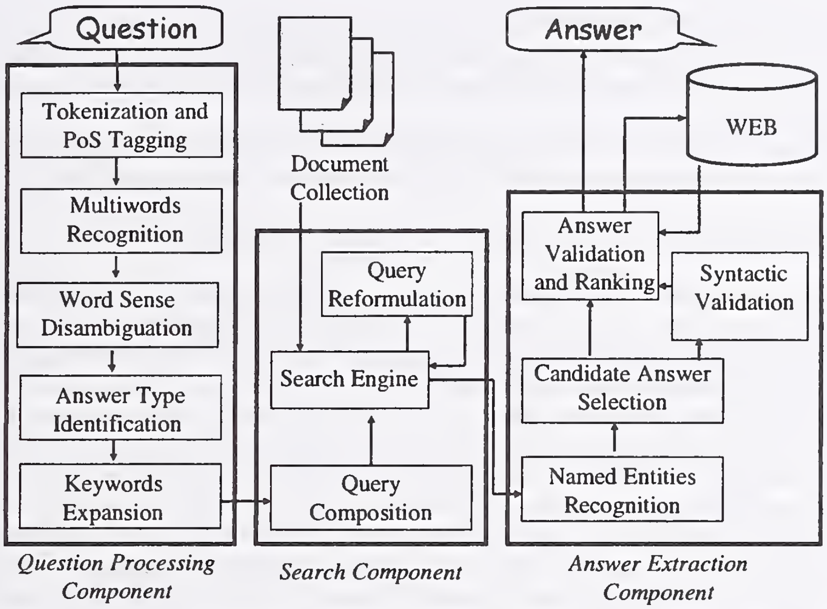

Question
Tokenization and PoS Tagging
Multiwords Recognition
Word Sense Disambiguation
Answer Type Identification
Keywords Expansion
Question Processing Component
Document Collection
Query Reformulation
Search Engine
Query Composition
Search Component
Answer
WEB
Answer Validation and Ranking
Syntactic Validation
Candidate Answer Selection
Named Entities Recognition
Answer Extraction Component

Figure 1. DIOGENE's Architecture.

3 Answering Definition Questions

In this year's TREC QA main task competition 10% of the questions belonged to the type definition. However, according to the evaluation scheme adopted, these questions contributed to the overall score up to 25%, thus forcing participants to invest on this particular aspect of research in QA. Our strategy relies on using patterns to extract the best text fragments where definitions are likely to occur (we call them "definition fragments") and then going to the Web to measure the co-occurrence between the question *focus* (*i.e.* the entity for which a definition is sought, such as "golden parachute" in the question "What is a golden parachute?") and the most important part of the definition (*i.e.* the so-called "definition core"), usually represented by an NP contained in the definition fragment.

3.1. Extraction and ranking of DEFINITION-FRAGMENTs

At the beginning of the process we use a small set of manually defined lexical patterns to extract and rank definition-like fragments. Being these patterns weighted, our technique resorts to calculating a score for every candidate fragment summing the weights from the matching.
For instance, the most used patterns for the extraction of candidate DEFINITION-FRAGMENTs are the following:

FOCUS {"who"|"what"|"which"} {"is"|"was"} DEFINITION-FRAGMENT
nonPrep FOCUS {"is"|"was"} DEFINITION-FRAGMENT

nonPrep FOCUS (DEFINITION-FRAGMENT)
nonPrep FOCUS, DEFINITION-FRAGMENT
DEFINITION-FRAGMENT “known as” FOCUS
DEFINITION-FRAGMENT “called” FOCUS

where “DEFINITION-FRAGMENT” stands for the part we take for further processing and “nonPrep” stands for any word which is not a preposition. We also used, as further clues, the presence of hypernyms of the focus and words from the WordNet gloss, in case the focus is found in WordNet hierarchy.
The following step consists of sorting all the DEFINITION-FRAGMENTs according to their score and passing them to the next module which is in charge of querying the Web.

3.2. Extraction and validation of definition cores (DEF-COREs)

At this stage of the process, we consider as possible appropriate answers to a definition question all the noun phrases that appear close to the question focus in the DEFINITION-FRAGMENTs retrieved from the document collection. For example, given the DEFINITION-FRAGMENT:
“… The Italian skier Alberto Tomba won the World Cup in 1993…”
the corresponding candidate answer phrases will be “Italian skier” and “World Cup”. We think that such noun phrases, to which we refer with the term DEF-CORE, represent the core of a good definition or at least an introduction to it. DEF-COREs are extracted from the candidate passages by means of the shallow parser Scoll (Abney, 1996).
Once the DEF-CORES have been extracted, their validation (*i.e.* the calculation of the corresponding *Answer Relevance Score*) is carried out automatically by means of the statistical answer validation technique outlined in Section 2. However, such technique allows that only one simple validation pattern, namely the generic pattern [Kq NEAR Ka], is considered. By definition, this pattern will lead to the number of pages where the keywords from the question (Kq) appear close to the keywords from the answer (Ka). However, for each specific question type it is possible to define one or more validation patterns which are much more efficient than the generic validation pattern. In particular, for the definition questions we can use the following list of more precise validation patterns (where Kq and Ka have been respectively substituted with FOCUS and DEF-CORE):

FOCUS “is” {“”|”a”|”an”|”the”} DEF-CORE
FOCUS “was” {“”|”a”|”an”|”the”} DEF-CORE
FOCUS “means” {“”|”a”|”an”|”the”} DEF-CORE
FOCUS “stands for” {“”|”a”|”an”|”the”} DEF-CORE
FOCUS “known as” {“”|”a”|”an”|”the”} DEF-CORE

These patterns are intended to present the typical lexical context used by an English speaker to introduce common notions when giving a definition for an entity. Moreover, even though some of them show a limited applicability with respect to some possible definition questions (*e.g.* patterns like “means” and “stands for” can not be applied to validate questions whose focus is a person name), all of them are completely domain independent. During the development we considered also other kinds of patterns, but we decided not to use them as they didn’t bring enough statistically relevant improvements to the performance.
In order a DEF-CORE to be taken into consideration, at least for one of the patterns the search engine should return relevant documents; this way, the number of pages where the focus and the DEF-COREs co-occurr is the combined number of the pages for all the patterns.
During the validation, the DEF-COREs are striped from determiners and are tested with all possible determiners. Often the DEF-COREs contain too many adjectives that make them receive

zero score when retrieving relevant documents. In this case we calculate the score for any of the sub-phrases of the DEF-CORE and take the maximum score obtained.
In light of these assumptions, the *Answer Relevance Score (ARS)* measure is specified in the following way:

$$ARS(def - core) = \frac{P(def - core \mid focus)}{P(def - core)^{2/3}} = \frac{\sum_{patterns} hits(pattern(def - core, focus))}{hits(focus) * hits(def - core)^{2/3}} * |EnglishPages|$$

which gives a score to the candidate DEF-COREs through the statistic measure of their co-occurrence with the focus.

3.3. Discussion, comments and future work

The proposed algorithm for extracting answers to definition questions gave us rather promising results. Even though the overall score for the definition part of the question set was not very high, with an average F score of 0.317 for the fifty definition questions of the competition, we think that the evaluation scheme that we have presented gives an appropriate framework for answering such type of questions. Our analysis of the results shows that on 76% of the questions the system has provided a correct answer. The major problems came from the relatively low recall (only 38% with respect to the vital nuggets selected by the NIST assessors as ideal answers). This is probably due to the fact that the methodology that we presented is more oriented towards canonical definition, rather than important facts and events related to the question focus which was the main objective of the TREC organizers when creating the appropriate nuggets. This leads us to the general conclusion that we need a more linguistically oriented approach, more focused on deep analysis of candidate answers. Another problem of our approach is related to the velocity with which the search engine returns the relevant documents. However our opinion is that using a large source of information as the Web is important to extract good answers to definition questions.

4 Experiment: Adding Parsing in the QA Loop

This year, we integrated in DIOGENE's architecture an algorithm for graph matching between syntactic structures in order to add structural-semantic criteria to the answer validation process which up to now was entirely based on techniques exploiting the Web redundancy. For every candidate answer, the graph matching algorithm gives a score which reflects structural, lexical and semantic similarities between its syntactic context and the question. The main assumption behind the use of a parser for answer validation is that often the question and the syntactic context of the answer have similar structures. Resorting to this assumption, besides our short-term goal of improving the answer validation process, our experiments represent a preliminary step towards the long-term goal of dealing with questions whose answer is not a named entity.
Given two parse trees, the main scope of the graph matching algorithm here presented is to find the best mapping among the two, considering similarities among their lexical content.
In the following explanation we will present the graph matching algorithm using as an example the question-answer_passage pair:

Question #1920: "When was 'Cold Mountain' written?"
Answer passage: "When the 'Cold Mountain' began rising to the top of bestseller lists in 1997…".

Our algorithm proceeds through the following steps:

0. The syntactic structures both of the question Q(V_Q,E_Q) (vertices V_Q and edges E_Q) and the candidate answer passage, CA(V_{CA},E_{CA}), are found. To this aim, we use dependency parse trees produced by the RASP (Carrol and Briscoe, 2002) parsing toolkit (in reality the structures are not trees but rather directed acyclic graphs).
1. An association graph is built A(V_A,E_A) with a set of vertices V_A and edges E_A. Such association graph generalizes the structure of both input graphs - Q and CA.

   1.1. Every vertex from the association graph A has two corresponding vertices from both graphs Q and CA. From any two vertices $v_Q \in V_Q$ and $v_{CA} \in V_{CA}$ we may form a vertex $v_A \in V_A$ if the lexical or part-of-speech tag labels on v_Q and v_{CA} are consistent and can be generalized. In such case the label on v_A is a generalization of the labels on v_Q and v_{CA}. For example, in Figure 2 the vertices from Q and CA labeled with "Cold Mountain" generate in the corresponding association graph a vertex with the same label. Moreover, being both of them verbs, the vertices "written" from Q and "rising" from CA generate a vertex labeled with "V" in A. In the model we have adopted, two words which have the same part of speech can also be generalized if they have a common hypernym in WordNet.

   1.2 We put an edge between any two vertices $v_A{}^1$ and $v_A{}^2$ from V_A if their corresponding vertices from Q and CA are linked in the same direction.
   Formally this means:

   $$IF\ \ v_A^1 = generalize\ \ (v_Q^1, v_{CA}^1)\ \ AND\ \ v_A^2 = generalize\ \ (v_Q^2, v_{CA}^2)\ \ THEN$$

   $$(v_A^1, v_A^2) \in E_A \Leftrightarrow (v_{CA}^1, v_{CA}^2) \in E_{CA}\ \ AND\ \ (v_Q^1, v_Q^2) \in E_{CA}$$

2. We have defined a function *weight* which gives a score to every syntactic structure obtained via generalization of two structures. In this way we can define for every substructure of A, a weight which is based on the number of edges and vertices and the labels they have. The best match is defined as the highest weighted sub-graph of A in which no vertices share common corresponding vertex in Q or CA. Considering the table in Figure 2, the last condition can be formulated in the following way: all the vertices in the matching sub-graph of A have to be distributed on different columns and rows. We call the resulting sub-graph the *best matching graph* of Q and CA.

3. From the question form we define the possible syntactic position of the exact answer (in Figure 2 the position is denoted with X). If the candidate exact answer matches the position X, then it takes the score of the matching between Q and CA.

4.1 Calculating the weights

As it was mentioned before, when two vertices are generalized in one vertex in the association graph, we assign a generalizing label to this *association vertex.* According to the differences in the labels of the vertices which have been generalized, we assign a score to the *association vertex.*

For every vertex in the association graph $v_A = generalize(v_Q, v_{CA})$ we calculate its weight by means of the following heuristically defined parameters:

$$weight(v_A) = \begin{cases} \bullet\ 300, \textit{if the lexical part of } v_{CA} \textit{ and } v_Q \textit{ is equal and they both represent names} \\ \bullet\ 100, \textit{if the lexical part of } v_{CA} \textit{ and } v_Q \textit{ is equal and it is not a name} \\ \bullet\ 100*k, \textit{ if } v_{CA} \textit{ and } v_Q \textit{ have a common hypernym in WordNet} \\ \textit{in this case k is defined from the number of vertices between the common} \\ \textit{hypernym and the generalized words} \\ \bullet\ 1.5, \textit{if } v_{CA} \textit{ and } v_Q \textit{ have the same part of speech} \\ \bullet\ 0, \textit{if none of the above conditions hold} \end{cases}$$

We define the weights of the edge e_A in the association graph as:

$$weight(e_A) = \begin{cases} \bullet\ 2, \textit{if both association vertices it connects represent a lexical item which is not} \\ \textit{a stop} - \textit{word} \\ \bullet\ 1, \textit{if only one of the vertices is a lexical item which is not a stop} - \textit{word} \\ \bullet\ 0, \textit{if none of the above conditions hold} \end{cases}$$

Using these definitions, we can define the weight of any syntactic sub-graph A'(V',E') of the association graph A(V,E) in the following way:

$$weight(A'(V',E')) = \left(\sum_{e\in E'} weight(e)\right)^2 \cdot \sqrt{\sum_{v\in V'} weight(v)}$$

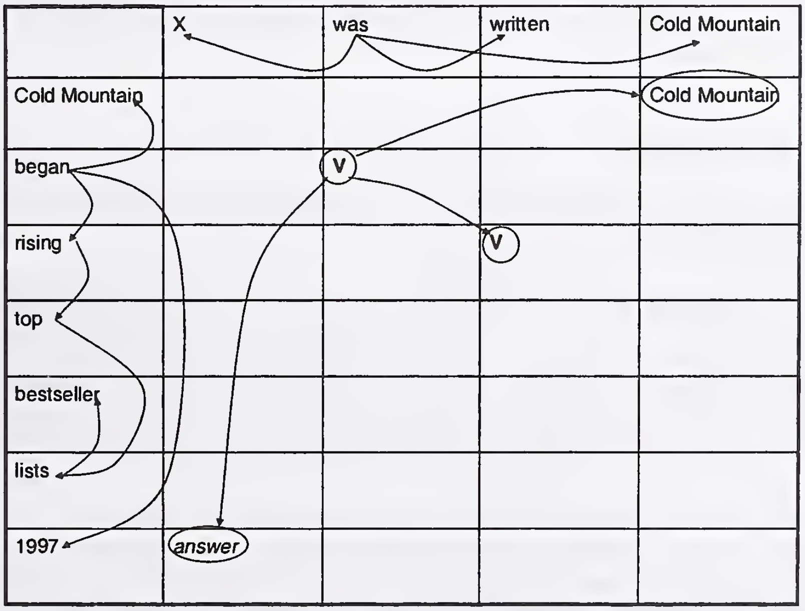


Figure 2 Matching among syntactic trees

Finally, we calculate the weight of every candidate answer passage by calculating the weight of the *best matching graph* between the sentence where it appears and the question. The algorithm gives this score only if the candidate answer matches the answer position in the question (denoted by X in Figure 2).

4.2 Problems and discussion on the syntactic graph matching

We did not have enough time to precisely define the parameters of the syntactic graph matching described so far; therefore, the application of the syntactic validation criteria gave no improvement in the overall result. Although, we did not perform complete error analysis, the following weak points of the current implementation can be pinpointed:
1. We are still not able to identify the position of the expected answer (X in Figure 2) with enough precision.
2. Calculation of the weights is not refined and parameters are only intuitively defined. For example, it would be much better to define the weight of the vertices considering their IDF value in a corpus.
3. Syntactic and lexical transformations can be integrated in the algorithm in order to make the matching of the structures more flexible (for instance, considering nominalization of verbs and active-passive transformations could improve our method).
4. Anaphora and ellipsis should be resolved before applying the syntactic structure matching; unfortunately, these techniques were not implemented in this version.
5. We did not normalize the question by translating it in affirmative form; this also influences the precision of the approach.
However, our empirical observations show that structural similarities between the question and the candidate answer passages often exist, and can be identified by inexact graph matching techniques. Therefore, fine tuning of the parameters of the matching algorithm will be necessary to identify these similarities and use them to improve both our answer ranking criteria and the overall answer extraction process.

5 Results and Conclusion

DIOGENE's performance has been evaluated over the three runs submitted to the TREC-2003 QA main task (see Table 1).

	Factoid							**List**	**Definition**	**FINAL SCORE**
	W	U	X	R	Accuracy	NIL Prec.	NIL Rec.	Average F	Average F	
irstqa2003w	300	10	6	97	0.235	0.121	0.267	0.076	0.317	**0.216**
irstqa2003p	305	11	5	92	0.223	0.132	0.167	0.074	0.315	**0.209**
irstqa2003d	343	4	4	62	0.150	0.111	0.067	0.067	0.318	**0.171**

Table 1: DIOGENE at TREC-2003

As for the 413 factoid questions, the total number of wrong (W), unsupported (U), inexact (X) and right (R) answers, together with the overall accuracy, the precision and the recall of recognizing NIL answers are reported for each run. Results for the list and the definition questions are only presented in terms of the average F-measure scores achieved over the total number of questions (respectively 37 and 50).

All these results have been obtained using the same overall architecture, but varying the validation technique to answer factoid and list questions. In particular:

irstqa2003w, our best run, has been obtained relying on the Web as the unique resource to accomplish automatic evaluation as in last year's best performing version of DIOGENE.

irstqa2003p results from the combination of the scores provided by the Web-based answer validation methodology and the graph-matching technique described in Section 4. Unfortunately, this did not bring any improvement to the system's performance. This is probably due to the weaknesses of the approach already mentioned in the same section. Nevertheless, in light of our empirical observations, parsing and other deeper linguistic analysis techniques (*e.g.* anaphora resolution, temporal and spatial reasoning, etc.) are deemed necessary to deal with the general QA problem and, more specifically, with the increasing difficulty level of the TREC competition.

irstqa2003d, surprisingly the worst result, has been obtained combining the Web-based answer validation technique with metrics that take into account the density of the query keywords within the retrieved passages. Our surprise, partially motivated by the higher difficulty of this year's TREC questions, comes from the fact that the same combined validation technique proved to be the most successful in the recent multiple-language QA track at CLEF-2003 (see Negri et al. 2003 for details).

A general conclusion that can be drawn in light of these results is that statistical approaches are relatively easy to implement and prove to be effective for some of the QA subtasks such as answer validation, allowing systems to reach reasonable performances with a limited effort. However, as they are limited to the statistically relevant knowledge that we can acquire from the Web or from an off-line corpus, deeper linguistic techniques seem to be a crucial step towards higher flexibility, coverage, and effectiveness of any QA system.

References

Abney, S.: Partial Parsing via Finite-State Cascades. Proceedings of the ESSLLI '96 Robust Parsing Workshop (1996)

Carroll, J., Briscoe, E.: High Precision Extraction of Grammatical Relations. Proceedings of the 19th International Conference on Computational Linguistics (COLING-2002), Taipei, Taiwan (2002).

Magnini, B., Negri, M., Prevete, R., Tanev, H.: Multilingual Question/Answering: the DIOGENE System. Proceedings of the Tenth Text Retrieval Conference (TREC-10), Gaithersburg, MD. (2001).

Magnini, B., Negri, M., Prevete, R., Tanev, H.: Mining Knowledge from Repeated Co-occurrences: DIOGENE at TREC-2002. Proceedings of the Eleventh Text Retrieval Conference (TREC-2002), Gaithersburg, MD. (2002a).

Magnini, B., Negri, M., Prevete, R., Tanev, H.: Is It the Right Answer? Exploiting Web Redundancy for Answer Validation. Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002), Philadelphia, PA. (2002b).

Negri, M., Tanev, H., Magnini, B.: Bridging Languages for Question Answering: DIOGENE at CLEF-2003. Proceedings of the Cross Language Evaluation Forum (CLEF-2003), Trondheim, Norway (2003).

Combining Methods for the TREC 2003 Robust Track

James Mayfield and Paul McNamee
Research and Technology Development Center
The Johns Hopkins University Applied Physics Laboratory
11100 Johns Hopkins Road, Laurel, Maryland 20723-6099 USA
{mayfield, mcnamee}@jhuapl.edu

Overview

The Johns Hopkins University Applied Physics Laboratory (JHU/APL) focused on the Robust Retrieval Track at this year's conference. In the past we have investigated the use of alternate methods for tokenization and applied character n-grams, with success, to tasks in ad hoc, filtering, and cross-language tracks.

For ranked retrieval, we have come to rely on a statistical language model to compute query/document similarity values. Hiemstra and de Vries describe such a linguistically motivated probabilistic model and explain how it relates to both the Boolean and vector space models [4]. The model has also been cast as a rudimentary Hidden Markov Model [7]. Although the model does not explicitly incorporate inverse document frequency, it does favor documents that contain more of the rare query terms. The similarity measure can be computed as

$$Sim(q,d) = \prod_{t \in q} \left(\alpha \cdot f(t,d) + (1-\alpha) \cdot f(t,C)\right)^{f(t,q)}$$

Equation 1. Similarity calculation.

where α is the probability that a query word is generated by a document-specific model, and $(1-\alpha)$ is the probability that it is generated by a generic language model. $f(t,C)$ denotes the mean relative document frequency of term t. We have observed that aggregate performance using this model is fairly insensitive to the precise value of α that is used; however, higher values of alpha tend to result in selecting documents that contain a greater number of the query terms.

Earlier work on the Query Track, held during the TREC-8 and TREC-9 evaluations, showed that different query formulations could result in substantially different retrieval performance on individual queries (see Buckley [1]). For example, deleting an important query term, adding an important word that was not initially present, the use of idioms in topic statements, and deleterious effects caused by inappropriate stemming (over or under aggressive) were each shown to demonstrably alter precision on particular topics. Furthermore, multiple reports have appeared in the literature suggesting that a combination of evidence from multiple, disparate approaches can be beneficial (*e.g.,* Savoy [9]).

From this we conclude that a scheme based on multiple document representations and multiple similarity metrics might exhibit increased robustness in query performance, and possibly higher aggregate performance as well. How different methods can best be combined is not clear. In this paper we report on our efforts attempting to: (1) merge disparate run files; and (2) devise an automated technique for learning query-specific run weights that can be used to create a single, robust run.

We built several indices to compare different tokenization methods. We have begun investigating the use of part-of-speech tagging and entity-tagging to transform the term space, with the hope of capturing semantic distinctions (*e.g.,* bat, a noun, versus bat, the verb, or, Washington, the place, instead of Washington, a person); however, we did not use these indices in this year's Robust Track. Summary information for the indices that we used is shown in Table 1.

Table 1. Index statistics for the Robust Track collection.

		# terms	index size
words	w	554751	373 MB
words preserving case	c	698786	410 MB
stems (Snowball)	s	455803	320 MB
4-grams	4	251694	1.39 GB
5-grams	5	1485406	2.22 GB
6-grams	6	6030289	3.22 GB
words + phrases	p	19141479	1.97 GB

Disparate Retrieval Approaches

In the previous section we enumerated a number of alternatives to tokenization that resulted in different index data files being created for the collection. In many cases different tokenizations lead to similar overall performance. For example, on the query "health food" the use of case-normalized and unnormalized words should produce similar ranked lists. On the other hand, the query "NRA" will likely produce different results for case-sensitive words and character 4-grams.

We considered several approaches to computing document-query similarity:

- Retrieval without the use of blind relevance feedback
- Retrieval with blind relevance feedback using *t* expansion terms
- Massive collection enrichment
- Weighting query terms by setting qtf = 1
- RIDF weighting [2][6]
- Adjusting values for alpha, the parameter which regulates the relative importance of seeing query terms in documents in our statistical language model
- Applying UC Berkeley's logistic regression retrieval model [3]
- Calculating similarity with a probabilistic model and Okapi BM25 weighting [8]
- Requiring certain query terms and/or prohibiting others
- Expanding queries using a human-constructed thesaurus
- Query expansion using a statistical thesaurus

We were unable to consider each of these, primarily due to a lack of time to implement each method or to empirically evaluate each method with each index. We ended up using seven different indexes and 11 different retrieval methods (listed below). We then sought to combine or select from the 77 runs produced. With some risk of overtraining, we measured our performance on the 150 queries used in the TREC-6, TREC-7, and TREC-8 evaluations. We restricted ourselves to the use of only the 'Description' portion of topic statements; however, we did submit one official run using 'Title', 'Description', and 'Narrative' sections expecting that it would better contribute to the relevance pools.

For each of the 7 indexes, we created runs that computed document relevance by:

- [4 runs] Adjusting alpha values in our statistical language model of retrieval. We considered values of 0.2, 0.5, 0.8, and 0.9. We thought that higher alpha values would lead to better precision at low recall levels. Pseudo relevance feedback was not applied.
- [3 runs] Adjusting alpha values as mentioned above; however, relevance feedback was applied, selecting 60 'terms' (whatever those terms might be (*e.g.*, n-grams, words, stems), from 20 top-ranked documents. Alpha values of 0.2, 0.5, and 0.8 were considered. We imagine that it might also be useful to use other parameter settings for automated feedback, such as, different methods for isolating feedback terms, different numbers of expansion terms, or different numbers of presumed positive and negative documents.
- [1 run] We used our statistical language model with alpha = 0.5 without feedback, and *without stopword removal.* Normally we represent documents using all terms, but omit query terms at run-time that have a relative document frequency greater than 0.2.
- [1 run] Using the logistic regression retrieval model described by UC Berkeley
- [2 runs] Using Okapi BM25 term weighting in a binary independence model, both with and without relevance feedback (as described above). We used values of 1.2 for k1, 500 for k3, 0.6 for b, and we assumed that the top 8 documents for each query were relevant and all others were not, for the purpose of term weighting.

We considered several metrics for robust performance, but chiefly examined the percentage of topics with at least one relevant document in the top 10 ranks (TopTen) and the area under the curve when topical average precision is averaged over a number of the worst scoring topics and plotted as a function of the number of worst topics examined (MAP-Hardest). We focused on the hardest 25% of topics, as suggested in the track guidelines. For whatever number of topics is considered, MAP-Hardest is most effected by the most difficult topics. If the worst 12 topics are examined, then about three-quarters of the weight is given to the hardest 6 topics and only about one-quarter of the weight is given to the next 6 hardest topics. This puts a premium on doing as well as possible on the absolute hardest topics. We also considered high mean average precision (averaged over all topics) to be desirable, but were primarily concerned with 'robust' measures of performance.

We were interested to know how well each combination might do. In particular, we wondered how much improvement could be obtained given an oracle that could perfectly select the single-best method to apply for each individual query. Using the known relevance judgments for the TREC-6 through TREC-8 topics we determined that a method that selected the single best method (from our set of 77) for each topic could improve each of the performance measures appreciably. If this was not the case, for example, if our different methods did not exhibit large variations in retrieval performance, then any machine-learning approach to select a query-specific method would be doomed to failure.

Of our 77 runs, the one with the highest mean average precision (over all topics) on the training set used stems as indexing terms with the statistical language model (with alpha=0.2) and with relevance feedback. However, when alpha=0.5 mean average precision was about the same, and the robust measures were improved. We compare these runs and an oracular 'best' run using the robust metrics in Table 2.

Table 2. Comparing an oracle-based run and two high-performing methods on the training set.

	TopTen	MAP-Hardest	MAP
stems-lm2-rf	0.7467	0.0052	0.2513
stems-lm5-rf	0.8067	0.0061	0.2489
oracular	0.9600	0.0364	0.3587

From Table 2 we observe that an oracle-based run can improve mean average precision by approximately 40% and MAP-Hardest by roughly 600%.

Selecting a Single Method

We would like to predict which tokenization and scoring methods will prove most effective given a particular query. As several years of training data are available for this collection, we are inclined to adopt a supervised learning approach. For this study we applied Support Vector Machines (SVMs) [5]. For any learning approach to succeed we need to identify features that might discriminate high performing retrieval configurations from low performing ones.

We envisioned using a large set of features, including:

- Number of query terms
- Length of query in characters
- Length of query in words
- Capitalization pattern
- Digit pattern
- IDF of ith term
- Mean IDF
- Variance of IDF
- Whether ith term is a known closed-class word and which kind
- Whether words are stop-structure

In practice we used only four types of features: those based on the query alone; features based on various statistics of an index; features based on scores of documents in particular runs; and statistics computed from an index and from a run file.

- *Query-based*: total number of terms; number of unique terms; number of unknown terms
- *Using index*: maximum, minimum, mean, and variance of both query term IDF and RIDF; the number of documents containing: (1) all of the query terms; (2) all the query terms excluding stopwords; (3) the two rarest query terms; (4) the 2 most common query terms; and (5) the two query terms with highest mutual information.
- *Run-based*: the ratio of the score between the highest ranked and rank (5, 10, 20, 100, 500) documents using the 'stems-lm5-rf' run
- *Run plus index*: the percentage of unique terms to the total number of terms observed in documents from a specified range using the 'stems-lm5-rf' run (ranges considered included 0-10, 11-50, 51-100, and 201-300); the percentage of query terms to total terms observed in the ranked documents of a given range – as described above; and, the mean RIDF value of all terms occurring in the documents of the ranges described above.

Using these features we attempted to train a support vector machine with a cubic kernel for each of our 77 runs. For each of the queries in a 100-query training set we used the top 10 scoring runs as positive examples and the bottom scoring runs as negative examples. We hoped the SVM could distinguish methods likely to achieve high mean average precision (*i.e.,* good performance) and those unlikely to do so.

Unfortunately the SVM was not generally able to learn this distinction. The training algorithm converged, but essentially memorized the data. It may be that our set of features was inadequate and that more semantically laden features are essential to such a task. Or possibly, many more training exemplars are required for this approach to succeed. We next turned our attention to an approach based on combining results from multiple runs.

Merging Multiple Methods

As an alternative to selecting a single method based on features about the query, or weighting several methods predicted to perform well, we examined combination of multiple methods to produce a single ranked list for each topic. Combination of disparate techniques has occasionally led to improved performance in TREC-style evaluations; many consider that the constituent runs should be chosen to maximize orthogonality with respect to one another. That is, it is hoped that they make independent mistakes and that run combination will reinforce selection of good documents and lower the ranks of documents that only appear to be of high quality using a single method. In the past we have used combination to reasonable advantage; we found that combination of n-gram based runs with stem or word runs can confer a 10% relative advantage in mean average precision [6].

Our preferred method for merging multiple runs is to first normalize score values for each individual run and then create an ordering based on these normalized scores. We view scores as masses, and divide individual scores by the sum of the masses of the top *k* documents (we use *k=1000*). Because our probabilistic calculations are typically performed in log space, and scores are therefore negative, we achieve the desired effect by using the reciprocal of a document's score as its mass.

Other Approaches

We thought up several other schemes that were not implemented for the evaluation.

One was to build a tagger that processes query words and assigns them to different categories. For example, some words are clearly query-specific, such as "find documents that"; others are modifiers of a key concept, like 'international' in 'international organized crime"; still others are keywords, like 'black bears' in "black bear attacks". This technique is unproven, but recent successes in tagging applications might be applied to the 1000+ available topics from past TREC, CLEF, and NTCIR conferences. Terms in different categories could be treated (*e.g.*, weighted) differently.

Official Submissions

We submitted five runs described below.

aplrob03a was a combination of two runs, one using stems and one using character 5-grams. Title, Description, and Narrative topic fields were used here, but only the Description field was used for our other official runs. Relevance feedback was applied and alpha was 0.5. We thought that this method would maximize mean average precision over all topics. This run typifies our traditional processing.

aplrob03b was designed to maximize the number of topics with a relevant document appearing in the top ranks (say up to rank 20 or so). We filled 'slots' by examining several run files and selecting what we though to be good documents from different methods. Some of the documents were selected based on which run files worked well on the training set; others were based by clustering the top 50 ranked documents (for a given run) and selecting the highest ranking document from each of 5 clusters.

We used a quadratic implementation of agglomerative clustering that started with the 50 individual documents and repeatedly combined clusters, one at a time. Our distance metric was Sim(c1,c2) + Sim(c2,c1) – Max(cardinality(c1), cardinality(c2)). Our language model similarity metric is not symmetric, so we added the similarity between 'query' and 'document' (both clusters of documents) and the similarity between 'document' and 'query'. We then subtracted the cardinality of the larger of the two clusters; this was done in an attempt to even the distribution of the number of documents per cluster. We also ignored both very common and very rare terms when clustering. This method found a relevant document in the top 10 ranks for 134 of 150 topics in the training set.

To achieve reasonable performance in mean average precision as well, we then extended this list of top-ranked documents using run aplrob03d (described below), taking care to remove duplicate documents when building the ranked list.

aplrob03c was an attempt to maximize MAP-Hardest. We combined results using all 77 runs as input. On the training data this method performed the best, achieving 0.0103 on the MAP-Hardest metric. This is nearly double the performance obtained with a single, well-performing run, but well below our 0.0364 theoretical maximum.

aplrob03d is analogous to aplrob03a, but different in using only the 'Description' portion of the topic statements. Like aplrob03a, we expected this run to achieve good mean average precision over all topics and to serve as a baseline for our other runs.

Finally, ***aplrob03e*** was an overtraining run that sought to optimize TopTen. On the training data, we were able to use the qrels to build a run by selecting the ith rank of the jth run for all topics. This method found a relevant document (in the top 10 ranks) for 143 out of the 150 training topics. Its use on novel data is questionable, but it is possible that the different runs selected represent a method for finding orthogonal methods. The following runs/ranks were employed:

stems-okapi	1
stems-logreg	3
stems-slm8	5
words-logreg	4
case-words-slm5	3
4-grams-okapi-rf	2
phrases-logreg	3
5-grams-slm8	5
stems-slm5	5
5-grams-okapi	2
4-grams-slm8-rf	5
words-nostop	5
4-grams-slm9	4

Examining Table 3 (below), we observe that run aplrob03c achieved a 0.0040 improvement in MAP-Hardest over our description-only baseline, aplrob03d; this is a 50% increase over the baseline. We interpret this as mild support for the hypothesis that combination of a very large number of methods can improve robustness. For the TopTen measure, run aplrob03e achieved a 90% success rate vs. 78% using our baseline method, a 15.4% relative improvement.

Table 3. Performance of officially submitted methods on all 100 topics. Run aplrob03d is a baseline for comparison against other methods.

	Fields	TopTen	MAP-Hardest	MAP
aplrob03a	TDN	0.8900	0.0238	0.2998
aplrob03b	D	0.8500	0.0113	0.2522
aplrob03c	D	0.8200	0.0120	0.2521
aplrob03d	D	0.7800	0.0080	0.2726
aplrob03e	D	0.9000	0.0096	0.2535

Conclusions

We attempted to determine whether selection or combination of diverse methods can improve the robustness of query processing. Our attempts to select preferred methods dynamically (*i.e.,* on a query-by-query basis) failed; however, we have not fully investigated this line of work. We did discover that combination of 77 runs (7 tokenizations and 11 similarity metrics) led to our best results for the MAP-Hardest measure. *A priori* selection of several diverse methods seemed to optimize the TopTen measure, though the small number of topics leaves it difficult to determine whether this result is valid. These results are based only on an examination of 'description-only' runs.

References

[1] C. Buckley, 'The TREC-9 Query Track'. Proceedings of the Ninth Text REtrieval Conference (TREC-9).

[2] Church, K. W., 'One term or two?' In Fox, E. A., *et al.*, eds., *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR-95),* pp. 310-318. 1995.

[3] W. Cooper, A. Chen, and F. Gey, 'Full Text Retrieval based on Probabilistic Equations with Coefficients fitted by Logistic Regression', Proceedings of the Second Text REtrieval Conference (TREC-2), pp. 57-66.

[4] D. Hiemstra and A. de Vries, 'Relating the new language models of information retrieval to the traditional retrieval models.' CTIT Technical Report TR-CTIT-00-09, May 2000.

[5] T. Joachims, Making large-Scale SVM Learning Practical Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.), MIT Press, 1999.

[6] P. McNamee and J. Mayfield, 'JHU/APL Experiments at CLEF-2001: Translation Resources and Score Normalization.' In: C Peters et al. eds., Evaluation of Cross-Language Information Retrieval Systems: Second Workshop of the Cross-Language Evaluation Forum (CLEF-2001), Darmstadt, Germany, pp. 193-208.

[7] D. R. H. Miller, T. Leek, and R. M. Schwartz, 'A Hidden Markov Model Information Retrieval System.' In the Proceedings of the 22nd International Conference on Research and Development in Information Retrieval (SIGIR-99), pp. 214-221, August 1999.

[8] S. E. Robertson, S. Walker, and M. Beaulieu, 'Okapi at TREC-7: automatic ad hoc, filtering, VLC and interactive track'. In E. M. Voorhees and D. K. Harman (eds) Proceedings of the Seventh Text REtrieval Conference (TREC-7).

[9] J. Savoy, 'Cross-language information retrieval: experiments based on CLEF 2000 corpora.' In Information Processing and Management, Vol. 39(1), pp. 75-115, 2003.

Report on the TREC 2003 Experiments using Web Topic-Centric Link Analysis

P. Ingongngam and A. Rungsawang
Massive Information & Knowledge Engineering
Department of Computer Engineering
Faculty of Engineering
Kasetsart University, Bangkok Thailand.
pain@csl.cpe.ku.ac.th, arnon@mikelab.net

Abstract

In TREC 2003, our experiments have been concentrated only on the topic distillation task. We first simply apply the term-based technique to the .GOV web collection, and then re-rank the retrieval results using a link analysis algorithm in order to boost the retrieval precision. Our link analysis has been inspired from the original PageRank, but focused on the web topic during the iterative score propagation. We hybridize the term-based retrieval scores with our link analysis approach. From the experiments, the results show that the combination of those scores still provides inadequate precision improvement.

1 Introduction

The information retrieval focuses on the quality of result the users obtain. However, the traditional information retrieval methods give the dissatisfying results for the web collections as the queries the users enter are highly ambiguous. In order to increase the precision of the retrieval results, many techniques have been developed. Link Analysis is one of the techniques extensively used in web-IR community. HITS [10] and PageRank [5] are the two well known algorithms that are developed based on link analysis technique, and widely studied by several TREC participants [9, 11, 6, 7, 12].

Hyperlinked-Induced Topic Search (HITS) has been developed by Kleinberg and implemented in the CLEVER Project [2], while PageRank [5] is the core mechanism of the most successful search engine, Google [1]. Both of these algorithms have different advantages and disadvantages. HITS algorithm calculates a page score based on the the user's topic. However, HITS must be computed on-line at query time. On the other hand, PageRank algorithm calculates a page score based on link relations. This calculation is done off-line only once, but the calculated scores are not related either to the topics of the web pages or to that the users are interested.

For the TREC 2003, we introduce another off-line link analysis that allows the web topic to influence the propagation of link scores, called "Topic-Centric" (TC) [8]. Analyzing the connectivity of the web graph in the same way as PageRank, TC algorithm iteratively propagates the portion of rank score of a source web page to the rank score of the destination one in accordance with the topics of both web pages. Following the hyperlink, the destination web page will then appropriately receive a high rank score when the topic of the source web page is really referred to that of the destination one.

The report is structured as follows. Section 2 reviews briefly the PageRank algorithm. Section 3 explains the new TC algorithm. Section 4 shortly provides the experimental setup and results. Finally, section 5 concludes the report.

2 Basic PageRank Algorithm

Brin and Page suggest a link-based search model called "PageRank" that evaluates the importance of each web page based on its citation pattern. The basic idea of PageRank is as follows. When a page u has a hyperlink to a page v, it is assumed that the author of the page u suggests the page v with some reasons, e.g., related context, individual favor, or popular reference. PageRank employs this hint to compute the page scores. Since it considers all web pages equally important, if we let N_u be the number of pages which page u points out, and $Rank(u)$ represent the rank score of a page u, then a hyperlink $u \rightarrow v$ confers $1/N_u$ units of rank to page v.

To compute the rank vector for all web pages, we then simply iteratively perform the following fixed-point computation. If we let B_v represent the set of pages pointing to page v, for each iteration, the successive rank scores of pages are recursively propagated from the previously computed rank scores of all other pages pointing to them:

$$\forall_v Rank_{i+1}(v) = \sum_{u \epsilon B_v} \frac{Rank_i(u)}{N_u} \tag{1}$$

In general, the web graph is not strongly connected, and this may lead the PageRank computation of some pages to be trapped in a small isolated cluster of the web graph. This problem is usually resolved by pruning nodes with zero out-degree, and by adding random jumps to the random surfer process [5]. This leads to the following modification of Equation (1) to:

$$\forall_v Rank_{i+1}(v) = (1 - \alpha) + \alpha \sum_{u \epsilon B_v} \frac{Rank_i(u)}{N_u} \tag{2}$$

where α, called "damping factor", is the value that we use to modify the transitional probability of the random surfer model of an underlying web graph.

3 Web Topic-Centric Algorithm

There is a big difference between PageRank and TC algorithms. PageRank treats every web page equally important. It iteratively propagates the link score from the source page to the destination one with the fraction of $\frac{1}{N}$ where N is the amount of outbound links of the former. Link score propagation with no regard to the web content, or the "topic" of that web page, may be inappropriate and independent to the user query. Besides, our TC algorithm propagates link scores by considering the similarity between the topics of the source and destination pages.

There are many ways to compute the page similarity, but we here only focus on the vector space approach [4]. The vector space is widely used in many information retrieval researches. The basic idea of the vector space is to imagine a web page as a vector. Each distinct word in that page is considered to be an axis of a vector in a space. The direction of a vector characterizes the content of a web page. Here, we define W_{uv} to be the page similarity score computed between the source page u and the destination page v, and calculate it using the following formula:

$$W_{uv} = \frac{\sum_{k \epsilon u \cap v} f_{ku} f_{kv}}{\sqrt{\sum_{k \epsilon u} {f_{ku}}^2 \sum_{k \epsilon v} {f_{kv}}^2}} \tag{3}$$

where f_{ku} and f_{kv} are the number of term k found in page u and page v, respectively. The rank of W_{uv} value is between 0 and 1, and the similarity increases as this value increases.

Since TC does not consider web pages as being equally important, the portion of a rank score that propagates from a page u to a page v should be dependent on their page contents or topics. We then appropriately modify the fraction of link score propagation in Equation (2) to:

$$\forall_v Rank_{i+1}(v) = (1 - \alpha) + \alpha \sum_{u \epsilon B_v} \left(\frac{W_{uv}}{\sum_{x \epsilon O_u} W_{ux}} Rank_i(u) \right) \tag{4}$$

Here, B_v represents a set of pages pointing to v, O_u represents a set of pages pointed by u, and W_{uv} represents the similarity score computed between the content of the page v and the content of the in-link page u.

4 Experimental Setup and Results

We use the LEMUR toolkit [3] as our vector space based retrieval system. We first process the .GOV documents using BM25 weighting scheme, with parameters $B = 0.9$, $K_1 = 2$ and $K_3 = 7$, respectively. We only use the title section of the TREC topic without any expansion, and examine the average retrieval precision at 5, 10, 15, 100 and 1000 retrieved documents. We hereafter call the result from this step, the "base" case. We then apply both TC and PageRank algorithms to compute the rank scores of web documents in the .GOV collection, and employ those scores to re-rank the search results obtained from the base case. Table 1 as follows provides the comparison between the average precision scores of the base case, the re-ranking results obtained from the TC (denoted by "TC"), and those obtained from the PageRank (denoted by "PR"), respectively.

Table 1: The average precision scores.

retreived docs	base	TC	PR
At 5 docs	0.0920	0.0880	0.0160
At 10 docs	0.0800	0.0760	0.0280
At 15 docs	0.0667	0.0680	0.0280
At 100 docs	0.0304	0.0294	0.0272
At 500 docs	0.0125	0.0122	0.0126
At 1000 docs	0.0069	0.0069	0.0069
Avg Precision	0.0855	0.0789	0.0164

5 Conclusion

Like PageRank, TC algorithm analyzes the link connectivity, and pre-computes the rank scores of web pages. During the computation, TC propagates the portion of rank scores of the source web pages to the destination web pages in accordance with the topics found in both ends. Therefore, we expect that the final computed rank scores will be more reasonable, and be more efficient to use to re-rank the search results obtained from the traditional vector space model.

The study concluded from the TREC experiments this year shows that TC algorithm does still not provide any significant improvement when it is used to re-rank the search results obtained from the standard vector space retrieval model. Comparing with PageRank, TC algorithm however gives better re-ranking results in our experiments. More study and experiments will be conducted, e.g., we will try several other vector space based weighting scheme in similarity computation, as well as the use of weighted inter-host and intra-host link score propagation.

References

[1] Google, http://www.google.com/.

[2] IBM Almaden Researcn Center. Clever Searching. http://www.almadenibm.com/cs/k53/clever.html.

[3] LEMUR, http://www-2.cs.cmu.edu/~lemur/.

[4] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. *Modern Information Retrieval.* ACM Press, Addison-Wesley, 1999.

[5] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[6] F. Crivellari and M. Melucci. Web document retireval using passage retrieval, connectivity information, and automatic link weighting – trec9 report. In *Proceeding of the TREC-9 Conference*, 2000.

[7] J. Gevrey and S.M. Rüger. Link-based approaches for text retrieval. In *Proceeding of the TREC-10 Conference*, 2001.

[8] P. Ingongngam and A. Rungsawang. Topic-centric algorithm: A novel approach to web link analysis. In *Proceeding of the* 18^{th} *AINA Conference*, 2004. (to be appeared).

[9] T. Kanungo and J.Y. Zien. Integrating link structure and content information for ranking web documents. In *Proceeding of the TREC-10 Conference*, 2001.

[10] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[11] J. Savoy and Y. Rasolofo. Report on the trec-9 experiment: Link-based retrieval and distributed collections. In *Proceedings of the TREC-9 Conference*, 2000.

[12] K. Yang. Combining text- and link-based retrieval methods for web ir. In *Proceeding of the TREC-10 Conference*, 2001.

Biomedical Text Retrieval System at Korea University

Young-In Song, Kyoung-Soo Han, Hee-Cheol Seo,
Sang-Bum Kim, Hae-Chang Rim

Natural Language Processing Lab., Dept. of CSE,
Korea University, Anam-dong 5-ga, Seongbuk-gu, 136-701, Seoul, Korea
{sprabbit, kshan, hcseo, sbkim, rim}@nlp.korea.ac.kr

Abstract. In this paper, we describe our retrieval system used for the primary task of genomics track at this year. Our primary goal in this task is to find a proper method for the domain-specific retrieval environment. To achieve the goal, we have tested several techniques such as a phrase indexing strategy, two query weighting methods, and two post-processing methods such as a document filtering method and a documents reranking method. According to the experimental results, query weighting methods and document filtering methods can improve the performance of the retrieval system, but there still remain a room for improvement.

1 INTRODUCTION

The primary task of Genomics track is a kind of conventional ad-hoc retrieval task, where the system is expected to retrieve relevant documents in response to a user's query. However, this task has some significant differences to previous ad-hoc tasks, because of its environment. Documents and queries in this task are limited to the biomedical domain.

The document collection used in this task consists of about 520,000 MEDLINE abstracts, which is a database of biomedical literature. Compared to a general news-wire document collection, it has a number of distinguished features such as frequent usage of spelling variants, long length of multi-word terms, and somewhat different lexical phenomena.

Query set in this task is also different in some respects. A typical query in traditional retrieval tasks almost consists of natural language sentences which are weakly structured using a tag such as <desc>, <title> or not structured, and there isn't any restriction about the query. However, the query in this task consists of not sentences but only several terms, formalized as a kind of table structure, and the user information need is limited to find documents relevant to 'basic biology' of gene X[Hersh 2003].

Our primary goal of this experiment, thus, is to explore methods and strategies which can reflect these differences of query and documents to improve retrieval performance of IR system, and especially we focus on following three issues:

1) Keyword extraction strategy for multi-word term.

2) Query weighting methods considering term variants and multi-word terms.

3) Post processing technique such as document reranking and filtering to satisfy restrictions of the structured query.

2. PRELIMANRY EXPERIMENT

In our preliminary experiment with the training data, we have tested basic techniques of information retrieval related to keyword extraction such as stemming, and we found some interesting points.

Table 1 and 2 show the results of our preliminary experiments.

	Avg Precision	R-Precision	Rel-ret
No Stemming	0.2274	0.2056	207
Porter	0.1944	0.1842	252
Lovins	0.2693	0.2408	265

Table 1. Experiments results according to various stemming methods at training data.

	Avg Precision	R-Precision	Rel-ret
Base line	0.2887	0.2680	271
Simple rule	0.3342	0.3112	274

Table 2. Experiments results of keyword extraction with /without simple rule

The retrieval performance of the Porter stemmer, which is one of the most widely used stemmer in retrieval systems, is much worse than the Lovins stemmer, and even worse than the case when any stemmer is not used. That result is contrary to the previous researches which reported that the Porter stemmer yields a similar or better performance than other stemmer including no stemming [Fuller 1998, Namba 2000].

In addition, we tested a simple key word extraction method for a word consisting of two numeric characters or one alphabet letter. They frequently occur in biomedical terms such as gene names, and sometimes they cause to fail in retrieving relevant documents. The simple heuristic rule is described as follows:

> Simple rule: if a word w_1 is a short length word and the adjacent word w_2 is not a short length word, w_1 and w_2 words are combined into a keyword as a canonical form.
> In this case, the adjacent word w_2 also is extracted as a keyword, too.
> E.g. [G protein, protein G -> protein, G:protein]

The result of adapting the simple rule to retrieval is shown in table 2. That simple method achieves about 15% improvement over the baseline.

3. INDEXING

Based on the observation of the preliminary experiments, keywords were extracted using the Lovins stemmer, and simple rules with case insensitive manner. Our system also did a stopword removal using a stop word list of PubMed [NCBI 2003].

Additionally, a phrase indexing strategy was also used to handle a multiword biomedical term.

3.1 The phrase indexing strategy using term boundary detection

The query and documents in this task have a lot of biomedical terms including multi-word terms, which often prevent a retrieval system from matching between a query and documents, so we tried to index phrases by identifying term boundaries.

Any keyword pair of adjacent non stopwords in order within a term boundary is regarded as a phrase. If a term consists of one word, it is also regarded as a phrase itself. To detect term boundaries in a document, we used a named entity tagger for biomedical domain [lee 2003]. Phrases are weighted with the same scheme as single terms.

4. DOCUMENT RETRIEVAL

In this section, we will describe the basic model of our retrieval system and two query weighting methods.

4.1 Basic retrieval model

All models used in our system are based on the probabilistic model with the BM25 weighting scheme of the Okapi system [Robertson 2000].

Equation (1) is the weighting formula of our basic model. We slightly modified K factor of the weighting function BM25.

$$\sum_{W \in Q} w^{(1)} \times \frac{(k_1+1)tf}{K+tf} \times QW \quad (1)$$

$$w^{(1)} = \log(\frac{N-n+0.5}{n+0.5})$$

$$K = (k_1 + \log(\frac{n}{totallikelihood}) - 4) \times ((1-b) + b(\frac{1+\log(dl)}{1+\log(avdl)})) \quad (2)$$

$$QW = \frac{(k_3+1)qtf}{k_3+qtf} \quad (3)$$

Where

Q is a query, containing word *W*,
N is the number of documents,
n is the number of documents containing the keyword,
$w^{(1)}$ is the Robertson / Sparck Jones weight[Robertson, et al 1976],
k_1, *b*, *k3* is the parameters which depend on the nature of queries and document collection. We fixed $k_1 = 1.5$, $b=0.6$, $k3= 1$ experimentally,
tf is the frequency of occurrence of the keyword within a document,
qtf is the frequency of the keyword within query,
dl and *avdl* are the document length and average document length.

4.2 Query weighting method

We have proposed two query weighting techniques for the genomics-track style queries: normalizing query weight and incorporating inverse query frequency.

4.2.1 Query weight normalization

Genomics-track style query consists of a number of subqueries including an official gene name, its official symbol and aliases, etc. Most of them are equally important to retrieve the relevant documents effectively. However, with the basic model of Equation (1), one critical problem can occur because of the long subqueries. For example, we can have two relevant documents: One contains a long official gene name "cyclin-dependent kinase inhibitor 1A (p21, Cip1)" and the other contains its official symbol "CDKN1A". In this case, it is obvious that two documents are equally relevant. With the base Okapi model, however, the former document appears at a higher rank since many term weights are added to the score of the former document.

One possible solution to alleviate this problem is ***query weight normalization*** according to the length of the subqueries. To do this, we modified the *QW* factor defined in Equation (3) as follows:

$$QW^1 = \sum_{q=1}^{|Q|} \frac{(qk+1)qtf}{qk((1-qb)+qb(ql))+qtf} \quad (4)$$

Where

$|Q|$ is the number of subquery within query *Q*,
qtf is the frequency of the keyword within subquery,
ql is the subquery length, and *qk* and *qb* is the parameters which depend on the documents and queries. We fixed $qk = 1.2$, $qb = 0.95$ experimentally.

We define the equation (4), QW^1, with a similar manner to document term weighting scheme of Okapi. The *ql* factor in equation (4) has an effect to balance weight of different length subqueries in a query.

4.2.2 Inverse query frequency

Each word forming a gene name can have different discriminative power. For example, while some words such as 'inhibitor', 'receptor', and 'kinase' occur within the various gene names, words such as 'p21', 'Cip1' occur only in some specific gene names. In other words, if 'Cip1' and 'receptor' occur in the same query, 'Cip1' is more useful query term than the common word 'receptor'.

Based on this observation, we define a new weight factor, inverse query frequency: the number of every possible query divided by the number of queries containing the specific term. For this task, we regard a set of every possible query as 15,000 gene names list obtained from the various web sites because only the gene names are assumed to be entered into our system.

Thus, the new query weight formula adopting inverse query frequency, QW^2, is represented by:

$$QW^2 = QW^1 \times \frac{QN+0.5}{qn+0.5} \quad (5)$$

Where

QN is the size of gene names list.
qn is the number of queries in the query set containing the keywords

We used equation (5) for submitted runs instead of the equation (3).

5. Reranking and Filtering

In the genomics track, two constraints must be satisfied. First, each retrieved document must be about 'basic biology' of the gene in a query or its protein product. Second, the gene in a document must be from the organism designated in the query. We reranked and filtered the initial retrieved documents to improve the performance of the system. The details are described in the following subsections.

5.1 Reranking using event verbs

We reranked documents using event verbs to increase the score of the documents about "basic biology". The event verb here means a verb widely used to represent interactions among the genes or proteins. We assume that documents containing many event verbs are likely to be about basic biology.

According to this assumption, we reranked documents by using the following new score function:

$$\textit{new score} = \textit{inital weight} + \textit{Additional weight} \qquad (6)$$

$$\textit{Additional weight} = \sum_{v=1}^{|V|} w^{(1)} \times \frac{(k_1+1)tf}{K+tf} \times \alpha$$

Where
initial weight is the weight between the query and the document, which is calculated at the initial retrieval,
v is the event verb and $|V|$ is the vocabulary size of a event verb list,
$w^{(1)}$,k_1, K, and tf are the same symbol used for equation (1).
α is the parameter depending on the reliability of reranking. We fixed it as 0.2.

The event verb list used in experiments consists of 182 verbs which is chosen by biologists for information extraction [Chun 2003].

5.2 Document filtering using MeSH

Unfortunately, many retrieved documents with the given query may have a lot of irrelevant documents, which focus on the basic biology of the query gene, but from another species.
To filter only the documents about genes from the species designated in given query, we used a simple heuristic using MeSH field in each document [NLM 2003] provided that the query gene from only the four species is given. The heuristic is as follows:

"If a document doesn't have a representative MeSH keyword for the species in the query, but has one of the representative keywords for other three species, remove the document from the list"

We choose four representative keywords for each species: 'human' for the human, 'rats' for the rat, 'mice' for the mice, 'drosophila' for the fruit fly.

6. EXPERIMENT AND EVALUATION

We have submitted two runs for the primary task of genomics track this year. The first run, KUBIO IRRAW, make use of simple rules for keyword extraction, query weighting using length normalization, and inverse query frequency, QW^2, reranking, and document filtering. The second run, KUBIOIRNE, uses one more strategy, phrase indexing method using a term boundary identification. Both of runs performed at or above the median in almost all queries, shown in table 3.

The results of table 4 show that there is little advantage of using phrase indexing strategy for keyword extraction. KUBIOIRNE shows a better performance than KUBIOIRRAW at all evaluation measures, but considering its cost, improvement is tiny.

	Avg Precision				Rel At 10 doc				Rel At 20 doc			
	Best	> Mid	=Mid	< Mid	Best	> Mid	=Mid	< Mid	Best	> Mid	=Mid	< Mid
KUBIOIRRAW	2	42	0	8	4	23	25	2	5	25	21	4
KUBIOIRNE	1	40	1	9	5	24	25	1	3	24	24	2

Table 3. Comparative results

	Avg Precision	R-Precision	Rel-ret	At 10 doc	At 20 doc
KUBIOIRRAW	0.2937	0.2696	541	0.2240	0.1690
KUBIOIRNE	0.2980	0.2837	532	0.2320	0.1710

Table 4. Retrieval results of submitted runs.

	Test Topics		Training Topics	
	Average Precision	Improvement over Baseline	Average Precision	Improvement over Baseline
Baseline	0.1619	+0.00%	0.3342	+0.00%
+ Phrase	0.1649	+1.85%	0.3197	-4.34%
QW^1	0.2011	+24.21%	0.3628	+8.56%
QW^2	0.2100	+29.71%	0.3797	+13.61%
+ Reranking	0.2121	+31.01%	0.3800	+13.70%
+ Filtering	0.2980	+84.06%	0.4201	+25.70%

Table 5. Retrieval performance at each step. Baseline represents the base model for retrieval including simple rules for keyword extractionis.

What is worse, performance of the phrase strategy with the basic model is lower than the baseline as shown in table 5.

Two possible reasons are as follows. One is the risk of a high inverse document frequency of phrase. Especially, some unsuitable phrases with abnormal high idf cause a trouble. Another reason is that when phrase strategy is used, long length terms of the query are more strongly favored. This tendency is proved indirectly in table 5. Improvement by the query weighting using length normalization, QW^1, is much bigger with the phrase indexing than without the phrase indexing.

Table 5 shows the relative improvement of the retrieval performance according to the additional techniques. Almost all our proposed methods for this task yield better results but one negative case, which use the base model with phrase indexing. Relatively, the phrase indexing and the document reranking method produce rather disappointing results, and query weighting methods and document filtering performed well.

The results of query weighting methods, QW^1 and QW^2, are fairly good as shown in table 5, table 6, and table 7. They achieved 17-29% improvement at test and training queries with any indexing methods.

The document reranking method makes just a little improvement. It achieved merely about 1% increase of average precision as shown in table 5, and table 7. We guess the reason is that the value of the parameter αused as 0.2 in experiments is too small to change a document rank, or our method for reranking documents was too heuristic.

The document reranking method makes just a little improvement. It achieved merely about 1% increase of average precision as shown in table 5, and table 7.

	Test (No filtering / Filtering)						Training (No filtering / Filtering)					
	No Phrase		Simple Rule		Phrase		No Phrase		Simple Rule		Phrase	
Base model	.1600	.2443	.1619	.2507	.1649	.2528	.2999	.3467	.3342	.3848	.3197	.3651
QW^1			.1822	.2691	.2011	.2843			.3496	.4086	.3628	.4070
QW^2			***.1966***	***.2929***	***.2100***	***.2957***			***.3586***	***.4164***	***.3797***	***.4195***

Table 6. Average Precisoin according to each methods. Bold is the best score.

	Test (No filtering / Filtering)						Training (No filtering / Filtering)					
	No Phrase		Simple Rule		Phrase		No Phrase		Simple Rule		Phrase	
Base model	.1312	.2188	.1399	.2200	.1416	.2306	.2773	.3073	.3112	.3569	.2734	.3350
QW^1			.1446	.2275	***.1647***	.2590			.3191	.3723	.3299	***.3767***
QW^2			***.1791***	***.2602***	.1639	***.2680***			***.3303***	***.3926***	***.3414***	.3719

Table 7. Recall-precision according to each methods. Bold is the best score.

	Test				Training			
	Average Precision		R-Precision		Average Precision		R-Precision	
	No filter	Filter	No filter	Filter	No filter	Filter	No filter	Filter
before Reranking	0.2100	0.2957	0.1639	0.2680	0.3797	0.4195	0.3414	0.3719
After Reranking	0.2121	0.2980	0.1709	0.2837	0.3800	0.4201	0.3396	0.3701

Table 8. Comparision between before and after reranking.

	Test		Training	
	Average Precision	R-Precision	Average Precision	R-Precision
Before filtering	0.2121	0.1709	0.3800	0.3396
After filtering	0.2980	0.2837	0.4201	0.3701

Table 9. Comparison between before and after filtering.

In spite of its simplicity, document filtering achieves the biggest improvement as shown in table 5, and table 9. It means that there are so many documents which are relevant but describe another species. In this task, satisfying the species constraint in a query seems to be important.

7. CONCLUSIONS

We have tried heuristic strategies which can reflect characteristics of this task. The strategies can be classified into three classes.

First one is the indexing strategy. To handle a lot of multi-word terms in biomedical literature, we have tried the phrase indexing method based on term boundary information. Named entity tagger was used for it, but it yields a rather disappointing result. We will have to devise a good phrase extraction method and a reliable phrase weighting scheme. It will be one of our future works.

Second one is the query weighting scheme. The query in this task is quite different from the other ad hoc task. We have developed two heuristic query weighting methods which can reflect the characteristics of the query, and the domain information, and they can increase performance of our system successfully.

Finally, we have used two post-processing methods. Our simple document filtering method works very well, but more analysis is required for documents reranking.

References

1. W. Hersh. http://medir.ohsu.edu/~genomics/
2. M. Fuller and J. Zobel. Conflation-based Comparison of Stemming Algorithms, In *Proceedings of the Third Australian Document Computing Symposium Sydney*, Australia, 21st August, 1998.
3. I. Namba, N Igata, H Horai, K Nittta, and K Matsui. *The Eight Text REtrieval Conference*, 2000.
4. K.J. Lee, Y.S Hwang, H.C. Rim. Two-Phase Biomedical NE Recognition based on SVMs, In *ACL'03 nlbio workshop,* 2003.
5. NCBI. http://www.ncbi.nlm.nih.gov/PubMed/
6. S.E. Robertson, S Walker. Okapi/Keenbow at TREC-8, In *The Eighth Text REtrieval Conference*, 2000.
7. S.E Robertson, and K Sparck Jones. Relevance weighting of search terms. *Jounal of the American Society for information Science 27*, May-June 1976, p129-146.
8. H.W.Chun, Y.S. Hwang, H.C.Rim. Unsupervised Event Extraction from Biomedical Literature using Co-occurrence Information and Basic Pattern, The *2nd annual conference of the Korean Society for Bioinformatics*, (To be appear), 2003.
9. NLM, http://www.nlm.nih.gov/mesh/

Answer Mining by Combining Extraction Techniques with Abductive Reasoning

Sanda Harabagiu, Dan Moldovan, Christine Clark, Mitchell Bowden, John Williams and Jeremy Bensley
Language Computer Corporation
Richardson, TX 75080

Abstract

Language Computer Corporation's Question Answering system combines the strengths of Information Extraction (IE) techniques with the vastness of axiomatic knowledge representations derived from WordNet for justifying answers that are extracted from the AQUAINT text collection. CICERO LITE, the named entity recognizer employed in LCC's QA system was able to recognize precisely a large set of entities that ranged over an extended set of semantic categories. Similarly, the semantic hierarchy of answer types was also enhanced. To improve the precision of answer mining, the QA system also relied on a theorem prover that was able to produce abductive justifications of the answers when it had access to the axiomatic transformations of the WordNet glosses. This combination of techniques was successful and furthermore, produced little difference between the exact extractions and the paragraph extractions.

Introduction

In 2003, the TREC QA track had two separate tasks: the *main task* and the *passage task*. LCC's QA system participated in both tasks. The main task combined three different question types: factoids, lists and definitions. Factoid questions seek short, fact-based answers in the document collections, e.g. Q1910: *"What are pennies made of?"*. Some factoid questions may not have an answer in the AQUAINT collection, and thus the correct answer in this case is *NIL*. Otherwise, the correct, exact answer is an entity, e.g. *steel* for Q1910. Factoid questions were evaluated similarly in the 2002 TREC QA track. This year however, the score of the main task was computed as a weighted average of the factoid score with the scores obtained for processing list questions and definition questions:

Main_task_score = $\frac{1}{2}$ × factoid_score + $\frac{1}{4}$ × list_score + $\frac{1}{4}$ × definition_score

This formula shows that in 2003, a QA system with very good performance (e.g. 76%) on factoid questions and with only 28% performance for list questions and 40% on definition questions would have achieved 55% overall performance. Another system, with much worse performance on factoid questions (e.g. 40%) but better performance for list questions (e.g. 60%) and definition questions (e.g. 80%) would have achieved the same performance. In general, a list question requests a <u>*set*</u> of instances of specified types, such as Q2014: *"List brands of pianos."* or Q1940: *"What grapes are used in making wine?"*. The response to a list question is a non-null, unordered and unbounded set of answer instances. In previous years, the cardinality of the set of list elements or instances was specified in the question. In the 2003 TREC, the list questions did not specify the target number of responses. The final answer set for any list question was created from the union of the distinct, correct responses returned by all participants plus the set of answers found by NIST assessors during question development. This final answer set was used for computing the F-measure of the question, which equally weighted the instance recall (IR) and the instance precision (IP). These measures were defined as:

- IR = #instances judged correct and distinct / #answers in the final set
- IP = #instances judged correct and distinct / #instances returned
- F = (2 × IP × IR) / (IP + IR)

The score for the list component was the mean of the F-scores of the list questions. The response to a definition question was also measured by the F-score, but the interpretation of the final set was different.

For each definition question, the assessor has created a list of acceptable information nuggets from the union of the returned responses and the information discovered during question development. Some of the nuggets are deemed essential, i.e. a piece of information that must be in the definition of the target in order to consider it a valid definition. The remaining nuggets in the list are acceptable. Once the list of acceptable nuggets are created, the assessor decides upon the acceptable and essential nuggets returned by each system for each question. Each nugget was matched only once. The definition questions were scored using nugget recall (NR) and an approximation of nugget precision (NP) based on length. These scores are combined using the F-measure in which NR is five times more important

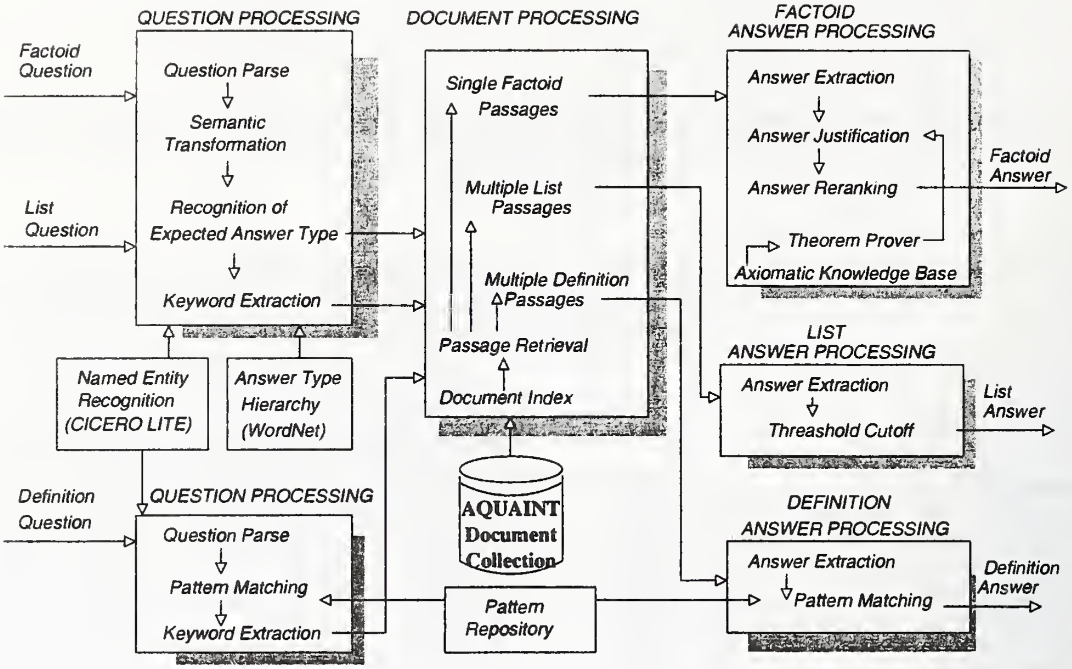


Figure 1: Architecture of LCC's QA system

than NP. The formulae used for measuring the definition score are:

- NR = #essential nuggets returned in response / #essential nuggets
- NP is defined using:
  - *allowance* = (100 × [#essential and acceptable nuggets returned]); and
  - *length* = (total #non-white space characters in answer strings):

  NP = 1, if length < allowance;
  NP = 1 - [(length - allowance)]/length, otherwise
- F = (26 × NP × NR) / (25 × NP + NR)

In TREC-2003, 413 factoid questions were evaluated; 37 list questions and 50 definition questions. Table 1 shows the distribution of the corresponding answers. The ideal system had to extract 383 exact answers for factoid questions, identify 30 NIL questions but also discover 549 list instances and 207 essential nuggets of definitions. If the list instances and the definition nuggets are approximated as exact answers, the factoid answers returned by this ideal system would have accounted only for 34% of the entire set of answers. The main-task score, although devised before knowing the cardinality of the final sets for the list and definition questions, attributes a 50% importance to factoid questions even for an ideal system. In other words, answering factoid questions was as important for the mai task of the 2003 TREC QA evaluation as answering list or definition questions.

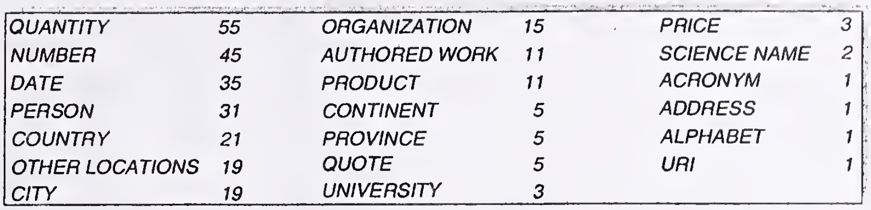
The passage task, in contrast, used only the factoid questions from the main task. A submission could contain only one response for each question, which could be either the string *NIL* or an extract from the document. A document extract is any text snippet of length smaller or equal to 250 bytes. This definition of he extract allowed the evaluations for this task to be performed by following the same procedure as in TREC-2001. A passage could be (a) incorrect, when it did not contain the answer; (b) unsupported, when it contained the answer but the document did not support the answer; or (c) correct. The final score was the fraction of the questions judged correct.

Answer type	*Count*
Answers to factoid questions	383
NIL-answers to factoid questions	30
Answer instances in List final set	549
Essential nuggets for Definition questions	207
Total nuggets for Definition questions	417

Table 1: Distribution of answers in TREC-2003

The architecture of the QA system

The architecture of LCC's QA system is illustrated in Figure 1. It is to be noted that the question processing module is identical for factoid and list questions, but different for definition questions. To process factoid or list questions, the QA system needs to identify the expected answer type encoded either as a semantic class recognized by Cicero Lite^{TM}, our Named Entity Recognizer, or in a hierarchy of semantic concepts, built using the WordNet hierarchies for verbs and nouns. The expected answer type is typically indicated by the head of one of the question phrases. The recognition of this



QUANTITY	55	ORGANIZATION	15	PRICE	3
NUMBER	45	AUTHORED WORK	11	SCIENCE NAME	2
DATE	35	PRODUCT	11	ACRONYM	1
PERSON	31	CONTINENT	5	ADDRESS	1
COUNTRY	21	PROVINCE	5	ALPHABET	1
OTHER LOCATIONS	19	QUOTE	5	URI	1
CITY	19	UNIVERSITY	3		

Figure 2: Name classes

head is based on syntactic dependencies as well as some semantic dependencies that are approximated from the question parse. For example, in the case of the factoid question Q1997: *"What American revolutionary general turned over West Point to the British?"*, the expected answer type is PERSON, a Named Entity class, determined by the noun *general* which is found in the hierarchy of humans in WordNet. The same expected answer type is found when processing the list question *"Who are professional female boxers?"*. But the main difference among the two kinds of questions stems from the fact that in the case of factoid questions, the system will search for a unique PERSON whereas for list questions, it will try to identify as many PERSONS as possible in any relevant passage.

When processing definition questions, the questions are parsed in order to detect NPs. Then NPs are then matched against a set of patterns. For example, question Q2041: *"What is Iqra?"* is matched against the pattern <What is Question-Phrase>, which is associated with the answer pattern <Question-Phrase, which means Answer-Phrase>. In this case, <Question-Phrase> is *Iqra* and the answer phrase will be extracted from *"Iqra , which means ' read ' in Arabic , was the first word that the arch - angel Gabriel is said to have spoken to Islam 's Prophet Mohammed"*.

The Document Processing module is the same for any of the three forms of questions, as it retrieves relevant passages based on the keywords provided by question processing. For factoid questions, it ranks the candidate passages after filtering out all passages that do not contain a concept of the expected answer type. In the case of list questions, it prefers passages having multiple occurrences of the expected answer type. In the case of definition questions, it allows multiple matches of keywords.

Answer extraction is performed differently for each form of questions. In the case of factoid questions, answers are first extracted based on the recognition of the answer phrase provided by CICERO LITE. If the answer is not extracted as a named entity, it is justified abductively by using a theorem prover that makes use of axioms derived from WordNet as well as other axioms approximating semantic relations or linguistic pragmatics. For example, for the factoid question Q2252: *"What apostle was crucified?"*, because apostles are classified in WordNet under PERSON, the CICERO LITE Named Entity Recognizer is not able to detect names of saints as PERSON, since it was not trained to do so. In the most informative paragraph, two names of apostles are found: the apostle Peter and the apostle Paul. Because the verb *crucified* is not used as a keyword, the candidate answer becomes *the apostle Paul.* But when abduction is performed, the correct answer, *the apostle Peter* is returned as the exact answer.

The extraction of definition answers relies on pattern matching. The answer phrase (AP) is identified based on the results of the parse. For the answer of question Q2041, two nuggets from the AP were evaluated as vital: *Arabic word for read* and *Gabriel's first word to Mohammed.*

List questions are extracted by using the ranked set of paragraphs and their corresponding exact answers. The paragraphs are processes with the goal of finding a cutoff measure based on the semantic similarity of answers. This cutoff measure determines the number of elements in each list answer. For example, the answer to question Q2014: *"List brands of pianos."* is the list *Ivers Pond*, *Baldwin*, *Boesendorfer*, *Steinway*, *Yamaha.*

Extracting answers for factoid questions

Our Question Answering system extracted 289 correct answers out of 500 factoid questions. Out of these, 234 correct answers were obtained by extracting the answer which was identified by the CICERO LITE system or recognizing it from the Answer Type Hierarchy. Table 2 illustrates some of the factoid questions that asked for city names as well as the answers returned by our system.

1898: What city is Disneyland in? Answer: Anaheim
1916: What city did Duke Ellington live in? Answer: Washington D.C.
1986: What city is Ole Mississippi University in? Answer: Oxford, Miss.
1912: In which city is the River Seine? Answer: Paris

Table 2: Factoid questions asking for city names

Similar to previous TREC QA evaluations, the Named Entity Recognizer had to identify a varied set of semantic classes. Figure 2 lists some of the semantic classes of the names as well as the number of times

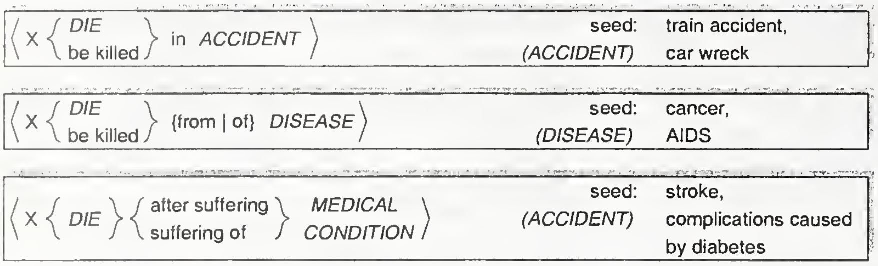


Figure 3: MANNER-OF-DEATH patterns

each of them were recognized correctly when extracting a factoid answer.

For locations, CICERO LITE distinguished among countries, cities, provinces, continents and other locations. After QUANTITY and NUMBER, the semantic classes associated with locations were the most numerous. A special class of names was trained for authored work, like names of books, songs or poems. Table 3 illustrates some of the factoid questions that were answered by this class of names.

1934: What is the play "West Side Story" based on? Answer: Romeo and Juliet
1976: What is the motto for the Boy Scouts? Answer: Be Prepared
1982: What movie won the Academy Award for best picture in 1989? Answer: Driving Miss Daisy
2080: What peace treaty ended WWI? Answer: Versailles
2102: What American landmark stands on Liberty Island? Answer: Statue of Liberty

Table 3: Questions asking for names of authored works

71% of the factoid questions were answered correctly because of a name that was recognized by CICERO LITE. The other 29% of the correctly answered factoid questions were answered by concepts that are represented in the answer type taxonomy employed by our system. The vast majority of these conceptual taxonomies classify concepts into such classes as: DISEASE, DRUGS, COLORS, INSECTS or GAMES.

There is one particular hierarchy that deserves more discussion. It is the MANNER-OF-DEATH category, developed because of previous TREC questions like *"How did Adolf Hitler die?"*. Text mining techniques for identifying such information were developed, based on lexico-semantic patterns from WordNet that were re-enforced in texts. For example, one such pattern is [kill#sense 1(verb) → cause → die#sense 1(verb)]. Some of the troponyms of the first sense of the verb *kill* are candidates for the MANNER-OF-DEATH hierarchy, e.g., *drown*, *poison*, *strangle*, *assassinate*, *shoot*. However, since not all MANNER-OF-DEATH are lexicalized as verbs, we set out to determine additional

1921: How did Virginia Woolf die? Sentence answer: When someone dies quoting from Virginia Woolf 's own *suicide* note , you think there must be even further options
1927: How did George Washington die? Sentence answer: Washington died from a *throat infection* at age 67 , almost three years after leaving the presidency
1928: how did Patsy Cline die Sentence answer: Who else died in the *plane crash* that killed Patsy Cline
1939: How did Einstein die? Sentence answer: Some 15,000 die from *ruptured abdominal aortic aneurysms* each year
2012: How did Marty Robbins die? Sentence answer: The late country - western singer , who died Dec. 8 , 1982 at age 57 after suffering a *heart attack* six days earlier at his Nashville , Tenn. , home , often would return home after a long day of rehearsing and still have enough voice left in him to deliver a private concert
2072: How did Brandon Lee die? Sentence answer: It was during filming of the original movie version of " The Crow " that actor Brandon Lee _ son of martial arts legend Bruce Lee _ was killed in an on - set shooting accident in 1993
2143: How did John Dillinger die? Sentence answer: On July 22 , 1934 , a man identified as bank robber John Dillinger was *shot to death* by federal agents outside Chicago 's Biograph Theater
2216: How did Dennis Brown die? Sentence answer: Initial reports suggested Brown died of *complications caused by respiratory problems* , but his cause of death had not yet been confirmed
2265: How did Marjorie Kinnan Rawlings die? Sentence answer: Alas , in that same year a *cerebral hemorrhage* dispatched her , only 57
2335: How did Harry Chapin die? Sentence answer: Chapin wrote some strong story - songs , but he was still a work in progress when he died in a *car accident* in 1981
2383: How did Jerry Garcia Die? Sentence answer: The Grateful Dead 's online ' OK ' will likely keep the buzz alive for a group that disbanded after lead singer Jerry Garcia died in 1995 of a *heart attack*
2386: How did Harry Houdini die? Sentence answer: In 1926 , magician Harry Houdini died in Detroit , suffering complications of a *ruptured appendix*

Table 4: Questions asking for MANNER-OF-DEATH

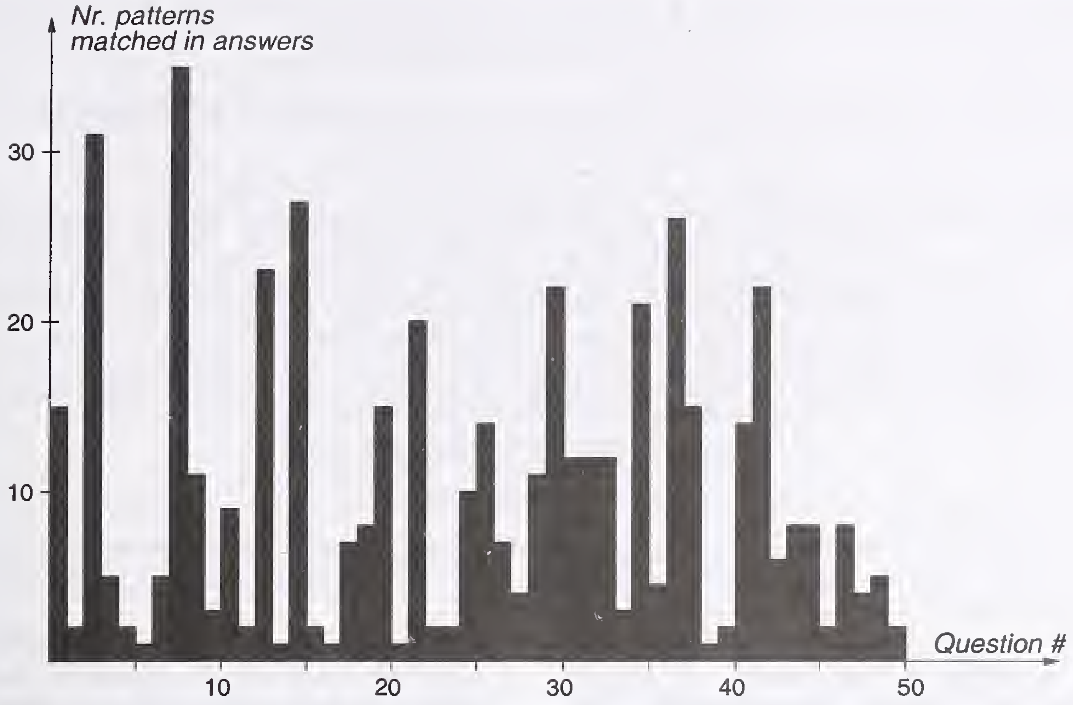


Figure 4: Number of answers extracted via patterns for each definition question

patterns that detect manners of death.

Especially for the cases when the cause of death is not lexicalized by a single noun or verb from the WordNet dictionary, we have developed a technique for acquiring (1) dictionaries for the cause-of-death; as well as (2) patterns that recognize manners of death. For this reason, we started with (a) a set of seed patterns and (b) a set of possible death causes. Figure 3 lists some of the seed patterns and their corresponding death causes.

By using the multi-level bootstrapping technique reported in (Riloff & Jones 1999) we populated this taxonomy with 100 concepts, which were manually verified. Table 4 lists factoid questions that are resolved by patterns extracted for MANNER-OF-DEATH questions.

CICERO LITE and the answer taxonomies alone are responsible for correctly extracting 234 answers. 65 additional correct answers are due to the theorem prover we employed, which was reported in (Moldovan *et al.* 2003). The role of the theorem prover is to boost the precision by filtering out incorrect answers, that are not supported by an abductive justification. For example, question Q2217: *"What country does Greenland belong to?"* is answered by *"Greenland, which is a territory of Denmark"*. Denmark is recognized as a COUNTRY name, therefore is extracted as an answer. Moreover, the gloss of the synset of {territory, dominion, province} is *"a territorial possession controlled by a ruling state"*. The logical transformation for this gloss is:

[control:v#1(e,x1,x2) & country:n#1(x1)
& ruling:a#1(x1) & possession:n#2(x2)
& territorial:a#1(x2)]

in which each lexeme has the format:

[root : part-of-speech # WordNet-sense].

All lexemes are predicalized, but verb lexemes have a special role: they have one argument *e* which stands for the eventuality of the event, state or action they represent (cf. Davidsonian treatment of actions) and their arguments stand for: x1=subject, x2=object. The subject and the object are recognized as predicates having the arguments x1 and x2 respectively. The same arguments are shared by the modifiers of the subjects and objects. Whenever the genus of the gloss was either one of the synset elements or one of its morphological variations (e.g. territorial for territory) the head of the genus indicates a specialization of the verbal predicate. In this case, the control is exercised by a possession, therefore the logical form of the gloss for sysnset {territory, dominion, province} can be specialized too:

[possess:v#2(e,x1,x2) & COUNTRY:n#1(x1)
& ruling:a#1(x1) & territory:n#2(x2)]

This specialized logical transformation also uses the unification between [control:v#1(e,x1,x2) & possession:n#2(x2) & territorial:a#1(x2)] and [possess:v#2(e,x1,x2) & territory:n#2(x2)]. Additionally, by using the logic form of the gloss of verb *belong*, which is *"be in the possession of"*, the predicate possess:v#2(e,x1,x2) may be replaced with belong:v#1(e,x1,x2), which resolves the abduction that proves question Q2217, if the answer is *Denmark*. The verb possess:v#2(e,x1,x2) and belong:v#1(e,x1,x2) express a form of meronymy which is not specifically

Id	*Pattern*	*Freq.*	*Usage*	*Question*
25	person-hyponym QP	0.43%	The doctors also consult with former Italian Olympic skier Alberto Tomba , along with other Italian athletes	1907: Who is Alberto Tomba?
9	QP , the AP	0.28%	Bausch Lomb , the company that sells contact lenses , among hundreds of other optical products , has come up with a new twist on the computer screen magnifier	1917: What is Bausch & Lomb?
11	QP , a AP	0.11%	ETA , a Basque language acronym for Basque Homeland and Freedom _ has killed nearly 800 people since taking up arms in 1968	1987: What is ETA in Spain?
13	QA , an AP	0.02%	The kidnappers claimed they are members of the Abu Sayyaf , an extremist Muslim group , but a leader of the group denied that	2042: Who is Abu Sayaf?
21	AP such as QP	0.02%	For the hundreds of Albanian refugees undergoing medical tests and treatments at Fort Dix , the news is mostly good : Most are in reasonably good health , with little evidence of infectious diseases such as TB	What is TB?

Table 5: Examples of definition patterns, usage and frequency of occurrence

encoded in WordNet. This form of meronymy corresponds to the semantics of territories being part of countries. The glosses of those verbs indicate that this meronymy is rather viewed as a form of possession, which due to the verb *control* from the gloss of *territory* shows preference to the more different relation of governing or ruling of territories by their countries. It is to be noted that Greenland is encoded in WordNet, its gloss *"a self-governing province of Denmark"* would have led to the same justification as the one determined by the text snippet retrieved from the AQUAINT corpus, creating a contradiction between the concepts *self-governing* and *control by possession.* Currently, the abductive processes implemented in the COGEX theorem prover (Moldovan *et al.* 2003) do not handle such contradictions. If the relation of provinces/territories belonging to countries would have been encoded as meronymy, then Greenland should have been encoded as part of Denmark in WordNet. (Currently, it is encoded as a part of the Atlantic Ocean)). The latter two meronyms show preference for geographical membership rather than country/nation possession.

For question Q2217, the absence of the abductive justification would have produced *Ethiopia* as the answer, because of the text snippet *"the high ice desert of Greenland and the tributaries of the Blue Nile in Ethiopia"*.

Extracting answers for definition questions

Our QA system extracted 485 answers in response to the 50 definition questions evaluated in TREC-2003. We submitted two runs, one of which consisted of exact answers and the other of the corresponding sentence-type answers. Out of 485 answers, the assessors have found a total of 68 (exact) and 86 (sentence) vital matches from a total of 207 they expected and a total of 110 (exact) and 144 (sentence) matches out of 417 they had in their final set. The definition questions evaluated in TREC-2003 can be classified in:

- questions asking about people;
- questions asking about other types of names; and
- questions asking about general concepts.

The questions asking about people started with the question stem *Who* and contained the name of a person. There were 30 such questions, 22 of which had the person name in the format *first name - last name*, e.g. *Aaron Copland*, *Allen Iverson* or *Albert Ghiorso*. One question had the name in the format *first name - last name1 - last name 2*, i.e. *Antonia Coello Novello.* Three questions had the name as a single word, signifying that they are very well known: *Nostradamus*, *Absalom* and *Abraham.* In the latter case, the context was also specified: *"Abraham in the Old Testament"*. Two other person names were names of old kings or princes: *Vlad the Impaler* and *Akbar the Great*, having the format *first name - the attribute.*

There were 14 questions asking about other types of names. Four asked about different organizations, e.g. *Bausch & Lomb*, *ETA*, *Friends of the Earth* or *Destiny's Child.* Two asked about cities, e.g. *the Hague* but also nicknames of cities, e.g. *Bollywood.* Two asked about medical or biology terms, e.g. *TB* or *Ph.* Three asked about words in foreign languages: e.g. *Iqra* in Arabic, *Schadenfreude* in German, and *Kama Sutra* in Sanskrit. Six definition questions asked about general concepts, e.g. *fractals*, *golden parachute* or *quasar.*

To produce answers for definition questions, our system uses 38 patterns. Out of these, 23 patterns had at least a match for the tested TREC questions. Table 5 illustrates the most popular patterns. Figure 4 illus-

Cecilia Bartoli broke her right ankle slipping on ice outside the Zurich Opera but still intends to sing her first Donna Elvira in Mozart 's "Don Giovanni" this Sunday
Sally Wolf gives a movingly tormented performance as Donna Elvira and is at her best in the great spleen - venting aria "i tradi quell" alma ingrata
Donna Elvira's cold fury seemed to emanate as much from the natural personality of the singer (*Veronique Gens*) as from the nature of the role
In the title role, the resonant bass Ferruccio Furlanetto led a strong cast that included the powerful bass Rene Pape as, surprisingly, a vocally agile Leporello; the exquisite soprano Renee Fleming as a Donna Anna to cherish; and the luminous - voiced soprano *Marina Mescheriakova* as Donna Elvira
A newcomer, the Norwegian soprano *Solveig Kringelborn*, sang Donna Elvira with a clean intensity
Lott is celebrated for her Mozart roles (and recorded her Fiordiligi in "Cosi fan tutte," the Countess in "The Marriage of Figaro," and Donna Elvira in "Don Giovanni"; earlier this season the Met broadcast her elegant portrayal of the Countess) and for her Strauss (she has taken her Arabella , Marschallin , Countess Madeleine , and *Christine Storch* to most of the major opera houses of the world)

Table 6: Answers for question Q2002

trates the number of answers extracted through pattern matching for each of the 50 definition questions.

Answering list questions

To answer the 37 list questions evaluated in TREC-2003, our QA system considered a threshold-based cutoff of the answers extracted. The general idea was that by using concept similarities between the candidate answers we could decide on the threshold value for submitting the elements of the list. Given that for a question we extract N list answers, we first compute the similarity between the first answer and the last answer, S_{1N}. In general, to compute the similarity between a pair of answers (A_i, A_j) we consider a window of three noun or verb concepts to the left and to the right of the exact answer: $W_i = (C^i_{-3}, C^i_{-2}, C^i_{-1}, C^i_{+1}, C^i_{+2}, C^i_{+3})$ and $W_j = (C^j_{-3}, C^j_{-2}, C^j_{-1}, C^j_{+1}, C^j_{+2}, C^j_{+3})$. Then we separate the concepts in nouns and verbs obtaining N_i, N_j, V_i, and V_j. The similarity is measured by the formula: $sim(A_i, A_j) = \frac{1}{2}(sim^N(N_i, N_j) + sim^V(V_i, V_j))$, where $sim^N(N_i, N_j) = \frac{1}{P_{(N_i,N_j)}} \sum sim^C(n_i, n_j)$, with $P_{(N_i,N_j)}$ representing all the possible pairs (n_i, n_j) in which $n_i \in N_i$ and $n_j \in N_j$; and $sim^V(V_i, V_j) = \frac{1}{P_{(V_i,V_j)}} \sum sim^C(v_i, v_j)$, with $P_{(V_i,V_j)}$ representing all the possible pairs (v_i, v_j) in which $v_i \in V_i$ and $v_j \in V_j$.

The concept-based similarity is computed as: $sim(n_i, n_j) = argmax \quad sim(c_i, c_j)$, where (c_i, c_j) are all possible combinations of the WordNet senses of n_i and n_j, and

$$sim(c_i, c_j) = \begin{cases} 1, & if \;\; c_i = c_j \\ 0, & if \;\; c_i \;\; and \;\; c_j \;\; do \;\; not \;\; belong \\ & \quad to \;\; the \;\; same \;\; hierarchy \\ sim_{LC}(c_i, c_j), & otherwise \end{cases}$$

where $sim_{LC}(c_i, c_j)$ is the Leacock-Chodorow similarity (Leacock & Chodorow 1998) defined as:

$$sim_{LC}(c_i, c_j) = \log \frac{len(c_i, c_j)}{2D}$$

where $len(c_i, c_j)$ is the shortest path between c_i and c_j and D is the overall depth of the WordNet taxonomy.

A threshold value of $Z = C \times S_{1N}$ is computed using the similarity between the first and the last concept answer multiplied with a constant C. The cutoff is determined as the largest value t that satisfies: $\frac{1}{t}\sum_{i=1}^{t} S_{1i} > Z$. When the cutoff is determined, it represents the length of the list of answers that is submitted.

The best precision and recall was obtained for Q2002: *"Name singers performing the role of Donna Elvira in performances of Mozart's "Don Giovanni""*. Table 6 lists the answers extracted for this question. There were 5 correct answers out of 6 submitted, corresponding to a precision of 0.833 and a recall of 0.625; the combined F-score was 0.714.

Performance evaluation

Table 7 summarizes the scores provided by NIST for our system. We have submitted two different runs. They differ only in the way definition answers were extracted. In the first submission, only exact answers were extracted whereas in the second the whole sentence containing the answer was submitted.

Table 7 illustrates the contribution of the factoid, list and definition components to the overall scores of the main tasks.

	factoid	list	definition	all
Main task submission 1	70.0%	39.2%	36.1%	53.8%
Main task submission 2	70.0%	39.6%	44.2%	55.9%
Passage task	68.5%	N/A	N/A	N/A

Table 7: Results in TREC-2003 evaluations

The score of the second submission was slightly higher than that of the first submission because of the better score obtained for the definition questions, which in this case were in the format of an entire sentence. This allowed more vital nuggets to be identified by the assessors, thus obtaining a better score. Another important observation stems from the fact that factoid questions in the main task were slightly better evaluated than in the passage task. We explain this fact by our belief that the passage might have contained multiple concepts similar to the answer, and thus produced a more vague evaluation context.

Acknowledgments

We would like to thank Drs. Mihai Surdeanu and Marius Paşca for their valuable contributions. This work, as well as our broader research on QA, is supported by the Advanced Research and Development Activity (ARDA)'s Advanced Question Answering for Intelligence (AQUAINT) Program.

References

Leacock, C., and Chodorow, M. 1998. *WordNet: An Electronic Lexical Database and Some of its Applications.* MIT Press. chapter Combining local context and WordNet similarity for word sense identification, 265–283.

Moldovan, D.; Clark, C.; Harabagiu, S.; and Maiorano, S. 2003. Cogex: A logic prover for question answering. In *Proceedings of the Human Language Technology and North American Chapter of the Association for Computational Linguistics Conference (HLT-2003)*, 87–93.

Riloff, E., and Jones, R. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, 474–479.

A non-functional prototype at TREC 2003

Brian D. Davison, Wei Zhang, and Josh Miller
Department of Computer Science & Engineering
Lehigh University, Bethlehem, PA 18015
{davison,wei,jamr}@lehigh.edu

Overview

As a first attempt at participation in the TREC competition, we built a system which produced some preliminary results, but was unable to generate the quality of results that we expected. While we were able to submit four base-line runs, bugs were discovered in the final hours before the deadline making it impossible to submit results using our intended implementation. We have since found additional coding errors, making our submitted results expectedly poor.

The size of our index dataset was approximately 3.8GB without compression. We did not use term position information nor any kind of phrasal indexing.

Topic distillation task

We submitted two runs for topic distillation. They employed both vector space and simple popularity-based link analysis techniques. Queries were down-cased and stop words were removed before ranking.

Term weights (both for terms in the main document, as well as terms in anchor text) were calculated as the $\log_{10}$ of (termfreq + 1).

For the 03wume206 run, the final document score was calculated as follows:

$$\begin{aligned}\text{docs[i].score} = {} & \log_{10}(\text{docs[i].termweight}+1) \\ & + \log_{10}(\text{docs[i].anchorweight}+1) \\ & + \text{docs[i].rlinkweight};\end{aligned}$$

where docs is an array of documents found to contain the queries, termweight is the number of times the keywords appear in this document, anchorweight is the number of URLs that contain query terms and link to the document, and rlinkweight stands for reverse link weight, which records how many other documents link to this page. Term and anchor weights are not normalized, but the reverse link weight is normalized by dividing by the sum of all incoming links to any document in the relevant set.

The `03wume359` run employed some slightly more sophisticated approaches. We used a different term weighting approach — a variant of Salton and Buckley's method [1], and a more subtle approach for calculating link weights. The final score still followed the equation above, but the term weight portion was calculated as (0.5 + (0.5 * termfreq)) * $\log_{10}$(docs/termdocs) where docs is the number of all documents containing at least one query term, termdocs is the number of all documents containing this term. Additionally, instead of simply counting the number of incoming links, rlinkweight was defined as the number of incoming links from this relevant subset divided by the total number of incoming links to this page. In this way we hoped to emphasize pages that were predominantly cited within this query topic.

Navigational task

We did not attempt a different approach for the mixed homepage and named page queries. All queries were treated in the same way as in topic distillation. These runs only employed vector space and anchor text. To obtain term weight and anchor weights, the same algorithm was used as in topic distillation. The only difference was a 20% reduction for standard term weights in the `03wume296` run.

$$\texttt{03wume296}\text{: docs[i].score} = \log_{10}(\text{docs[i].termweight}+1) * 0.8 \\ + \log_{10}(\text{docs[i].anchorweight}+1)$$

$$\texttt{03wume298}\text{: docs[i].score} = \log_{10}(\text{docs[i].termweight}+1) \\ + \log_{10}(\text{docs[i].anchorweight}+1)$$

Results after bug fixes

After fixing a number of bugs (after the competition was complete), but without changing the logic, we re-ran our system on both tasks. The performance metrics of the original and corrected system are shown in Table 1. The corrections almost tripled our system's performance on the navigational task, and improved performance on the topic distillation task by approximately 60%.

While the relative score improvement was large for the navigational task, the overall performance was still low, and would only change our relative ranking by a couple of positions (assuming all others stayed the same). In contrast, the smaller relative improvement in the topic distillation translates to a movement of 16 positions in the system rankings.

Topic distillation task

Rank	R-Prec	MAP	P@10	Group	Run	D	A	L
(70)	0.0636	0.0517	0.0380	lehighu	03wume206corrected	-	A	L
86.	0.0395	0.0343	0.0280	lehighu	03wume206	-	A	L
(89)	0.0357	0.0295	0.0160	lehighu	03wume359corrected	-	A	L
91.	0.0204	0.0225	0.0180	lehighu	03wume359	-	A	L

Navigational task

Rank	MRR	S@10	Group	Run	D	A	L
(69)	0.189	28.0	lehighu	03wume298corrected	-	A	-
71.	0.067	9.3	lehighu	03wume298	-	A	-
73.	0.065	8.7	lehighu	03wume296	-	A	-

Table 1: Original and corrected scores for topic distillation and navigational tasks.

In the end, however, while all improvements are welcome, the corrected scores are still not particularly competitive, and point to the need for fundamentally better algorithms.

Conclusion

Even after coding errors were corrected, the performance of this simplistic implementation was not competitive. However, it does provide a foundation on which we expect future work to build.

References

[1] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1998.

LexiClone

lexical cloning system

The Role and Meaning of Predicative and Non-predicative Definitions in the Search for Information.

Ilya S Geller
LexiClone Inc.

Borrowings and Adaptations.

First of all, the author would like to clarify the meanings of the terms "predicative" and "non-predicative". As is well known, *praedicatum* in Late Latin means "what has been said (previously)". In Aristotelian and subsequent forms of traditional logic [8,9,11,12,13,14,15,16] a predicate was understood to be one (the one in which something is said about the subject of speech) of the two terms for the judgment of a subject. In his treatment the author counts as predicative any definition of a subject or object in which something is said about an observable subject or object as it changes. In addition, the author proposes as the sole measure of change the movement of a subject or object with acceleration [5,6,7,17]: if a subject or object is immobile or moves evenly, it cannot be observed and, consequently, cannot be defined predicatively [1].

More: as is well known, Bertrand Russell introduced the notion of a "non-predicative" definition, in which what is to be defined is brought in through its relation to a class of which it is an element. For example: "the set of all sets that are not elements of themselves". It is said that the use of "non-predicative" definitions leads to paradoxes, so they should be dealt with carefully.

The author adopts Russell's definition [27,28,29,30,31], but in a new mode: one counts as non-predicative any definition of a subject or object in which something is said about a subject or object that is unchanging. For example, according to the author there are no closed sets in the world of change - a set can strive to be the set of all sets, but no more.

The Text.

A text is made up of words. But what is a word? First of all, a word is made up of letters, which are, in practice, meaningless if separate. And yet a word, as the joining together of several letters, already, beyond any doubt, has a certain meaning. But the existence of synonymy makes the meaning of words, taken separately, vague and lacking in concreteness and separate words are declared to be non-predicative definitions. For example, the word “red”, taken by itself, can mean anything: beginning with a colour and ending with a pejorative name for a Communist. In order to understand the “true” meaning of a word one must first identify in what minimal lexical construction of speech and in what slang a given word is being used; where:

I. A minimal lexical construction of speech is a predicative definition: the articulation of three words, relating to three parts of speech - substantives, verbs and adjectives - in the context of a sentence. All other parts of speech, with the exception of prepositions and interjections, can be, in some way or other, taken to be substantives, verbs, and adjectives, where:

1. A substantive has the meaning of the abstract Name of certain points of accumulation[1];
2. A verb defines the abstract Name of an action;
3. An adjective is the abstract Name describing points of accumulation in the process of change.

II. A slang is an aggregate of predicative definitions used with a strictly specialized meaning, particular to each and every type of human activity. [2,3,4,22,26,33,36]

This triad of non-predicative definitions is indispensable to defining the subjective evaluation of a fact, when faced with the possibility and the need to include the objects and subjects of the fact

[1] A point of accumulation is the boundary point of the set M - the point x of the topological space X>M, of which any vicinity contains an indefinite number of points of the set M. Everything – all things, animals, humans, etc. – are “accumulation points”. [8,9,11,12,13,14,15,16,35,36,38].

within the certain context. In other words, a person has to evaluate a sandwich from all sides: he has to understand that it's a sandwich, and to decide whose it is, whether he should eat it or not, whether it's fresh and tasty, etc.

Moreover, the presence of at least one predicative definition is absolutely necessary and sufficient for the creation of a sentence, even if it's missing one or more words from the substantive/verb/adjective triad. Such a word or words can be reconstructed on the basis of the context and subtext of the predicative definition; where:

1. The context consists of those predicative definitions where a substantive is used as the abstract Name of points of accumulation and abstractions;
2. The subtext consists of those predicative definitions where pronouns and interjections are used as the abstract Name of points of accumulation and abstractions.

For example, having said the word “unfresh”, one can reconstruct the words “sandwich” and “exists” if we know in what context and subtext the word “unfresh” appears. And if we don't know the context and subtext of a given predicative definition, then the word “unfresh” can be used with, for example, the words “fish” and “smells”. Only a text, being a collection of predicative definitions grouped together in meaningful sentences, can provide, more or less identically, the context and subtext of every one of these predicative definitions. That is, a text is considered to be completed in so far as its context and subject are, more or less identically, defined.

Processing the Text.

The task comes down to extracting all the predicative definitions from every sentence of the text, and then counting how many times each one occurs in the text. Such a collection is termed a summary; the number of times each predicative definition occurs in the text is referred to as its weight. (NLP uses the notion of predicative definitions not counting weights of them.) A summary, being an ordered list of triads, is susceptible to rapid processing by computer. The margin of error in the cloning is lessened in proportion to the amount and size of the texts being used.

Examples of Summary.

The entire summary of George Bernard Shaw, created on the basis of his books as found on the Internet at the URL http://promo.net/pg/ , contains a little over 320,000 triads occurring more than once. The first triad - it-be-in - occurs 4 755 times; the second in order of frequency - i-be-in - occurs 2 534 times [18,19,20].

Similarly, at the URL http://lexiclone.com/SummarySample_Fyodor_Dostoevsky.htm ,
the reader can see an extract from Fyodor Dostoevsky's summary (a part of which is reproduced below), created on the basis of his book The Brothers Karamazov (the numbers to the right represent the frequency of each triad-phrase's occurrence in the text):

it - be - in : 1 466
i - be - in : 1 347
it - have - in : 996
you - be - in : 936
you - be - your : 798
i - have - in : 664
all - be - in : 657
it - will - in : 535
my - be - in : 496
all - have - in : 473

Clearly, it is subtext that dominates in Dostoevsky's text. In another summary - that of our Patent #6.199.067 - it is context that dominates [32]:

one - say - least : 1 447
segment - say - least : 1 124
datum - item - plural : 1 025
system - say - remote : 950
datum - say - plural : 888
computer - say - remote : 845
datum - item - linguistic : 845
system - say - least : 844
computer - say - least : 818

one - say - remote : 805

It appears that the preponderance of subtext can be explained by the fact that certain texts concentrate on the Ethical component in the process of a person's becoming whole[2] - on the question, what will happen if certain points of accumulation are included in the vicinity of a given? - rather than on the Aesthetic component. On the other hand, texts of the kind we might call "technical" are primarily concerned with the Aesthetic component, examining not the consequences of an action but rather its mechanism, and therefore context predominates in them. The summary of this article is:

weight - be - in : 592
weight - be - summary : 436
text - be - in : 418
it - be - in : 306
text - be - summary : 265
weight - weight - in : 214
i - be - in : 210
weight - weight - summary : 194
weight - say - in : 182
say - weight - in : 182
triad - be - in : 176
each - be - in : 176
all - be - in : 169
say - say - remote : 168
say - say - in : 164
text - weight - in : 158
phrase - be - in : 156
text - weight - summary : 134
one - be - in : 134
say - say - plural : 124
text - say - in : 116
say - be - least : 114
each - be - summary : 114
triad - be - summary : 109
triad - weight - summary : 61

To show summaries in their entirety would be impossible because of their extremely large (up to 3M) size [34].

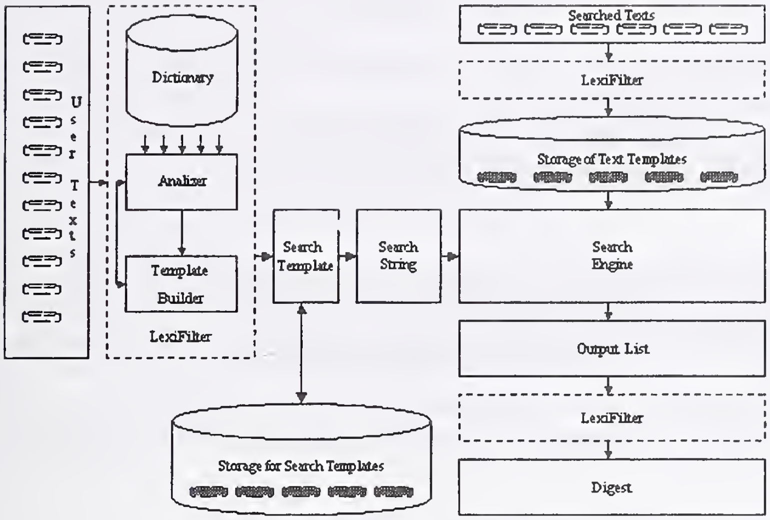
The Search Engine.

The author, as an immigrant in the lair of unbridled capitalism with its limitless opportunities to "make" money, naturally put the theory into practice at once and created a search engine for finding textual information in electronic form: the program is demonstrated on the Internet at the URL http://www.lexiclone.com/Products.htm

The system works as shown in the following diagram:

[2] The author assumed that a point of accumulation (an open set) always strives to become a/the material point (a/the closed set); and that this striving is the motivation for the universe to "spin" around, to change: the point x of the topological space X>M strives to include (for the sake of closeness) some other points of accumulation in its neighborhood. But after becoming complete-closed a set (a point of accumulation) is to lose all its qualities that make it unique and distinct. Here, the author sees the well-known Russell's paradox: how to distinguish something that has exactly the same quality? In other words: if something is "red", how to distinguish this "red" from all other "reds" (a complete-closed set does not change and cannot be predicatively defined)? [35,36,37,38]

Figure 1



The program, called UniSearch, does the following:
It fixates the searcher's summary, as a network of predicative definitions within limits; compares it with the summaries of authors, taken within limits, for a certain collection of texts. The result of the operation of the program consists of several (sufficiently few) sentences or predicative definitions, distilled out of the collection of texts, containing the desired information and called "digests".
These are some examples of searches' results: the author asked clones *How should we prosecute terrorists, murderers who kill innocent people? What should we do with them? Should we kill them in return?* The clone of Plato said: *How should we answer him?* The clone of the Bible said: (4.0% Ephesians) *For we are his workmanship, created in Christ Jesus unto good works, which God hath before ordained that we should walk in them.* The clone of Mohamed said: *Thee do we serve and Thee do we beseech for help.* The clone of Bernard Shaw (20th century) answered: *Who are we that we should judge them?* Fedor Dostoevsky said - *Well, what can we do?*
For the normal functioning of the program it is desirable, although not necessary, to create a summary of the user. Then,
1. The user puts in a search text, of any size;
2. On the basis of that text, the program selects those texts in the collection that were created by authors whose own summaries are closest to the summary of the person seeking the information, as extracted from the search text.

Compatibility.

The choice of one or another text is made through the medium of a standard formula, which is called a “controller” in Reinforcement Learning, and later named Compatibility by the author:

Figure 2

$$\text{Compatibility} = \left(\frac{\text{Sum}\left(\text{Weight-SU} * \text{Weight-ST}\right)}{\text{sqrt}\left(\text{Sum}\left(\text{Weight-each-SU}^{2}\right) * \text{Sum}\left(\text{Weight-each-ST}^{2}\right)\right)} \right) *100$$

where:
Weight-SU is the weight of the triads that are common to the summaries of the searcher and of the text,
Weight-ST is the weight of the predicative definitions that are common to the user’s summary and the summary of the text,
Weight-each-SU is the weight of each triad in the user’s summary,
Weight-each-ST is the weight of each predicative definition in the summary of the text [23].

References.

[1] Aristotle, A new Aristotle reader, Princeton University Press, Princeton, 1995
[2] A. Ayer, The Foundations of Empirical Knowledge, Macmillan, 1940
[3] A. Ayer, Logical Positivism, The Free Press, Glencoe, Illinois, 1960
[4] A. Ayer, Language, Truth and Logic, Dover Publications, Inc., NY, 1952
[5] N. Bohr, Atomic Physics and Human Perception, Moscow, 1962
[6] N. Bohr, Can Quantum Mechanical Description of Physical Reality Be Considered Complete? Moscow, 1992
[7] Bohr N., Continuity, Determinism and Reality”, Moscow, 1992
[8] F.H.Bradley, The Principles of Logic, Oxford University Press, London, 1883
[9] F.H.Bradley, Appearance and Reality, Oxford at the Clarendon Press, 1959
[10] Brill E., Susan Dumais and Michele Banko. An Analysis of the AskMSR Question-Answering System
[11] G. Hegel, Science of Logic, Vols.1-3, Moscow, 1972
[12] G. Hegel, Phenomenology of Mind, George Allen&Unwin Ltd., London
[13] G. Hegel, Lectures on the History of Philosophy, Moscow, 1932
[14] G. Hegel, On Art, Religion, Philosophy, Harper Tourchbooks, Harper&Row, Publishers, NY, 1980
[15] G.Hegel, The Encyclopedia Logic, Hacket Publishing Company, Inc., Indianapolis, 1991
[16] G. Hegel, The Essential Writings, Harper Torchbooks, Harper&Row, Publisher, NY
[17] W. Heisenberg, Problems of Philosophy, Moscow, 1953
[18] D. Hume, Dialogues Concerning Natural Religion, Hacket Publishing Company, Cambridge, 1984
[19] D. Hume, Enquiries concerning the human understanding and concerning the principles of morals; Oxford, Clarendon P., 1969
[20] D. Hume, Treatise of Human Nature, Oxford at the Clarendon Press, 1951
[21] R. Geyer, Alienation theories; Oxford, 1980
[22] W. James, Pragmatism, New York, 1902
[23] H. Lebesque, About Measurement of Measure, Moscow, 1969
[24] L. Peshkin, Reinforcement Learning by Policy Search (dissertation)
[25] L. Peshkin, Edwin D. de Jong. Context-based policy search: transfer of experience across problems
[26] Pierce, Collective papers. Vol.5, Cambridge, 1960
[27] B. Russell, Scientific Method in Philosophy, Oxford, 1914; Moscow, 1988
[28] B. Russell, Introduction to Mathematical Philosophy, London, 1953; Moscow,

1989
[29] B. Russell, Principles of mathematics; Bradford & Dickens, London, 1950
[30] B. Russell, The Problems of Philosophy, Hacket Publishing Company, Inc., Indianapolis, 1996
[31] B. Russell, My Philosophical Development, Ruskin House, George&Unwin Ltd, London, 1959
[32] G. Ryle, Dilemmas, Cambridge at the University Press, 1954
[33] D. Sack, Human Territoriality; Cambridge University Press, Cambridge, 1986
[34] P. Turney, Mining the Web for Lexical Knowledge to Improve Keyphrase Extraction: Learning from Labeled and Unlabeled Data

Some Extra References.

[35] P. Aleksandrov, "An introduction in the theory of sets and general topology", Moscow, 1977
[36] I. Geller, New Mechanics: The Foundation, http://lexiclone.com/NEWONE1_BODY.html
[37] I. Geller, In Search of a Living Language, http://lexiclone.com/insearchofalivinglanguage.htm
[38] G. Cantor, Works on the theory of sets, Moscow, 1985

AnswerFinder in TREC 2003

Diego Mollá
Centre for Language Technology
Macquarie University
Sydney, NSW 2109
Australia
Tel. +61 2 9850 9531
Fax +61 2 9850 9551
diego@ics.mq.edu.au

Abstract

In this our first participation in TREC we have focused on the passage task of the question answering track. The main aim of our participation was to test the impact of various types of linguistic information in a simple question answering system. In particular, we have tested various combinations of word overlap, grammatical relations overlap, and overlap of minimal logical forms in the final scoring module of the system. The results indicate a small increase of accuracy with respect to a baseline system based on word overlap. Overall, given the short time available for developing the system, the results are satisfactory and equal or surpass the median.

1 Introduction

This is the first time that the Centre for Language Technology at Macquarie University participates in TREC. Due to strong time restrictions we decided to implement a simple and functional system for the passage task of the Question Answering track. The final estimated time of development was about 55 person-hours (plus execution time) distributed among three months of intermittent work.

Section 2 describes the general architecture of the system. Our main focus was the exploration of sentence similarity measures for question answering. The measures used were based on word overlap, grammatical relations, and minimal logical forms. The two latter measures are described in Sections 3 and 4, respectively. Section 5 presents the final results and discussion. Finally, Section 6 describes a post-submission extension that incorporates question classification and named-entity recognition.

2 System Architecture

The system is fairly straightforward (Figure 1). A **document preselection** stage returns the documents preselected by NIST. A subsequent **sentence preselection** stage splits the documents into sentences and ranks the sentences according to word overlap. A final **scoring** stage analyses the top-ranking sentences returned by the previous stage and re-ranks the sentences according to a similarity measure. The top-ranking sentence is returned as the answer, possibly truncated if it is longer than the limit of 250 characters. Note that all questions are treated the same way. In other words, there is no question classification stage. Also, no attempt to detect NIL answers was made.

It is worth noting that the complete process was done when the question was processed. All the data structures (except, of course, the documents provided by NIST) were built on the fly.

To determine the number of documents to preselect, during the development of the system we analysed the questions used in TREC 2002 and the answers available from the TREC web pages. In particular, we used the regular expressions provided by Ken Litkowsky to determine if an arbitrary document contained the answer of a specific question. The results of this analysis are

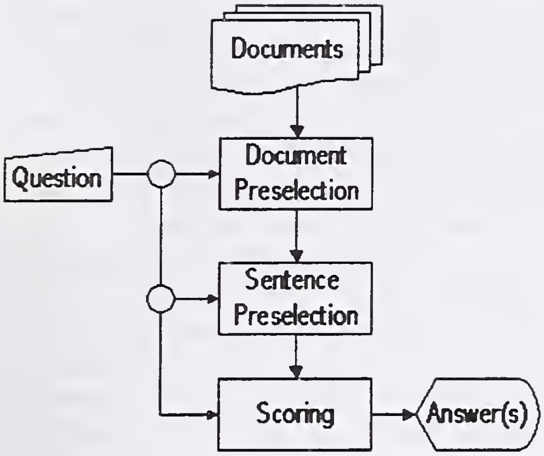


Figure 1: Architecture of AnswerFinder.

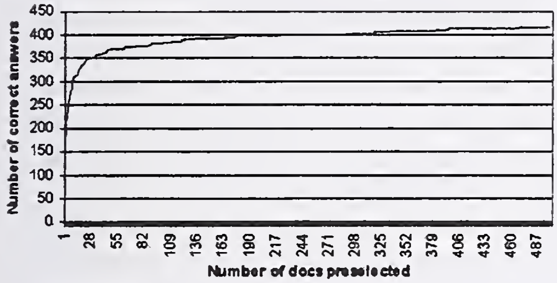


Figure 2: Relation between preselected documents and number of correct answers.

not necessarily accurate for two reasons. First of all, good answers phrased in unfamiliar terms may not be covered by the regular expressions. Second, some text may happen to match a regular expression by coincidence but still the document may fail to support the answer. Still, the results are indicative for our purposes. The results of our analysis is summarised in Figure 2, which shows the number of questions that have an answer within the top X documents. We can see that there is little gain by preselecting many documents. For practical reasons we set the threshold of documents to preselect to $X = 50$. For that threshold, 74% of the questions would find a document that satisfies the regular expression of the answer.

To split the documents into sentences we used a simple regular expression that detects punctuation characters as end-of-sentence markers:

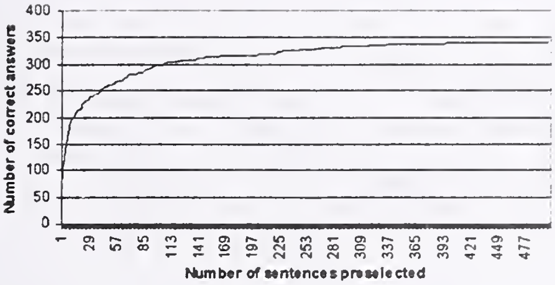


Figure 3: Relation between preselected sentences and number of correct answers (using the top 50 documents).

```
'(?:\.|\?|!|;|<.*?>)+'
```

The resulting sentences are ranked according to word overlap. After several experiments we found that the optimal measure of word overlap is obtained by using a list of stop words[1] and ignoring repeated words in the answer candidate. We arbitrarily set a threshold of 100 sentences to be sent to the final scoring stage. Figure 3 shows the relation between the sentences preselected and the number of correct answers. For the threshold of 100 sentences and using the 2002 question set, 59.4% of the questions would have a string that satisfies the regular expression of the answer. This is therefore the expected upper limit of accuracy that the scoring module can achieve.

The final scoring module combines several types of information. Apart from word overlap we experimented with other types of overlap that use various types of linguistic information. In particular, we used grammatical relations and minimal logical forms, as described in the following sections.

3 Grammatical Relations

The grammatical relations by Carroll et al. (1998) were devised to enable comparative evaluations of parsers. Following their evaluation methodology, the output of the parsers to evaluate is converted into sets of grammatical relations, thus enabling the representation of the parser output in a uniform way. In order

[1] We used the list of stop words available from http://www-fog.bio.unipd.it/waishelp/stoplist.html

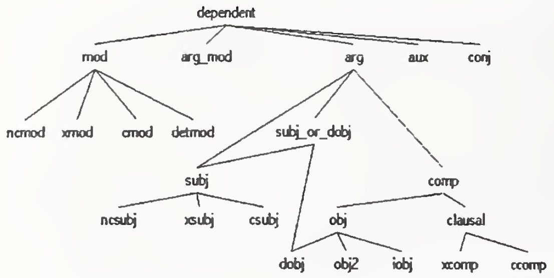


Figure 4: Hierarchy of grammatical relations (Briscoe and Carroll, 2000).

to be able to accommodate parsers with different granularity in the output, the grammatical relations are classified hierarchically (Figure 4). Different types of parser output are represented with different types of grammatical relations and therefore a wide range of parsers can be easily compared.

Table 1 lists the grammatical relations used in the examples of this paper and the final implementation. For further detail about grammatical relations see (Briscoe and Carroll, 2000).

For example, the grammatical relations for the sentence *The man that came ate bananas and apples with a fork without asking* are:

```
DETMOD(_,man,the),
CMOD(that,man,come),
SUBJ(come,man,_),
SUBJ(eat,man,_),
DOBJ(eat,banana,_),
DOBJ(eat,apple,_)
CONJ(and,banana,apple),
NCMOD(fork,eat,with),
DETMOD(_,fork,a),
XCOMP(without,eat,ask)
```

Briscoe and Carroll's grammatical relations are different from the dependency arcs used in dependency grammar formalisms (Mel'čuk, 1988). Consider *The man that came ate bananas and apples with a fork.* Figure 5 (a) shows the graphical representation of the structure returned by Conexor FDG, a dependency-based parsing system (Tapanainen and Järvinen, 1997). For comparison, Figure 5 (b) shows a simplified graphical representation of the grammatical relations. In dependency grammar a unique head is assigned to each word, thus the head of *man* is *ate*. However *man* is the dependent of more than one grammatical relation, namely `SUBJ(eat,man,_)` and `SUBJ(come,man,_)`. Furthermore, in dependency grammar a word can have at most one dependent of each argument type, and so *ate* can have at most one object. But the same is not true for grammatical relations, and we get both `OBJ(eat,banana,_)` and `OBJ(eat,apple,_)`. Thus, grammatical relations can theoretically provide a sentence representation that is closer to the semantic contents of a sentence than the representation provided by dependency arcs. In practice, the representational power of the grammatical relations depends on the output of the parser used.

Grammatical relations can be used to introduce parser-independent syntactic information in a question-answering system. Since the grammatical relations are expressed as *lists of relations*, a score measure can be implemented by simply computing the *overlap* of grammatical relations between the question and the answer candidate. In theory, to compute the overlap we must use the hierarchical organisation of the grammatical relations to decide if two grammatical relations are related. For example, `SUBJ(eat,man,_)` can be subsumed by `SUB_OR_DOBJ(eat,man)`. However, since the same parser was used for both the question and the answer, the granularity of grammatical relations between questions and answer candidates will be practically the same. Thus, each grammatical relation can be seen as an unstructured token and the scoring module can simply count the number of common tokens, very much like counting the overlap of words. This was the approach used in our QA prototype.

4 Minimal Logical Forms

Flat logical forms have been used in several NLP systems, including question-answering systems (Harabagiu et al., 2001; Lin, 2001; Mollá et al., 2000, for example). The flat logical forms that we use in our QA system are borrowed from (Mollá et al., 2000), who uses reification to flatten out nested expressions. For example, the logical form of *The cp command will quickly copy files* is:[2]

[2] For illustration purposes, the logical forms used in this paper are slight variants of the ones shown in the literature.

Relation	*Description*
CONJ(type,head+)	Conjunction
MOD(type,head,dependent)	Modifier
CMOD(type,head,dependent)	Clausal modifier
NCMOD(type,head,dependent)	Non-clausal modifier
DETMOD(type,head,dependent)	Determiner
SUBJ(head,dependent,initial_gr)	Subject
OBJ(head,dependent,initial_gr)	Object
DOBJ(head,dependent,initial_gr)	Direct object
XCOMP(head,dependent)	Clausal complement without an overt subject

Table 1: Grammatical relations used in this paper.

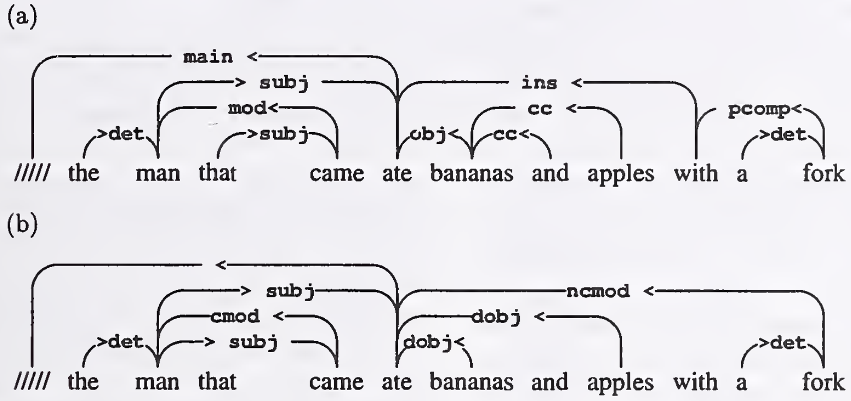

(a)
main <
> subj
mod<
>det
>subj
ins <
cc <
obj<
cc<
pcomp<
>det
///// the man that came ate bananas and apples with a fork
(b)
<
> subj
cmod <
>det
> subj
ncmod <
dobj <
dobj<
>det
///// the man that came ate bananas and apples with a fork

Figure 5: (a) Dependency structure of a sample sentence; (b) grammatical relations.

```
holds(e6),
object('cp',o2,[x2]),
object('command',o3,[x3]),
compound_noun(x2,x3),
prop('quickly',p5,[e6]),
evt('copy',e6,[x3,x7]),
object('file',o7,[x7])
```

The logical form above says that there are two entities x2 and x3 that represent two objects for the compound noun *cp command*. There is an entity x7 (a file); there is an entity e6, which represents a copying event where the first argument is x3 (the object introduced by the head of the compound noun) and the second argument is x7; there is an entity p5 which states that e6 is done quickly, and the event e6 (the copying) holds.

The above expression does not aim to express the complete logical form of the sentence. For example, there is no information about quantification, tense, aspect, and plurals. In essence, only the main relations among open words and determiners is expressed. This is why our logical forms are called *minimal logical forms*: only information that is minimal for the task of question-answering is encoded.

An advantage of the use of flat logical forms over nested logical forms is that, again, sentence similarity can be measured as a type of overlap. The only additional complexity is that the question now contains variables. For example, the minimal logical form of *Which command copies files?* is (the symbols in uppercase indicate variables):

```
object('command',O1,[X1]),
evt('copy',E2,[X1,X2]),
object('file',O2,[X2])
```

If this logical form is to match that of the sentence *The cp command will quickly copy files* above, the scoring module needs to instantiate the variable O1 in the question with the constant o3

Answer candidate	**Minimal Logical Form**
John saw Mary	object('john',o1,[x1]), object('mary',o3,[x3]), evt('see',e2,[x1,x3])
Question	**Minimal Logical Form**
Did John see Mary?	**object('mary',O,[X]), evt('see',E,[Y,X]), object('john',O2,[Y])**
Did Mary see John?	object('john',O,[X]), **evt('see',E,[Y,X])**, object('mary',O2,[Y])

Table 2: Question answering using flat logical forms. Overlap shown in bold.

in the answer candidate, `X1` with `x3`, and so on. In our implementation we have used Prolog unification. Basically, the logical form of the answer candidates is stored as Prolog data, and a simple Prolog program computes the overlap of the logical forms of the answer candidates with the logical form of the question. Also, the scoring module ignores the `holds` term in the question logical form. Otherwise, since almost all questions and candidate answers contain a `holds` term in the logical form, two completely unrelated sentences would have an overlap of 1 and this is counterintuitive.

Since there are several plausible combinations of variable instantiations, the scoring module finds the set of instantiations that provides the highest overlap of logical forms.

Table 2 shows the minimal logical forms of questions that differ solely in the argument positions, the minimal logical form of an answer candidate, and the resulting overlaps.

5 Results and Discussion

To test the impact of grammatical relations and minimal logical forms, we experimented with different scoring modules corresponding to word overlap, grammatical relations, and minimal logical forms, using the TREC 2002 questions. The results (Table 3) show that, remarkably, word overlap is best.

Formula	*Accuracy*
Word overlap	14.8%
Grammatical relations	09.0%
Minimal logical forms	10.8%

Table 3: Experiments with TREC 2002 data.

Due to the limited time available we decided to postpone any cause analysis. Instead we ran several experiments combining the linguistic information available. Table 4 shows the final runs submitted to TREC, the results of our experiments with data from TREC 2003, and the final results returned by NIST.

The data in the 2002 column show that the runs submitted produce slightly better results than simple word overlap. Interestingly, The evaluation provided by NIST (2003 column) gives noticeably better results than our home evaluation with the TREC 2002 data (2002 column). This may be due to the fact that the regular expressions provided by Litkowsky do not attempt to cover all possible formulations of correct answers, or it may be indeed the case that the questions asked in TREC 2003 are easier to process. As we will see in next section, the former is more likely.

6 Adding Named Entities

During the development of the system we tried to integrate the named entity recogniser bundled with GATE.[3]

First of all, we implemented a question analyser module that classifies the question into the type of expected answer. The classifier uses 29 regular expressions to allocate one of the following types to the question: person, date, location, money, number, city, date, organization, location, percent,country, state, river, name, and unknown. The regular expressions were based on the questions used in TREC 2002. The final module has an accuracy of 78.6% (393 from a total of 500 questions were correctly classified).

Since GATE's named entity recogniser can detect a reduced number of named entity types (person, location, date, money, and organization only), a simple mapping was necessary between the question classification and the final list of answer types (Table 5).

[3] `http://gate.ac.uk/`

Run	*Formula*	*2002*	*2003*
answfind1	$3wo + gro$	16.8%	19.1%
answfind2	$9wo + 3gro + mo$	16.8%	18.6%
answfind3	$9wo + 3mo + gro$	15.6%	18.2%

Table 4: Runs submitted to TREC 2003. The 2002 column indicates the results of an automatic self-evaluation with data from TREC 2002. The 2003 column indicates the evaluation results returned by NIST. The formula components are: wo – word overlap; gro – overlap of grammatical relations; mo – overlap of minimal logical forms.

Question Type	*Answer Type*
country, city, state, river	location
percent	number
name	person OR organization OR location
Any other question type yields same answer type	

Table 5: Mapping between question type and answer type.

2002	*Without NEs*	*With NEs*
answfind1	16.8%	19.1%
answfind2	16.8%	19.3%
answfind3	15.6%	18.4%
2003	*Without NEs*	*With NEs*
answfind1	18.2%	16.2%
answfind2	17.4%	15.7%
answfind3	17.2%	15.5%

Table 6: Results of integrating the named entity recogniser.

The information regarding answer type is used during the sentence preselection stage. Thus, the score given to a sentence is the sum of word overlap with the question (as described above) plus a reward of 10 points if the sentence contains an entity of the expected answer type.

We developed a Java interface to GATE's named entity recogniser. However, the steep learning curve required to learn Java, together with unexpected execution errors prevented us from integrating the NE module in the final version. The final system therefore did not use the named entity recogniser and as a consequence the question classifier became useless and therefore it was disabled. Subsequent work on the Java interface enabled us to compute the named entities of the top 50 documents preselected for the TREC 2002 and TREC 2003 questions. With these named entities computed off-line we ran AnswerFinder and obtained the results shown in Table 6.

The results show a noticeable improvement of accuracy in the runs with the TREC 2002 question set. In contrast, accuracy *decreases* in the runs with the TREC 2003 question set. A plausible explanation to the results with the TREC 2003 question set is that we used the new regular expressions provided by Litkowsky for the evaluation. The regular expressions are based on the set of answers returned by the systems competing in TREC 2003. Possibly, some of the answers returned by the version with named entities are paraphrases of the correct answer that do not match the regular expressions and therefore they are erroneously classified as wrong. In fact, note that the results without named entities reported in Table 6 are slightly worse than the ones returned by NIST (Table 4) due to inaccuracies in the regular expressions.

7 Conclusions and Further Research

The short time available only allowed us to build a baseline system for the passages task of the Question Answering track. Still, we were pleased to find that the results were better or equal than the median of all 21 submissions to the task. Overall, our experiments suggest that simple word overlap gives better performance than simple overlaps based on grammatical relations or minimal logical forms alone. These findings confirm the work by (Mollá, 2003), who used a similar question answering system for the Reading Comprehension corpus (Hirschman et al., 1999).

Further work will include:

Error analysis. This will be the first step to do. We will determine if different types of questions are more or less likely to produce good results with the different overlap measures and identify methods of combining word overlap, grammatical relations, and minimal logical forms for each type of question.

NE integration. We will finalise the integration of the named entity recogniser and identify entity types that are useful for the task of question answering. Besides using the named entities to determine the answer candidates, we will also explore ways to include NE information in the parsing modules and semantic interpreter. This way we hope to obtain more accurate grammatical relations and logical forms.

Logical forms. We will explore ways to leverage the use of logical forms by using more sophisticated measures. For example, we will look into adding weights to the logical forms. We will also explore the possibility of using weighted abduction methods.

Extract the exact answer. Logical forms may be useful to determine the exact answer of the question. For example, the original ExtrAns system (Mollá et al., 2000) generates a predicate of the form `object(_,_,_)` that represents the object asked about by the question word. ExtrAns also keeps track of what words produces what predicates in the minimal logical form. All this information can be used to determine the exact part of the sentence that matches the concept being asked about.

Complex questions. We will also work on methodologies to answer questions that require the fusion of output from several documents, such as list and definition questions.

By introducing the above and other extensions, in future participations in the question answering track we hope to increase the accuracy of the system and to participate in the main task of the track.

References

Ted Briscoe and John Carroll. 2000. Grammatical relation annotation. On-line document. `http://www.cogs.susx.ac.uk/lab/nlp/carroll/grdescription/index.html`.

John Carroll, Ted Briscoe, and Antonio Sanfilippo. 1998. Parser evaluation: a survey and a new proposal. In *Proc. LREC98*.

Sanda Harabagiu, Dan Moldovan, Marius Paşca, Mihai Surdeanu, Rada Mihalcea, Roxana Gîrju, Vasile Rus, Finley Lăcătuşu, and Răzvan Bunescu. 2001. Answering complex, list and context questions with LCC's question-answering server. In Ellen M. Voorhees and Donna K. Harman, editors, *Proc. TREC-10*, number 500-250 in NIST Special Publication. NIST.

Lynette Hirschman, Marc Light, Eric Breck, and John D. Burger. 1999. Deep Read: A reading comprehension system. In *Proc. ACL'99*. University of Maryland.

Jimmy J. Lin. 2001. Indexing and retrieving natural language using ternary expressions. Master's thesis, MIT.

Igor Mel'čuk. 1988. *Dependency Syntax: Theory and Practice*. State University of New York Press.

Diego Mollá, Rolf Schwitter, Michael Hess, and Rachel Fournier. 2000. Extrans, an answer extraction system. *Traitment Automatique des Langues*, 41(2):495–522.

Diego Mollá. 2003. Towards semantic-based overlap measures for question answering. In *Proc. ALTW 2003*, Melbourne, Australia.

Pasi Tapanainen and Timo Järvinen. 1997. A non-projective dependency parser. In *Proc. ANLP-97*. ACL.

Meiji University Web and Novelty Track Experiments at TREC 2003

Ryosuke Ohgaya, Akiyoshi Shimmura and Tomohiro Takagi
Department of Computer Science, Meiji University
{ohgaya, sinmura, takagi}@cs.meiji.ac.jp

Akiko Aizawa
National Institute of Informatics
akiko@nii.ac.jp

1. Introduction

This year we participated in TREC for the first time. We submitted runs for Novelty track and the topic distillation task of Web track.

2. Conceptual Fuzzy Sets

To represent the meaning of a word, we proposed conceptual fuzzy sets (CFS) [1][2]. In CFS, the meaning of a word is represented by the distribution of the activation of other words and dynamically changes reflecting context. The image of CFS is shown in Figure 1.

We used two different implementation of CFS in each track.

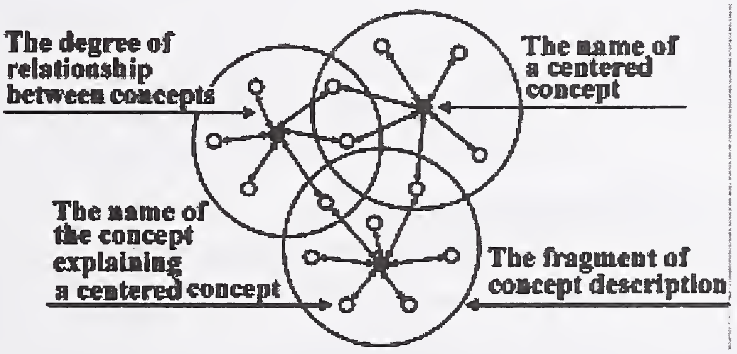

The degree of relationship between concepts
The name of a centered concept
The name of the concept explaining a centered concept
The fragment of concept description

Figure 1. Image of CFS

In Figure 1, white surrounding concepts explain the centered gray concept. The strength of the links between concepts reflects their degrees of relationship. The centered concept and its connected concepts constitute a fragment of concept description. A CFS is generated by overlapping the fragments of the activated concept description. A CFS expresses the meaning of a concept by the activation values of other concepts in these fragments.

3. Web Track

We submitted five runs for the topic distillation task. Our system is based on vector space model with tf-idf weighting. To create a document vector, we used the contents of a target page and those of its neighboring pages in the run *meijihil3*, *meijihil4* and *meijihil5*.

Searching procedure is:

1. Expand query using conceptual fuzzy sets (in *meijihil2*, *meijihil4* and *meijihil5*)
2. Calculate similarities
3. Rerank search results based on out-degree (in *meijihil5*)
4. Aggregate pages from the same server into one

Table 1 shows the description of each run.

Table 1. Evaluation results of submitted runs

Run	Query expansion	Inlinks & Outlinks	Reranking	R-Prec	P@10
meijihil1				0.0918	0.0920
meijihil2	O			0.0614	0.0700
meijihil3		O		0.0902	0.1060
meijihil4	O	O		0.0687	0.0700
meijihil5	O	O	O	0.0523	0.0620

3.1. Using the contents of inlinks and outlinks

In the World Wide Web, a web page and its neighboring pages are likely to be on the same topic. We evaluated whether incorporating the contents of neighboring pages in that of a target page improve search accuracy.

We create the document vector of a target page as follows:

1. Create the word vector of each page using only its contents with tf-idf weighting.
2. Aggregate the word vector of the target page and those of its neighboring pages:

$$V_i = \alpha V_{d_i} + \beta \sum_j V_{d_j} + \gamma \sum_k V_{d_k}$$

where V_{d_i} is the word vector of the target page, V_{d_j} is the word vector of a page that is linking to the target page and V_{d_k} is the word vector of a page that is linked to by the target page.

In the run *meijihilw3*, *meijihilw4* and *meijihilw5*, we set α, β and γ to be 1.0.

3.2. Query expansion using conceptual fuzzy sets

We used CFS to expand queries. To construct CFS, we need a dictionary in which the meanings of words are represented by other words and their degree of relationship.

3.2.1. Dictionary for conceptual fuzzy sets

To create the dictionary, we used a method proposed in [3] in which overlapping clusters of terms are generated based on co-occurrence (Actually, documents and other related information are also clustered simultaneously with terms, but we used only term clusters for the dictionary). A term cluster is composed of a representative word and related words with their degrees of relationship and is considered as a word vector that represents the concept of the word. We refer to this word vector as concept vector.

3.2.2. Expansion procedure

The similarities between the input vector and each concept vector are calculated using cosine measure:

$$S_i = \frac{V_q \cdot V_{C_i}}{|V_q||V_{C_i}|}$$

where V_q is the input vector and V_{C_i} is the i th concept vector.

The expanded query vector is the weighted sum of the concept vectors:

$$V_q' = \sum_i S_i \cdot V_{C_i}$$

3.3. Similarity calculation

We used cosine measure to calculate the similarity between input vector and each document vector. Document structure and proximity of query terms are also used: a document gets an additional score if the query terms appeared in title (<title>) or headings (<h*n*>) field or if the query terms appeared closely in the document.

3.4. Out-degree reranking

A key resource is expected to have links to many relevant pages. Thus we reranked initial search results based on out-degree as follows:

$$Sim'_{d_i} = Sim_{d_i} + \alpha \frac{1}{n} \sum_{j}^{n} Sim_{d_j}$$

where Sim_{d_i} is the initial score of the document d_i, Sim_{d_j} is the initial score of the document d_j that is pointed to by d_i and n is the number of outlinks in d_i. This technique is used in the run *meijihilw5* and α is set to be 1.0.

3.5. Site aggregation

Initial search results often give higher rank to pages from the same server. We simply merged them into one that has the shortest URL.

3.6. Results

Results are shown in Table 1. Query expansion and reranking failed to improve R-precision and P@10. Incorporating the contents of neighboring pages on the other hand showed some improvements.

4. Novelty Track

In Novelty Track, our main challenge is conceptual expansion of profiles and sentences. Expanding them using CFS can calculate similarities more correctly than only using word frequency.

We regarded sentences as very short documents, and converted them to word vectors. In the conversion phase, we removed stop words, stemmed words using Porter's algorithm and assigned weights to them using tf-idf.

4.1. Conceptual Expansion

We constructed the network shown in Figure 2 to implement CFS.

Concept vector C_i(fragment of the concept description) is created by clustering documents in Reuters corpus. The weights between concept layer and output layer are also trained using Reuters corpus.

An input is expanded as follows:

1. Calculate similarities between input vector X and each concept vector C_i:

$$S_i = \frac{X \cdot C_i}{|X||C_i|}$$

2. Expanded vector Y is calculated by propagating the similarities:

$$Y_j = \sum_i (a_{ij} \times S_i)$$

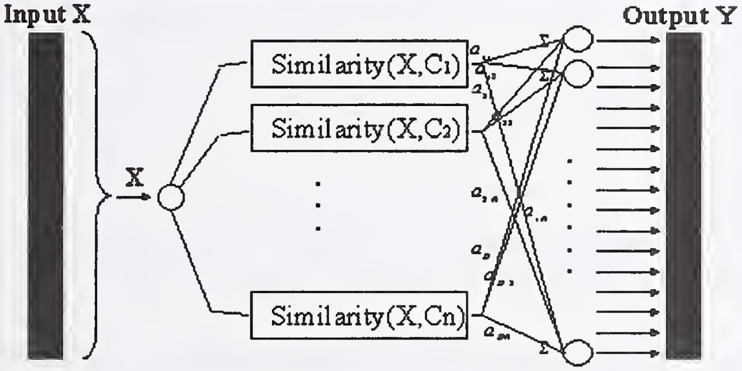


Figure 2. Network structure for CFS

4.2. Relevant Sentence

4.2.1. Relevancy Detection System Description

To identify relevant sentences, we used an information-filtering-based approach. Initial profiles, which are made with the topic descriptions, are expanded conceptually. If the cosine similarity between an

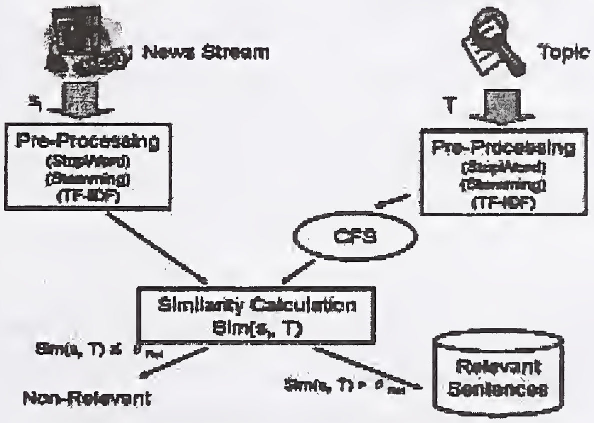


Figure 3. Architecture of relevancy detection system

expanded profile and the word vector of a sentence exceeds a threshold, the sentence is regarded as relevant. Figure 3 shows the architecture of our relevancy detection system. The title, description and narrative field were used to adjust profiles. Only the topic profile was expanded.

4.2.2. Threshold Learning

In this system, we must set an appropriate threshold to distinguish Relevant sentences from Non-Relevant ones. The threshold was trained by using the corpus of TREC2002 Novelty Track (min_qrels.relevant and max_qrels.relevant). We adopted the threshold where the F measure was maximized.

The number of New sentences in a Relevant sentence set decreases inevitably as the recall becomes low. Therefore, the threshold where the recall is 0.7 was also used.

4.2.3. System Variation

We had three system variations to identify Relevant sentences as shown in Table 2. The profiles were expanded by CFS in R1 and R2, but were not expanded in R3 to compare accuracy with R1 and R2.

Table 2. Relevancy detection system variation

	CFS Expansion	Threshold Learning
R1	O	Maximum F-Measure
R2	O	Recall=0.7
R3	X	Maximum F-Measure

4.3. New Sentence

4.3.1. Novelty Detection System Description

To identify new sentences, we used two measures: sentence score and redundancy score. 1) For calculating sentence score, we used N-window-idf to consider the time window. Local sentence score is calculated by using document frequency for the past N documents. 2) Redundancy score of a sentence is the maximum similarity between the sentence and ones judged to be new in the past. Figure 4 shows the architecture of our novelty detection system.

4.3.1.1. Sentence Score

The sentence score is calculated based on sentence weight proposed by Zechner [4]. We improved it so that it might take novel feature. If news documents are streaming in chronological order, they have the feature that a specific topic concentrates in a small range. Therefore, in order to judge novelty, it is effective not to consider globally, but to consider locally. We used local rarity of a word to use this feature. It is calculable using N-window-idf which is document frequency in past N documents. By using N-window-idf, weights of frequent words decrease and sentence weights represent local information.

$$SentenceScore(s) = \sum_{i} tf(t_i) \times N - window - idf(t_i)$$

$$N - window - idf(t) = \log \frac{N}{N - window - df(t)}$$

where tf(t_i) is the frequency of the word t_i in the sentence s, N is the window size, and N-window-df(t) is the document frequency of the word t in past N documents.

4.3.1.2. Redundancy Score

To calculate the redundancy score, we used maximum similarity of sentences which are already identified as novelty. The similarity is calculated by cosine measure.

$$RedundancyScore(s) = \underset{NovSi \in NoveltySentences}{Max} Similarity(NovSi, s)$$

4.3.1.3. Novelty Score

We used the sentence score and the redundancy score to identify the novelty. We thought that novelty sentences must have higher information weight and differ from pre-selected novelty sentences. Therefore, we combined these scores:

$$NoveltyScore(s) = \lambda \times SentenceScore(s) - (1 - \lambda) \times RedundancyScore(s)$$

If the NoveltyScore exceeds a threshold, the sentence is regarded as novelty.

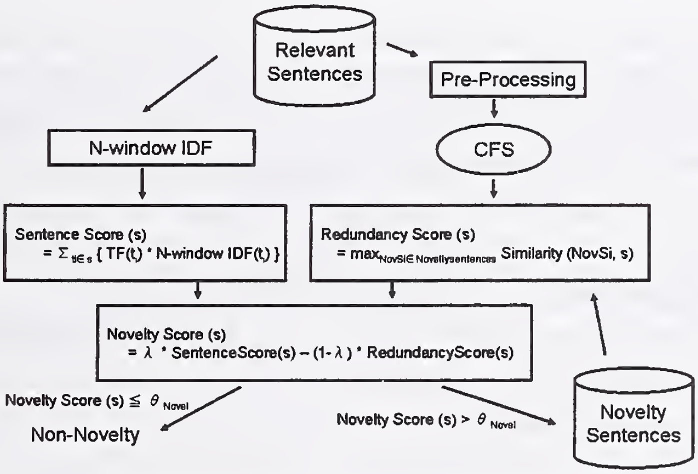

Relevant Sentences
Pre-Processing
N-window IDF
CFS
Sentence Score (s) = Σ ti∈s { TF(ti) * N-window IDF(ti) }
Redundancy Score (s) = max NovSi∈ Noveltysentences Similarity (NovSi, s)
Novelty Score (s) = λ * SentenceScore(s) – (1- λ) * RedundancyScore(s)
Novelty Score (s) ≦ θ Novel
Non-Novelty
Novelty Score (s) > θ Novel
Novelty Sentences

Figure 4. Architecture of novelty detection system

4.3.2. Parameter Setting

To identify new sentences, we had to set up three parameters:

1) window size: N
2) ratio of sentence score to redundancy score: λ
3) threshold for judging whether the input sentence is New or not: θ

We set the widow size to 200 based on the number of sentences to a news document. λ and θ were determined by learning using Trec2002 Novelty Track data (min_qrels.new, max_qrels.new) as well as Relevancy Detection System. We adopted λ and θ from which F measure becomes the maximum.

4.3.3. System Variation

Four variations were prepared (Table 3).

Table 3. Novelty detection system variation

	N-window-idf	CFS Expansion
N1	O	O
N2	O	X
N3	O	-
N4	X	-

In N-window-idf column, [O] means N-Window-idf is used to calculate sentence scores and [X] means basic idf is used instead of N-Window-idf. The df values of basic idf were calculated using about 810,000 news documents in Reuters corpus. In Expand column, [O] means CFS expansion is used to calculate redundancy scores, [X] means expansion is not used and [-] means redundancy scores are not used.

4.4. Result and Discussion

We submitted for Task 1-4. Table 4-5 shows the results. In the Relevancy Detection phase, the validity of expansion by CFS has been shown. Moreover, we presented the validity of N-window-idf, which considered locality, in the Novelty phase.

Table 4. Result of Task 1 and Task 3

				Relevant			New		
				Recall	Precision	F-Measure	Recall	Precision	F-Measure
Task1	MeijiHIIF11	R1	N1	0.64	0.63	[redacted]	0.10	0.52	0.151
	MeijiHIIF12		N2				0.13	0.51	0.176
	MeijiHIIF13	R2	N1	0.84	0.52	[redacted]	0.16	0.60	0.199
	MeijiHIIF14		N2				0.22	0.49	0.280
	MeijiHIIF15	R3	N2	0.55	0.57	0.496	0.22	0.56	0.270
Task3	MeijiHIIF31	R1	N1	0.58	0.54	0.540	0.26	0.44	0.310
	MeijiHIIF32		N2				0.25	0.46	0.301
	MeijiHIIF33	R2	N1	0.49	0.54	0.495	0.26	0.46	0.310
	MeijiHIIF34		N2				0.27	0.47	0.320

Table 5. Result of Task 2 and Task 4

			New		
			Recall	Precision	F-measure
Task2	MeijiHIIF21	N1	0.77	0.66	0.708
	MeijiHIIF22	N2	0.76	0.69	0.713
	MeijiHIIF23	N3	0.99	0.65	[redacted]
	MeijiHIIF24	N4	0.96	0.65	0.765
Task4	MeijiHIIF41	N1	0.73	0.65	0.672
	MeijiHIIF42	N2	0.72	0.66	0.675
	MeijiHIIF43	N3	0.98	0.62	[redacted]
	MeijiHIIF44	N4	0.96	0.49	0.634

5. References

[1] T. Takagi, A. Imura, H. Ushida, and T. Yamaguchi, "Conceptual Fuzzy Sets as a Meaning Representation and their Inductive Construction", *International Journal of Intelligent Systems*, Vol.10, pp.929-945, 1995.

[2] T. Takagi, A. Imura, H. Ushida, and T. Yamaguchi, "Multilayered Reasoning by Means of Conceptual Fuzzy Sets", *International Journal of Intelligent Systems*, Vol.11, pp.97-111, 1996.

[3] A. Aizawa, "A Method of Cluster-Based Indexing of Textual Data", *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*, pp.1-7, 2002.

[4] K. Zechner, "Fast Generation of Abstracts from General Domain Text Corpora by Extracting Relevant Sentences", *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, pp.986-989, 1996.

MICROSOFT RESEARCH ASIA AT THE WEB TRACK OF TREC 2003

Ji-Rong Wen, Ruihua Song, Deng Cai, Kaihua Zhu, Shipeng Yu, Shaozhi Ye and Wei-Ying Ma

Microsoft Research Asia
5f, Beijing Sigma Center
No.49, Zhichun Road Haidian District
Beijing 100080, P. R. China

INTRODUCTION

This is the first year that our group participates in the Web track of the TREC conference. Here we report our system and methods on the topic distillation task and the home/named page finding task.

All of our experiments are conducted on a Web search platform we designed and developed from scratch. We originally want to use an existing retrieval system such as Okapi or the full text search mechanism of SQL Server. But we soon find the limitations of such a strategy – these systems cannot fully support some important Web search functions such as link analysis and anchor text, and they also lack of the flexibility to arbitrarily adjust some parameters or add new ranking functions. So we decided to design and implement a research platform to let researchers to test various algorithms or new ideas easily and, also, to conduct the TREC experiments easily. We will introduce the framework of this system in the "Platform" section.

We feel that this year's topic distillation is more close to the real Web search scenarios. The target is to find a list of key resources for a particular (broad) topic and "key resources" are defined as the entry pages of websites. So, different from the previous years, we think that link analysis may play a positive role on identifying key resources in this year. As a consequence, we focus on using different link analysis techniques to enhance the relevance ranking. In particularly, we propose a novel block-based HITS algorithm to solve the noisy link and topic drifting problems of the classic HITS algorithm. The basic idea is to segment each Web page into multiple semantic blocks using a vision-based page segmentation algorithm we developed before. Then the main steps of the HITS algorithms, such as getting the seeds, expanding the neighbors using inlinks and outlinks, and calculating hub/authority values, can be performed at the block level instead of at the page level. Thus the noisy link and topic drifting problems can be effectively overcome. We will detail these techniques in the "Page Layout Analysis" and "Block-based HITS" section.

To our understanding, the biggest difference of this year's topic distillation task from last year is that, in general, only one most "suitable" page for each website should be returned as a top-ranked result. Any other page at the same website should not be included in the results or ranked highly since it is a "part of a larger site also principally devoted to the topic", despite that the page also "is principally devoted to the topic". Therefore, we construct a hierarchical site map for each website by building up the parent-children relationships of Web pages in the .GOV dataset. Then we apply a site compression technique to select the most suitable entry pages for websites among the retrieval results and return these entry pages as top-ranked pages. This site compression method has proved to be quite effective to increase the p@10 precision if used appropriately, and will be introduced in the "Site Compression" section.

We totally submitted 5 runs for the topic distillation task and 3 runs for the home/named page finding task. In the "Experimental results" section, we will introduce these runs and their evaluation results.

PLATFORM

We built a Web search platform from scratch since we found traditional IR-like platforms cannot meet the requirements of comprehensive Web search algorithms. The architecture of the platform is shown at Figure 1.

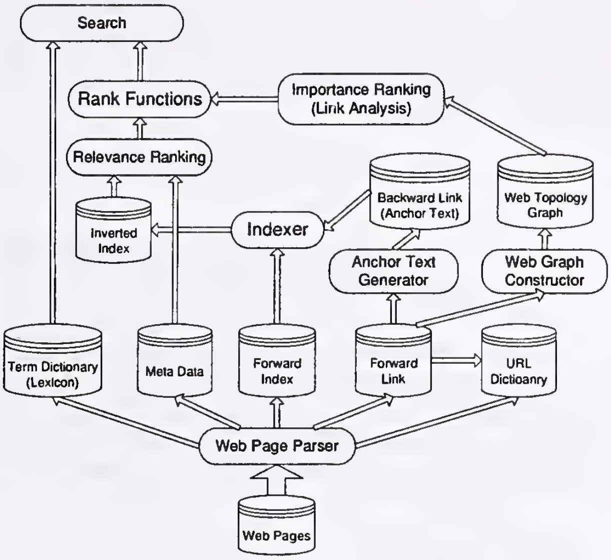


Figure 1. Architecture of our Web search platform

WEB PAGE PARSER

We built a robust Web page parser to extract the following information from each Web page:

1. Term hits – the type, format, position of each occurrence of each term in the page. Each term is assigned a unique term ID and a lexicon is built to store the term-ID mapping information during the parsing process. Similar to Google, we classify terms into five types:

   - Title: words in <title>…</title>
   - Meta: keywords extracted form <meta keyword="…">
   - URL: words that occur in hyperlinks
   - Anchor text: words in anchor texts from other pages and can only be obtained through a post-processing of all anchor texts

✧ Plain text: all the rest content words, and they are further divided into 6 categories:

- H1_2: words in H1 and H2 tags
- H3_6: words in H3 to H6 tags
- STRONG: words with font type "bold", "italic", "underlined", etc.
- Large: words with large font size
- Medium: words with medium font size
- Small: words with small font size

All of the above information are extracted and put into a storage called "forward index".

2. URL and anchor text – all hyperlinks and their corresponding anchor texts are extracted and stored in a "forward link" storage. Also, each URL is assigned a unique ID and a URL dictionary is built. Two important functions rely on the forward link storage. One function is to add anchor texts to pages which the hyperlinks point to. Another is to construct Web graph and facilitate link analysis such as PageRank and HITS.

3. Meta data – some important meta data about each page, such as size, date and layout structure (will introduce later) are recorded in this storage.

INVERTED INDEX

There is a significant difference of the structure of our inverted index from general ones. Since we use an algorithm to segment web pages into semantic blocks, we add a block-level structure into the inverted index, as illustrated by Figure 2. Through this structure, we can identify each term hit at each block of each Web page. This structure is very critical to support our block-level search methods. Also, this structure occupies nearly the same (or less if we omit hits) storage space as page-level index and is compatible with page-level index.

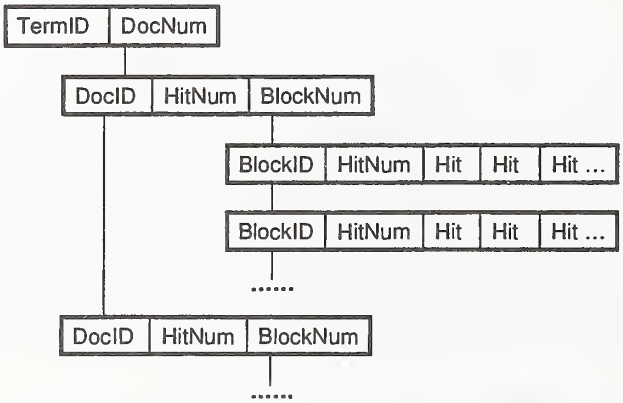


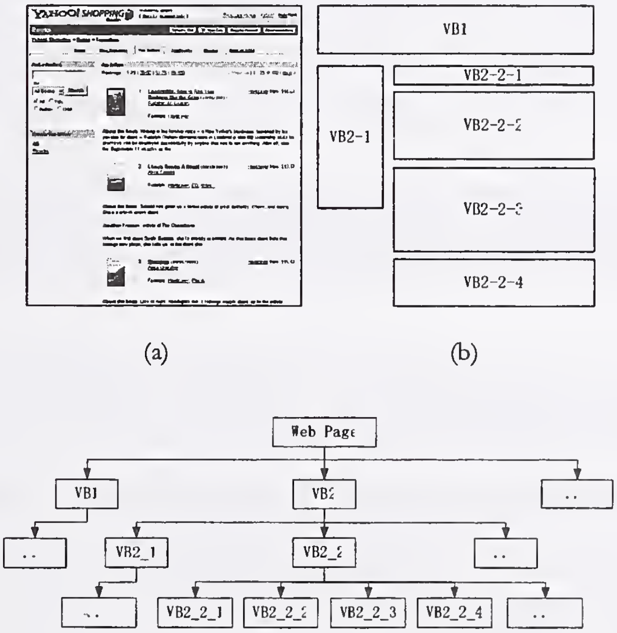
Figure 2. Structure of the inverted index

RANK FUNCTION

We use Okapi's BM2500 as our fundamental relevance ranking function. Considering the characters of Web search, we made some important modifications and augmentations to BM2500. First, it is allowed to set different weights to different term types and formats. For example, we can assign high weight to terms in titles of pages if we find title is more important for relevance ranking. Second, we use term proximity to adjust the relevance scores since it is observed that the distributions of query terms in a page significantly affect the relevance judgment.

PAGE LAYOUT ANALYSIS

As mentioned before, one distinguished feature of our platform is that each Web page is segmented into multiple semantic blocks. We think that a web page as a whole is not a good information unit for search because it often contains multiple topics and a lot of irrelevant information from navigation, decoration, and interaction parts of the page. We use a VIsion-based Page Segmentation (VIPS) algorithm to detect the semantic content structure in a web page based on visual cues such as color, line, font size, image, etc. For the sample page shown in Figure 3(a), the visual blocks are detected are shown in Figure 3(b) and the global layout structure is shown in Figure 3(c).



(a) (b)

(c)

Figure 3. Vision-based layout structure for the sample page

VIPS can help to remove noisy information within a web page and detect multiple topics, and therefore it can be very beneficial to web information retrieval. In [8], VIPS is used in pseudo-relevance feedback to improve the quality of top ranked documents or blocks. About 27% improvement is achieved and it also wins over the DOM-based approach.

BLOCK-BASED HITS

Noisy link and topic drifting are two main problems in the classic HITS algorithm. Some links such as banners, navigation panels, and advertisements, can be viewed as "noise" with respect to the query topic. Generally, noisy links do not carry human editorial endorsement, which is a basic assumption in topic distillation. Also, hubs may be "mixed", which means that only a portion of the hub content may be relevant to the query. Most link analysis algorithms treat each Web page as an atomic, indivisible unit with no internal structure. This leads to false reinforcements of hub/authority calculation.

By segmenting a Web page into separate semantic blocks, we implement a modified HITS algorithm at block level. We hope to verify that page segmentation is an effective way to overcome the noisy link and topic drifting problems of HITS, to some extent. Below are the main steps of the block-based HITS algorithm.

Step 1: A *start set* of pages matching the query is fetched by a block retrieval algorithm (described below), and the top 200 pages are used.

Step 2: The start set is augmented by its neighbors, with no limitation on inlinks. When the neighborhood graph is expanded using outlinks, only those outlinks in blocks with high block ranks are used to expand the *neighborhood set.*

Step 3: Prune the neighborhood blocks by a query expansion algorithm (also described below). First, we use the query expansion algorithm to get a list of blocks with relevance scores. Then these blocks are intersected with the blocks in the *neighborhood set.* The blocks in the intersection set are labeled using the scores of the query expansion step. Finally, we use the median value as a threshold to prune the nodes whose weights are below this threshold. The resulting set is used as a new neighborhood set.

Step 4: In the neighborhood set, if there are k edges from pages on a first website to a single page on a second website, we assign each edge an authority value of $1/k$. This weight is used when computing the authority score of the page on the second website. If there are t edges from a single page on a first website to a set of pages on a second website, we assign each edge a hub weight value of $1/t$. Finally we remove isolated nodes from the graph.

Step 5: Only each root block (i.e. the whole page) can be assigned an authority score. Every leaf block has its hub score. So we build up a $m \times n$ matrix in which m stands for the number of blocks in the neighborhood set and n stands for the number of pages in the neighborhood set.

Generally, the calculating results of our block-based HITS algorithm have a very "sharp" distribution – generally only the first 15~30 authority or hub values are greater than zero. And all of the remaining pages have a zero value. Therefore, only those most densely linked collection of hubs and authorities in the neighborhood graph can be detected and distinguished by the algorithm. But this is not a serious problem for us since P@10 is used as the evaluation measure this year. So such a value distribution is sufficient to provide the differentiated capability for the top 10 results.

BLOCK RETRIEVAL

Similar to passage retrieval, block retrieval performs the retrieval task on the block level and aims to adjust the rank of documents with the blocks they contain. The block retrieval algorithm contains the following steps:

Step 1. Initial Retrieval: An initial list of ranked web pages is obtained by using the general page level retrieval method and a page-level rank called *PR* is obtained.

Step 2. Page Segmentation: In this step, the page segmentation algorithm VIPS is applied to partition all of the retrieved pages into blocks. All the extracted blocks form a block set.

Step 3. Block Retrieval: This step is similar to Step 1 except that pages are replaced by blocks. The same queries are used to produce a block-level rank called *BR* for each block.

For each page, the block with the highest *BR* rank is selected and its rank is called *BRMax*. In our experiments, a combination of *PR* and *BRMax*, formatted as $\alpha \cdot rank_{PR}(d) + \beta \cdot rank_{PR\_BRMax}(d)$, is used as the final rank of each page.

QUERY EXPANSION

In our experiments, we use pseudo-relevance feedback as a basic query expansion method. Its basic idea is to extract expansion terms from the top-ranked pages to formulate a new query for a second round retrieval. The effect of query expansion method strongly relies on the quality of selected expansion terms. Since our VIPS algorithm can group semantically related content into a single block, the term correlations within a block will be much higher than those within a whole page. With the improved term correlations, high-quality expansion terms can be extracted from blocks and then used to improve retrieval performance. The query expansion algorithm contains the following steps:

Step 1 – Step 3 are the same as those of block retrieval. We get a page rank *PR* for each page and a block rank *BR* for each block after these steps.

Step 4. Expansion Term Selection: Top blocks are used for expansion term selection. We use an approach similar to the traditional pseudo-relevance feedback algorithm to select expansion terms. All terms except the original query terms in the selected blocks are weighted according to the following term selection value *TSV*:

$$TSV = w^{(1)} * r / R$$

where $w^{(1)}$ is Robertson/Sparck Jones weight [7]. *R* is the number of selected blocks, and *r* is the number of blocks which contain this term. In our experiments, top 10 terms are selected to expand the original query.

Step 5. Final Retrieval: The weights for the expanded query terms are set as the following:

- For original query terms, new weight is $qtf \cdot 2$ where *qtf* is its term frequency in the query;

- For expansion query terms, new weight is $1-(n-1)/m$ if the current term ranks nth in *TSV* rank. *m* is the number of expansion terms and is set to 10 in our experiments.

Then the expanded query is used to retrieve the data set again for the final results.

SITE COMPRESSION

It is a normal phenomenon that multiple Web pages from the same sites are ranked highly for a given query. This is not a problem if the objective is to find relevant pages. But this year's topic distillation task targets to find a list of key resources for a particular topic. Key resources are defined as the entry pages for websites and other pages should be discarded. So we should try to find as many different websites (represented by their entry pages) as possible within the first ten results. So finding a method to detect the entry page for each website is very important.

We construct a hierarchical site map for each website by building up the parent-children relationships of Web pages in the .GOV dataset. Then we apply a site compression technique to select the most suitable entry pages for websites at the retrieval results and return these entry pages as top-ranked pages. Figure 4 is a sample site map of the "fitness.gov" website.

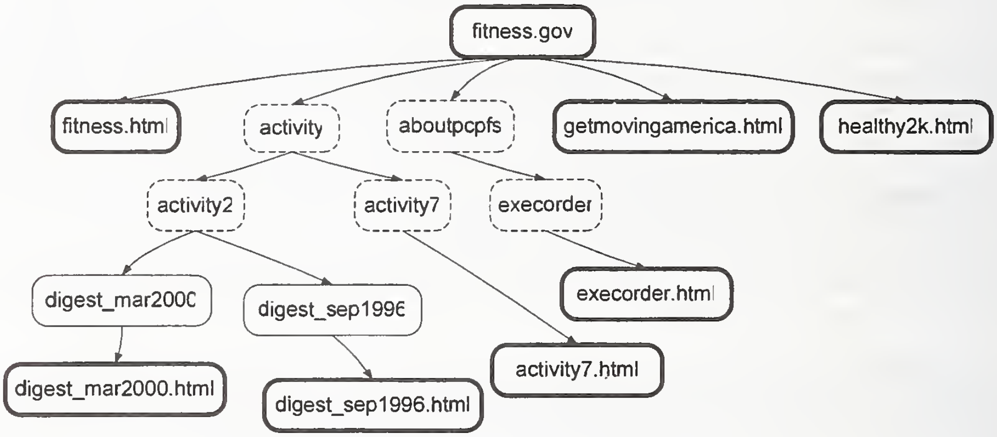


Figure 4. Site map

The solid boxes represent a page appearing in the search results and the dashed boxes represent virtual directories which do not appear in the results. Take the topic of "physical fitness" as an example, below are the pages and their ranks from the "fitness.gov" website among the top 1000 results.

2. http://fitness.gov/activity/activity2/digest_mar2000/digest_mar2000.html

3. http://fitness.gov/aboutpcpfs/execorder/execorder.html

16. http://fitness.gov/

21. http://fitness.gov/getmovingamerica.html

23. http://fitness.gov/healthy2k.html

……

To judge if a parent page (solid box) can represent all of its children results, we check the following two conditions:

- If the parent page has more than three child pages (with solid box) in the top 1000 results
- If the parent page has more solid children than dashed children

If any of the two conditions is met, the children can be represented by their parent page, and the rank of the parent page is assigned with the maximum of its rank and its children's ranks. The above rules are applied to each site maps in the results recursively from bottom up. This site compression method proved quite effective in terms of increasing P@10 if used appropriately.

EXPERIMENTS

We totally submitted 5 runs for the topic distillation task and 3 runs for the home/named page finding task. Below is the list of the runs:

- MSRA1001 – The augmented BM2500 (by adding term weights and term proximity) is used as the rank function. Site compression is used to post-process the ranking results and only one page from each website is kept in the top 10 results.
- MSRA1002 – Similar with MSRA1001 only except that the site compression step allows at most 2 pages from each website are kept in the top 10 results. Notice that we do not use any link analysis technique in MSRA1001 and MSRA1002. We want to use these two runs as our baseline to compare them with the runs with link analysis.
- MSRA3 – MSRA1001 is first used to get a result list. Then the importance value calculated by PageRank is used to re-rank the results.
- MSRA4002 – MSRA1002 is first used to get a result list. Then the authority value calculated by our block-based HITS algorithm is used to re-rank the results. Notice that the site compression step in MSRA1002 is executed after the HITS step.
- MSRA4003 – MSRA1001 is first used to get a result list. Then the authority value calculated by our block-based HITS algorithm is used to re-rank the results. Also, the site compression step in MSRA1001 is executed after the HITS step.

  The above 5 runs are related to the topic distillation task.
- MSRANP1-3 – These are the 3 runs related to home/named page finding task. The rank function is similar with MSRA 1001, except that no site compression is used and term weight settings are different.

TOPIC DISTILLATION

Table 1 shows results of our five topic distillation runs and the techniques used in each run are listed in Table 2.

We found that term weight settings are important for relevance ranking. We use a greedy algorithm to automatically learn weights for different term types and format by using the data of TREC'02. In our experiments, PageRank do not show significant improvement on P@10. But we found that the pages returned are perceived well when browsing. Therefore, we have no conclusion on whether PageRank is useful based on the experiments. Block-based HITS shows a steady improvement on retrieval performance. The two runs containing block-based HITS, MSRA4002 and MSRA4003, got the best two p@10 and average precision. We also find that the performance of block-based HITS is significantly better that PageRank at the .GOV dataset, which further affirms that PageRank is not suitable to be used in a relatively small or moderate dataset.

Run	Precision@10	Average precision	R-Precision
MSRA1001	0.0960	0.0699	0.1027
MSRA1002	0.1100	0.0824	0.1078
MSRA3	0.1040	0.0933	0.1016
MSRA4002	0.1160	0.1027	0.1354
MSRA4003	0.1140	0.0946	0.1052

Table 1: Topic distillation results of 5 runs

Run	Term Weight	Term Proximity	PageRank	Block-based HITS	Site Compression*
MSRA1001	Y	Y			Y (1)
MSRA1002	Y	Y			Y (2)
MSRA3	Y	Y	Y		Y (2)
MSRA4002	Y	Y		Y	Y (2)
MSRA4003	Y	Y		Y	Y (1)

Table 2: Techniques used in topic distillation runs

* Y(1) means that a run uses site compression and keeps only 1 page from a site in the top 10 results, while Y(2) means that a run uses site compression and keeps at most 2 pages from a site in the top 10 results.

Site compression is proved to be quit effective in terms of increasing P@10 (Figure 3). However, to retain only one page for each website may be risky since the site compression method cannot identify the entry pages with 100% accuracy. So we can see that MSRA1001 only achieve a slight increase of P@10 in comparison with the baseline (without site compression). But if we adapt a conservative strategy to retain at most two pages for each site, a significant increase of P@10 of MSRA1002 is shown.

Run	Precision@10	Average precision	R-Precision
Baseline (without site compression)	0.0940	0.0966	0.0995
MSRA1001	0.0960	0.0699	0.1027
MSRA1002	0.1100	0.0824	0.1078

Table 3: Effect of site compression

Finally, run MSRA4002, which combines augmented BM2500, block-based HITS and conservative site compression, achieve the best performance on P@10, average precision and R-Precision.

NAMED PAGE FINDING

Run	Average Reciprocal Precision	Named pages in top 10	Named pages not found	Features

MSRANP1	0.651	253 (84.3%)	27 (9.0%)	Anchor_weight = 3
				Title_weight = 1.1
				Other weights = 1

MSRANP2	0.540	214 (71.3%)	56 (18.7%)	Linear combined with term proximity
MSRANP3	0.556	218 (72.7%)	51 (17.0%)	Anchor, title and url only

Table 4: Named page finding results and features

For the named page finding task, we mainly focus on setting proper weights for different term types and the weights are learned based on the data of TREC'02 named page finding task. We discovered that anchor plays the most important role in this task as its weight is highest, and title is the second one. Run MSRANP1 is the baseline with term weight settings as shown in Table 4. MSRANP2 is a run by combining MSRANP1 and term proximity linearly. But its average reciprocal precision decreases quite a few. In run MSRANP3, only anchor, title and url are used in the rank function and the result indicates that it lost 8% more named pages than the run uses all fields.

REFERENCES

1. Bharat, K. and Henzinger, M., Improved Algorithms for Topic Distillation in a Hyperlinked Environment, Proceedings of 21st ACM International Conference on Research and Development in Information Retrieval, 1998
2. Brin, S. and Page, L., The Anatomy of a Large-Scale Hypertextual Web Search Engine, In the Seventh International World Wide Web Conference, Brisbane,Australia, 1998.
3. Callan, J. P., Passage-Level Evidence in Document Retrieval, In Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Dublin, 1994, pp. 302-310.
4. Chakrabarti, S., Joshi, M., and Tawde, V., Enhanced topic distillation using text, markup tags, and hyperlinks, In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval , ACM Press, 2001, pp. 208-216.
5. Kleinberg, J., Authoritative sources in a hyperlinked environment, In Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms, 1998, pp. 668-677.
6. Robertson, S. E. and Walker, S., Okapi/Keenbow at TREC-8, In The Eighth Text REtrieval Conference (TREC 8), 1999, pp. 151-162.
7. Robertson, S. E. and Sparck Jones, K., Relevance weighting of search terms, *Journal of the American Society of Information Science*, Vol. 27, No. May-June, 1976, pp. 129-146.
8. Yu, S., Cai, D., Wen, J.-R., and Ma, W.-Y., Improving Pseudo-Relevance Feedback in Web Information Retrieval Using Web Page Segmentation, In Proceedings of the Twelfth International World Wide Web Conference (WWW2003), May 2003.

Microsoft Cambridge at TREC–12: HARD track

S E Robertson[*] H Zaragoza[†] M Taylor[‡]

1 Summary

We took part in the HARD track, with an active learning method to choose which document snippets to show the user for relevance feedback (compared to baseline feedback using snippets from the top-ranked documents). The active learning method is described, and some prior experiments with the Reuters collection are summarised. We also invited user feedback on phrases chosen from the top retrieved documents, and made some use of the 'relt' relevant texts provided as part of the metadata. Unfortunately, our results on the HARD task were not good: in most runs, feedback hurt performance, and the active learning feedback hurt more than the baseline feedback. The only runs that improved slightly on the no-feedback runs were a couple of baseline feedback runs.

2 Overview

The present team at Microsoft Cambridge may be regarded as the descendant of the Okapi team, working first from City University London and then from Microsoft. A summary of the contributions to TRECs 1–7 is presented in [4]. In these TRECs on various adhoc tasks we had concentrated on the weighting schemes and pseudo relevance feedback (blind feedback), and had developed the successful BM25 weighting function. However, we also took part in most of the early interactive tracks, and also developed iterative relevance feedback strategies for the routing task. Following up on the routing work, in TRECs 7–11 we took part principally in the adaptive filtering track (summarised in [6]). This work included developing alternative feature selection strategies, and also extensive analysis of thresholding; one outcome of the latter was a method of calibrating the BM25 score into an estimate of the probability of relevance.

For this year's TREC, we have entered only the HARD track. We have concentrated on the use of the clarification forms (one-shot interaction with the originator of the topic).

Since moving to Microsoft we have been working in part with a successor to Okapi, the Keenbow evaluation environment. The work reported in this paper was undertaken entirely with this new system, which is described in outline below.

3 System

Keenbow is built in part using components from the MSSearch system, used in various Microsoft products including the SharePoint Portal Server. Although MSSearch maintains its own index of a traditional inverted file type, Keenbow can work with collection indexes stored as SQL tables; the distinction is largely a matter of performance (efficiency). That is, for large collections/indexes, it may be necessary for performance reasons at search time to use the native inverted file indexing system, while for smaller collections everything can be done within SQL. Clearly 'large' and 'small' are relative to the current hardware and low-level system state-of-the-art. In practice, all the experiments described here came into the 'small' category, and were run using Keenbow on a Microsoft SQL Server, running on an Intel Quad 700MHz Xeon with 3GB RAM.

The basic ranking algorithm in Keenbow is the usual Okapi BM25. The collection was preprocessed in a standard manner, using a 126 stop-word list and the Porter stemmer. In the context of query expansion (from relevance or blind feedback), feature selection is again based on usual Okapi methods – normally, the absolute term selection criterion described in [5]. As before, relevance feedback involves selecting a small number of terms from the known relevant documents, and weighting all selected terms (including the original topic terms) by the usual BM25 methods.

Currently Keenbow indexes predefined passages (we have not yet implemented in Keenbow the arbitrary window retrieval that we had in Okapi). For these experiments we defined passages at a level which comes somewhere in between paragraph and sentence – in other words, documents are broken into non-overlapping passages, each consisting of one or a few sentences.

4 HARD

The particular aspect of the HARD track which appealed to us was the opportunity to invoke a user-interaction phase actually involving the assessor who originated the topic. This is clearly highly artificial if we want to see it as a simulation of a genuine interactive system; however, it is the first time in TREC that we have had the opportunity to interact with the assessors, and it provides scope for some interesting experiments on what kinds of information might be elicited from users, and to what effect they might be put. We were less concerned with the metadata aspect of the track: we made minimal use of metadata.

[*] Microsoft Research Ltd, 7 J.J.Thomson Avenue, Cambridge CB3 0FB, UK, and City University, London, UK. email ser@microsoft.com

[†] Microsoft Research Ltd, 7 J.J.Thomson Avenue, Cambridge CB3 0FB, UK. email hugoz@microsoft.com

[‡] Microsoft Research Ltd, 7 J.J.Thomson Avenue, Cambridge CB3 0FB, UK. email mitaylor@microsoft.com

An outline of the system is as follows. We put the original topic to the system in the usual fashion, and obtain the top-ranked retrieved documents. From these we select some to show to the user/assessor. The baseline system shows the top five documents, but the major experimental version shows five selected from the top 30 according to an active learning principle, as discussed below. What we show the user in each case is a short passage extracted from the document in a query-specific fashion: a query-specific snippet. In addition, we show the user some (max 15) 2-word phrases selected from the snippets according to a statistical measure, again described below. We invite the users to make 'relevance' judgements on each snippet and on each phrase (the form of the question is discussed below). The clarification form submitted to the user is made up out of these snippets and phrases.

On receipt of the completed clarification forms, we have made various runs using various parts of the returned information in different ways. We also make limited use of some of the metadata. Some of these runs were submitted as our official returns, and others have been evaluated since.

4.1 Basic methods

Okapi BM25 is used with the following parameters: $k_1 = 0.4$; $b = 0.75$; $k_3 = 0$ (the last means that duplicate query terms were ignored).

This procedure is used with the feedback obtained from the clarification forms, as discussed below. It is also used in the active learning stage, when we hypothesise various combinations of relevance judgements which the assessor might make on the documents presented.

Essentially the procedure is as described in many previous TREC reports and elsewhere. Each relevant document (or piece of text) is parsed to extract all terms as indexed. A table of statistics for the complete merged set of terms (including all original topic terms) is generated, a term selection value is calculated, and the top terms according to this value are selected for inclusion in the query. The various parameters for this process are as follows:

- Term selection function: Absolute function described in [5].
- Threshold for term selection: -8.
- Treatment of original topic terms: forced inclusion.
- Weighting after selection: original topic terms were given a boost in the expanded query by assuming that they occur in *rload* out of *Rload* mythical relevant documents, to be added to the r and R respectively concerning the actual relevant items. These parameters were set to *Rload* = 20 and *rload* = 19.

5 Active learning: document selection

In the usual *probabilistic learning* setting we are tying to estimate some function $f(x)$ from a collection of values $(x_i, f(x_i))$ (the training sample). However, in the *active learning* setting [3, 2] there is no pre-existing training sample, but rather we get to ask or *query* the function $f(x)$ with our chosen x values. In general, rather than querying points at random, it is much more advantageous to query points for which i) our incertitude is greatest and ii) obtaining an answer to our query will change our present model of the function the most. Active learning algorithms are used to choose these values in some optimal manner, exploiting properties of the function f to obtain the most information in the least number of points.

This problem is reminiscent of our problem in HARD. Here we wish to learn the probability of relevance of a document with respect to a query, $P(r = 1|d, q)$, were $r \in \{0, 1\}$ is a relevance indicator function, and d and q are the indexed documents. From a probabilistic learning perspective we would then need a collection of data points (q, d, r). We cannot do probabilistic learning as such yet, since we do not have any such data points[1], but perhaps we could *query* the judge for such values. We would need to present the judge with the query and a carefully selected batch of documents and ask him to reveal if the documents are or are not relevant. We would then use this information to i) update our approximation of the function of interest $P(r|d, q)$ and ii) select the next batch of points.

However, two things stand in the way of such an approach. The first problem is that active learning algorithms are tailored to each learning algorithm. Probabilistic active learning exploits properties of the learned function [3, 2]; in particular one needs to compute analytically how the introduction of a data point will change the approximation of the function. But in the case of Okapi feedback this is *a priori* unknown (as described above).

The second is that we only get a single chance to ask the judge! So iterative procedures are out of the question. All we can do is exploit the knowledge available in the query to form our initial approximation of $P(r = 1|d, q)$ and then select a batch of documents to be used as queries.

So in fact active learning will be used weakly, more as an inspiration than as a rigorous application of its principles. For this reason we call the resulting feedback algorithm *active feedback*.

5.1 Algorithm

We assume that we have the following:

- a query q,
- an indexed document collection $D := \{d_i\}_{i=1..N}$
- a retrieval function $\pi(d, q, F)$ which scores a document d for a given query q and a given *feedback set* of documents $F \subset D$.

[1] In fact the extra documents (metadata items with the tag *relt*) provided by the judges could be considered as such data-points, but unfortunately these were not available at form-generation time.

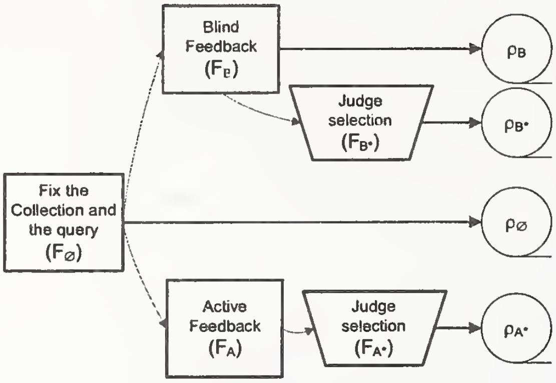


Figure 1: *Boxes and trapezes indicate automatic and manual procedures respectively. Feedback set names are indicated in parenthesis. Circles indicate the resulting collection rankings. Initially, a document collection and a query are fixed. The feedback set is empty and the resulting rank $\rho_\emptyset$ is the usual ad-hoc rank. By selecting first documents of this rank as the feedback set (F_B) we obtain the usual blind-feedback ranking ρ_B. If the judge is presented the documents in F_B she will select the relevant ones only (F_{B*}) obtaining an improved blind-feedback ranking ρ_{B*}. Finally, using the proposed active feedback procedure, we present the set F_A to the judge, who selects the relevant documents only (F_{A*}). This results in the ranking ρ_{A*}.*

We will assume that the function π models the probability of relevance of the document, given the query, the collection statistics and the feedback set F: $\pi(d,q,F) \approx P(r = 1|d,q,F,D)$. We do not assume any further knowledge about the retrieval function. In particular, we do not assume known the way in which the feedback set F modifies the query or the scoring function.

This *black-box approach* has the advantage of remaining quite general; in particular we will exploit the Okapi feedback framework which we know to be discontinuous with respect to the function parameters (and so difficult to analyze *inside the box*). However, the cost of this generality will be high: later, we will only obtain a heuristic algorithm instead of the usual provably optimal active learning algorithms.

Now we note that after fixing D, π and q, each feedback subset $F \subset D$ will implicitly define an *ordering* $\rho_F = (d_{(1)}, ..., d_{(N)})$ of the documents on the collection, where $d_{(i)}$ is the document with rank i under $\pi(d,q,F)$.

Initially no human judgements are known and so $F = \emptyset$. This results in the baseline ad-hoc ranking of the collection, $\rho_\emptyset$. The usual *blind feedback* method sets F to the k highest scored documents. Let us denote this set $F_B := \{d_{(i)} | i \leq k\}$ for some value of k. The resulting collection ranking is denoted ρ_B (see Figure 1).

In the HARD setting, however, we can ask the user the relevance of a number of documents before defining the feedback set. Specifically, we could ask the user to verify the documents in the set F_B and select the relevant ones, forming the new set F_{B*}. This would result in the new (and hopefully improved) ranking ρ_{B*}.

We consider this "verified blind-feedback" method to be our HARD baseline. We believe we can improve on this selection because we feel that the top scoring documents, while they are the most *likely relevant* documents, are the *least informative* relevant documents, for two reasons: i) they will probably be very alike (the top documents are likely to be very redundant) and ii) the relevance of the documents is well explained by the query already.

Introducing the human judge as a filter has a crucial effect: we do not need to fear introducing irrelevant documents, since the human judge will eliminate them before retrieval! If we call F_A the set automatically selected for feedback, then the set used for feedback will be the human-verified set $F_{A*} \subseteq F_A$.

This allows us to look for more exotic documents that may be false but, *if they were relevant*, would carry a lot of new information on the query. Of course, since the judges have little time, we need to be slightly conservative or we risk not using any relevant documents (e.g. $F_{A*} = \emptyset$ if no relevant documents are selected in F_A) .

Therefore, we will argue that we need to detect not only the most *relevant* documents but also the most *informative* ones, or in other words, the ones that would produce the bigest change (or update) in the retrieval function if they were relevant. Unfortunately we do not have a good way to define the information gained by the introduction of a particular document in F. This is because we are considering a general retrieval function (a black-box), and therefore cannot analyse the effect of a relevant document in the function itself. All we can observe is the output of the black-box: the change induced in the ranking of the collection, ρ_F. For this reason we define the following function of the difference between two orderings:

$$\delta(\rho_A - \rho_B) := \frac{3}{\pi^2} \sum_{l=1}^{|\rho_A|} \left(\frac{1}{\rho_{l,A}} - \frac{1}{\rho_{l,B}} \right)$$

where $\rho_{l,A}$ indicates the lth coordinate of the vector ρ_A, and the constant $3/\pi^2$ is a normalisation factor which keeps δ in the [0,1] range. This function is chosen on the basis of the following criteria:

- it should have the value 0 if the two rankings are identical, and approach 1 if they are very different;
- it should depend more on the top end of the ranking than on items further down.

This second point is achieved by using the reciprocal ranks instead of the ranks themselves (in the same way that known-item search tasks are often evaluated using mean reciprocal rank). Note that it does *not* behave like the inverse of a rank correlation coefficient, specifically in that it has no notion of a reverse correlation.

We can finally state our objective: we need to choose the set F_{A*} (of some fixed size k) that maximises the quantity $\delta(\rho_\emptyset, \rho_{A*})$.

Unfortunately, we do not know which documents will be chosen by the judge, and so we do not have access to F_{A*}. So we will revert to maximising the *expected* change of rank over all possible judge selections (weighted by the probability that these selections are relevant under π). For this, let us denote by $\mathcal{F}$ the power set[2] of F and by $\mathcal{F}_k$ the set of all subsets of F with size k or less including $\emptyset$. With this, we can define the *expectation* of δ over the set of documents F under $\pi(d, q, \emptyset)$ as:

$$\mathsf{E}\left[\delta_F\right] := \sum_{F' \in \mathcal{F}} \left[\delta\left(\rho_\emptyset, \rho_{F'}\right) \prod_{d_i \in F'} \pi\left(d_i, q, \emptyset\right)\right]$$

Finally, we define the *active feedback set* F_A as the subset of the collection D of some fixed size k which maximises $\mathsf{E}\left[\delta_{F_A}\right]$:

$$F_A = \arg\max_{F \in \mathcal{D}_k} \mathsf{E}\left[\delta_F\right]$$

In practice the size of $\mathcal{D}$ is too large to exactly compute 5.1. But we notice that for most documents their probability of relevance is so low that they would bring to zero any expectation in which they are considered as candidates. Therefore it is safe to consider only the most relevant documents as candidates. We do this simply by considering only the documents in D with highest $\pi(d, q, \emptyset)$ values. For our the HARD 2003 runs we considered only the top 30 documents. The calculation of $\delta\left(\rho_A - \rho_B\right)$ is based on comparing the rankings of the top 500 documents (as indicated, it is most strongly affected by changes at the top of the ranked list).

6 Phrase selection

The two-word phrases to be shown to the user in the clarification forms were selected as follows:

We considered each pair of adjacent words in every snippet shown to the user. For each such pair, we calculated the following *plausibility* measure (originally used in [1]): If s and t are two terms with frequencies $n(s)$ and $n(t)$ respectively the plausibility of the adjacent pair st is $n(st) \times C/(n(s)n(t))$, where C is the total number of tokens in the collection. For randomly collocated terms we would expect this measure to be around 1; we set a high threshold on it to select words which are collocated considerably more often than that. The selection threshold chosen was 20. We also chose phrases with a reasonable frequency of occurrence ($n(st) > 10$). Finally, we calculated the offer weight or term selection value, on the blind feedback assumption that the snippets chosen are all relevant (this is of course before we have user judgements). Thus the complete criterion was:

- Select phrases with plausibility$>$ 20;
- From these, select those with $n(st) > 10$;
- Sort these by term selection value;
- Accept the top 15, or those with term selection value> 3, whichever is the less.

The resulting phrases mostly looked like reasonable phrases; some not. An example list from Topic HARD-033 is: *antimicrobial drug; APHIS regulations; hog cholera; intestinal tract; contagious disease; Endangered Wildlife; Nacional de; golden eagle; occurring outside; animal drugs; drug resistant; animal product; Shanxi Province; draft guidance; wild animals.*

7 Clarification forms

7.1 Retrieved items

As indicated, our principal aim was to obtain relevance feedback data from the assessors. However, given the various limitations (screen real estate and time taken to complete) on the clarification forms, it was not feasible to present the assessors with anything like complete documents. In a reasonable compromise between document numbers and amount of information per document, we decided to present up to four lines from each of up to five documents.

At the time of indexing, each document is partitioned into predefined, non-overlapping passages. Each passage is a single sentence or a small number of contiguous sentences. We therefore presented the best-matching passage from each of the selected documents in the form. In cases where the selected passage was too long, it was arbitrarily truncated. Most passages presented would include at least some of the query terms, but some would not, because of this arbitrary truncation. We considered including the complete passage in a small scrollable window in these cases, but rejected this idea, both for technical reasons (the version of Netscape being used by the assessors) and because it seemed counter to the principle of a restricted clarification form.

The issue of what question to ask the assessors about each document was an interesting one. Perhaps unlike many users of IR systems, they can be expected to have a rather clear idea about what 'relevant' might mean, given that they either have already made, or will in the near future be making, official TREC relevance judgements. On the other hand, the official judgements they will be making will be on the basis of reading (or at least being able to read) the entire document being judged. It seems a little hard to ask them to make an equivalent judgement on the basis of the snippet presented.

One of our interests is in the use of indirect evidence such as click-through as a form of feedback. We therefore decided to present the relevance question to the assessors as a click-through question:

[2] that is the set of all subsets of F including $\emptyset$. It is usually noted F^* but this clashes with our notation.

Assume that you have issued a query on the above topic to your search engine, which has responded with the following list.
Would you click through to any of these documents?
Check as many or as few as you like.
* If you can answer your question from the snippet alone, please check "No need".

The radio buttons beside each item were:

- Yes
- Perhaps
- No
- No need *

The default button was 'No'. The 'Perhaps' was included primarily for the comfort of the assessors who might find it difficult to make a definite answer in some cases, but allows us to try the relevance feedback with or without the *Perhaps* responses included as relevant. The 'No need' button was included on the basis that some of the questions could be answered with a sentence or phrase which might actually be in the snippet. These were counted as relevant (although in such cases relevance feedback seems a bit superfluous).

The responses were coded 3 (No need), 2 (Yes), 1 (Perhaps), 0 (No) for the experiments discussed below.

7.2 Phrases

Phrases were selected from the snippets chosen for the documents shown to the assessor (but before truncation). Up to 15 were selected. The question asked was:

Do any of the following phrases help to describe what you are looking for? Check as many or as few as you like.
* If you think a phrase is indicative of a document you do not want to see, please check "Neg".

The radio buttons for each phrase were:

- Yes
- No
- Neg*

The default button was 'No'. The 'Neg' (negative) button was included on the grounds that 'No' was neutral (*No* phrases would simply be ignored), but some phrases seem to indicate an incorrect context, and might therefore be treated in a more strongly negative fashion, as providing positive evidence *against* the relevance of the document. This was quite a popular button among the assessors, but raises interesting questions of how the negative evidence should be used, discussed further below.

These responses were coded 1 (Yes), 0 (No), -1 (Neg) for the experiments discussed below.

There was no necessary reason to choose the phrases from the chosen snippets – we could have chosen them from the (whole) chosen documents, or from some other set of documents. The data we have collected from the experiment allows us to simulate two more possibilities, by using the phrases selected for the baseline run with the snippets selected for the main experimental run, and vice versa.

8 Use of feedback data and metadata

When we have received the assessors' responses to the clarification forms, we have various forms of data that might be used in various ways and in various combinations in feedback. We have tried a few of these combinations as officially submitted runs, and some additional combinations are also evaluated in this paper.

8.1 Evaluated snippets and *relt* items

Snippets evaluated as relevant (in the click-through sense) are to be used for relevance feedback. In common with most other relevance feedback experiments, we make no use of items judged not relevant – they are simply ignored (instead, statistics from the whole collection, excluding those documents known to be relevant, are taken to represent the non-relevance class). Furthermore, we use the items judged relevant only in the usual relevance feedback algorithm: although it is likely that these items rise in the ranking as a result of the feedback, there is no necessary reason why they should rise to the top, and we do not force them to do so.

In the present circumstances, there is a choice between taking as the texts of the relevant items just the snippets judged relevant by the assessors, or the entire documents from which they come. We have chosen to take just the snippets themselves, on the grounds that those are the items of text actually judged (but in the cases where the snippet was truncated for display, we take the entire snippet). It may be argued that this approach does not fit very well with the theory on which the relevance feedback algorithm is based, which involves counting documents containing each term. This is an issue for further work.

One of the metadata items to which we now have access is the 'relt' item – that is, any texts provided as relevant by the assessor in advance of the search. One issue associated with these *relt* items, interacting with the issue just mentioned, is their length – they are typically quite long, certainly much longer than our snippets, and probably comparable in length to the documents. In the experiments where we have included the *relt* items, we have treated them in the same way as the relevant snippets. However, it seems likely that some differentiation should be made.

8.2 Positive phrases

It would be possible to treat any phrase as if it were a (new) single term, and give it a weight on the same basis that a term would be weighted. However, this ignores the fact that the phrase may contain terms that are themselves in the query. In this case, the danger is that a document will be overweighted because it gets the weight of the phrase and also the weight of the single term contained in the phrase. To put it another way, the probabilistic model makes independence assumptions, but in this case we have an extreme dependence situation: the presence of the phrase implies the presence of any constituent single term.

Since the constituent terms may or may not be in the query, we have a set of cases to deal with. Also, a phrase has a 'natural' weight of its own (the usual RSJ weight which is the document-independent part of the BM25 formula, which reduces to a tf*idf weight in the absence of relevance information but is a relevance weight when we have such information). This 'natural' weight may or may not exceed the combined weights of the constituent terms.

Thus our algorithm looks like this. We consider only 2-term phrases ab, and w(x) is the natural weight of x, which can be single term or phrase. wPhrase will be the weight to be given to the phrase.

```
wPhrase $\leftarrow$ w(ab)
IF (a $\in$ query) THEN wPhrase $\leftarrow$ (wPhrase - w(a)) ENDIF
IF (b $\in$ query) THEN wPhrase $\leftarrow$ (wPhrase - w(b)) ENDIF
IF (wPhrase $< 0$) THEN wPhrase $\leftarrow 0$ ENDIF
```

8.3 Negative phrases

Negative phrases present some of the same problems as positive ones – namely, any of the constituent terms may or may not be in the query. In addition, there is another general problem about using negative weights. The probabilistic theory that is the basis for BM25 is quite at home with negatively-weighted terms – essentially any term whose presence in a document is evidence against relevance – but for several practical reasons, negative weights have been avoided in almost all work with BM25. The normalisation of BM25 is designed to ensure that an absent term contributes nothing to a document's score, which means that documents containing none of the query terms (usually the vast majority of documents) have zero score. This is a big advantage in a system based on inverted files. Furthermore, if the query contains only positively weighted terms, then this large set of zero-scored documents is necessarily at the bottom of the ranking. Thus a ranking of all the non-zero (and therefore positive) scores implies in a very straightforward way a ranking of the complete collection (and of course no user ever ventures into the large mass of zero-scored documents tied at bottom rank). The usual term selection algorithms that form part of relevance feedback tend to select only positively weighted terms.

Introducing negative term weights potentially complicates this picture. In practice, however, small negative weights for a small number of terms may be accommodated (we would presumably only ever look at documents with resulting positive score, and ignore not only the zeros but also the net negatively scored documents).

In the light of these considerations, the proposed treatment of *Neg* phrases is as follows. The principle is that if either (or both) of the constituent terms is in the query, occurrences of that term in the document *as a constituent of the phrase* should be ignored (that is, should not contribute to the s, but other occurrences of the term on its own should continue to count positively. There is a slightly complex interaction here with the tf factor which is the other bit of BM25, and the proposed algorithm does not deal very elegantly with this interaction, but may serve as a first approximation. In addition, the presence of the phrase in a document should somewhat reduce the score of the document. The 'natural' (quite likely positive) weight of the phrase does not figure in this algorithm; however, we begin by assigning the basic amount by which the phrase should reduce the score. This might be a small positive constant, or perhaps half the average weight of the single query terms, or the weight of the least-weighted single query term. Then we consider the cases.

```
define small wDown $> 0$
wPhrase $\leftarrow$ - wDown
IF a $\in$ query THEN wPhrase $\leftarrow$ (wPhrase - w(a)) ENDIF
IF b $\in$ query THEN wPhrase $\leftarrow$ (wPhrase - w(b)) ENDIF
```

There is clearly scope for many experiments here. In the event, because of the generally negative results from the other experiments discussed (and our efforts to understand them), we have not yet conducted any experiments on these negative phrases.

8.4 Topic description and metadata

As a guiding principle, we tried to limit the amount of information required *a priori* form the user. To this end, we used only the *Title* of the topic description (discarding the topic's description and narrative) and discarded most of the topic's metadata. The two exceptions were:

GRANULARITY If the value was SENTENCE or PHRASE, we returned the best-matching passage as the passage-definition in the retrieved document (after ranking the documents by the usual document score).

RELATED-TEXT We used these texts in the same way that we used fragments returned in the clarification forms as relevant (see experiments below).

9 Experiments

9.1 Preliminary experiments

Before deciding on the methods to be used for HARD, we made a series of runs based on the active learning idea with the Reuters RCV1 corpus (as used in recent years for the adaptive filtering track), with the topics generated for last year's filtering track. We did not have the possibility of interaction with the assessors in this case, so the experiments simulated user feedback (or rather an upper bound) by assuming that the user would recognise as relevant the chosen snippet from a document that was officially judged as relevant.

In other respects these experiments were similar to those conducted for HARD – that is, for the active learning proce-

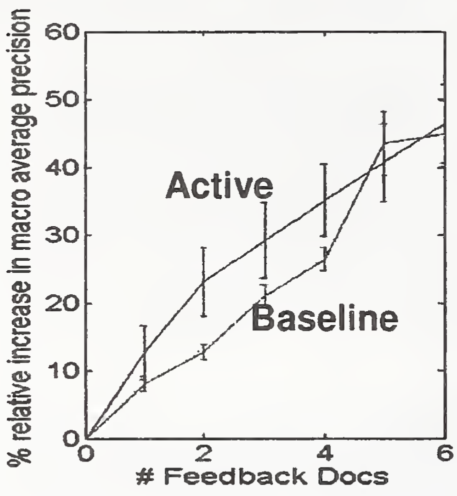


Figure 2: *Results on Reuters RCV1 corpus, TREC 2002 filtering topics. Performance is shown after feedback on 1–6 documents. In the case of the Baseline, these are the top-ranked documents; in the case of active learning, they are those selected from the top 30 by the active learning algorithm.*

dure we chose the best snippets according to the above algorithm, without reference to relevance judgements. Having chosen the snippets, we looked up their relevance judgements in lieu of actually consulting a user, and used the snippets from relevant documents to expand the query. The baseline here was to choose the top-ranked documents to provide the snippets and the relevance judgements.

These experiments and results are not described in detail here, but Figure 2 shows some results. Both active learning and baseline feedback improve on the baseline without feedback. On the whole the active learning procedure does better than the baseline feedback with few judged documents; this advantage may have disappeared by the time five documents have been used for feedback. Nevertheless, the results from these experiments were sufficiently encouraging for us to adopt the active learning method in our HARD experiments.

9.2 Variables and runs

The initial run, submitted before the clarification forms, is called MSRCbase. This is a straight BM25 baseline run on the topic titles only, and was the basis for the construction of the clarification forms.

[Actually, we believe that the run we submitted as baseline run was not the correct one. The submitted run was somewhat better than the 'real' baseline. The results reported below include the correct baseline run. They do not, however, change the generally negative results of this paper.]

As indicated above, we submitted two sets of clarification forms, one based on snippets from the top 5 ranked documents on the baseline run, and the other on the items selected by the active feedback analysis described above. (However, we attempted to remove duplicates from the baseline run snippets, after selection of the top 5, so that we often presented less than 5 snippets. The active feedback algorithm could be expected to remove duplicates anyway.)

Thus we had the following main variables to experiment on:

- use of the snippets in relevance feedback;
- use of the relt texts from the metadata in a similar fashion;
- and use of the phrases.

Our official runs were coded MSRCsXeXpX and MSRCsXeXpXB where the Xs are defined below and the B indicates use of the baseline clarification forms (rather than the Active Feedback ones). The use of snippets is coded s1, s2 or s9 – s9 means no snippets were used in feedback, s2 means that only the 'Yes' and 'No need' snippets were used (referred to below as *best* snippets), and s1 means that the 'Perhaps' snippets were also used (referred to as *good* snippets). e1 means the extended (relt) texts from the metadata were used, e0 that they were not. p1 indicates that the positive phrases were used, p0 that they were not (the negative phrases were not used in the official runs). We submitted these runs:

Run	CFs	Snippets	relt texts	phrases
MSRCs1e1p1	AF	good	yes	positive
MSRCs1e0p1	AF	good	no	positive
MSRCs1e0p0	AF	good	no	no
MSRCs9e1p1	AF	none	yes	positive
MSRCs2e0p1	AF	best	no	positive
MSRCs9e1p0	none	none	yes	no
MSRCs1e1p1B	base	good	yes	positive
MSRCs1e1p0B	base	good	yes	positive
MSRCs1e0p0B	base	good	yes	positive

We have since completed additional runs with other combinations of these variables.

9.3 Results

Unfortunately, our results have been almost exclusively negative. That is, we failed to improve significantly on the baseline with any of our methods; most of them degraded performance. Furthermore the active learning methods degraded performance more than the baseline feedback runs. The main results are in Table 1. The only run that outperforms the baseline uses the top 5 best snippets only, no phrases or relt texts.

We wished to test the hypothesis that the difference from our earlier Reuters experiments had to do with the fact that we used official relevance judgements in the Reuters experiments. We therefore made some runs on the HARD topics based on the selected snippets, but looking up official relevance judgements rather than using the feedback provided to the clarification forms. However, although this gave slightly better performance than our official runs, we still do not get anything like the increases observed in the Reuters experiments (see Table 2).

Table 1: Main results

Run	MAP	P@10	Notes
[MSRCbase]	.285	.496	Our corrected version, not as submitted
MSRCs1e0p0	.239	.467	Feedback from active learning snippets
MSRCs1e0p1	.215	.421	– plus phrases
MSRCs1e1p1	.255	.488	– plus relt texts
MSRCs1e1p0*	.251	.454	– relt texts but no phrases
MSRCs1e0p0B	.282	.490	Feedback from top 5 snippets
MSRCs1e0p1B*	.251	.446	– plus phrases
MSRCs1e1p1B	.277	.492	– plus relt texts
MSRCs1e1p0B	.291	.494	– relt texts but no phrases
MSRCs2e0p0*	.259	.488	Active learning best snippets only
MSRCs2e0p0B*	.297	.504	Top 5 best snippets only
MSRCs9e1p0	.251	.452	Relt texts only, no feedback
Note: The results here differ slightly from the official ones. This is probably due to a small difference in our method of calculation of the measures from trec_eval. We will be attempting to locate and remove this difference.			
Note 2: Runs marked * are additional to the official runs.			

Table 2: Feedback using official relevance judgements

Run	MAP	P@10	Notes
MSRCs1e0p0-R	.265	.488	Active learning snippets, official rels
MSRCs1e0p0B-R	.273	.485	Top 5 snippets, official rels

10 Conclusions

We are obviously disappointed at the results obtained. They suggest that our basic feedback methods are fragile with regard to some or all of the following: the collection, the nature of the documents, the use of snippets for feedback, the topics... Given that feedback on the top five documents (baseline feedback) hurts us, it is perhaps not surprising that active learning feedback hurts us more. We have some serious work to do!

References

[1] M M Beaulieu et al. Okapi at TREC–5. In E M Voorhees and D K Harman, editors, *The Fifth Text REtrieval Conference (TREC–5)*, pages 143–165. Gaithersburg, MD: NIST, 1997. NIST Special Publication 500-238.

[2] David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. Active learning with statistical models. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 705–712. The MIT Press, 1995.

[3] D. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, 1992.

[4] S E Robertson and S Walker. Okapi/Keenbow at TREC–8. In E M Voorhees and D K Harman, editors, *The Eighth Text REtrieval Conference (TREC–8)*, pages 151–162. Gaithersburg, MD: NIST, 2000. NIST Special Publication 500-246.

[5] S E Robertson and S Walker. Microsoft Cambridge at TREC-9: Filtering track. In E M Voorhees and D K Harman, editors, *The Ninth Text REtrieval Conference (TREC–9)*, pages 361–368. Gaithersburg, MD: NIST, 2001. NIST Special Publication 500-249.

[6] S E Robertson, S Walker, H Zaragoza, and R Herbrich. Microsoft Cambridge at TREC 2002: Filtering track. In E M Voorhees and D K Harman, editors, *The Eleventh Text REtrieval Conference, TREC 2002*, pages 439–446. Gaithersburg, MD: NIST, 2003. NIST Special Publication 500-251.

Integrating Web-based and Corpus-based Techniques for Question Answering

Boris Katz, Jimmy Lin, Daniel Loreto, Wesley Hildebrandt,
Matthew Bilotti, Sue Felshin, Aaron Fernandes, Gregory Marton, Federico Mora
MIT Computer Science and Artificial Intelligence Laboratory
Cambridge, MA 02139

1 Introduction

MIT CSAIL's entry in this year's TREC Question Answering track focused on integrating Web-based techniques with more traditional strategies based on document retrieval and named-entity detection. We believe that achieving high performance in the question answering task requires a combination of multiple strategies designed to capitalize on different characteristics of various resources.

The system we deployed for the TREC evaluation last year relied exclusively on the World Wide Web to answer factoid questions (Lin et al., 2002). The advantages that the Web offers are well known and have been exploited by previous systems (Brill et al., 2001; Clarke et al., 2001; Dumais et al., 2002). The immense amount of freely available unstructured text provides data redundancy, which can be leveraged with simple pattern matching techniques involving the expected answer formulations. In many ways, we can utilize huge quantities of data to overcome many thorny problems in natural language processing such as lexical ambiguity and paraphrases. Furthermore, Web search engines such as Google provide a convenient front-end for accessing and filtering enormous amounts of Web data. We have identified this class of techniques as the *knowledge mining* approach to question answering (Lin and Katz, 2003).

In addition to viewing the Web as a repository of unstructured documents, we can also take advantage of structured and semistructured sources available on the Web using *knowledge annotation* techniques (Katz, 1997; Lin and Katz, 2003). Through empirical analysis of real world natural language questions, we have noticed that large classes of commonly occurring queries can be parameterized and captured using a simple object–property–value data model (Katz et al., 2002). Furthermore, such a data model is easy to impose on Web resources through a framework of wrapper scripts. These techniques allow our system to view the Web as if it were a "virtual database" and use knowledge contained therein to answer user questions.

While the Web is undeniably a useful resource for question answering, it is not without drawbacks. Useful knowledge on the Web is often drowned out by the sheer amount of irrelevant material, and statistical techniques are often insufficient to separate right answers from wrong ones. Overcoming these obstacles will require addressing many outstanding issues in computational linguistics: anaphora resolution, paraphrase normalization, temporal reference calculation, and lexical disambiguation, just to name a few. Furthermore, the setup of the TREC evaluations necessitates an extra step in the question answering process for systems that extract answers from external sources, typically known as *answer projection.* For every Web-derived answer, a system must find a supporting document from the AQUAINT corpus, even if the corpus was not used in the answer extraction process.

This year's main task included definition and list questions in addition to factoid questions. Although Web-based techniques have proven effective in handling factoid questions, they are less applicable to tackling definition and list questions. The data-driven approach implicitly assumes that each natural language question has a unique answer. Since a single answer instance is sufficient, algorithms were designed to trade recall for precision. For list and definition questions, however, a more balanced approach is required, since multiple answers are not only desired, but necessary. We believe that the best strategy is to integrate Web-based approaches with more traditional question answering techniques driven by document retrieval and named-entity detection. Corpus- and Web-based strategies should play complementary roles in an overall question answering framework.

2 List Questions

For answering list questions, our system employs a traditional pipeline architecture with distinct stages for document retrieval, passage retrieval, answer extraction, and duplicate removal (see Figure 1). The general idea is to successively narrow down the AQUAINT corpus, first to a candidate list of documents, then to manageable-sized passages, and fi-

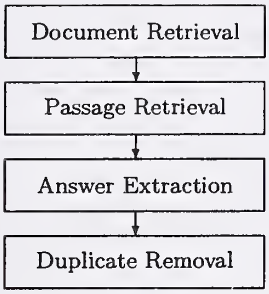


Figure 1: Architecture for answering list questions.

nally employ knowledge of fixed lists to extract relevant answers. The following subsections describe this process in greater detail.

2.1 Document Retrieval

In response to a natural language question, our document retriever provides a set of candidate documents that are likely to contain the answer; these documents serve as the input to additional processing modules. As such, the importance of document retrieval cannot be overstated: if no relevant documents are retrieved, any amount of additional processing would be useless.

For our document retriever, we relied on Lucene, a freely available open-source IR engine.[1] Lucene supports a weighted boolean query language, although the system performs ranked retrieval using a standard *tf.idf* model. We have previously discovered that for the purposes of passage retrieval, Lucene performs on par with state-of-the-art probabilistic systems based on the Okapi weighting (Tellex et al., 2003).

An often effective way to boost document retrieval recall is to employ query expansion techniques. In our TREC entry this year, we implemented two separate query generators that take advantage of linguistic resources to expand query terms. Lucene provides a structured query interface that gives us the ability to fine-tune our query expansion algorithms. In the following subsections, we describe these two techniques in greater detail.

2.1.1 Method 1

Our first query generator improves on a simple bag-of-words query by taking inflectional and derivational morphology into account: queries are a conjunction of disjuncts, where each disjunct contains morphological variants of a single term. Base query terms are extracted from the natural language question by removing all stopwords. Assuming we have three query terms, A, B, and C, arranged in increasing *idf*, our first query method would generate the following queries:

$$\begin{array}{l} A \wedge B \wedge C \\ e(A) \wedge e(B) \wedge e(C) \\ e(B) \wedge e(C) \\ e(C) \\ e(A) \wedge e(B) \\ e(B) \\ e(A) \end{array}$$

where

$$e(x) = x \vee \text{inflect}(x)^{0.75} \vee \text{derive}(x)^{0.50}$$

where $\text{inflect}(x)$ and $\text{derive}(x)$ represent a disjunct of inflectional and derivational morphological forms of x, respectively. The first query is simply a conjunction of all non-stopwords from the question. The second query is a conjunction where each of the conjoined elements is a disjunct of the morphological expansions of a query term. Inflectional variants are generated with the assistance of WordNet (to handle irregular forms). Derivational variants are generated by a version of CELEX that we manually annotated. Using Lucene's query weighting mechanism, inflected forms are given a weight of 0.75, and derivational forms a weight of 0.5. To generate subsequent queries, the system successively drops disjuncts starting with the disjunct associated with the lowest *idf* term until all disjuncts have been dropped—this has the effective of query relaxation. After that, the highest *idf* disjunct is dropped, and the generator starts a fresh cycle of successively dropping the lowest *idf* disjuncts.

Our document retriever is given a target hit list size, and successively executes queries from the query generator until the target number of documents has been found. This ensures that downstream modules will always be given a consistently-sized set of documents to process.

2.1.2 Method 2

Our second query generation algorithm takes advantage of named-entity recognition technology and other lexical resources to chunk natural language questions so that query terms are not broken across constituent boundaries. To identify relevant named entities, we use Sepia (Marton, 2003), an information extraction system based on Combinatory Categorial Grammar (CCG). In particular, personal names are recognized so that inappropriate queries are never generated; for example, a name such as "John Fitzgerald Kennedy" can produce legitimate queries involving "John F. Kennedy", "John Kennedy", and "Kennedy", but never "John Fitzgerald" or simply "John". For certain classes of named-entity types, we have encoded a set of heuristic rules that generates the acceptable variants. Our

[1] `jakarta.apache.org/lucene/docs/index.html`

query generator takes advantage of Lucene's ability to execute phrase queries to ensure that the best matching documents are returned.

Our second query generator also leverages WordNet to identify multi-word expressions that should not be separated in the query process. Multi-token collocations such as "hot dog" should never be broken down into `hot` and `dog`, since the meaning of hot dog cannot be compositionally derived from the individual words. Because these multi-word expressions cannot be predicted syntactically (e.g., compare "hot dog" with "fast car"), one practical solution is to employ a fixed list of such lexical items. If a query term is neither a recognized entity nor a multi-word expression, our second query generator expands the term with inflectional and derivational variants using the same technique as the first method.

We found that our first query generation method traded off precision for recall with its elaborate term dropping strategy—often, the first few queries are too restrictive, and because of this, most of the documents are retrieved by overly general queries. The result is often a hit list that has been "padded" with irrelevant documents; it appears that loose queries with few terms aren't precise enough to retrieve good candidate documents. As an alternative, we implemented a slightly different strategy for our second query generator. It drops query disjuncts in order of increasing *idf* until no terms remain, and then stops. As a simple example, if the query has three (non-stopword) terms, A, B, and C, arranged in increasing *idf*, our second query generator would produce the following queries:

$$e(A) \wedge e(B) \wedge e(C)$$
$$e(B) \wedge e(C)$$
$$e(C)$$

where $e(x)$ represents the expansions of an individual query term, as described in this section.

2.2 Passage Retrieval

The next stage in the processing pipeline for answering list questions is passage retrieval, which attempts to narrow down the set of candidate documents to a set of candidate passages, which are sentences in our architecture.

In a separate study of passage retrieval algorithms (Tellex et al., 2003), we determined that IBM's passage scoring method (Ittycheriah et al., 2000; Ittycheriah et al., 2001) produced the most accurate results. To determine the best passage (sentence in our case), our system breaks each candidate document into sentences and scores each one based on the IBM algorithm.

The IBM passage retrieval algorithm computes a series of distance measures for each passage. The "matching words measure" sums the *idf* values of words that appear in both the query and the passage. The "thesaurus match measure" sums the *idf* values of words in the query whose WordNet synonyms appear in the passage. The "mis-match words measure" sums the *idf* values of words that appear in the query and not in the passage. The "dispersion measure" counts the number of words in the passage between matching query terms, and the "cluster words measure" counts the number of words that occur adjacently in both the question and the passage. These various measures are linearly combined to give the final score for a passage.

We modified the IBM passage scoring algorithm to take into account linguistic knowledge provided by our query generator. The modified algorithm includes scores for matching hyponyms, inflectional variants, derivational variants, and antonyms (negative weight). In addition, our modified algorithm takes advantage of multi-word expressions tokenized from the question, that is, occurrences of "hot" and "dog" within a passage will not match "hot dog".

One of our goals is to determine the effects of additional linguistic knowledge on performance, and for our TREC submissions, we set up a matrix experiment with two query generators and two passage retrievers (the original IBM method and our modified algorithm). The results will be discussed later in Section 5.

2.3 Answer Extraction

The first step of the answer extraction process is to determine the question focus—the word or phrase in the question that is used to identify the ontological type of the entity we are looking for (i.e., the target type). For this, we enlisted the parser of the START question answering system (Katz, 1997). In addition, we have also constructed a mapping from question focus to target type. Consider a question such as "List journalists that have won the Pulitzer Prize more than once?": START would recognize *journalist* as the question focus, and PERSON as the target type (since we don't have a specific category for journalists in our ontology).

Separately, we have compiled offline a large knowledge base of entities, mostly in the form of fixed lists, that correspond to the various target types. For example, we have gathered lists of U.S. states, major U.S. cities, major world cities, countries, person names, etc. If the target type is among one of these categories for which we have a fixed list, our answer extractor simply extracts instances of the target type from the top ranking passages collected by the previous stage.

As an example, consider the following question:

> In which U.S. states have there been fatalities caused by snow avalanches? (q2183)

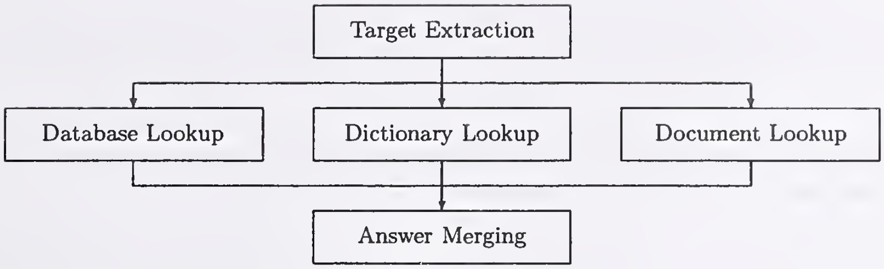


Figure 2: Architecture for answering definition questions.

Our system correctly identifies the question focus as "U.S. state" (corresponding to the target type US STATE) and extracts all instances of U.S. states from top ranking passages. Since the passage retrieval algorithm returns passages that already have occurrences of terms from the question, instances of the target type are likely to be the correct answer.

If the target type is not in our knowledge base, we employ two backoff procedures. Occasionally, answers to list questions have the question focus directly embedded in them (e.g., "littleneck clam" is a type of clam), and in the absence of any additional knowledge, noun phrases containing the question focus are extracted as answer instances. Finally, if no noun phrases containing the question focus can be found, our answer extraction module simply picks the noun phrase closest to the question focus in each of the passages and returns that as the answer.

After collecting all the answer candidates, we discard ones with query terms in them. Noun phrases containing keywords from the query typically repeat some aspect of the original user question and make little sense as answers. This heuristic has worked well in our previous question answering system (Lin and Katz, 2003).

2.4 Duplicate Removal

Answer instances extracted from the previous stage typically contain duplicates, which our system removes using a thresholded edit-distance measure. Finally, the system computes the number of answer instances to return based on a relative thresholding scheme. Each answer candidate is given a score equal to the score of the passage from which it was extracted, and all candidate answers below 10% of the maximum score are discarded. The remaining instances are returned as the final answers.

3 Definition Questions

Our architecture for answering definition questions is shown in Figure 2. The target extraction module first analyzes the natural language question to determine the unknown term. Once the target term has been found, three parallel techniques are employed to retrieve relevant nuggets that "define" the term: lookup in a database of relational information created from the AQUAINT corpus, lookup in a Web dictionary followed by answer projection, and lookup directly in the AQUAINT corpus with information retrieval techniques. Answers from the three different sources are merged to produce the final system output. The following subsections briefly describe each of these techniques; please refer to our forthcoming paper (Hildebrandt et al., 2004) for more details.

3.1 Target Extraction

We have developed a pattern-based parser to analyze definition questions and extract the target term using simple regular expressions. If the natural language question does not fit any of our patterns, the parser heuristically extracts the last sequence of capitalized words in the question as the target. Our simple definition target extractor was tested on definition-style questions from the previous TREC evaluations and performed quite well on those training questions.

3.2 Database Lookup

The use of surface patterns for answer extraction has proven to be an effective strategy for question answering. Typically, surface patterns are applied to a candidate set of documents that have been returned by traditional document retrieval systems. While this strategy may be effective for factoid questions, it generally suffers from low recall. In the case of factoid questions, where only one instance of an answer is necessary, recall is not a primary concern. However, definition questions require a system to find as many relevant nuggets as possible, making recall very important.

Instead of using surface patterns post-retrieval, we

copular pattern: A **fractal** *is* a pattern that is irregular, but self-similar at all size scales
appositive pattern: The **Aga Khan**, Spiritual Leader of the Ismaili Muslims
occupation pattern: *steel magnate* **Andrew Carnegie**
verb pattern: **Althea Gibson** *became* the first black tennis player to win a Wimbledon singles title
parenthesis pattern: **Alice Rivlin** *(*director of the Office of Management and Budget*)*

Figure 3: Sample nuggets extracted from the AQUAINT corpus using surface patterns. The target terms are in bold, the nuggets underlined, and the pattern landmarks in italics.

employ an alternative strategy: by applying a set of surface patterns offline, we are able to "precompile" from the AQUAINT corpus knowledge nuggets about every entity mentioned within it. In essence, we have automatically constructed an immense relational knowledge base, which, for each entity, contains all the nuggets distilled from every article within the corpus. Once this database has been constructed, the task of answering definition questions becomes a simple database lookup.

Our surface patterns operate both at the word level and the part-of-speech level. We utilize patterns over part-of-speech tags to perform rudimentary chunking, such as marking the boundaries of noun phrases. Our system uses a total of thirteen patterns, some of which are described below (Figure 3 shows several examples):

- **Copular pattern.** Copular constructions often provide a definition of the target term. However, the pattern is a bit more complex than finding the verb *be* and its inflectional variants; in order to filter out spurious nuggets (e.g., the progressive tense), our system throws out all definitional nuggets that do not begin with a determiner; this ensures that we only get "NP_1 be NP_2" patterns, where either NP_1 or NP_2 can be the nugget.
- **Appositive pattern.** Commas typically provide strong evidence for the presence of an appositive. With the assistance of part-of-speech tags, identifying "NP_1, NP_2" patterns is relatively straightforward. Most often, NP_1 is the target term and NP_2 is the nugget, but occasionally the positions are swapped. Thus, we index both NPs as the target term.
- **Occupation pattern.** Common nouns preceding proper nouns typically provide some relevant information such as occupation, affiliation, etc. In order to boost the precision of this pattern, our system discards all common noun phrases that do not contain an "occupation" such as *actor*, *spokesman*, *leader*, etc. We mined this list of occupations from WordNet and Web resources.
- **Verb pattern.** By statistically analyzing a corpus of biographies of famous people, we were able to compile a list of verbs that are commonly used to describe people and their accomplishments, including *became*, *founded*, *invented*, etc. This list of verbs is employed to extract "NP_1 *verb* NP_2" patterns, where NP_1 is the target term, and NP_2 is the nugget.
- **Parenthesis pattern.** Parenthetical expressions following noun phrases typically provide some interesting nuggets about the preceding noun phrase; for persons, it often contains birth/death years or occupation/affiliation.

Typically, our patterns identify short nuggets on the order of a few dozen characters. In answering definition questions, we decided to return responses that include additional context. To accomplish this, we simply expand all nuggets around their center point to encompass one hundred characters. We found that this technique enhances the readability of our responses: many nuggets seem odd and out of place without context and surrounding text is often necessary for disambiguation. Furthermore, returning a longer answer means that our responses sometimes contain additional relevant nuggets that are not part of the nugget matched by the original pattern. In definition questions, these "fortuitous" nuggets serve as an additional source of answers.

One drawback to our knowledge base of nuggets is the tremendous amount of redundancy contained within it. Because we compiled all patterns from all entities within the entire AQUAINT corpus, common nuggets are often repeated. In order to deal with this, we employed a simple heuristic to remove duplicate information: if any two responses share more than sixty percent of their keywords, one of them is randomly thrown out.

3.3 Dictionary Lookup

Another component of our system for answering definition questions utilizes an existing Web-based dictionary for nuggets. Obviously, such an approach cannot be applied directly, because all nuggets must originate from the AQUAINT corpus. To address this issue, we developed answer projection techniques to "map" dictionary definitions back onto AQUAINT documents. The mapping component is based on the idea that if you already know the answer, it is much easier to find relevant nuggets in the corpus.

Given the target term, our dictionary wrapper goes online to the Merriam-Webster website and fetches the term's definition. Keywords from the definition are used as the query to the Lucene document retriever. Once a set of candidate documents has been returned, we break each document into sentences and score each sentence based on its keyword overlap with the dictionary definition. The sentences with the highest scores are retained and, if necessary, shortened to one hundred characters centered around the target term.

3.4 Document Lookup

As a last resort (i.e., if no answers are found with the first two techniques), our system employs standard document retrieval to extract relevant nuggets. The target term is used as a Lucene query to gather a set of candidate documents. These documents are chunked into separate sentences and those sentences containing the target term are retained as responses. As before, these sentences are shortened appropriately if needed.

3.5 Answer Merging

The input to the answer merging stage is a series of one hundred character responses from each of the sources: database, dictionary, and corpus. The responses are arranged according to an ad-hoc priority scale we developed based on the accuracy of each approach. For example, we found that verb patterns generally return very good nuggets, and copular constructions are often less accurate. The priority of dictionary answers lies somewhere between the best and worst patterns, ordered such that some dictionary responses (if any) would always be returned in the final answer. Responses extracted directly from document lookup are used only if the two other methods return no answers: document lookup is considered a strict back-off method used only as a last resort. See (Hildebrandt et al., 2004) for a more detailed description of our ordering algorithm.

Finally, the answer merging stage of our system also decides the number of one hundred character responses to return. Since the length penalty for returning long answers is not very steep, we decided to return longer answers in hopes of including more relevant nuggets. Given n responses, we calculated the final number of responses to return as:

$$\begin{array}{ll} n & \text{if } n \leq 10 \\ n + \sqrt{n - 10} & \text{if } n > 10 \end{array}$$

This strategy ensures that our system will always return a generous number of nuggets, and has proven to work well empirically.

4 Factoid Questions

Our system for answering factoid questions was largely unchanged from last year. We employed the Aranea question answering system (Lin et al., 2002; Lin and Katz, 2003), which embraces two different views of the World Wide Web: as a heterogeneous collection of unstructured documents and as a source of carefully crafted and organized knowledge about specific topics.

Aranea's approach is primarily motivated by an observation that the distribution of user queries qualitatively obeys Zipf's Law—a small fraction of question types accounts for a significant fraction of all question instances. Large classes of commonly-occurring questions translate naturally into database queries and are handled by Aranea using a technique we call *knowledge annotation*, which allows our system to access semistructured and heterogeneous data as if it were a uniform database. In addition, we have discovered that a simple object–property-value data model captures the content of both Web resources and natural language questions (Katz et al., 2002). To take advantage of these observations, we have built a framework of site-specific wrappers that provide uniform access to knowledge contained in a variety of Web resources. These wrappers are connected to natural language questions through parameterized schemata.

As with all Zipf curves, there is a broad tail where individual instances are either unique or account for an insignificant fraction of total questions. To answer questions that cannot be easily classified into common categories or grouped by simple patterns, Aranea employs what we call redundancy-based *knowledge mining* techniques. Knowledge mining leverages the massive amounts of information available on the Web to overcome many thorny problems associated with natural language processing. The insight is simple: the more data available, the greater the chance that the answer to a natural language question is stated as a reformulation of that question. In such cases, simple pattern matching techniques suffice to accurately extract answers.

The setup of the TREC evaluation requires each answer to be supported with a document from the AQUAINT corpus. Since Aranea does not use the AQUAINT corpus in the question answering process, Web-based answers must then be "projected" back onto AQUAINT documents. Answer projection is accomplished in a two-step process: first, a set of candidate documents is gathered; then, a modified passage retrieval algorithm scans the documents to pick the best document. For obtaining the set of candidate documents, we tried three different approaches: using the NIST-supplied PRISE documents, using documents generated by our first query generation algorithm (Section 2.1.1), and using documents generated by our second query generation algorithm (Section 2.1.2).

After a set of candidate documents has been gathered, the answer projection module applies a modified window-based passage retrieval algorithm to score the documents. Each 140-byte window is given a score equal to the number of times keywords from both the question and candidate answer appear, with the restriction that at least one keyword from the question must appear in the passage. The score of a document is simply the score of the highest scoring passage. The highest scoring document is paired with the Web-derived candidate answer as the final response unit.

5 Results

A summary of our results at this year's TREC evaluation is shown in Table 1. Out of twenty-five groups, we ranked sixth in factoid questions, third in list questions, and eighth in definition questions. Our final weighted score ranked us sixth out of all the participating groups. For factoid questions, the query generation algorithms used for answer projection in each of the runs are shown in Table 2. For list questions, the query generator and passage scoring algorithms used for each of the runs are also shown in Table 2. For definition questions, all three submissions were exactly the same.

5.1 List Results

In this section, we discuss our results for answering list questions with respect to query generation, passage retrieval, and the question focus/target type.

5.1.1 Query Generation

Our second query generation method performed slightly better than our first query generation method. In particular, tokenization of multi-token expressions had the biggest positive impact on performance. Consider the following question:

> What countries have had school bus accidents that resulted in fatalities? (q2180)

The second query generation algorithm correctly identified "school bus" as a collocation and thus never broke up the expression into "school" and "bus".

5.1.2 Passage Retrieval

In general, the modified IBM passage scoring algorithm performed slightly worse than the original IBM algorithm. However, since they returned exactly the same responses most of the time, it is difficult to determine if the score differences are above the margin of error inherent in human judgments. In retrospect, we believe that our modified IBM algorithm was too lax in matching various forms of expansions (too high a score was given to variants). It is a well-known result that uncontrolled expansion of lexical-semantic relations (e.g., synonyms and hyponyms) results in lower performance (Voorhees, 1994). It has likewise been shown that inflectional and derivational expansion does not significantly increase performance. However, these previous experiments were focused solely on document retrieval, using queries that were typically much longer than TREC-style natural language questions. For the question answering task, we believe that linguistically-motivated query expansions will have a positive impact on performance. While our experiments have not yet shown a significant overall positive effect, we attribute this to implementational deficiencies in our overall system, rather than conceptual shortcomings.

As an illustrative example, we present a case where matching of expanded terms did increase performance:

> What countries still have royalty? (q2250)

The original IBM passage retrieval algorithm did not return any correct answers, whereas the modified version returned four correct answers. The performance increase can be directly attributed to looser query matching of expanded terms. The original algorithm found passages related to the economic sense of royalty, whereas the modified algorithm retrieved passages with the correct sense related to monarchy.

5.1.3 Question Focus and Target Type

Our strategy for answering list questions crucially depends on correctly identifying the question focus and the associated target type (the ontological type of entity sought after). For a few questions, our system was unable to correctly determine the question focus, resulting in a score of zero for those questions. To address this shortcoming, we will improve START's ability to recognize question focus.

Although identifying the question focus helps in answering a question, care is needed to map the focus word into a corresponding target type (the specific ontological category). Consider the following questions:

> List the names of cell phone manufacturers. (q2096)
> Name recipients of funds given by the various foundations of Bill and Melinda Gates. (q2291)

Our system correctly identified "manufacturer" as the question focus in the first question, but chose the

Task	MITCSAIL03a	MITCSAIL03b	MITCSAIL03c	best	median	best rank
Factoid	0.293	0.295	0.291	0.7	0.149	6th
List	0.13	0.118	0.134	0.396	0.053	3rd
Definition	0.309	0.282	0.282	0.555	0.189	8th
weighted total	0.256	0.248	0.250	0.559	0.135	6th

Table 1: Summary of MIT CSAIL submissions.

	MITCSAIL03a	MITCSAIL03b	MITCSAIL03c
List questions:			
Query generator	method 1	method 2	method 2
Passage retriever	IBM	IBM	modified IBM
Factoid questions:			
Answer projection	PRISE	method 1	method 2

Table 2: Variations in each of the TREC runs.

wrong sense for the target type. The term was on our list of professions, so the system incorrectly looked for personal names. The second question demonstrates that not all focus terms, even when correctly identified, are useful. "Recipients" are so general that they can be anything: people, companies, organizations, and even countries.

Not surprisingly, our system performed well for questions whose target type had corresponding fixed lists in our knowledge base. Since we had exhaustive lists for entities like cities, countries, and U.S. presidents, all our answers were at least of the correct type. However, since our system ignored syntactic relations within the passage, it often overgenerated wrong answers. Consider the following question:

> What countries have won the men's World Cup for soccer? (q2346)

Since our system returned all countries found near the relevant keywords, most of the answers were countries that played in the World Cup, not winners of it. As a result, we obtained high recall, but poor precision, on this question. This is certainly a case where the use of syntactic relations can dramatically improve question answering performance (Katz and Lin, 2003).

Our backoff method of looking for the question focus in candidate answers worked for the following question:

> What grapes are used in making wine? (q1940)

The system extracted correct answers like "Chardonnay Grapes". However, the same technique didn't work when the question focus was "team" or "food" because journalists typically do not write "X team" or "Y food".

5.2 Definition Results

Although the responses were identical in each of our three submitted runs for definition questions, the scores were not; that is, given the same exact answer string, assessors came up with different judgments some of the time. Out of the 317 responses we submitted for the 50 definition questions, there were 19 responses which were not judged the same over all three runs. However, 7 of these were cases where assessors found the same nugget in different responses for a question. In addition, there are clear instances where an answer nugget is in one of our responses and the assessors missed it, even when the nugget was present word for word. Voorhees' analysis (2003) of the definition results indicates that the margin of judging error was 0.043, i.e., scores for pairs of identical runs differed by as much as 0.043 (F-measure). Furthermore, due to the small testset size, a score difference of at least 0.1 in F-measure is required in order for two evaluation results to be considered statistically different (at 95% confidence).

Target term extraction was the single biggest source of error in answering definition questions. If the target term is not correctly identified, then all subsequent modules have little chance of providing relevant nuggets.

We did not anticipate the presence of stopwords in names. Consider the following questions:

> What is Bausch & Lomb? (q1917)
> Who is Vlad the Impaler? (q1933)
> Who is Akbar the Great? (q1955)

Our naive pattern-based target extractor identified "Lomb", "Impaler", and "Great" as the target terms for the above questions, respectively. Fortunately, "Impaler" is such a rare word that we actually returned nuggets concerning "Vlad the Im-

	MITCSAIL03a		MITCSAIL03b		MITCSAIL03c	
	PRISE		Method 1		Method 2	
Right	121	29.30%	122	29.54%	120	29.06%
Inexact	18	4.36%	15	3.63%	15	3.63%
Unsupported	26	6.30%	21	5.08%	21	5.08%
Wrong	248	60.05%	255	61.74%	257	62.23%
Total	413		413		413	

Table 3: Detailed analysis of factoid questions.

paler". Similarly, "Lomb" so frequently co-occurs with "Bausch & Lomb" that our system was able to provide relevant nuggets. However, since "Great" is a very common word, our definitions for "Akbar the Great" were meaningless.

The system's inability to parse certain forms of names is related to our simple assumption that the final consecutive sequence of capitalized words in a question is the target. However, this turned out to be an incorrect assumption:

> Who was Abraham in the Old Testament? (q1972)
> What is ETA in Spain? (q1987)
> What is Friends of the Earth? (q2222)

Our pattern-based target extractor marked "Old Testament", "Spain", and "Earth" as the targets for those questions, respectively. The inability to correctly identify the target term resulted in the system's failure to return relevant nuggets.

Another problem our target extractor encountered is apposition. Take the following example:

> What is the medical condition shingles? (q2348)

The target extractor incorrectly identified "medical condition shingles" as the target term. As a result, our system did not identify a single relevant nugget. To better extract target terms for definition questions, we will employ START and Sepia in the future, which we were unable to utilize for definition questions this year for technical reasons.

5.3 Factoid Results

Table 3 shows a detailed analysis of factoid questions. As in previous years, answer projection appears to be the Achilles' heel in our Web-based question answering strategy, as shown by the relatively large fraction of unsupported and inexact answers (in comparison to typical results of other teams). Furthermore, it does not appear that any of our more advanced query generation algorithms had any significant impact of the final score of factoid questions.

6 Conclusion

The focus of our research this year was to integrate Web- and corpus-based question answering techniques under a unified framework. This falls under our general research agenda of employing linguistic techniques, at the lexical, morphological, syntactic, and semantic levels, in conjunction with statistical techniques when appropriate. Although our TREC experiments have yet to show significant benefits from linguistically-informed processing techniques, we believe that high performance in the question answering task can only be achieved through fusion of multiple strategies and multiple resources.

References

Eric Brill, Jimmy Lin, Michele Banko, Susan Dumais, and Andrew Ng. 2001. Data-intensive question answering. In *Proceedings of the Tenth Text REtrieval Conference (TREC 2001).*

Charles Clarke, Gordon Cormack, Thomas Lynam, C.M. Li, and Greg McLearn. 2001. Web reinforced question answering (MultiText experiments for TREC 2001). In *Proceedings of the Tenth Text REtrieval Conference (TREC 2001).*

Susan Dumais, Michele Banko, Eric Brill, Jimmy Lin, and Andrew Ng. 2002. Web question answering: Is more always better? In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002).*

Wesley Hildebrandt, Boris Katz, and Jimmy Lin. 2004. Answering definition questions with multiple knowledge sources. In *Proceedings of the 2004 Human Language Technology Conference and the North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT-NAACL 2004).*

Abraham Ittycheriah, Martin Franz, Wei-Jing Zhu, and Adwait Ratnaparkhi. 2000. IBM's statistical question answering system. In *Proceedings of the Eighth Text REtrieval Conference (TREC-9).*

Abraham Ittycheriah, Martin Franz, and Salim Roukos. 2001. IBM's statistical question answering system—TREC-10. In *Proceedings of the Tenth Text REtrieval Conference (TREC 2001).*

Boris Katz and Jimmy Lin. 2003. Selectively using relations to improve precision in question answering. In *Proceedings of the EACL 2003 Workshop on Natural Language Processing for Question Answering.*

Boris Katz, Sue Felshin, Deniz Yuret, Ali Ibrahim, Jimmy Lin, Gregory Marton, Alton Jerome McFarland, and Baris Temelkuran. 2002. Omnibase: Uniform access to heterogeneous data for question answering. In *Proceedings of the 7th International Workshop on Applications of Natural Language to Information Systems (NLDB 2002).*

Boris Katz. 1997. Annotating the World Wide Web using natural language. In *Proceedings of the 5th RIAO Conference on Computer Assisted Information Searching on the Internet (RIAO '97).*

Jimmy Lin and Boris Katz. 2003. Question answering from the Web using knowledge annotation and knowledge mining techniques. In *Proceedings of Twelfth International Conference on Information and Knowledge Management (CIKM 2003).*

Jimmy Lin, Aaron Fernandes, Boris Katz, Gregory Marton, and Stefanie Tellex. 2002. Extracting answers from the Web using knowledge annotation and knowledge mining techniques. In *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002).*

Gregory A. Marton. 2003. Sepia: Semantic parsing for named entities. Master's thesis, Massachusetts Institute of Technology.

Stefanie Tellex, Boris Katz, Jimmy Lin, Gregory Marton, and Aaron Fernandes. 2003. Quantitative evaluation of passage retrieval algorithms for question answering. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003).*

Ellen M. Voorhees. 1994. Query expansion using lexical-semantic relations. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR-1994).*

Ellen M. Voorhees. 2003. Overview of the TREC 2003 question answering track. In *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003).*

MITRE's Qanda at TREC-12

John D. Burger
The MITRE Corporation
john@mitre.org

Introduction

Qanda is MITRE's TREC-style question answering system. This year, we were able to apply only a small effort to the TREC QA activity, approximately one person-month. As well as some general improvements in Qanda's processing, we made some simple attempts to handle definition and list answers.

1. TREC-12 system description

Catalyst architecture

Qanda uses a general architecture for human language technology called *Catalyst*, (Burger & Mardis 2002, Nyberg et al., to appear). Developed at MITRE for the DARPA TIDES program, the Catalyst architecture is specifically designed for fast processing and for combining the strengths of Information Retrieval and Natural Language Processing into a single framework. Catalyst uses a dataflow architecture in which standoff annotations are passed from one component to another, with the components connected in arbitrary (acyclic) topologies. The use of standoff annotation permits components to be optimized for just those pieces of information they require for their processing.

Major system components

Qanda has a by now familiar architecture—questions are analyzed for expected answer types, documents are retrieved using an IR system and are then processed by various taggers to find entities of the expected types in contexts that match the question. Below we describe each of the major components in turn.

Question analysis: This component is run after the question has been subjected to POS and named entity tagging. It uses a simple grammar, currently hand-written, to identify important components of the question—see Section 2 below for more detail.

IR wrappers: Catalyst components have been written for several IR engines, taking the results of the question analysis and formulating an IR query. For continue to use the Java-based Lucene engine (Apache 2002). Lucene's query language has a phrase operator, and also allows query components to be given explicit weights. Qanda uses both of these capabilities in constructing queries from the information extracted from the question. For TREC-12, the top 25 documents were retrieved.

Passage processing: Retrieved documents are tokenized, and sentence boundaries are detected. Because some downstream components run more slowly than the rest of the system, Qanda assigns a preliminary score to each sentence by summing the log-IDF (inverse document frequency) of each word that overlaps with the question. Those sentences with a low score are filtered out and not processed by most of the system.

Named entity tagging: Qanda uses Phrag (Burger et al. 2002), an HMM-based tagger, to identify named persons, locations and organizations, as well as temporal expressions. Phrag is also used as a POS tagger for question analysis.

Numeric tagging: A simple pattern-based tagger identifies measures, currency, percentages and other numeric phrases.

Specialized taggers: We have a simple facility for constructing taggers from fixed word- and phrase-lists. These were used to re-tag many named locations more specifically as cities, states/provinces, and countries. Qanda also identifies various other (nearly) closed classes such as precious metals, birthstones, various animal categories, etc.

Overlap: The question is compared to each sentence, and a number of overlap features are computed, some in terms of various WordNet relations (see Section 3).

Answer formation and ranking: Candidates are identified and merged, a number of features are collected, and a score is computed (Section 3).

For factoid questions, we simply use the top-scored candidate. For definition and list questions, we apply some other processing to generate answer strings, as detailed in Section 4

2. Questions analysis

Phrag, our HMM-based tagger, first annotates questions using separate models for part-of-speech and named entities. Qanda also runs a simple lookup-based tagger that maps head words to answer types in Qanda's ontology using a set of approximately 6000 words and phrases, some extracted semi-automatically from WordNet, some identified by hand. Based on these annotations, Qanda's main question analysis component uses a parser with a simple hand-optimized grammar to identify the following aspects of each question:

Answer type: a type in Qanda's (rather simple) ontology, e.g., PERSON or COUNTRY.

Answer restriction: an open-domain phrase from the question that describes the entity being sought, e.g., first woman in space.

Salient entity: What the question is "about". Typically a named entity, this corresponds roughly to the classical notion of topic, e.g., Matterhorn in What is the height of the Matterhorn?

Geographical restriction: Any phrase that seems to restrict the question's geophysical domain, e.g., in America.

Temporal restriction: Any phrase that similarly restricts the relevant time period, e.g., in the nineteenth century.

These features are emitted as annotations on the question, and are then available for other components to use.

3. Answer ranking with log-linear models

Qanda only examines sentences that match the question sufficiently, based on the IDF-weighted overlap described above. It collects candidate answers by gathering phrasal annotations from all of the semantic taggers, and identifies a number of features. These are combined using a conditional log-linear model trained from past TREC QA data sets. This approach has been used by several TREC participants, e.g., Ittycheriah et al (2001).

Many of the features used in the log-linear model reflect particular kinds of overlap between the question and the context in which the candidate answer is found:

Context IDF Overlap: Described above.

Context Unigram Overlap: Raw count of words[1] in common with the question.

Context Bigram Overlap: Raw count of word bigrams in common with the question.

Context Question Restriction Overlap: Raw count of words from the restriction phrase of the question (see Section 2).

Context Salient Overlap: Raw count of words considered especially salient by question analysis (see Section 2).

Context Synonym Overlap: Raw count of words that could be synonymous with questions words.

Context Antonym Overlap: Raw count of words that could be antonyms of question words.

The synonym and antonym features are computed with respect to WordNet (Fellbaum 1998).

Several features are computed based on the candidate itself, or its location in the context sentence:

Candidate Overlap: Raw count of words in common between the candidate itself and the question, to bias against entities from the question being chosen as answers.

Candidate Overlap Distance: Number of characters between the candidate and the closest (content) question word in the context.[2]

Candidate Question Restriction Distance: Number of characters between the candidate and a word from the restriction phrase of the question.

The only document-level feature currently used is the following:

IR Ranking of the source document by the IR system.

Candidates with similar textual realizations are merged, with the combined candidate retaining the highest value for each feature. Accordingly, an additional feature is maintained:

Merge Count

A number of boolean features are also computed that compare the question's expected answer type derived by analysis with the semantic type of the candidate:

[1] All of the "raw count" features described in this section omit stop words.

[2] Words would arguably be a more intuitive unit for this feature.

Type Same: Boolean, true if the candidate and expected answer types are identical.

Type Consistent: True if the candidate's type is "similar" to the expected answer type.

For the most part, candidates are only considered for a question if their types are consistent. For example, Where questions lead to an expected answer type of *LOCATION*, which is consistent with *COUNTRY* candidates; *How much* questions lead to *QUANTITY*, consistent with *PERCENTAGE*.

Ideally, Qanda would consider all candidates for all questions, but, if nothing else, performance considerations justify limiting this. We do not even represent all consistent pairs as explicit features. Instead, a small set of approximately 20 combinations was chosen by hand, as indicated in Figure 1. These represent particular biases or preferences that we feel justified in trying to acquire from the training data. In addition, some of these pairwise features represent exceptions to the consistency requirement, e.g., *PERSON* is not consistent with *COUNTRY*, but we wish to consider such candidates anyway. Similarly, we wish to consider certain named entity types as candidates, even when question analysis was unsuccessful in divining an expected answer type (*unknown*).

Question expected answer type	Candidate type.
PERSON	ORGANIZATION
PERSON	COUNTRY
NAME	PERSON
NAME	ORGANIZATION
NAME	LOCATION
CITY	LOCATION
DATE	YEAR
DATE	YEAR
ORGANIZATION	*other*
AMBIGLONG	DURATION
AMBIGLONG	LENGTH
AMBIGBIG	LENGTH
AMBIGFAST	SPEED
MEASURE	MASS
MEASURE	MONEY
MEASURE	MISCMEASURE
MEASURE	*other*
QUANTITY	PERCENT
unknown	LOCATION
unknown	ORGANIZATION
unknown	PERSON

Figure 1: Type-pair features used in evaluating answer candidates

After all of the (merged) candidates have been acquired, the raw feature values described above are normalized with respect to the maximum across all candidates for a particular question, resulting in values between 0 and 1.[3] Features normalized in this way are more commensurate across questions, especially word overlap and related features (Light et al. 2001).

The normalized features are then combined using the weights assigned by the log-linear model during training. This year, we trained the model using the questions from TREC 1999 and 2000. We also used a development test set composed of TREC 2002 factoid questions, the 25 AQUAINT definition evaluation questions, and the lists questions from 2001.

4. Special question types

This year, in addition to the usual factoid questions, systems had to deal with two additional types of questions, definition and list questions.

Definition questions

Qanda has no real facility for processing definition questions as such. Instead, we attempted to leverage our factoid question processing, which for the most part only considers named and other entities as candidate answers. Of course, very few definition answers were named entities, per se, but we noticed that certain kinds of named entities were involved with some definition answers, as indicated in the example below:

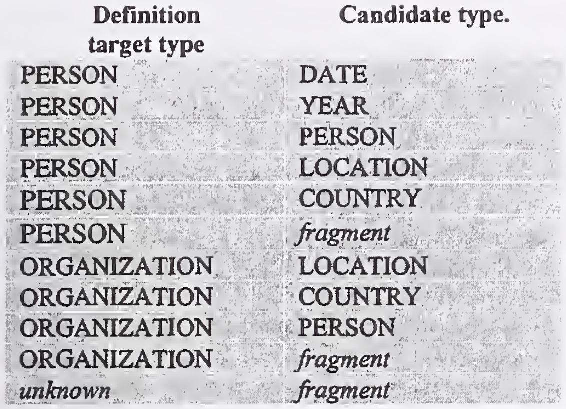
Who is Gunter Blobel?

Is at Rockefeller University
1999 Nobel prize in Medicine
was born in 1936
was born in Waltersdorf, Silesia, Germany

Qanda's question analysis component could already identify the semantic type of the definition target (e.g., *PERSON*, above). Since definition answers did not need to be exact, we decided to allow Qanda to consider certain entity types as "answers" to definition questions, deriving the actual answer string by extracting approximately 90 characters around the putative candidate.

We used the type-pair features described in Section 3 to license certain combinations of definition target type and candidate type, as shown in Figure 2.

[3] The normalized values are computed so that the intuitively "best" feature value is 1, the worst 0—this is primarily for the developers' convenience, but also so weights are all positive, and more easily reasoned about.



Definition target type	Candidate type.
PERSON	DATE
PERSON	YEAR
PERSON	PERSON
PERSON	LOCATION
PERSON	COUNTRY
PERSON	*fragment*
ORGANIZATION	LOCATION
ORGANIZATION	COUNTRY
ORGANIZATION	PERSON
ORGANIZATION	*fragment*
unknown	*fragment*

Figure 2: Type-pair features used in evaluating answer candidates

Additionally, we injected some non-entity candidates using crude heuristics for identifying short fragments occurring in appositional contexts. Our hope was that the type-pair features, as well as the candidate count feature, would allow the system to find some definition answers. As training data, we used 24 questions from TREC 1999 and 2000 that we determined were essentially definition questions.

We had some limited success with this approach. In the following examples, we indicate with underlining the actual "core" candidates that Qanda considered:

> 2112: *Who is Antonio Coello Novello*?
>
> *general, as New York's health commissioner on Thursday, drawing cries of betrayal from the abortion.*
>
> 2385: *What is the Kama Sutra?*
>
> *the famous Indian manual of sexual knowledge, and his pursuit of Lalita is a kind of realization*

These answers are hardly well-formed, but apparently relevant. It remains to be seen whether such a crude approach can be refined sufficiently to be useful, or if more sophisticated approaches are necessary.

List questions

Qanda currently treats list questions no differently from factoid questions, except that more than one answer is generated. We might have simply picked some fixed cutoff, say 3, and generated exactly three answers to every list question. We decided to attempt something slightly more sophisticated.

Since Qanda's candidate evaluation mechanism is probabilistic in nature, we decided to choose how many answers to generate dynamically, based on the expected value of the score we might receive. Each list question was to be scored using F-measure, the harmonic mean of precision and recall:

$$F = \frac{2PR}{(P+R)} = \frac{2c}{(n+r)}$$

Here, n is the number of answers we choose to generate, c is the number of correct answers we generate, and r is the total number of correct answers possible. We do not know in advance whether an answer is correct, but we can use Qanda's probabilistic score for the answers as the basis for an expectation of c. We have no real hope of estimating r, the number of correct answers possible, so we simply fixed this at the magic number 3.

Thus, our algorithm for generating list answers was to add each of the candidates to the answer set in turn, increasing n by one each time, and calculating the following expectation:

$$E[F] \approx \frac{2\sum_{i=1}^{n} s_i}{(n+r)}$$

Here, s_i is the score assigned to candidate i. We stop generating candidates when this expectation begins to decrease. On this year's evaluation, this admittedly crude mechanism performed slightly better than simply generating one candidate per list question (F=0.07 vs. 0.05).

5. Conclusion

As well as the requisite description of this year's system architecture, we have discussed Qanda's question analysis and our use of log-linear models for answer selection. We intend to improve the latter with better features, feature combination, and more sophisticated models. We also presented some initial results with crude mechanisms for generating definition and list answers. We intend to improve these components substantially as well.

References

Apache Software Foundation, 2002. "Jakarta Lucene—Overview". http://jakarta.apache.org/lucene/.

John D. Burger, John C. Henderson, William T. Morgan, 2002. "Statistical named entity recognizer adaptation", *Proceedings of the Conference on Natural Language Learning.* Taipei.

John D. Burger, Scott Mardis, 2002. "Qanda and the Catalyst architecture", *AAAI Spring Symposium on Mining Answers from Texts and Knowledge Bases.*

Christiane Fellbaum, ed. *WordNet: An Electronic Lexical Database.* MIT Press, 1998

Abraham Ittycheriah, Martin Franz, Salim Roukos, 2001. "IBM's statistical question answering system", *Proceedings of the Tenth Text Retrieval Conference (TREC-10).*

Marc Light, Gideon S. Mann, Ellen Riloff, Eric Breck, 2001. "Analyses for elucidating current question answering technology", *Natural Language Engineering* 7(4).

Eric Nyberg, John D. Burger, Scott Mardis, David Ferrucci, to appear. "Software Architectures for Advanced Question Answering", in *New Directions in Question Answering*, ed. Mark Maybury. AAAI Press.

Methods for accurate retrieval of MEDLINE citations in functional genomics

Mehmet Kayaalp,[a] Alan R. Aronson,[a] Susanne M. Humphrey,[a] Nicholas C. Ide,[a] Lorraine K. Tanabe,[a]
Lawrence H. Smith,[a] Dina Demner,[a,b] Russell R. Loane,[a] James G. Mork,[a] Olivier Bodenreider[a]

[a]National Library of Medicine, Bethesda, Maryland
{Kayaalp, Alan, Humphrey, Ide, Demner, Loane, Mork, Olivier}@nlm.nih.gov;
{Tanabe, LSmith}@ncbi.nlm.nih.gov

[b]University of Maryland, College Park, Maryland

Abstract

The lack of discipline and consistency in gene naming poses a formidable challenge to researchers in locating relevant information sources in the genomics literature. The research presented here primarily focuses on how to find the MEDLINE® citations that describe functions of particular genes. We developed new methods and extended current techniques that may help researchers to retrieve such citations accurately. We further evaluated several machine learning and optimization algorithms to identify the sentences describing gene functions in given citations.

Keywords: Genomics; MEDLINE; MeSH; Information Retrieval; Propositional Logic; Decision Lists; Machine Learning; Bayesian Networks; Model Averaging; Probabilistic Inference.

1 Introduction

Genomics research has created a wealth of information in a relatively short period of time. A downside of this rapid growth has been the inability of the research community to establish a disciplined and consistent labeling system for labeling new information (such as naming new genes and proteins). In the absence of such a systematic information labeling discipline, accessing certain genomic information might be insurmountable for researchers who are not in the circle of that particular genomics research.

To better understand the problem and perhaps to devise some remedies, the National Library of Medicine® (NLM®) and University of Maryland (UMD) teamed up to participate in the genomics track of the 12th Text Retrieval Conference (TREC-12) in 2003.

After describing the primary task of the genomics track in the next section, we introduce three different approaches to solving the problem in separate sections. In the subsequent section, we explain how the outcomes of these methods were combined. In Section 7, the secondary task and several methods towards the possible solutions are discussed. Our conclusions can be found in Section 8.

2 Primary Task

The primary task of the genomics track was defined as ad hoc information retrieval of MEDLINE citations that contain descriptions of a function of a gene given in the query of interest.

The provided corpus consisted of over a half million MEDLINE citations indexed between April 1, 2002 and April 1, 2003. The training query set consisted of 50 queries. Each query corresponded to a gene and was composed of a set of gene identifiers such as *official gene name*, *official symbol*, *alias symbol*, *preferred product*. The test query set contained the same type of information for another 50 genes.

The source of the gene information was the curated genes represented as NLM's LocusLink (LL) database. An LL record contains links in the form of unique identifiers to MEDLINE citations found in NLM's bibliographic resource known as PubMed®. Such a link, along with a brief description of gene function from the MEDLINE article, comprises a GeneRIF (Gene References Into Function). In the example shown in Table 1, the unique PubMed identifier (PMID) 11859139 and the passage next to it are a GeneRIF in the LL record for Interleukin-5 of *Mice*.

The first GeneRIF citation of the example shown in Table 1 is

> Mishra A, Hogan SP, Brandt EB, Rothenberg ME.: *IL-5 promotes eosinophil trafficking to the esophagus.* J Immunol. 2002 Mar 1;168(5):2464-9. PMID: **11859139** [PubMed - indexed for MEDLINE]

In this case, the description in the GeneRIF corresponds to the title of the MEDLINE citation. We informally call such a citation a GeneRIF citation.

Table 1: A Small Portion of LocusLink Record[1]

[illegible]	
Il5	interleukin 5
LocusID	16191
Locus Type	gene with protein product, function known or inferred
Product	interleukin 5
Alternate Symbols	Il-5
[illegible]	
PMID	*GeneRIF*
11859139	IL-5 promotes eosinophil trafficking to the esophagus
11960640	Role of IL-5 during primary and secondary immune response to acetylcholine receptor.

GeneRIF citations were considered the gold standard given that they were the most reliable information resource practically available at the time of the primary task design.

For training purposes, the GeneRIF citations associated with training queries were provided. For test queries, participants were asked not to retrieve and use associated GeneRIF citations. They were expected to develop methods/tools that would label all citations either as positive (i.e., relevant) or negative for every test query, and to submit all positive documents in rank order.

3 An Information Retrieval Approach

In this portion of our study, we used an in-house IR tool called Search Engine (SE) to identify GeneRIF citations. SE was developed at NLM to enable consumers of *ClinicalTrials.gov* to locate information relevant to their needs (McCray, Ide, Loane, & Tse, 2004).

To objectively evaluate SE's performance, Inquery (Callan, Croft, & Harding, 1992) was used to establish a baseline for the evaluation of SE. The best performance we could obtain using Inquery on the training dataset was 0.34 in mean average precision.[2]

3.1 Search Engine

The corpus was parsed by SE by (1) identifying XML fields, (2) tokenizing words, numbers, and non-alphanumeric characters, and (3) indexing all tokens associated with XML fields.

3.1.1 Tokenization

The retrieval was case insensitive. All letters were converted to lower case. All consecutive white spaces were collapsed to a single white space. Tokens containing both alphabetic and numeric characters such as *JAK2* were also searched for their hyphenated variants such as *JAK-2*.

Queries were preprocessed before scoring documents. Commas in queries were treated as separators of independent query terms. Parenthetical expressions in the gene names delimited by white space were considered as separate query terms; however, any other parenthetical expression such as 1(2)*gd2* was considered as a single token.

3.1.2 Scoring

The final document score was a conjunction of three part scores:

1. on species of interest,
2. on query terms, and
3. on key terms that occurred frequently in GeneRIF citations

If the exact species name or a variant of the term denoting the organism of interest (i.e., org^+) was not found in the <MeshHeading> field, then the likelihood of the document was of interest (i.e., d^+) was lowered drastically. Namely,

$$\frac{P(d^+|org^+)}{P(d^+|org^-)} = 1000 \tag{1}$$

Each query term t_i (e.g., *Slowpoke binding protein*) observed in an XML field (e.g., $xml(t_i)$ = <ArticleTitle>) was associated with a subjective probability $P(xml(t_i))$, which reflects our belief how likely a document is of interest given that the query term was observed in $xml(t_i)$.

$$\begin{aligned} P(d^+, t_i) &\equiv P(d^+, xml(t_i)) \\ &= P(d^+)P(xml(t_i)|d^+) \end{aligned} \tag{2}$$

The values of $P(xml(t_i)|d^+)$ for fields <ArticleTitle>, <MeshHeading>, <NameOfSubstance>, and <AbstractText> are 0.9, 0.6, 0.6, and 0.4, respectively, and $P(d^+) = 0.8$.

If the observed term t_i was not an exact query term, but an inflectional variant $lex(t_i)$ or a synonym $syn(t_i)$ of the exact query term, then

[1] An actual LocusLink record of *Locus:16191* contains a larger set of GeneRIF records among other entries such as Gene Ontology terms.

[2] This result was obtained by using the *sum#* and *n#* operators of Inquery , where $n = |query\ term\ tokens| + 2$, and indexing MeSH fields separately (Callan et al., 1992).

$$\begin{aligned} P(d^+,t_i) &\equiv P(d^+,xml(lex(t_i))) \\ &= P(d^+)P(xml(lex(t_i))|d^+) \\ &= P(d^+)P(xml(t_i)|d^+)P(lex(t_i)|d^+) \end{aligned} \tag{3}$$

The values of $P(lex(t_i)|d^+)$ and $P(syn(t_i)|d^+)$ are 0.9 and 0.8, respectively.

There were a number of other factors considered in scoring query terms, such as:

1. The specificity of the query term $spc(t_i)$ to the gene of interest. The term $P(spc(t_i)|d^+)$ was a multiplicative factor similar to $P(lex(t_i)|d^+)$ and had different probability assignments for preferred gene names (0.9), official gene names (0.8), preferred symbols (0.7), official symbols (0.6), symbol aliases (0.5), preferred product names (0.3), product names (0.3), protein alias (0.1), and derived terms[3] (0.01).
2. Multiword terms with $P(spc(t_i)|d^+) > 0.5$ were subject to phrase relaxation: If all tokens of a multiword phrase were found in arbitrary locations of an XML field of a document, then
$$\begin{aligned} P(d^+,t_i) = P(d^+)P(xml(t_i)|d^+) \\ P(rlx(t_i,n,m)|d^+) \end{aligned} \tag{4}$$
where $P(rlx(t_i,n,m)|d^+)$, n, and m denote conditional probability of the phrase relaxation score, the number of subphrases and number of tokens in term t_i and have the following characteristics:
$$\begin{gathered} 0.01 \le P(rlx(t_i,n,m)) = 0.01^{\frac{n-1}{m-1}} \le 1 \\ 1 \le n \le m \ne 1 \end{gathered} \tag{5}$$
3. Names of species were mapped into corresponding Medical Subject Headings (MeSH®) term equivalent:
   Homo sapiens → *Human*
   Mus musculus → *Mice*
   Rattus norvegicus → *Rats*
   Drosophila melanogaster → *Drosophila*
4. The query term *Rats* would match both MeSH terms "*Rats, Mutant Strain*" and "*Rats*." The latter is considered as an exact match. Query terms that exactly matched to the MeSH terms of interest contributed to the overall score with higher conditional probabilities.

When multiple query terms were observed in a given document, the probability that the document was of interest was computed as the union of the probabilities of all occurrences. For example, let d be a document, in which the query terms t_i and t_j were observed once. The probability that d was of interest was computed as follows:

$$\begin{aligned} P(d^+,t_i \cup t_j) = P(d^+,t_i) + P(d^+,t_j) \\ - P(d^+,t_i \cap t_j) \end{aligned} \tag{6}$$

The probability term of the co-occurrence $P(d^+,t_i \cap t_j)$ was computed with the assumption that t_i and t_j were independent from each other.

$$P(d^+,t_i \cap t_j) = P(d^+,t_i)P(d^+,t_j) \tag{7}$$

If a term or its lexical variant $lex(t_i)$ was observed in a given XML field of a document J times, the probability of the document relevancy was computed as follows:

$$\begin{aligned} &P\left(d^+,\bigcup lex(t_i)\right) \\ &\quad = P(d^+)4\sum_{j=1}^{J} P(lex(t_i)|d^+)5^{-j} \end{aligned} \tag{8}$$

Earlier research suggested that key context terms (i.e., with unusually high frequency counts only in that context) may be beneficial to filter in documents of interest from large corpora (Tanabe et al., 1999). A new set of key context terms $\{k_1,\ldots,k_{46}\}$ frequently occurred in GeneRIF citations, such as *genetics*, *gene expression*, and *sequence*, was collected ad hoc. The third part of the document probability was computed by evaluating the document with respect to all lexical variants $\{lex(k_i)\}$ of these terms.

$$P(d^+,xml(k_i)) = P(d^+)P(xml(k_i)|d^+) \tag{9}$$

where $k_i \in \{\{lex(k_1)\},\ldots,\{lex(k_{46})\}\}$.

The final document probability was obtained as

$$\begin{aligned} &P\left(d^+,org,\bigcup t,\bigcup k\right) \\ &\quad = P(d^+,org)P\left(d^+,\bigcup t\right)P\left(d^+,\bigcup k\right) \\ &\quad = P(d^+,org)\bigcup_{t_i} P(d^+,t_i)\cdot\bigcup_{k_i} P(d^+,k_i) \end{aligned} \tag{10}$$

3.1.3 Results and Analysis

The performance of SE was evaluated as 0.4168 in mean average precision using the trec_eval program provided by TREC organizers. In order to understand which heuristics played the major factor in obtaining this result, SE was rerun by excluding a different heuristic at each run. The largest performance drop (0.085) was observed when the organism names on the test queries were not mapped to the corresponding MeSH terms. A similar performance drop (0.081) was observed when the differentiations among XML fields were disregarded. When one-to-one (exact) mapping between MeSH terms and query terms was not taken into account, the retrieval performance dropped by 0.06. Performance drops

[3] A derived term is a portion of a query term that is delimited by commas or parentheses.

through the exclusions of any other heuristics (searching hyphen variants, frequently co-occurring terms, phrase relaxation, and synonymy) were insignificantly small (between 0.03 and 0.003).

The relatively low performance contributions of phrase relaxation and synonymy were surprising since these features provide empirically significant benefits in *ClinicalTrials.gov* environment. The difference might be due to the relatively well structured queries and the low counts of gene name synonymy found in the current version of the Unified Medical Language System® (UMLS®).

The possible performance differences between SE and other IR approaches might partially be due to SE's tokenization and scoring methods. For example, SE conserves parenthetical information in terms such as 1(2)*gd*2 and includes them in the search while others might search *gd*2 only. Unlike many other IR approaches, SE does not use an inverse document frequency statistics in evaluating the importancc of a token, since some commonly occurring words (e.g., *rat*, *fat*, *human*) may play crucial roles under specific conditions, as in the current task and should not be devaluated due to their high frequencies in the corpus.

4 A Rule-Based Approach

This portion of our research focuses on how effectively we can utilize MeSH resource. MeSH is a controlled vocabulary consisting of medical terms organized in a set of broader-narrower hierarchies.[4] A decision list composed of 18 nodes each of which was a propositional logic clause (i.e., a rule-based search statement consisting of MeSH terms using Boolean logic connectors) was developed to identify GeneRIF citations in the MEDLINE test corpus. A decision list (Rivest, 1987) is a special case of decision trees where each internal decision node has only one direct descendent (see Figure 1).

The traversal on the decision list is terminated when the propositional clause (*PC*) represented on the currently visited decision node is satisfied. If none of the *PC*s were satisfied, the document would be labeled as *negative* and would not be retrieved. The approach of using a sequence of models (such as *PC*s) and switching between them in this manner is sometimes called model switching (Kayaalp, Pedersen, & Bruce, 1997).

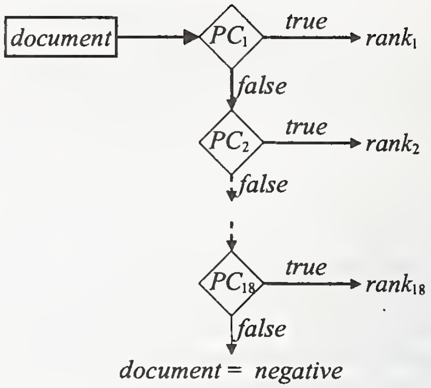


Figure 1: A Decision List with 18 *PC* Nodes

4.1 Controlled Vocabulary Search

In contrast to our other two approaches, this part of the study mainly relies on search on the structured data portion (MeSH) of the corpus. Since MeSH is a controlled vocabulary (as opposed to free text), the presented search method is called controlled vocabulary search (CVS).

CVS is composed of three subsequent phases:

1. Identification of MeSH terms in queries through pattern matching against MeSH records
2. Transformation of each query into a decision list of 18 *PC*s with MeSH terms
3. Search against the MeSH fields of each MEDLINE citation, and output the search result.

The first phase consisted of pattern matching between query terms and MeSH terms that are found in MeSH *regular descriptor* file and MeSH *supplementary concept records* file. Techniques were used for query expansion, tokenization, and eliminating results due solely to matching an acronym on the query side with an acronymic MeSH term. Names of species provided in query terms were converted to the corresponding MeSH terms as stated in Section 3.1.2.[5]

The second phase consisted of constructing a decision list of 18 *PC*s for each query. Each *PC* is a conjunction of three parts: (1) species name, (2) gene name, (3) MeSH qualifiers describing biomedical functions, with the exceptions of $PC_{16} - PC_{18}$, which were composed of only (1) and (2). These parts correspond to how indexers would likely index a document discussing function of a particular gene in a particular species. A simplified example of *PC* is

[4] More information about MeSH can be found in http://www.nlm.nih.gov/mesh/meshhome.html.

[5] This approach was considered a manual run since the algorithms identifying MeSH terms for gene names were subsequently modified for the test queries before the test run.

shown in Proposition (11), which corresponds to gene query 18, where the species name is *Mice*, gene name is *Interleukin-5*, and a set of MeSH qualifiers *genetics*, *physiology*, and *metabolism*:

$$\begin{aligned}&PC_i\,(Query = 18):\\&\quad Mice \wedge Interleukin\text{-}5 \qquad (11)\\&\quad \wedge (genetics \wedge (physiology \vee metabolism))\end{aligned}$$

Any MEDLINE citation indexed under *Mice* and one of the following terms would be retrieved (i.e., be satisfied by Proposition (11)):

- Interleukin-5/genetics/physiology
- Interleukin-5/genetics/metabolism
- Interleukin-5/genetics/physiology/metabolism

The order of the propositional clauses (*PC*s) was designed to maximize retrieval precision.

Ad hoc analysis of the training dataset yielded a set of nine MeSH qualifiers (e.g., *genetics*, *physiology*, *metabolism*) and 16 MeSH hierarchical nodes (e.g., *Cell Physiology*, *Gene Expression*, *Neoplasm Protein*) that occurred frequently in GeneRIF citations.

The last phase was the search for relevant citations against the test corpus and output of retrieval results. A citation was retrieved if at least one of the *PC*s was satisfied. As illustrated in Figure 1, the rank of a document was determined by the decision list order of the *PC* that retrieved the document first.

4.2 Results and Analysis

The retrieval performance of CVS was 0.34 (in mean average precision) on the training query set and 0.23 on the test query set.

We also tried an alternative ranking strategy based on the number of functional keywords (Tanabe et al., 1999) contained in the retrieved documents. The strategy improved the retrieval performance by 0.04, which however was not sustained when CVS was used along with SE and a collocation network to retrieve GeneRIF citations as explained in Section 6.

Retrospective analysis of results suggest that recall rate might have been improved if an additional set of *PC*s corresponding to a single *gene* conjunct was appended to the existing *PC*s.

As expected, many GeneRIF citations contained MeSH terms of proteins that were associated with the genes of interest. The current CVS algorithms for matching genes with MeSH protein terms found a reasonable match in 92% of cases, suggesting that a maintained crosswalk between genes and MeSH protein terms would be a valuable knowledge resource for the CVS method to answer gene queries posed to MEDLINE.

5 A Machine Learning Approach

Conventionally, supervised learning involves a classification task and a training dataset. A training corpus is usually evaluated as a bag of words associated with a specific class. After the model is learned on training corpus, it is used to classify a new corpus (test set). For example, given all MEDLINE citations published before 2003, a model may be learned to identify whether a citation is about a particular gene (e.g., *Interleukin*-5). Then that model may be used to classify any MEDLINE citation published in 2003 with the same classification criterion (e.g., whether a citation is about *Interleukin*-5 or not).

In contrast to the usual classification task, the current problem does not yield a persistent class that remains the same in both training and test cycles. Each test of the primary task involves a new class that was not available during the training phase. Obviously, an *Interleukin*-5 model (i.e., its *class* = *Interleukin*-5, its *feature set* = $\{Interleukin, IL, \ldots\}$, and parameters defined on them) would be useless in classifying whether a document is a citation on *Tropomyosin*-1(*alpha*). In other words, our higher-level question becomes:

> *How can we solve a classic information retrieval problem through machine learning methods?*

This part of the study was exploratory in nature; we looked for empirical evidence as to whether abstraction could be an answer to the above question. The class assignment was abstracted from a particular gene name to a Boolean decision of relevancy; i.e., $class = +$ denotes that a document is about a gene of interest.

Abstracting only class (from a particular instance of gene name such as *Interleukin*-5 to a generic gene of interest) would yield a model that learns a set of genes. Such a model may be trained to discriminate whether a document is a GeneRIF citation. The features of such a model may be a set keywords that would be specific to GeneRIF domain as the ones used in SE or MedMiner (Tanabe et al., 1999) but not specific to a particular gene. Obviously, such a system cannot identify whether a citation is about a particular gene of interest, which however was required by the primary task. In this study, in addition to abstracting the class of interest, the feature set was also abstracted from a set of phrases to a set of phrase containers, *n*-grams, of different sizes. For example, a gene name *Indian hedgehog* would be abstracted to

2-gram, and if a document contains the term then the variable 2-gram = + .

The premise was that if the training and test corpora were randomly sampled from the same population of citations, then the characteristics of $(class, \{n\text{-gram}\})$ distribution could be informative for inferring the document class of interest. For example, it is expected that a document that has a three-word phrase as part of the original query term is more likely to be a document of interest compared to another document that only has a two-word phrase of the query term.

$$\begin{aligned} &P(class = + | 2\text{-gram} = +, 3\text{-gram} = -) \\ &< P(class = + | 3\text{-gram} = +, 4\text{-gram} = -) \end{aligned} \tag{12}$$

Thus, two documents each of which satisfies two different conditions in (12) would be assigned different probability scores and ranked accordingly. The training corpus would serve as a means of learning the values of these probability mass functions.

5.1 Collocation Networks

A collocation network is a Bayesian network whose structure reflects the dependency hierarchy of morphological (e.g., lexical) and/or conceptual (e.g., semantic) collocations observed in corpora of interest. In this work, the focus is on lexical collocations. The root of the (collocation) network (see Figure 2) is the class *C* representing whether a given document *d* is of interest. Two other morphological constructs, titles and abstracts, were also considered. Their representations in the network were based on the assumption that the presence of lexical structures in titles is independent of the presence of lexical structures in abstracts, given *C*.

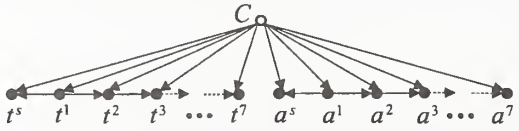


Figure 2: The Collocation Network Structure

Each descendent of the root is a member of an ordered set of *n*-grams, where $1 \leq n \leq 7$. Let w_i and w_j denote two different words, and (w_i, w_j) denote a two-word phrase in which w_i is followed by w_j. Since $P((w_i, w_j)|d)$, the probability that (w_i, w_j) is present in a given document *d*, depends on the presence of both w_i and w_j in *d*, a natural order for the *n*-gram dependency relationships may be

$$w^{n-1} \rightarrow w^n \tag{13}$$

where w^{n-1} is an $(n-1)$-gram and w^n is an *n*-gram containing w^{n-1}. The topological order in (13) was assumed in the network. In Figure 2, t^n and a^n where $1 \leq n \leq 7$ denote *n*-grams obtained from gene names and symbols that were observed in titles and abstracts, respectively. The nodes t^s and a^s represent gene symbols, which usually are single words, from titles and abstracts, respectively.

The *n*-gram variables of the network were generated by conserving the word order in gene names and gene symbols. For example, the gene name *Slowpoke binding protein* yields the following three *n*-grams in the first pass:

$$\begin{aligned} &1\text{-gram} = \{(slowpoke), (binding), (protein)\} \\ &2\text{-gram} = \{(slowpoke, binding), \\ &\qquad\qquad (binding, protein)\} \\ &3\text{-gram} = \{(slowpoke, binding, protein)\} \end{aligned} \tag{14}$$

In the second pass, each *n*-gram set is populated with the lexical variants of its elements using the SPECIALIST Lexicon (McCray, 1998). The lexical variants of *slowpoke*, *binding*, and *protein* are *slowpokes*, *bindings*, *bind*, *binds*, *bound*, *bounded*, *bounding*, *bounds*, and *proteins*, based on which all combinations are generated for each *n*-gram. If a given document abstract contains a three word phrase which is a member of the 3-gram set {(*slowpoke*, *binding*, *protein*),...,(*slowpokes*, *bounds*, *proteins*)} , then the variable a^3 would be labeled as positive for that document. All variables were labeled accordingly and this protocol was followed for each query (i.e., for each gene) separately.

5.2 Results and Analysis

Unlike SE, the collocation network presented in this study was in an early phase of its development. Even though it was not ready to be used as a stand-alone IR tool, our preliminary results on the training set (using leave-one-out cross-validation) indicated that the collocation network in its current state of development might improve overall retrieval performance if its results were combined with the results of SE and CVS using model averaging as explained in Section 6. For the training dataset, the retrieval performance of the collocation network was 0.24 in mean average precision using leave-one-out cross-validation. For the test queries, the retrieval performance of the collocation network was 0.11. The retrieval performance of the system combined with the other two systems is analyzed in Section 6.

Given the training results were obtained through leave-one-out cross-validation, which is known to be a very conservative measurement, the degradation of the performance indicates that the parameters learned on the training set were not as applicable to the test set.

The underlying assumption that was made in retrieving documents for the test queries was that training and test queries were selected randomly from the same population of queries. Any selection bias (such as elimination of certain queries through post-processing) may have caused degradation of the results. Had we had a larger set of training queries (a larger sample size), parameter learning and retrieval results might have been more robust.

This portion of the study was intended to evaluate the value of collocation information. The results suggest that a collocation network based IR system using MeSH indices may be useful in classification and retrieval of genomic information.

6 Model Averaging

In this study, we combined outputs of SE, CVS, and a collocation network through model averaging using uniform priors. We call the resulting system SCC. Model averaging is in line with the Bayesian approach, which suggests using all possible models in making inference. Since it is generally intractable, the approach is usually relaxed to model selection or selective model averaging (Heckerman, Meek, & Cooper, 1999).

Every document *d* in the corpus was labeled by each system as negative or as positive for a given query. Every positive label was associated with a rank order. Given a query and a document *d*, a composite relevancy score $crs(d)$ was obtained as follows:

$$crs(d) = c - \sum_{s \in \{SE, CVS, coNet\}} \alpha_s \cdot rank_s(d) \qquad (15)$$

where $rank_s(d)$ is the rank order of *d* according to system *s*; α_s is the prior and was set to $1/3$ for every *s*; $c \geq \sum_s \max(rank_s(d))$ is a constant and was set to 3000 in this study; and *coNet* denotes the collocation network. If *d* was evaluated as negative by *s*, then $rank_s(d) = c$. The rank order of the document was determined by a decreasing order of $crs(d)$, where

- the most relevant document has the highest $crs(\cdot)$, and
- documents with equal $crs(\cdot)$ share the same rank.

6.2 Results and Analysis

The retrieval performance of the combined system (SCC) was measured in mean average precision as 0.52 on the training set and 0.40 on the test set. Analysis of the test set results of SCC and SE reveals that SE was more precise in ranking documents but SCC was more robust in recall where the retrieval maximum was set at 1000 documents. SCC not only recalled all positive cases that SE identified but also recalled additional cases that SE missed. The difference in robustness is more obvious in Figure 3, where AUCs denoting the areas under the receiver operating characteristics (ROC) curves were plotted per query for both SE and SCC. As seen in this plot, SCC consistently performed with an $AUC > 0.7$.

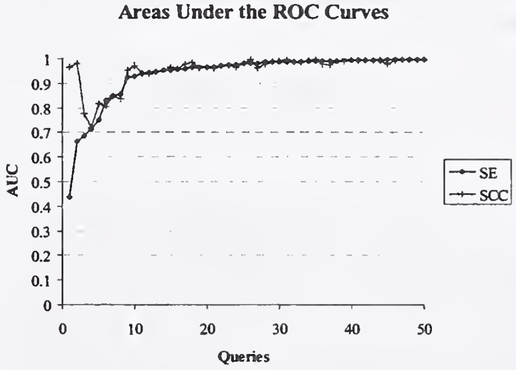


Figure 3: Retrieval performances of SE and SCC compared using ROC metric with queries sorted in increasing order of AUC of SE.

The difference between SE and SCC in terms of AUC is statistically significant: the mean AUC score of SE remains outside of the 95% confidence interval of AUC of SCC (see Figure 4). The values were obtained through bootstrapping with 10,000 samples.

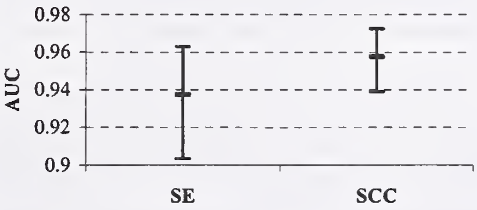


Figure 4: The mean AUC of SE remains outside of the 95% confidence intervals of AUC of SCC.

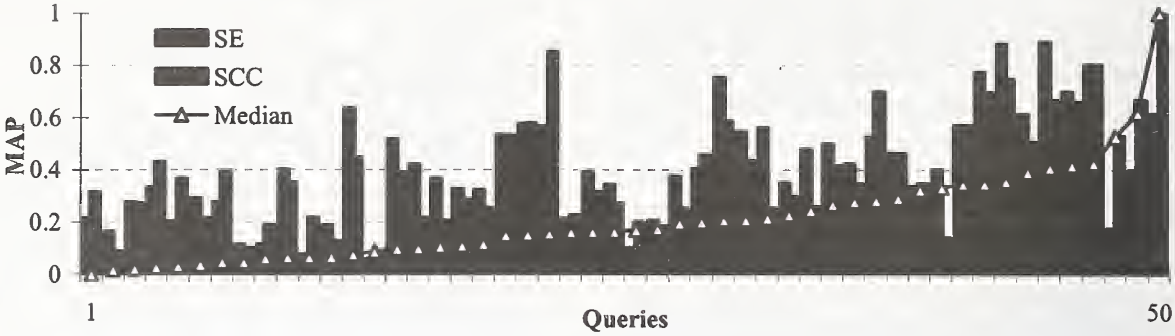


Figure 5: Retrieval performances of SE and SCC compared using mean average precision (MAP) against median performance values of other systems with queries sorted by Median in ascending order.

In Figure 5, the retrieval performances of SE and SCC were also compared against the median retrieval performance of other systems participated to the genomics track in 2003.

Except in one case, the retrieval performance of SCC did not drop below the median performance level.

7 Secondary Task

The secondary task of the genomics track was an information extraction (IE) task and its goal was defined as reproducing the GeneRIF annotations from MEDLINE citations. Our initial analysis revealed that 95% of GeneRIF annotations were taken form titles or abstracts of the corresponding MEDLINE citations, 42% of which were direct quotations. Thus, we decided that finding best sentences in the corresponding MEDLINE citations might serve the purpose of the secondary task.

We used a set of 9,403 recent MEDLINE documents associated with LocusLink GeneRIF records. After excluding 133 abstracts associated with the TREC test set, the documents were divided into a training set and a development test (*devtest*) set. The documents were segmented into sentences from the titles and abstracts so that each annotation A_i was associated with a set of candidate sentences $C_{i,1},\ldots,C_{i,n_i}$. The Dice score $s(C_{i,j})$ was computed for all *i* and *j*.

Our objective was to find a selection function $sel(A_i)$, which returns a sentence $C_{i,m}$ that maximizes the Dice score; i.e., $s(C_{i,m}) \ge s(C_{i,j})$ for all *j*. If the best possible candidate was selected, the average Dice score on the training set was 78.6%. Restricting candidate sentences to either titles or abstracts only, the best possible average Dice scores were 55.0% and 62.2% respectively. Selecting the first sentence from each title yielded 54.7%, which we used as baseline.

7.1 Methods and Results

We assumed that the optimal selection function depends on textual features of the candidate sentences. A very broad range of features familiar to the field of information retrieval and text summarization was considered, as summarized in Table 2. Feature values $f_j(C)$ were computed for each feature f_j and candidate sentence *C*. Two forms for the selection function were considered: (A) a weighted linear form, and (B) a predicate calculus form. In both cases, machine learning algorithms were used to search for the optimal instantiation.

A. In the weighted linear formula, weights w_j were associated with each feature, and a selection score was computed for each candidate sentence by the formula $sel\_score = \sum_j w_j f_j(C)$. The selection function was then defined by selecting from the candidate sentences the one with the largest selection score. Two indirect methods were employed to quickly compute w_j and obtain an estimate for *sel_score*:

1. Linear regression was used with $f_j(C)$ to predict $s(C)$, and this gave an average Dice score of 48.1%, with the highest weight assigned to the ABS feature (which indicates whether a sentence is in the abstract or title).
2. The CMLS algorithm (Zhang & Oles, 2001) was used to predict the candidate from each document with the highest Dice score (the objective is 1 for the candidate with the highest Dice score and 0 for all other candi-

Table 2: Lexical Features Used in Secondary Task

ABS	1 if sentence is in abstract
ALAST	1 if sentence is last sentence of abstract
ANUM	the sentence number for sentences in the abstract
GENE	1 if the sentence contains a gene name, determined by Abgene (Tanabe & Wilbur, 2002)
GOOD-LEN	1 if the sentence is in the title and has 5 or more words, or it is in the abstract and has 40 or fewer words
GOOD-NUMS	1 if sentence is in title, or number 7, 8, or 9 in abstract
HD*w*	1 if word *w* occurs as the head word of a noun phrase in the sentence (for 92 most frequent stemmed words in the training set)
LEN	number of characters in sentence
MAX-SENT	maximum Dice coefficient of this sentence to some other sentence in the document
MM	number of MedMiner keywords in the sentence (Tanabe et al., 1999)
MM*k*	1 if MedMiner keyword *k* occurs in the sentence (for the 78 most frequent keywords in the training set)
NBW	number of words that also occurred in the Brown or Wall Street Journal corpus
NUM-CAPS	number of capital letters occurring in the sentence
NUMDIGITS	number of digits occurring in the sentence
NUM-WORDS	number of words occurring in the sentence
REL	the relevancy score based on most frequent words in a document (Ishikawa, Ando, & Okumara, 2001)
ST*n*	the score for semantic index *n* (for 129 different types (Humphrey, Rindflesch, & Aronson, 2000))
T1*t*	1 if POS tag *t* is the first tag of the sentence (for 34 most frequently occurring first POS tags in the training set). For example, T1*DT* = 1 if the first word of the sentence is a determiner.
TDICE	the maximum Dice coefficient of the sentence with a sentence in the title (for abstract sentences only)
TNUM	the sentence number for sentences in the title
TNM	the sentence number of sentences in the abstract or the sentence number of the sentence in the title plus the number of sentences in the abstract
WORDS	sum of Bayesian weights of words in sentence

dates). The best Dice score obtained with this approach was 53.3% using only the ABS indicator, and could not be significantly increased by adding other features.

Finally, we implemented an incremental search algorithm to maximize the Dice score directly, one feature at a time. With weights trained on the training set and feature combinations selected based on the average Dice score of the *devtest* set, the best selection score function obtained was

$$sel\_score(C) = -f_{\mathrm{ABS}}(C) + 0.19 f_{\mathrm{REL}}(C) \\ -0.22 f_{\mathrm{T1}DT}(C) \qquad (16)$$

which gave an average Dice score of 54.42% on the *devtest* set. The features ABS, REL, and T1 are described in Table 2.

B. We also sought a predicate calculus formula to decide if a given title sentence was a best candidate GeneRIF. We used the Aleph inductive logic programming (ILP) system (Muggleton, 1995; Srinivasan, 2000) to induce a Prolog program (ILP theory) to find good titles using the features REL, NUMWORDS, NUMDIGITS, NUMBROWNWSJ, MM*n*, MM, and NUMCAPS. Title sentences that had minimum and maximum Dice scores among all candidates were used as negative and positive training examples, respectively. Consistent with all other findings, the induced program almost always selected one of the title sentences. If the ILP theory rejected a title, an abstract sentence covered by the ILP theory was selected. This resulted in an average Dice score of 48.6%, comparable to the linear regression result. In the TREC test set, all but seven GeneRIF candidates were indicated by ILP to be selected from titles.

Based on the average Dice score on the *devtest* set, the best performing method was based on the linear selection score shown in Equation (16). We used this to obtain our TREC submission for the secondary task, which had a average classic Dice of 50.36% on the TREC testing set. We pointed out the large negative weight assigned to the ABS feature in Equation (16) caused all sentences to be selected from titles.

Only five of these sentences were the second sentence in the title and the remaining 134 were the first. For comparison, selecting the first sentence from each title gives an average classic Dice score of 49.83%.

8 Conclusions

The primary task of the track was ad hoc retrieval of GeneRIF citations from a subset of MEDLINE corpus. We applied three approaches: (1) a conventional information retrieval approach using SE, (2) a rule-based decision making approach using CVS, and (3) a machine learning approach using a collocation network. The SE's test results and the test result of SCC, which is a combination of these three systems, were submitted.

Empirical results suggest that SE performed with high precision in most of the cases while SCC consistently showed robust performance.

SE is a deployed system servicing to the on-line users of *ClinicalTrials.gov*. Both CVS and collocation networks are in early phase of their development and can be improved significantly based on the experience that we gained in this study.

The results on the IR portion of the problem were submitted in two sets: (1) the documents retrieved by SE alone and (2) the documents retrieved by the ensemble called SCC, whose retrieval performances were measured in mean average precision as 0.42 and 0.40, respectively. The mean average precision of SE was slightly better than that of SCC, which was statistically not significant. SCC was evaluated as a more robust retrieval system than SE, where the difference of mean AUCs of two systems was statistically significant.

In the information extraction portion of the problem, experiments with different models yielded similar results that are comparable to selecting titles as GeneRIFs with a Dice-coefficient performance measure of 50%. A method capable of selecting the best GeneRIF candidate sentence from a document can achieve a Dice score exceeding 70%. A wide range of lexical features and several machine learning algorithms were applied to select candidates, yet the best result selected all candidates from titles and performed only 0.53% better than selecting the first sentence from the title. A deeper linguistic analysis at semantic or discourse level might give better performance; however, the heterogeneity of GeneRIF choices suggests even more elusive pragmatics and/or cognitive modeling may be required to achieve optimal results.

References

Callan, J. P., Croft, W. B., & Harding, S. M. (1992). The INQUERY Retrieval System. Proceedings of the *Third International Conference on Database and Expert Systems Applications*, 78–83.

Heckerman, D., Meek, C., & Cooper, G. F. (1999). A Bayesian Approach to Causal Discovery. In C. Glymour, & G. F. Cooper (Eds.) *Computation, Causation, & Discovery*, 141–165. Menlo Park, CA: AAAI/MIT Press.

Humphrey, S. M., Rindflesch, T. C., & Aronson, A. R. (2000). Automatic Indexing by Discipline and High-Level Categories: Methodology and Potential Applications. Proceedings of the *11th ASIST SIG/CR Classification Research Workshop*, 103–116.

Ishikawa, K., Ando, S., & Okumara, A. (2001). Hybrid Text Summarization Method Based on the TF Method and the LEAD Method. Proceedings of the *Second NTCIR Workshop*.

Kayaalp, M., Pedersen, T., & Bruce, R. (1997). A Statistical Decision Making Method: A Case Study on Prepositional Phrase Attachment. Proceedings of the *1997 Meeting of the ACL SIG in Computational Natural Language Learning* (CoNLL97), 33–42.

McCray, A. T. (1998). The nature of lexical knowledge. *Methods of Information in Medicine*, 37(4-5), 353–360.

McCray, A. T., Ide, N. C., Loane, R. R., & Tse, T. (2004). Strategies for Supporting Consumer Health Information Seeking. Accepted for publication in the proceedings of *MedInfo 2004*.

Muggleton, S. (1995). Inverse Entailment and Progol. *New Generation Computing*. Special Issue on Inductive Logic Programming, 13, 245–286.

Rivest, R. L. (1987). Learning Decision Lists. *Machine Learning*, 2, 229–246.

Srinivasan, A. (1999). The Aleph Manual. Available at http://www.comlab.ox.ac.uk/oucl/research/areas /machlearn/Aleph/.

Tanabe, L., Scherf, U., Smith, L. H., Lee, J. K., Hunter, L., & Weinstein, J. N. (1999). MedMiner: An Internet Text-Mining Tool for Biomedical Information, with Application to Gene Expression Profiling. *BioTechniques*, 27(6), 1210–1217.

Tanabe, L., & Wilbur, W. J. (2002). Tagging Gene and Protein Names in Biomedical Text. *Bioinformatics*, 18, 1124–1132.

Zhang, T., & Oles, F. J. (2001). Text Categorization Based on Regularized Linear Classification Methods. *Information Retrieval*, 4, 5–31.

Finding Gene Function using LitMiner

Berry de Bruijn and Joel Martin

Institute for Information Technology
National Research Council of Canada
berry.debruijn@nrc.gc.ca, joel.martin@nrc.gc.ca

Abstract

NRC (National Research Council, Canada) submitted 2 sets of results for the primary task in the TREC Genome track. The systems that generated these results were tuned primarily to achieve very high recall (above 90%) and secondarily to minimize the number of documents retrieved. Both submitted sets were the outputs of automatic systems (non-interactive, non-supervised) with a modular architecture.

The TREC evaluation confirmed that recall for both submissions was extremely high: 543 out of 566 target documents (0.9594) were returned. In addition, these systems returned far fewer documents than were allowed by the genomic track rules. They returned an average of 196 documents per query across the 50 queries, with a median value of only 100 documents.

For the first submission, the system was entirely based on Information Retrieval techniques, tuned to achieve very high recall and fair precision. Averaged precision was 0.3941 for the first submission. This first submission ranked third out of 49 runs submitted by all participants.

For the second submission, reranking was done based on the outcome of an information extraction module, tuned towards the task of identifying gene function papers. This module identified 539 documents as highly promising; 121 of these turned out to be target documents, 418 weren't. All in all this caused the averaged precision to drop slightly to 0.3771 - contrary to our expectations. This second submission ranked fifth out of all 49 runs.

1. Introduction

Scientists reviewing literature in their field often hope for exhaustive searches that return all the relevant documents. The cost of missing an important document is high, so less than perfect precision is accepted from real-world (less than perfect) systems if close to full recall is still guaranteed. Of course, since the scientist would have to scan every article returned by the search, a system returning 100 results is far better than one returning 1000. This is the type of system we envisioned when taking up the TREC task.

After a genomics 'pre-track' in 2002, a full genomics track was added to the TREC setup for the 2003 edition. This development is in full agreement with the strong attention that Information Extraction from biomedical literature has recently received (see for instance De Bruijn and Martin, 2003). The genomics track presents interesting issues to the text retrieval research community because of the combination of working with large-scale document collections and the specifics of the application field with its esoteric jargon.

The National Research Council (NRC) is the Government of Canada's premier organization for research and development. For a number of years, researchers at its Institute for Information

Technology have been working on language processing technologies, including methods and tools to process biomedical literature. Multiple technologies are now being bundled into an integrated toolbox for literature access and management, named LitMiner.

This paper gives an overview of the architecture that we used for our TREC-genomics submissions, including a discussion on how such a high recall was achieved. It includes a further analysis of those queries where performance was disappointing or under par. Design and performance of the information extraction module are discussed.

2. System

2.1 system architecture

For storage, a MySQL database was used under Linux/Intel586. The documents were stored in one table, and an index on terms to document identifiers (PMIDs) was created in a separate index table. The index contained lists of PMIDs for all non-stop words, along the position of the word in that text. A title word index was included in a separate column in the same table; other columns contain the word frequencies and document frequencies. Also stored in the index table were entries for the complete RN and MeSH terms which can be multiple word terms.

The NRC system finally consisted of 7 modules, working in sequence. The modular architecture is sketched in figure 1. The functions of the various modules is as follows:

[1] *basic retrieval:* database/index lookup of all articles containing one or more of the query gene names verbatim

[2] *morphological term expansion:* based on a number of rules, variations of the original gene names are generated and articles containing these expanded query terms are added to the retrieved set

[3] *acronym disambiguation:* this module removes those articles from the result set where the letter combination clearly has an incorrect meaning, for instance, "MVP" should mean "major vault protein" and not "mitral valve prolapse".

[4] *relevance feedback:* this is done on the RN field of the Medline records; records are added to the retrieved set if they contain RN terms that are relatively 'overrepresented' in the set so far. Such a relevance feedback (RF) loop is done twice.

[5] *organism matching:* since the target organism is given for each query, only articles are returned that actually mention the organism name.

[6] *reranking:* articles are reranked based on the occurrence of more gene aliases or more significant ones; aliases occurring in the title give an extra boost.

[7] *function phrase finding:* (second submission only) - documents that score high on a high-precision 'function phrase finding' test get an additional boost up.

Modules 1,2 and 4 are designed to achieve optimal recall, while modules 3, 5, 6 and 7 are designed to improve precision. The various modules are described in detail in the next section.

2.2 Technical description of each of the modules

2.2.1 query term retrieval

Basic document retrieval / index lookup: retrieves the document identifiers for those documents that contain that word (for single-word terms) or that phrase (for multi-word terms). Matching was case-independent and on full, untruncated terms. Since stop words were excluded from indexing, query terms that double as common English stop words (such as AND) would not pose a problem. Stop words were allowed to be part of query phrases.

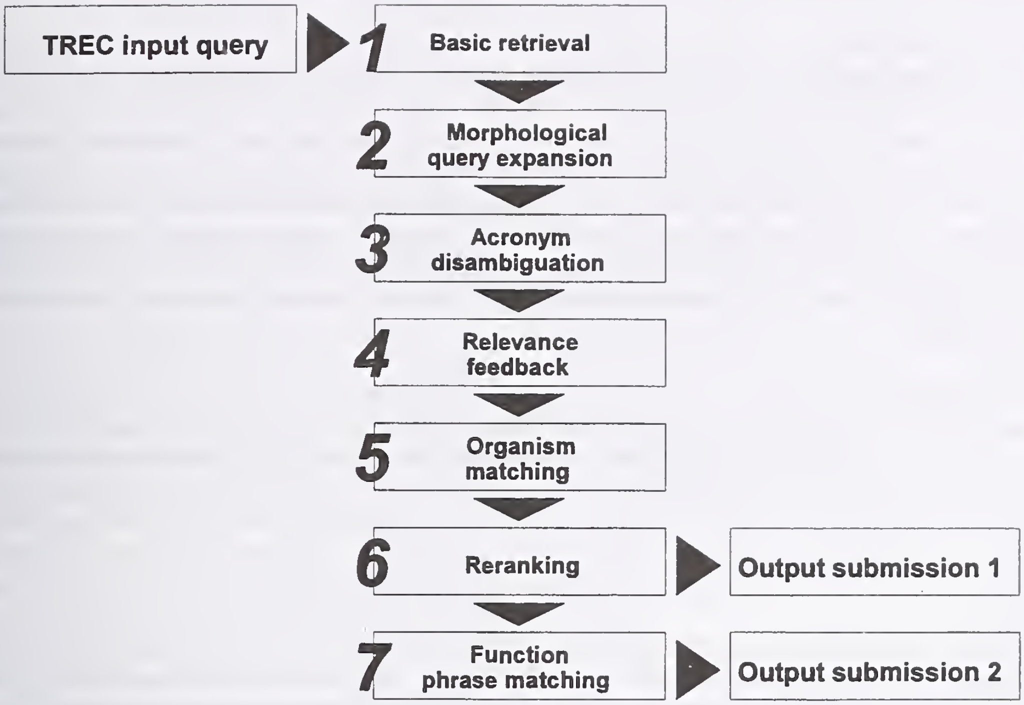


Figure 1: architecture of the NRC retrieval system

2.2.2 morphological term expansion

It was observed by us and our domain experts that authors do not always use the preferred molecule name, or in other words, the list of aliases is not complete in listing all possible molecule names that authors decide to use. Some of the used names are minute variations on one of the official symbol/alias names and can be found by creating a limited number of morphological variations on the official names. This module includes handcrafted rules that make these variations. A complete list of the rules is available from the authors; some examples of morphological variations that this module deals with, are:

'ATF2' could be referred to as 'ATF-2', 'ATF 2', 'ATF-II'
'PPARG' could be referred to as 'PPAR Gamma'
'FGFR' could be referred to as 'FGF receptor'.

2.2.3 acronym disambiguation

Acronyms are fairly likely to be ambiguous. For instance, using the query term "MVP" with the intention retrieve documents on major vault protein will also return irrelevant documents where the same acronym stands for 'mitral valve prolapse', 'microvascular pressure', or 'Midwifery Ventouse Practitioners'. This module disambiguates the meaning by first creating a list of possible meanings for that acronym from Medline documents, in a routine analogous to e.g. Pustejovsky (2001), or Schwartz and Hearst (2003). For each of the meanings in the list, it is determined whether the meaning is correct or incorrect in the context of the query. Then documents under consideration are tested for the presence of any of the possible acronym meanings, and if only an incorrect meaning is found in a document then it is excluded from the retrieved set.

2.2.4 relevance feedback

After retrieval with the query terms, and after disambiguation, an unsupervised relevance feedback step is done (twice). For all documents thus far retrieved, the values for the RN field are retrieved. The RN field in the Medline data lists Registration Numbers to chemicals and substances involved in the study, including Enzyme Commission numbers and names. Then other (so far unretrieved) documents are added to the retrieved set if they contain RN terms that are strongly represented in the retrieved set. Per feedback iteration, the number of terms used was at most 10 and at least five provided that each added term occurred in at least 20% of the previously retrieved documents. The number of feedback iterations was set to two. These parameters were set based on runs with the training material.

2.2.5 organism matching

This step rejects articles where the correct gene might be discussed but where it relates to another organism. For that, a synonym shortlist is used, containing the synonyms for the most frequently researched organisms (human, drosophila, mouse, rat, cow, yeast, zebrafish, e-coli, c. elegans and xenopus). If the organism from the query is not referred to in the retrieved article then that article is discarded.

2.2.6 reranking

A TF*IDF weighting scheme is used to rerank the retrieved articles, so that articles that contain characteristic query terms will end up higher in the results list. Articles that mention multiple query terms get an additional boost. Articles where the title contains query terms get an additional boost as well.

2.2.7 phrase matching and boosting

A phrase matching module was used to identify sentences likely to include descriptions of gene function and boost those documents up in the rankings. The module was trained with a supervised machine learning method on training data, while the test phase took place without supervision. In this method, seed sentences known to be GeneRIFs from the training material were automatically generalized to include similar phrases that appeared in other abstracts. The expansion included identifying words that could be substituted in the sentence and the gradual shortening of the phrase to increase the number of abstracts in which it appeared. The specific algorithm used produced sentence identification with very high precision in the training set. When a sentence or phrase was identified in a title or an abstract, that abstract could be promoted in the results ranking. This act of promotion could lead to large errors. To prevent that, the system further restricted promotion to ensure high precision. This promotion was only used when the phrase was in the title of the article AND clearly contained a reference to the gene.

3. Experiment

3.1 Task

The aim of the competition in the Primary Task of the Genomics track was to retrieve documents that contain a description of the function of a gene, given the gene names and the organism under consideration. Result lists should have the (more) relevant documents at the top and any less/not relevant documents near the tail. Result lists can contain at most 1000 documents per query.

3.2 Material

The document set consisted of 525,938 article abstracts, or one year's worth (2002), from

Medline/PubMed. These are article citations from every kind of biomedical discipline. The abstracts or citations or records contain various fields including title, abstract, author names, affiliation, publication info (joumal name, issue), category terms from the Medical Subjects Headings (MeSH) hierarchy, and RN entries. The RN field contains Registration Numbers and official names of chemical substances, as well as official Enzyme Commision names and numbers of molecules involved in the study.

Fifty queries were supplied in a training set and another fifty in a test set. The test set was kept in isolation away from all system development and all developers (conforming to the competition rules). The training set was useo to tune system parameters and make other performance decisions. No other data source was used (such as LocusLink, which was specifically off-limits, or Gene Ontology), with the exception of a background collection of older Medline abstracts for the Acronym Disambiguation module (more about that in section 3.3). The training material as well as the test material originated from NCBI's LocusLink database, specifically the GeneRIF field (RIF = Reference in Function) and the nomenclature field (for the query terms). Each query contained a (TREC) query number, a LocusLink identifier (not used), the organism, and a number of gene identifiers, including the official symbol name, alias names, and product names.

For the training queries, the answers were available, for the test queries, answers were made available after the submission deadline. The number of target documents per query ranged between few (0) to many (53) in the training set, median=4, average=5.83. In the test set, the range turned out to be 2 to 66, median=7, average=11.32.

3.3 Evaluation metrics

Systems were evaluated with the common Information Retrieval metrics, as well as a metric named Averaged Precision: at each point in the results list where a target document is positioned, the precision is calculated and averaged over the number of target documents. For unretrieved target documents, precision=0.

4. Results and discussion

4.1 Evaluation of the entire system

Submission 1, overall performance: this was the submission as produced with the system without module 7 (the phrase searching / document boosting module). This system returned 9824 documents over the 50 queries; recall was 543 target documents found out of 566 target documents max, or 95.9% recall. Average precision for all relevant documents (averaged over queries) was .3941. The average number of documents returned per query was 196.5, but the distribution over queries was very skewed and the median number was 100 documents per query.

Submission 2, overall performance: this system included the phrase searching / document boosting module. The module identified 539 documents as highly promising, 121 of these were indeed target documents while 418 were not. For this system, the submission-1 results were only re-ranked so the returned set remained 9824 documents with 95.9% recall. Since quite some non-target documents got boosted, the average precision score dipped to .3771.

The results for submissions 1 and 2 are summarized in table 1.

With a median number of 100 documents per query and a recall of nearly 96%, exhaustive (not exhausting) searches are realistic with this system. Scientists often want to find all the relevant documents without having to discard too many false positives. The NRC system successfully restricts the number of documents returned while maintaining a high recall. The NRC system achieved a very competitive score in the TREC competition, falling just short of matching the highest scores from one other participant.

One explanation of the lower score for our second submission, is the sparsity of GeneRIF's. The module used for that submission tried to capture characteristics of function descriptions. But not every statement of a gene's function is currently a GeneRIF. Different systems that find the same document will likely favour different valid statements of a gene's function and will produce different measures of precision based on GeneRIFs. Unlike the estimated measure for recall, an estimate of precision based on a small sample of the true documents will not necessarily be a reliable estimate of the precision across the entire population. Only one in four of the documents that were promoted by the Function Phrase finding module in our second system, were in fact GeneRIF's. That does not mean that three in four are false positives, only that we do not know their true status.

The performance of this phrase matching module was strikingly different between the training and the test data sets. Besides the problem of a sparse gold-standard, it is also possible that the learning method overfitted the training data. Only the GeneRIFs in the training set were used to infer the structure of phrases indicating function. As a result, those structures may have worked differentially well when applied to reranking training documents.

In addition to a solution to the sparsity of the GeneRIFs, we would like to propose another metric to assess the recall. For one example, the average precision could be averaged over the reading cost to give a benefit per unit of effort. Another measure would divide the recall (a measure of benefit), by the number of documents returned (a measure of the cost). Although, the NRC systems would do well on such measures, other systems might also do well by deleting the documents beyond the first 100. This modification would only produce a fair comparison if the system itself is able to determine that the number of documents should be 100.

Table 1: performance of the system and its separate components

System **(or modules included)**	**Retrieved (before capping at 1000)**	**Recall**	**averaged precision**
nrc1 (modules 1-6)	9824 (9837)	543/566 = 95.9%	.3941
nrc2 (modules 1-7)	9824 (9837)	543/566 = 95.9%	.3771
module 1 +6	13975 (25737)	433/566 = 76.5%	.2051
modules 1+2 +6 *(morph. query expansion)*	17466 (34188)	511/566 = 90.3%	.2355
modules 1+4 +6 1 loop relevance feedback 2 loops rf 3 loops rf	 16086 (28612) 17370 (32616) 18540 (34263)	 479/566 = 84.6% 495/566 = 87.5% 498/566 = 88.0%	 .2198 .2220 .2198
modules 1+2+4 +6 (mqe + 2 loops rf)	20813 (40692)	546/566 = 96.5%	.2397
modules 1+2+3+4 +6 (+ acronym disambiguation)	20636 (40185)	546/566 = 96.5%	.2399

4.2 Evaluation of each of the modules

Additional runs were done with some modules switched on and others switched off, to determine the separate contributions of the modules. The results of these runs can also be found in table 1.

- *Basic retrieval* with most modules switched off: only query term retrieval is included, and reranking is applied. This retrieves 25737 articles, but if the retrieved set is capped at 1000 articles per query ('cap-1000', as per TREC guidelines), 13,975 articles remain. Recall is 76.5% (433 target documents out of 566), with .2051 average precision. These measures confirm that basic retrieval alone does not cut it.

- *Morphological query expansion* increases the size of the retrieved set to 34188 articles or 17466 after cap-1000. Recall improves to 90.3% (511 documents), average precision is .2355. The strong increase in documents retrieved indicates that many irrelevant documents are also added. In this stage of the process, where high recall is the main objective, that is less important.

- *Relevance feedback* (RF) applied as an alternative to morphological query expansion comes close to the same scores but not quite. One RF loop retrieves 28612 documents (16086 after cap-1000); recall is 84.6% (479 documents); average precision is .2198. A second loop increases recall to 87.5% (495 documents) with 17370 retrieved (32616 before cap-1000); average precision .2220. A third RF loop does very liitle more: 88.0% recall (498 documents) with 18540 retrieved (34263 before cap-1000) and average precision of .2198. This confirms the decision that was based on the training material that two RF iterations would be the best setting.

- Relevance feedback did complement morphological query expansion. With both modules in place, the recall was near to perfect with 96.5% (546 documents) with 20813 retrieved (40692 before cap-1000); average precision here was .2397.

- *Acronym disambiguation* did not help much on the test material. While this module did oust 507 documents compared to the previous setting, with no incorrect discardings so no loss in recall, most of these ousted documents were at the tail of the document rankings anyway and wouldn't have survived the cap-1000 step, or wouldn't harm the average precision. While the overall improvement was small, this module did prove very useful in a few cases - including some cases in the training material. Since it this module is intuitively appealing and wasn't hurting performance we decided to leave it in.

- *Organism matching* proved a powerful method to drastically limit the number of retrieved items while doing very little harm to recall. This module cut the size of the retrieved set in four - from 40185 to 9837 documents. Among the discarded abstracts were a mere 10 target documents. After cap-1000, the final result set was 9824 with no further loss of precision. This step gives the average precision a terrific boost: from .2399 without the module to .3941 with the module. Of the ten discarded target documents, eight mention a different organism than the target organism and make a dubious or poor GeneRIF. One more document lists a broader organism category ('mammalian') rather than the target document (human). Finally, one article did not list an organism, but the Medline entry for that article has been updated after the TREC data was collected, and currently does list 'human' in the MeSH field.

4.3 Failure analysis

In the following section, we analyse a number of queries where our system returned poor results.

Query 4: *"guanine nucleotide binding protein (G protein), alpha activating activity polypeptide, olfactory type"*. The system retrieved 984 documents with both target documents retrieved, in ranks

18 and 95 (submission 1), or 54 and 125 (submission 2). The results set after module 1 had 2 documents, including one of the two target documents. The morphological query expansion module included a rule that would cause the phrase “G-protein” to be used as a query term in itself, while this would be too general a query term to be appropriate in this case. A system with module 2 disabled would have retrieved five documents with total recall and .5 average precision on this query. Disabling the single rule that caused the over-generalization showed harmful to the performance on other queries.

Query 22: *“arginine vasopressin”.* This is a fairly frequently researched molecule and many top retrieved articles seem on-topic and relevant, albeit not strictly target documents. This might be a case where the evaluation metrics undervalue the results.

Query 41: *“CASP8 and FADD-like apoptosis regulator”.* This query includes three query terms that could double as common English words: ‘FLAME’, ‘FLIP’ and ‘CASH’. These terms contaminate the result set and even though 100% recall was achieved, precision suffered dramatically. An added rule that would prohibit common English words from being used as query terms would not help: on that query, recall dropped to an unacceptable 0.1 (1 retrieved, 10 targets). Allowing English words as long as capitalization of that word is uncommon (a capital letter somewhere after the first letter of the word) did reduce the number of retrieved documents from 323 to 101, kept the recall on 10/10 and raised the average precision from .1502 to .2972. That rule, however, caused a severe degradation on two other queries from the test set (26 and 27) and on one in the training material (14). With no major improvements on other queries, it would give an overall worse performance.

Query 49: *“T-cell receptor alpha chain”.* This query gave a failure of a different kind: even though a limited number of articles was retrieved (80) and two target documents were returned in positions 4 and 12 of the result list, five more target documents were not retrieved. That made this query the only one where recall for our system was <50% and it accounted for 5 out of 23 missed target documents over the entire task.

These failure analyses and other observations gave us no reason to drastically revise the overall architecture or details of the design of the system.

5. Conclusions

The LitMiner system used for the primary TREC-genomics task, was almost entirely based on established Information Retrieval techniques. It performed very well in absolute terms as well as in comparison with other systems in this TREC track. The two submissions from LitMiner ranked third and fifth out of 49 submitted runs by all participants.

We designed our system in such a way that an early stage would return a limited document set with high recall and fair precision. The second stage would be allowed to be more computationally expensive if it would be capable of increasing precision while retaining the high recall. We must admit that the second stage performed less well than we expected. This has to do with the slightly vaguer definition of what a GeneRIF would stand for. Future work will explore the use of additional training examples and hopefully more effective information extraction.

On the other hand, the first stages succeeded very well in getting the high-recall results while limiting the size of the result set. In fact, searching literature with a >95% recall in a typical result set size of a mere 100 documents provides a very practical tool for biologists.

While interactive systems are planned to be part of future TREC-genomics tasks, the setting of this year's task was a static environment. It differs from the regular, interactive search environment that characterizes our search toolbox LitMiner. Supervised relevance feedback techniques and result navigation tools are likely to add power to the current design and allow the user to quickly home in on the desired results.

Acknowledgments

We would like to thank Brandon Smith, Roy Walker, Hung Fang, Maria Moreno, and Annie Law from the NRC Institute for Biological Sciences, as well as Lynn Wei and Jeffrey Demaine (NRC-IIT), for their input during our joint meetings and on-line.

References

De Bruijn B and Martin J: Getting to the (c)ore of knowledge: mining biomedical literature. Int J Med Inf 2002 Dec;67(1-3):7-18.

Pustejovsky J, Castano J, Cochran B, Kotecki M, Morrell M: Automatic extraction of acronym-meaning pairs from MEDLINE databases. Medinfo 2001;10(Pt 1):371-5.

Schwartz AS, and Hearst MA: A simple algorithm for identifying abbreviation definitions in biomedical text. Pac. Symp. Biocomput. 2003; 451-462.

Experiments in TREC-2003 Genomics Track at NTT

Hirotoshi Taira, Tomonori Izumitani, Tsutomu Hirao,
Hideki Isozaki, Hideto Kazawa, Eisaku Maeda
NTT Communication Science Laboratories
2-4, Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0237, Japan
{taira,izumi,hirao,isozaki,kazawa,maeda}@cslab.kecl.ntt.co.jp

In TREC-2003, we participated in Question Answering and Genomics Tracks. Since the QA system was essentially the same as the past years' systems[1, 2], we describe our results with the Genomics Track in this paper.

1 Genomics Track Primary Task

Our system consists of two steps. The first step retrieves documents using a keyword search, and the second step scores each document retrieved in the previous step and creates an output file for the TREC submission.

The database provided by TREC consists of more than 500,000 PubMed abstracts. However, less than 50 documents are relevant for most queries. Applying scoring methods to all 500,000 abstracts would create a lot of noise. In the first step, we refined the document set with a simple keyword search.

For the second step, we developed two methods. The first method (Method 1) uses a heuristic scoring system that simply counts the number of verbs and their derived words, which are important to specify the function of a query gene or its product. The second method (Method 2) uses a machine learning technique to score documents.

1.1 Method

1.1.1 Document Retrieval using Keyword Search

TREC provides categories for each query. Namely, official/alias gene/product names, symbols and species. Although species are not necessarily described on documents, "names" or "symbols" should be written in relevant documents. We retrieved all documents that include at least one "name" or "symbol" for each query. They are scored in the next step.

Symbols are represented in various ways in various documents. For example:

- An alias symbol between parentheses follows an official name, such as "p21(Cip)".
- Some symbols are connected by slashes, such as "p21/Waf1/Cip1/Sdi1".
- A combination of the above two cases, such as "p21(WAF/CIP1)".

Additionally, symbols could be written by uppercase characters, lowercase characters or a mixture of both. In this step, we searched for symbols between spaces or marks, such as '-', '/', '(' or ')', without distinction between uppercase and lowercase characters.

8,538 documents for TREC training queries and 18,084 documents for TREC test queries were retrieved in this step.

1.1.2 Method 1 : Heuristic Scoring System

In the previous step, documents that could be relevant to each query gene were obtained. The problem is whether the documents refer to the function of the query gene or a product of it. In this step, all of the retrieved documents are scored for this purpose.

¿From the analysis of all relevant documents for the TREC training data, we found that common verbs or their derived words, such as "express", "bind" or "inhibition", are often used to describe functions of genes. These words are located adjacent to keywords (query names or symbols). We manually extracted 97 kinds of verbs or their derived words from the vicinity of keywords. We, then, generated a list of words that includes their inflected forms and derived words. Here, we call these words "function words". The list of function words consists of 595 words. The following are parts of this list.

```
bind binds binding bound
control controls controlling controlled
express expresses expressing expressed
expression expressions
indicate indicates indicating indicated
indication indications indicator indicators
...
```

To score each document retrieved in the previous step, a set of words is made using five words before and after the keywords. Then, the score is simply calculated by counting the number of "function words" in the list, allowing for duplication.

1.1.3 Method 2 : Scoring System using SVM

In Method 1, important information for scoring might be lost because of its simplicity and heuristics. We adopted a machine learning techniques to automatically reflect such information to the scoring system.

Machine learning methods such as the Perceptron or Support Vector Machine (SVM) generate discriminant functions whose inputs are mainly vectors and whose outputs are real values. While these methods are usually used as classifiers that output the sign of the discriminant functions, many applications adopt the real value outputs of discriminant functions as confident scores. In this task, we use this value and the SVM as a machine learning method.

Making Vectors from Documents

Representing each document by vector is necessary to make inputs of an SVM [1]. We used the classical "bag of words" model for vectors.

To make vectors, all five words before and after keyword query gene names or symbols are extracted, as well as Method 1. All words except stopwords are used as features of vectors. To decide the values for these vectors, we tried some weighting methods such as TFIDF (term frequency inverse document frequency) and TF in addition to simple binary vectors. However, these weighting methods did not improve the performance. We therefore used binary vectors for all experiments.

Feature Selection

All features of high dimensional vectors are not always effective for discriminant functions. Some features appear in very few documents or have no information for discrimination. The features satisfying the following conditions are eliminated.

- The document frequency is less than Θ_{min}.
- The ratio of positive (relevant) documents to negative (irrelevant) documents is less than Θ_{ratio}.

1.2 Experiment for TREC Training Set

The 8,538 retrieved documents included 233 relevant documents that are 78.5 % of 297 documents provided by TREC[2].

We evaluated Methods 1 and 2 using this data. We divided the data into two sets to create the training and test data for Method 2. The first set is made from queries 1 to 25 and the second is made from the rest. We call the former "Set1" and the latter "Set2". Documents corresponding to queries 21, 35 and 49 were eliminated because they do not include any relevant documents. Set1 consists of 4,675 documents, Set2 consists of 3,560 documents. Set1 and Set2 are used for training and testing, respectively, in Method 2. For Method 2, 12,494 features were extracted from the 4,675 training data.

Table 1 shows the results of Method 1. The method was applied to Set1, Set2 and the whole TREC training set independently, because Method 1 does not need training. The evaluation was performed by mean average precision (MAP) using the "trec_eval"program.

[1] Recently, some methods that calculate values of the discriminant functions directly from character strings or more complicated structures have been developed using kernel methods.

[2] 38 documents in "training-qrels.txt" are not included in the Medline database file, "medline.txt".

Table 1: Mean average precisions of Method 1

Data set	MAP
Set1 (4,675 docs.)	0.250
Set2 (3,560 docs.)	0.322
Whole TREC training set	0.285

Table 2: Mean average precisions of Method 2

Data set	MAP
Set1 (Training set)	0.610
Set2 (Testing set)	0.323
Whole TREC training data	0.573

In Method 2, two kernels, the first and second order polynomial kernels, were applied and various kinds of parameters were examined, namely, Θ_{min} and Θ_{ratio} for feature selection and the SVM soft margin parameter (C). The best parameters, $\Theta_{min} = 2$, $\Theta_{ratio} = 4$ and $C = 0.01$, were decided by comparing the mean average precision of Set2.

Table 2 shows the results of Method 2. The result for the whole TREC training set was calculated using the TREC training set for SVM training. Set1 and the whole TREC training set have a much higher mean average precision since they are also used for training. Therefore, only the result of Set2 may be an estimation of the TREC test. Methods 1 and 2 yield almost even performances, even though Method 1 utilizes only 595 words in contrast with more than 10,000 words by Method 2. This indicates that verbs and their derived words are crucially important to specify documents that describe the functions of genes or their products.

1.3 Results for Test Set and Discussion

In the first step, 18,084 documents were extracted from the test queries provided by TREC. We applied both Methods 1 and 2 to this data and made two files for submission. Dummy PubMed IDs were filled for queries 7 and 26 because no documents were retrieved in the first step.

Table 3: Mean average precisions for the TREC test set

Method 1	Method 2	Best	Median
0.148	0.153	0.567	0.212

TREC returned average precision scores for each query. The scores of the best, median and worst system were also provided for each query. Table 3 shows the mean average precisions of Method 1 and Method 2 compared with the best and median systems submitted to TREC. The results for Method 1 and Method 2 are almost even, which is consistent with the evaluation for the training set (Subsection 1.2). However, both methods have a little worse than mean average precision.

Figure 1 shows the distribution of the average precisions of Method 1 and Method 2 compared to the best and median systems submitted to TREC. The horizontal axis denotes the average precision and the vertical axis denotes the number of queries. The best scores are significantly high because they do not necessarily come from only one system. The score of the median systems could be a good indicator for average systems. Although the form of distributions are similar among Method 1, Method 2 and the median systems, Methods 1 and 2 have too many low scores less than 0.2. Actually, Method 1 has nine queries whose average precisions are zero and Method 2 has seven queries, of which eight queries are the same for both methods.

This comes from the fact that very few documents were retrieved in the first step. For seven zero score queries, only less than ten documents were retrieved in the first step. Extending queries for the first step considering variations of the description of the gene names, or integrated scoring systems that consider whether a given document describes a query gene and its function simultaneously, will be necessary to improve the performance of our system.

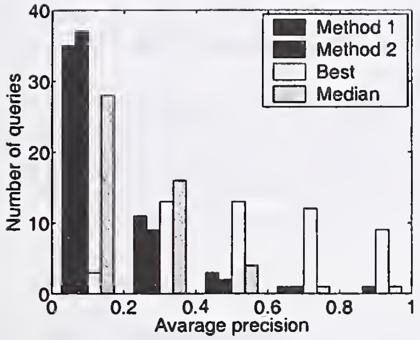


Figure 1: Comparison of Method 1, Method 2, the best systems and the median systems

2 Secondary Track : Automatic Functional Phrases Extraction

We extracted the sequence patterns of the characteristic words (more correctly, the characteristic stems) in the sentences described the gene functions in the training data, in order to generate automatically the phrase that describes the function of a gene.

Next, we scored the test sentences using the information criteria of the sequence patterns.

Last, we output the sentence with the highest score as the phrase explaining the gene's function.

2.1 Labeling Positive and Negative Labels to Training Set

First, as preparation for calculating the information criteria, we gave positive or negative labels to the training sentences according to whether their sentences are close to a correct answer or not. After we divided articles into sentences by our sentence boundary detector, we selected sentences with a small Edit Distance to the actual GeneRIF used as correct answers out of the training set. We gave these positive labels and gave the others negative labels.

More precisely, we labeled sentences whose Edit Distances were 30 % or less than the length of the GeneRIF and the sentence with the smallest Edit Distance as positive. We labeled the other sentences as negative.

2.2 Specification of Gene Name

The information about whether a sentence includes gene names is important for judging whether the sentence includes descriptions about a target gene. We, therefore, replaced the query gene name to "<QUERY_GENE>" tag, and the other gene names to "<SUBSTANCE>" tag.

Although various methods for extracting gene names have already been proposed, these methods need a lot of training data. Therefore, we used the following techniques.

We used gene names and abbreviated gene names registered in the LocusLink and GOA database [3] for searching gene names.

Moreover, we applied the following experiential rules to determine gene names and abbreviated gene names.

- words that are constructed from 3 to 8 characters and are not DNA base pair sequences.

Next, we detected word sequences not satisfied with the following condition in the word sequences that begin with 'the' and end with '(consonant)+ase', '(consonant)+in', '-tor' or '-ssor' as gene names, except for the following case.

- containing Stopwords (Stopwords at PubMed[4] and our original stopwords).
- containing '-ing', '-ed', '.', ';', etc.
- containing only one parenthesis, '(' or ')'.

2.3 Stemming Process

Pattern extraction is possible also from the surface word sequence; however, in the case of, for example, "inhibition of A" and " inhibitor A", these phrases will be treated as different phrases.

In order to avoid this, we extracted stem patterns after stemming to the word using the Porter stemmer [7].

For example, the following sentence,

[3] http://www.ebi.ac.uk/GOA/

[4] http://www.ncbi.nlm.nih.gov/entrez/query/static/help/pmhelp.html#Stopwords

Regulation of Fas-associated death domain interactions by the death effector domain identified by a modified reverse two-hybrid screen.

is stemmed to the following stem sequence.

<regul> <fas-associ> <death> <domain> <interact> <SUBSTANCE> <domain> <identifi> <modifi> <revers> <two-hybrid> <screen>

2.4 Pattern Extraction with Tidal PrefixSpan

We utilized a hyper geometry distribution score (hgs) for extracting stem sequence patterns that appear exclusively to positive examples.

Hisamitsu et al. [3] have proposed a method of weighting words by which the given document set is characterized using an hgs. They showed that words selected by the hgs are effective for standing for the contents of artciles compared with TF-IDF, etc.

Here, a definition of the score using this supergeometry distribution (hgs) is the probability that more than y samples are positive, when x samples are taken without duplication out of the sample set of n containing positive samples of m.
We used $-$ log (hgs) as statistical criteria.

We extracted patterns using the Tidal PrefixSpan [4] for improvement of speed. PrefixSpan [6, 5] is a high-speed extraction method that can extract high-frequency appearance patterns allowed skips that was proposed by Pei et al.

For example, from the following sentences,

1. I should point out that we need ...
2. I must point out that it is important ...,

PrefixSpan can extract the pattern

"I"-"point"-"out"-"that"

at a high speed.

However, since the original PrefixSpan only takes out high frequency patterns, it is necessary for it to be devised to take out the pattern with high information criteria. Here, we can utilize Tidal SMP (Tidal Statistical Metric Pruning) [8]. Tidal SMP is a technique to accelerate counting the number of patterns with an information criteria.

We used Tidal PrefixSpan, which is a technique of applied Tidal SMP to PrefixSpan, for finding significant patterns with statistically meaning. We used the value of $-\log(hgs)$ divided by the pattern length ($= 1, 2, 3, \ldots$) as statistical criteria and scoring points.

2.5 Functional Phrase Output

We scored all the sentences that included test articles by summing up stem pattern scores. Next, we extracted the sentence with a high score for every part (title, abstract, body and caption parts) of the article. Then, we finally selected the output sentence from four sentences by rescoring with weight. Output sentences are basicaly one sentence. If the sentence was long, we outputed a head part of less than 256 characters of the sentence.

2.6 Experimental Result

We scored patterns with a length of three or less and a frequency of two or more in the training data. We then extracted the top 800 patterns with high hgs values using the Tidal PrefixSpan.

Stem patterns that appear two or more times extracted by Tidal PrefixSpan are shown in Table 4.

<crystallin> (crystallin), <len> (lens), etc., which seldom generally appear, were extracted from the training set. This is because patterns with low frequency may often get a high value of ($-\log(hgs)$ / pattern length).

We show the patterns extracted with a higher rank in Table 5 that appear 100 or more times. This indicates that our method can extract patterns that are likely to appear also in the test data and which are generalized. This shows that the generalized patterns can be extracted with the combination of the cut-off point by frequency and the value using the hyper geometry distribution.

We evaluated the output results by four improvement Dice coefficients. By average of a total of 139 questions, their scores are CD (Classical Dice) : 48.78%, MUD (Modified Unigram Dice) : 50.39%, BD (Bigram Dice) : 31.49% and BP (Bigram Phrase) : 33.79%.

Part of the concrete results is shown in Table 6. This is the result of the higher 1, 5, 10, 50 and 100 ranked when

Table 4: Extracted Stem Patterns (higher 30 pattern, existing more than one frequency).

Pattern	Pos. frq.	Neg. freq.	$-\log(hgs)$ / pattern length
<crystallin>	13	28	44.40
<regul>	31	1095	29.25
<crystallin> <gene>	13	8	27.85
<len>	7	25	21.15
<crystallin> <express>	10	7	21.15
<human>	27	1264	19.45
<signal>	26	1180	19.37
<gene>	33	1887	18.76
<QUERY_GENE>	50	3818	18.71
<SUBSTANCE> <crystallin>	9	10	17.75
<crystallin> <gene> <express>	10	4	15.08
<pathwai>	19	826	15.01
<regul> <SUBSTANCE>	22	511	14.34
<recognit>	7	83	14.09
<SUBSTANCE>	135	18437	13.99
<suggest>	20	1022	13.29
<suffici>	8	139	13.20
<conclud>	6	61	13.08
<gene> <len>	5	0	13.00
<express>	46	4081	12.86
<SUBSTANCE> <crystallin> <gene>	8	4	11.82
<moieti>	4	19	11.78
<co-activ>	5	46	11.53
<pyrophosph>	4	21	11.43
<crystallin> <crystallin>	5	3	10.99
<gtp-bound>	4	27	10.55
<necessari> <gtp-bound>	4	0	10.39
<level> <crystallin>	4	0	10.39
<human> <moieti>	4	0	10.39
<gene> <crystallin>	4	0	10.39

sorting with the results of the Classic Dice coefficient in 139 questions.

Even if the output is apparently close to the correct answer, for example, the 50th problem, a low score can be obtained, because predicted phrases are evaluated only until bi-gram.

These evaluation methods are also a future work.

3 Conclusion

In this paper, we showed characteristic word sequences allowed skips are effective for extracting sentences that described the function of genes in medical documents and showed that scoring by the characteristic word sequence that allows the skip is effective.

Moreover, we showed that the characteristic word sequence that allows the skip can be extracted by Tidal PrefixSpan at a high speed.

Concerning the secondary track, improvement of the evaluation method is greatly required for grasping the

Table 5: Extracted Stem Patterns (higher 30 pattern, existing 100 or more than frequency).

Pattern	Pos. frq.	Neg. freq.	$-\log(hgs)$ / pattern length
<regul>	31	1095	29.25
<human>	27	1264	19.45
<signal>	26	1180	19.37
<gene>	33	1887	18.76
<QUERYGENE>	50	3818	18.71
<pathwai>	19	826	15.01
<regul> <SUBSTANCE>	22	511	14.34
<SUBSTANCE>	135	18437	13.99
<suggest>	20	1022	13.29
<suffici>	8	139	13.20
<express>	46	4081	12.86
<evid>	9	273	10.35
<function>	17	988	9.86
<gene> <express>	16	419	9.73
<regul> <cell>	12	208	9.60
<role>	15	835	9.30
<provid>	9	321	9.15
<transcript>	19	1267	9.09
<drosophila>	6	147	8.39
<necessari>	6	153	8.18
<novel>	6	153	8.18
<cancer>	8	294	8.07
<interact>	17	1177	7.82
<modul>	7	238	7.65
<taken>	5	110	7.64
<SUBSTANCE> <regul>	15	501	7.62
<SUBSTANCE> <SUBSTANCE>	82	8888	7.56
<essenti>	7	244	7.51
<SUBSTANCE> <express>	30	1901	7.45
<high>	9	408	7.43

deeper meaning of sentences.

References

[1] H. Kazawa, H. Isozaki and E. Maeda. NTT question answering system in TREC 2001. *Proc. of The Tenth Text REtrieval Conference (TREC2001)*, 2001.

[2] H. Kazawa, T. Hirao, H. Isozaki and E. Maeda. A machine learning approach for QA and novelty tracks: NTT system description. *Proc. of The Eleventh Text REtrieval Conference (TREC2002)*, 2002.

[3] T. Hisamitsu and Y. Niwa. Topic word selection using a method of word weighting based on conbinatorial probability. *IPSJ SIG-NL, 2000-NL-140 (in Japanese)*, pages 85–90, 2002.

[4] H. Isozaki, T. Hirao and J. Suzuki. On selection criteria of combinatorial features for machine learning. *IPSJ SIG-NL, 2003-NL-158 (in Japanese)*, 2003.

[5] T. Kudo, K. Yamamoto, Y. Tsuboi and Y. Matsumoto. Text mining using linguistic information. *IPSJ SIG-NLP, 2002-NL-148 (in Japanese)*, pages 65–72, 2002.

Table 6: Extracted higher 1, 5, 10, 50, 100th rank phrases.

Rank 1st(TREC ID 9 : **CD:100.00%, MUD:100.00%, BD:100.00%, BP:100.00%**)
Answer : Regulation of intracellular pH mediates Bax activation in HeLa cells treated with staurosporine or tumor necrosis factor-alpha
Predicted : Regulation of Intracellular pH Mediates Bax Activation in HeLa Cells Treated with Staurosporine or Tumor Necrosis Factor-alpha

Rank 5th (TREC ID 10 : **CD:100.00%, MUD:100.00%, BD:100.00%, BP:100.00%**
Answer : Apocytochrome c blocks caspase-9 activation and Bax induced apoptosis
Predicted : Apocytochrome c Blocks Caspase-9 Activation and Bax-induced Apoptosis

Rank 10th (TREC ID 90 : **CD:96.55%, MUD:95.24%, BD:94.74%, BP:92.31%**
Answer : Activity in the nucleus accumbens shell controls gating of behavioral responses to emotional stimuli.
Predicted : CREB activity in the nucleus accumbens shell controls gating of behavioral responses to emotional stimuli

Rank 50th (TREC ID 69**CD:51.28%, MUD:48.00%, BD:17.39%, BP:14.29%**
Answer : there is a mechanically coupled transcriptional circuit that promotes binding of p38 to Sp1 in the nucleus
Predicted : Interaction of p38 and Sp1 in a Mechanical Force-induced, beta1 Integrin-mediated Transcriptional Circuit That Regulates the Actin-binding Protein Filamin-A

Rank 100th (TREC ID 33 :**CD:31.82%, MUD:32.26%, BD:13.79%, BP:23.53%**
Answer : the JH2 domain contributes to both the uninduced and ligand-induced Jak-receptor complex, where it acts as a cytokine-inducible switch to regulate signal transduction
'redicted : The Pseudokinase Domain Is Required for Suppression of Basal Activity of Jak2 and Jak3 Tyrosine Kinases and for Cytokine-inducible Activation of Signal Transduction

[6] J. Pei, J. Han, B. Mortazavi-Asl, and H. Pinto. Prefixspan: Mining sequential patterns efficiency by prefix-projected pattern growth. In *Proc. of the International Conference on Data Engineering (ICDE)*, pages 215–224, 2001.

[7] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[8] J. Sese and S. Morishita. Answering the most correlated n association rules efficiently. In *Proc. of the 6th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)*, pages 410–422, 2002.

SVM Approach to GeneRIF Annotation

Wen-Juan Hou, Chun-Yuan Teng,
Chih Lee and Hsin-Hsi Chen

Department of Computer Science and Information Engineering
National Taiwan University
Taipei, Taiwan

E-mail: {wjhou, cydeng, clee}@nlg.csie.ntu.edu.tw;

hh_chen@csie.ntu.edu.tw

Abstract

In the biological domain, to extract the newly discovered functional features from massive literature is a major challenging issue. To automatically annotate GeneRIF in a new literature is the main goal in this paper. We try to find function words and introducers in the training corpus, and then apply such informative words to annotate the GeneRIF. The experiments showed that 48.15%, 49.78%, 32.31%, and 35.63% for the measure of Classic Dice, Modified unigram Dice, Modified bigram Dice, and Modified bigram Dice phrases. After applying SVM learning mechanism combing new weighting scheme and position information, we get much better performance.

1 Introduction

Information explosion in molecular biology and biomedicine is evolving rapidly, and becomes one of challenging problems in the new information era. How to obtain relevant information, for example, gene/protein functions, from a large amount of data collection is indispensable for bioinformatics researchers and experts. In the past, researchers in biomedicine have already constructed large scale of databases such as UMLS [1], Gene Ontology [2], SwissProt [3], GenBank [4], DIP [5], SNOMED [6], and LocusLink [7] *etc.*, which are useful for researches to capture and organize information. However, creating and maintaining the knowledge bases requires enormous work. For example, if the paper includes a sentence like "*probably exist a binding between gene x and gene y*", we cannot assert that the paper is related to the molecular function. Thus, it needs careful judgment to add new information into a knowledge base. In other words, if we want to retrieve the relevant data from the massive literatures automatically, it needs a lot of efforts.

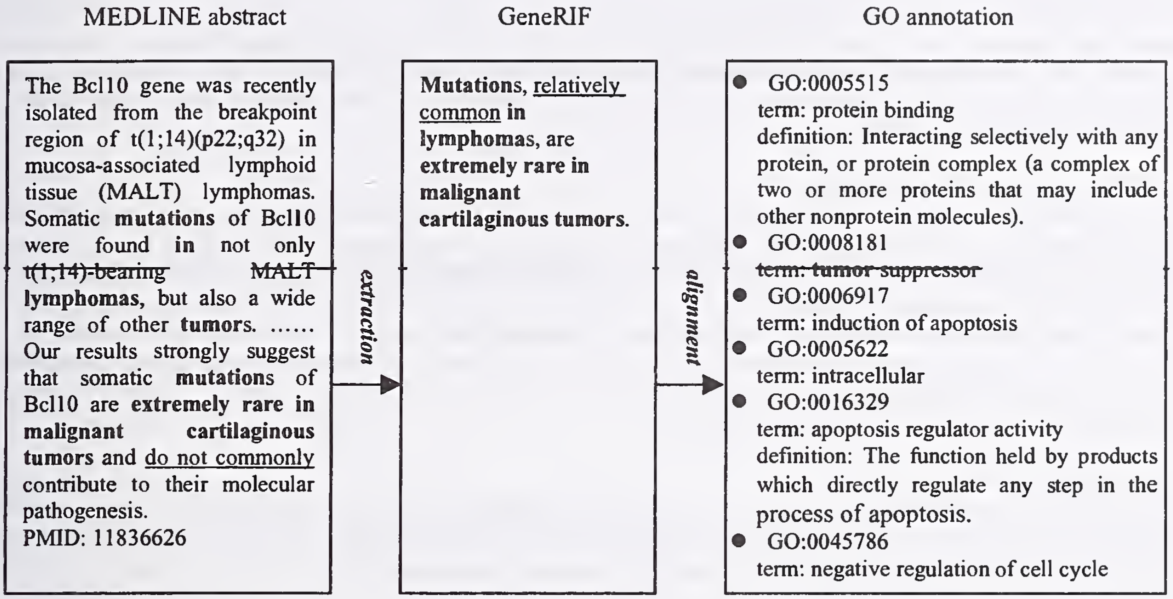
MEDLINE is a massive biomedical corpus for information extraction and knowledge discovery. Biomedical experts explore new development of some special topics by retrieving relevant documents from MEDLINE through search engines or information retrieval (IR) systems. These systems only return documents satisfying users' information needs instead of locating the relevant sentences denoting the specific functions. For example, during exploring molecular functions, users have to go through the whole documents to find the relevant information, and align it to a suitable database entry. To solve the above problem, some efforts have been made to extract functional relations [8, 9, 10, 11, 12]. Those only extract protein or gene interactions rather than the whole functions in the text.

This paper investigates how to extract molecular functions from the literatures. More precisely, the particular goal is to reproduce the GeneRIF annotation as stated in the secondary task of TREC 2003 Genome Track [13]. The Gene References into Function (GeneRIF) exists in LocusLink database [14] and it provides a simple mechanism to allow scientists to add functional annotation of loci. The rest of the paper is organized as follows. Section 2 presents the architecture of our extracting procedure. The basic idea and the experimental methods in this study are introduced in Section 3. Section 4 shows the results and makes some discussions. Finally, Section 5 concludes the remarks and lists some future works.

2 Architecture and the Extracting Method

2.1 Background

Generally, a gene name may have several aliases, and different functions may be discovered in different literatures. A complete annotation system consists of two major stages, including extraction of molecular function for a gene from a literature and alignment of this function with a GO identifier. Figure 1 shows an example. The left part is an MEDLINE abstract with the function description highlighted. The middle part is the corresponding GeneRIF, which is extracted from the last sentence of the abstract. The matching words are in bold, and the similar words are underlined. The right part is the GO annotation. This figure shows the feasibility of maintaining the knowledge bases and ontology using natural language processing technology. To complete this annotating procedure, we have to deal with the first stage automatically since the coverage of GeneRIF records in LocusLink depends on human experts and it cannot come up with the speedy growth of the literatures.



The Bcl10 gene was recently isolated from the breakpoint region of t(1;14)(p22;q32) in mucosa-associated lymphoid tissue (MALT) lymphomas. Somatic **mutations** of Bcl10 were found **in** not only ~~t(1;14)-bearing MALT~~ **lymphomas**, but also a wide range of other **tumors**. ...... Our results strongly suggest that somatic **mutations** of Bcl10 are **extremely rare in malignant cartilaginous tumors** and do not commonly contribute to their molecular pathogenesis.
PMID: 11836626

Mutations, relatively common **in lymphomas**, are **extremely rare in malignant cartilaginous tumors**.

- ● GO:0005515
  term: protein binding
  definition: Interacting selectively with any protein, or protein complex (a complex of two or more proteins that may include other nonprotein molecules).
- ● GO:0008181
  ~~term: tumor suppressor~~
- ● GO:0006917
  term: induction of apoptosis
- ● GO:0005622
  term: intracellular
- ● GO:0016329
  term: apoptosis regulator activity
  definition: The function held by products which directly regulate any step in the process of apoptosis.
- ● GO:0045786
  term: negative regulation of cell cycle

Figure. 1. An Example of Complete Annotation from the Literature GO

A GeneRIF contains a few sentences that describe the function introduced in the scientific document identified by PMID. But how could we recognize the sentence exactly contains such information? We introduce two cues in this paper: function words and introducers. The details will be explained in Section 3.

2.2 Overall architecture

The overall architecture of the extraction from Medline to candidate GeneRIF is shown in Figure 2.

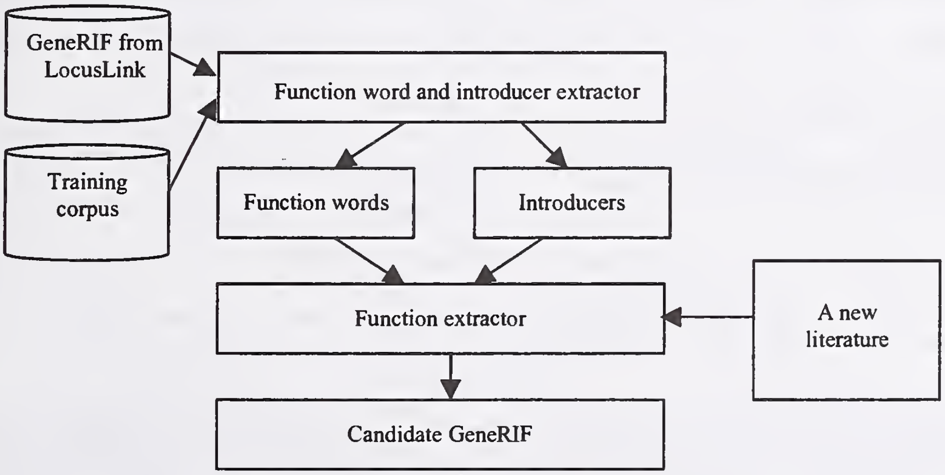


Figure 2. Architecture of Extracting Candidate GeneRIF

For getting the informative words, i.e., function words and introducers in this paper, from training data, we gather GeneRIF from LocusLink. Those are mutually exclusive with testing data in Genome Task and our testing data. And then, the system will compute the score that functions words and introducers contributed to. After applying the function extraction algorithm, the candidate GeneRIF is generated.

3 Function Extraction Approach

As described in Section 2, the score for each sentence depends on function words and introducers. The key issue is how to get function words and introducers and how to measure such scores. First, we prepare the training data and testing data, including those GeneRIFs existed in LocusLink and the corresponding Medline abstracts. We divided the corpus into three parts: training corpus, testing corpus, and testing topics for TREC 2003. The training corpus included 27,236 abstracts and the testing corpus including 9,005 abstracts. The details of our function extraction approach are illustrated as follows.

3.1 Training material preparation

Since GeneRIFs are written by human experts, some parts may include opinions of humans and/or some parts may be cited from papers. We focus on the latter parts. GeneRIF is not directly cut from papers but it has some relationship with paper content. Because the goal is to reproduce GeneRIF automatically, we find the most similar sentence with the corresponding GeneRIF in this paper. The measure is achieved by matching stemmed words between GeneRIF and each sentence. The sentence of matching the most number of words with GeneRIF is selected as the training data for the next stage. However, if more than one sentence matches the most number of words with GeneRIF, this abstract will be aborted because we cannot tell out which is correct. In this way, we get 27,236 sentences extracted from Medline abstracts.

3.2 Function words extraction

We call the matched words between GeneRIF and the selected sentence as *function words* in this paper. Function words form the favorite vocabularies that human experts used to describe the gene functions. Applying stopped word removal and stemming procedure, there are 22,275 function words extracted.

3.3 Introducers extraction

In the training data, there exists some important information except function words and we call it as the *introducer*. Function words are those words that human experts usually adopt to describe gene functions while introducers are the words that often co-occur with function words. In our approach, introducers are words appearing in the selected training sentence but not appearing in the other parts of the abstracts. Under such constraints, we get 621 introducers.

3.4 Compute the weight for each function words, weight(w_i)

Let $|w_{i:}|$ denote the frequency of the function word $w_{i:}$ in the training corpus. Then, weight(w_i)=$|w_i|/\sum_{1}^{n}|w_j|$, where n is the total number of function words. In this way, we can give the weight for each function words extracted in Section 3.2.

3.5 Compute the score for each sentence in the testing abstract

For the testing abstract, we compute two scores for each sentence using the weight defined in Section 3.4. The first score of sentence k, Score(S_k), is shown as follows.

Score(S_k)= $\sum \text{weight}(w_j)$, where w_j appear both in S_k and the set of function words.

To avoid the preference for the long sentence, we normalize score of sentence k, Score(Norm(S_k)), by the sentence length. The second score is defined as follows.

Score(Norm(S_k))= Score(S_k)/$|S_k|$, where $|S_k|$ is the total number of words where stop words have been removed from sentence k.

3.6 Function extraction algorithm

When a new literature comes in, we use the function extraction algorithm to annotate the candidate GeneRIF in the literature. This algorithm employs function words and introducers mentioned before. Besides, the statistics show that GeneRIF is often cited from the title or the first/the last sentence of the abstract. We adopt position information as the heuristic cues. The function extraction algorithm is illustrated as follows.

```
For each sentence $k$ in test document $d$
    Compute Score($S_k$);
        Sort Score($S_k$) in descending order;
    Since we cannot guarantee the sentence with the highest score is the candidate GeneRIF, we remain sentences with minor difference with the highest score where minor difference is gotten from the training data so that the reported set can cover the correct answer.
    If there is only one sentence remained
        Produce this sentence as candidate GeneRIF
    Else
        Count the number of matched words with introducers in the sentence;
        If there is only one sentence with the highest matched numbers
            Produce this sentence as candidate GeneRIF
        Else
            Produce the sentence with the following precedence
                1.  The title sentence.
                2.  The first sentence in the abstract.
                3.  The last sentence in the abstract.
                4.  Other position in the abstract.
```

The above algorithm compute the score with Score(S_k). If we compute the score with Score(Norm(S_k)), we get another set of candidate GeneRIF.

4 Results and Discussions

4.1 Results of official runs

We sent two runs to Genome Track on secondary task. The first run is called "we" and the score is computed with Score(S_k). The second run is called "nwe" and the score is computed with Score(Norm(S_k)). The evaluation result is shown in Table 1. The first column shows the measure criteria. "Classic Dice" is the classic Dice formula using a common stop word list and the Porter stemming algorithm. "Modified unigram Dice" gives added weight to terms that occur multiple times in both strings. "Modified bigram Dice" gives some addition weights to proper word order. Instead of measuring the Dice coefficient on single words it measures it on bigrams. For "Modified bigram Dice phrases", this measure only includes bigrams that have not had intervening stop words filtered. The second column "we" and the third column "nwe" denote the average performance for each measure and each run. The fourth to sixth columns represent the average score performed by 24 submissions from 14 groups attended in the secondary task.

Table 1. Experiments with "we" and "nwe"

Measure	we	nwe	best	median	worst
Classic Dice	48.15%	47.62%	57.83%	49.31%	9.42%
Modified unigram Dice	49.78%	49.37%	59.63%	51.30%	14.20%
Modified bigram Dice	32.31%	31.61%	46.75%	33.62%	0.15%
Modified bigram Dice phrases	35.63%	34.80%	49.11%	36.99%	0.17%

Compared with the other submissions, we summarize the number of topics performed as the best, between best and worst, and the worst. The result is shown in Table 2.

Table 2. Analysis with other submissions

Measure	we			nwe		
	best	between	worst	best	between	worst
Classic Dice	54	122	2	53	123	1
Modified unigram Dice	53	118	3	51	121	2
Modified bigram Dice	57	124	30	56	124	31
Modified bigram Dice phrases	61	122	35	59	123	37

From Tables 1 and 2, we find that although the performance is below the average median, we achieve the best results among 139 topics. This shows much effort should be made for further improvement.

4.2 Results with different weight schemes

To improve the result, we try different weight schemes used in Section 3.4 as follows.

- wga/nwga: Let $|w_{i,g:}|$ denote the frequency of the function word w_i in the GeneRIF and $|w_{i,a}|$ denote the frequency of the function word w_i in all articles. Then, weight(w_i)=$|w_{i,g}|/|w_{i,a}|$. We call this weight as wga. If normalization is applied, we call it as nwga.
- wgn/nwgn: Let $|w_{i,g:}|$ denote the frequency of the function word w_i in the GeneRIF and $|w_{i,ng}|$ denote the frequency of the function word w_i in all articles but not in the GeneRIF. Then, weight(w_i)=$|w_{i,g}|/|w_{i,ng}|$. We call this weight as wgn. If normalization is applied, we call it as nwgn.

Replacing the weight used in Section 3.4, new results are expressed in Table 3. Compared with Table 1, although "nwga" is improved, the others are reduced. It shows new weight schemes are not good enough.

Table 3. Experiments with "wga", "nwga", "wgn" and "nwgn"

Measure	wga	nwga	wgn	nwgn
Classic Dice	35.56%	50.18%	35.56%	48.53%
Modified unigram Dice	35.23%	46.71%	35.23%	50.38%
Modified bigram Dice	19.23%	33.47%	19.23%	36.72%
Modified bigram Dice phrases	21.76%	38.83%	21.76%	37.24%

4.3 Results with SVM method

Marcotte *et al.* [15] incorporated a weight-based method and a Bayesian approach in detecting abstracts discussing protein interactions. Several most-discriminating words are first identified by the *p*-score of each word assuming the number of occurrences of a word in an abstract conforms to a Poisson distribution under known dictionary frequency of this word. We therefore investigated the performance of this weighting scheme on GeneRIF sentences and Non-GeneRIF sentences. The weight for each word is calculated by taking negative log of the following probability density function.

$p(n\,|\,N,f)=e^{-Nf}\dfrac{(Nf)^n}{n!}$, where *n* is the number of occurrence of a given word in an abstract of *N* words, and *f* is the dictionary frequency of this word.

According to some preliminary study of the secondary task, it was observed that the position of a sentence in the abstract is an important clue to determine where the answer sentence is. Inspired by the work by the highest-scored team [16] in TREC 2003 Genome secondary task, we also combined sentence positions in our weighting scheme. With our weighting scheme of –log p, given an abstract, we first compute the scores of the title, the first three and the last five sentences, and then this feature vector is fed to a support vector machine (SVM) [17] to make the final decision.

Further, we'd like to know how SVM performs on the features used by the highest-scored team, which we called it sentence-wise bag-of-word model. In this case, 10,506 words were used, and therefore, the feature vector is 94,554 in length. For comparison, we design experiments called "wlog" and "nwlog" which did not contain SVM model, i.e., pure weight scheme used in Section 3.2. As usual, "nwlog" is the normalization version of "wlog". The results are shown in Table 4. It shows the SVM method really works well.

Table 4. Experiments with "wlog", "nwlog", "- log p" and "sentence-wise bag-of-word model"

Measure	wlog	nwlog	- log p	sentence-wise bag-of-word model
Classic Dice	31.55%	48.23%	56.86%	58.92%
Modified unigram Dice	30.14%	50.38%	58.81%	61.46%
Modified bigram Dice	16.11%	32.52%	45.08%	47.86%
Modified bigram Dice phrases	19.17%	36.03%	48.10%	50.84%

5 Concluding Remarks

This paper proposed automatic approaches to extract gene function in the literature. It is helpful to the work of conducting the GeneRIF in LocusLink database. The result shows that 48.15%, 49.78%, 32.31%, and 35.63% for the measure of Classic Dice, Modified unigram Dice, Modified bigram Dice, and Modified bigram Dice phrases in one of our official runs.

Combining the sentence position information, new weighting scheme, and SVM learning mechanism, the performance is improved significantly, i.e., 56.86%, 58.81%, 45.08%, and 48.10% for the measure of Classic Dice, Modified unigram Dice, Modified bigram Dice, and Modified bigram Dice phrases in "-log p" weighting scheme. It directs us to consider another training method for the next stage.

References

[1] Humphreys, B.L., Lindberg, D.A., Schoolman H.M. and Barnett G.O. (1998) "The Unified Medical Language System: an Informatics Research Collaboration," J*ournal of American Medical Information Association*, **5**, pp.1-11, 1998.

[2] Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M. *et al.* (2000) "Gene Ontology: Tool for the Unification of Biology," *Nature Genetics*, **25**, pp. 25-29, 2000.

[3] Bairoch, A. and Apweiler, R. (2000) "The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000," *Nucleic Acids Research*, **28**, pp. 45-48, 2000.

[4] Benson, D.A., Karsch-Mizrachi, I., Lipman, D.J., Ostell, J., Rapp, B.A. and Wheeler, D.L. (2000) "GenBank," *Nucleic Acids Research*, **28**, pp. 15-18, 2000.

[5] Xenarios, I., Fernandez, E., Salwinski, L., Duan, X.J., Thompson, M., J., Marcotte, E.M. and Eisenberg, D. (2001) "DIP: the Database of Interacting Protins: 2001 update," *Nucleic Acids Research*, **29**, pp. 239-241, 2001.

[6] SNOMED, http://www.snomed.org.

[7] Pruitt, K.D., Katz, K.S., Sicotte, H. and Maglott, D.R. (2000) "Introducing RefSeq and LocusLink: Curated Human Genome Resources at the NCBI, " *Trends Genet*, **16**(1): pp. 44-47, 2000.

[8] Blaschke, C., Andrade, M.A., Ouzounis, C. and Valencia, A. (1999) "Automatic Extraction of Biological Information from Scientific Text: Protein-Protein Interactions," *Proceedings of 7th International Conference on Intelligent Systems for Molecular Biology*, pp. 60-67.

[9] Park, J.C., Kim, H.S., and Kim, J.J. (2001) "Bidirectional Incremental Parsing for Automatic Pathway Identification with Combinatory Categorial Grammar," *Proceedings of Pacific Symposium on Biocomputing*, **6**, pp. 396-407, 2001.

[10] Rindflesch, T.C., Hunter, L. and Aronson A.R. (1999) "Mining Molecular Binding Terminology from Biomedical Text," *Proceedings of AMIA Symposium*, pp. 127-131, 1999.

[11] Rindflesch, T.C., Tanabe, L., Weinstein, J.N. and Hunter, L. (2000) "EDGAR: Extraction of Drugs, Genes, and Relations from Biomedical Literature," *Proceedings of Pacific Symposium on Biocomputing*, **5**, pp. 517-528, 2000.

[12] Sekimizu, T., Park, H.S. and Tsujji, J. (1998) "Identifying the Interaction Between Genes and Gene Products Based on Frequently Seen Verbs in Medline Abstracts," *Genome Information*, **9**, pp. 62-71, 1998.

[13] TREC 2003 Genome TRACK, http://medir.ohsu.edu/~genomics/.

[14] LocusLink database, http://www.ncbi.nlm.nih.gov/LocusLink/.

[15] Marotte, E.M., Xenarios, I., and Eisenberg, D. (2001) "Mining Literature for Protein-protein Interactions," *Bioinformatics*, **17**(4), pp. 359-363, 2001.

[16] Jelier, R., Schuemie, M., Eijk, C.V.E., Weeber, M., Mulligen, E.V., Schijvenaars, B., Mons, B., and Kors, J. (2003) "Searching for geneRIFs: concept-based query expansion and Bayes classification," *TREC 2003 work notes,* 2003.

[17] Hsu, C.W., Chang, C.C., and Lin, C.J. "A Practical Guide to Support Vector Classification." Available at: http://www.csie.ntu.edu.tw/~cjlin/libsvm/index.html.

Approach of Information Retrieval with Reference Corpus to Novelty Detection

Ming-Feng Tsai, Ming-Hung Hsu and Hsin-Hsi Chen

Department of Computer Science and Information Engineering
National Taiwan University
Taipei, Taiwan
E-mail: {mftsai, mhhsu}@nlg.csie.ntu.edu.tw; hh_chen@csie.ntu.edu.tw

1 Introduction

According to the results of TREC 2002, we realized the major challenge issue of recognizing relevant sentences is a lack of information used in similarity computation among sentences. In TREC 2003, NTU attempts to find relevant and novel information based on variants of employing information retrieval (IR) system. We call this methodology IR with reference corpus, which can also be considered an information expansion of sentences. A sentence is considered as a query of a reference corpus, and similarity between sentences is measured in terms of the weighting vectors of document lists ranked by IR systems. Basically, we looked for relevant sentences by comparing their results on a certain information retrieval system. Two sentences are regarded as similar if they are related to the similar document lists returned by IR system. In novelty parts, similar analysis is used to compare each relevant sentence with all those that preceded it to find out novelty. An effectively dynamic threshold setting which is based on what percentage of relevant sentences is within a relevant document is presented. In this paper, we paid attention to three points: first, how to use the results of IR system to compare the similarity between sentences; second, how to filter out the redundant sentences; third, how to determine appropriate relevance and novelty threshold.

2 Procedure

The flow of IR with reference corpus is illustrated in Figure 1, which contains an IR system and a reference corpus inside. To begin with, each sentence from the known relevant documents is treated as a query to a certain IR system that retrieves documents from the reference corpus. Then, a sentence can be transformed into a vector that uses each unique document retrieved by the IR system as one dimension and set the relevant weight assigned by the IR system as the weight of each dimension. An IR system, for instance, may retrieve top *m* documents from the reference corpus for a query. Therefore, a sentence can be regarded as a vector of *m* dimensions of weights assigned by the IR system. Finally, similarity metric is applied to measure the similarity between vectors, and the threshold is also applied to the following operations, retrieval or filter. Below we discuss this approach in detail.

2.1 IR System and Reference Corpus

In the experiments, the document sets used in TREC-6 text collection (Voorhees and Harman, 1997) were considered as a reference corpus. It consists of 556,077 documents. Okapi IR system (Robertson, Walker and Beaulieu, 1998) is adopted to experiment this approach. In the initial experiments, Okapi was in the option of *bm25*, and had average precision 0.2181 on TREC-6 text collection.

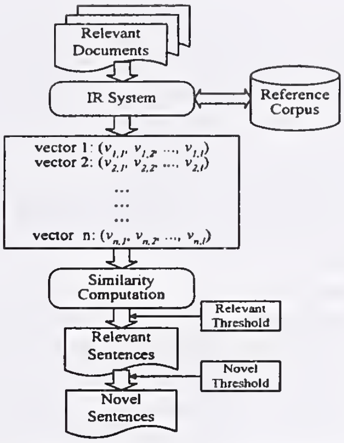

Relevant Documents
IR System
Reference Corpus
vector 1: $(v_{1,1}, v_{1,2}, \ldots, v_{1,l})$
vector 2: $(v_{2,1}, v_{2,2}, \ldots, v_{2,l})$
...
vector n: $(v_{n,1}, v_{n,2}, \ldots, v_{n,l})$
Similarity Computation
Relevant Threshold
Relevant Sentences
Novel Threshold
Novel Sentences

Figure 1. Flow of IR with Reference Corpus Approach

2.2 Similarity Computation

The cosine similarity computation is considered in our task. The metric is shown as follows.

$$\cos(s_i, s_j) = \frac{\sum_{k=1}^{l} v_{i,k} \times v_{j,k}}{\| s_i \| \cdot \| s_j \|} \tag{1}$$

where s_i is represented as a sentence-vector $(v_{i,1}, v_{i,2}, \ldots, v_{i,l})$, l denotes the number of documents retrieved from the reference corpus by IR system; and s_j is another sentence-vector.

2.3 Threshold Setting

We consider what percentage of sentences is relevant within a document. In TREC 2002, Larkey *et al.* showed that about 5% of the sentences contained relevant materials for average topic. We also discovered the percentage of relevant sentences gets less when total number of given sentences is more. Therefore, we used logarithmic regression as follows to simulate the relationship between total number of the given sentences and percentage of the relevant sentences.

A dynamic threshold setting model is proposed as follows. Assume normal distribution with mean μ and standard deviation σ is adopted to specify the similarity distribution of the given sentences with a topic. We compute the cosine of a topic vector T and a given sentence vector $S_i (1 \le i \le m)$, where m denotes total number of the given sentences. The percentage n denotes that top n percentages of the given sentences will be reported. Similarity thresholds ($\mathrm{TH}_{\mathrm{relevance}}$) are determined by these percentages.

$$\mu = \frac{\sum_{i=1}^{m} \cos(T, S_i)}{m} \tag{2}$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{m} (\cos(T, S_i) - \mu)^2}{m}} \tag{3}$$

$$\mathrm{TH}_{\mathrm{relevance}} = \mu + z\sigma \tag{4}$$

$$\phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-y^2/2} dy = 1 - n \tag{5}$$

We first compute the percentage n, and then derive z by Formula (5). Finally, $\mathrm{TH}_{\mathrm{relevance}}$ is computed by Formula (4). Therefore, the relevance threshold is determined by the total number of given sentences.

In the novelty part, a threshold of novelty decision, $TH_{novelty}$, determines the degree of redundancy. If the similarity score of two sentences is larger than $TH_{novelty}$, then one of them has to be filtered out depending to their temporal order. In this way, the redundant sentences are filtered out and only the novel sentences are kept. The remaining sentences are the result of the novelty detector. Two algorithms are proposed as follows. Assume there are r relevant sentences, $s_1, s_2, \ldots, s_r$ for topic t.

(1) Static threshold approach

Let T be a set containing novel sentences found up to know. Initially, $T=\{s_1\}$. For each relevant sentence s_i $(2 \leq i \leq r)$, if there exists a sentence in T whose similarity with s_i is larger than a predefined threshold, then s_i is not a novel sentence and is removed; otherwise, s_i is kept in T.

(2) Dynamic threshold approach

Assume s_1 is a novel sentence. Compute the similarities between s_1 and s_i $(2 \leq i \leq r)$. Determine the novelty threshold, $TH_{novelty}$, in the same way as $TH_{relevance}$. Filter out the top n% of sentences with the higher similarities with s_1. Let R be the remaining sentences. If the number of sentences in R is less than 30[1], then regard these sentences as novel sentences and stop. Otherwise, select the first sentence in R, regard it as a novel sentence and repeat the same filtering task.

3 Experiments

3.1 Finding Relevant Sentences

This part is to give the set of 25 relevant documents for each topic and to identify all relevant sentences. We first treated each given sentence as a query to IR system, and then get a vector of document weight assigned by IR system. Next, we applied the cosine function to measure similarity between sentences. In the part of threshold setting, we used the statistics of TREC 2002 novelty track to simulate the relation of total number of given sentences and percentage of relevant sentences. Formula 6 and Figure 2 show the trend. Because some topics may get less percentage, we apply a parameter to multiply the percentage calculated by Formula (6) to retrieve more sentences. Take Ln-4 for example. That means that it multiplies 4 to the calculated percentage.

$$n = -2.4938Ln(x) + 23.157 \tag{6}$$

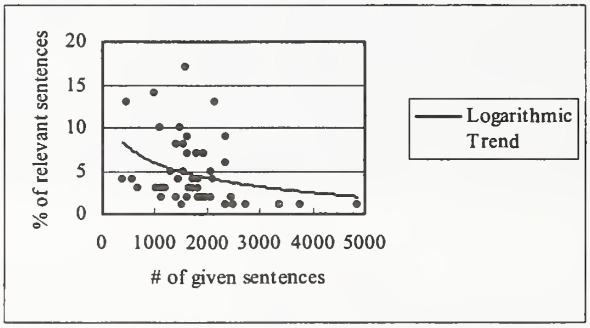


Figure 2. An illustration of Logarithmic Trend

Figure 3 shows the experimental results of relevance detection. These results are totally different to those of last year, because the number of qrels of relevance information is dramatically more than that of last year. Last year, the percentage of relevant sentences within the whole given sentences was about 5%, but this year some topics even has about 50 percent of relevant sentences. Therefore, our average recall gets lower since our relevance threshold is too high. That demonstrates the issue of identifying an appropriate threshold in the novelty detection is very important.

[1] A sample size of at least 30 has been found to be adequate for normal distribution

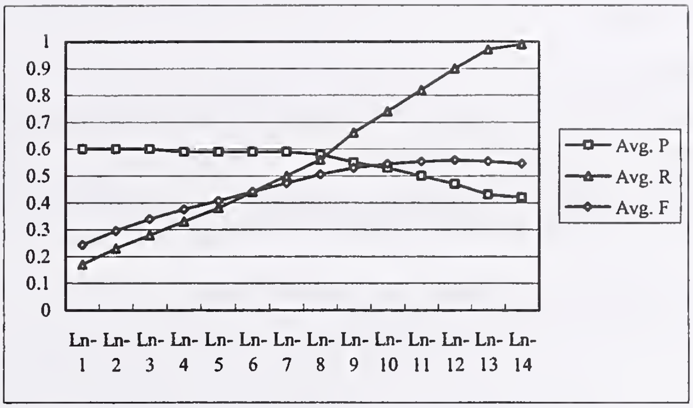


Figure 3. Experimental Results of Relevance Detection

3.2 Finding Novel Sentences

This part is to identify sentences that include new information among the relevant sentences. In other words, this part will filter out the redundant sentences. The key issue of finding novel sentences is how to differentiate the meaning of sentences accurately. We extend the idea, i.e., employing IR with reference corpus approach to expand a sentence, to find novel sentences. We experiment two novelty threshold setting algorithms, i.e., static and dynamic settings. In order to test this model, we use the perfect relevance results to experiment. And the number of consulted documents retrieved by IR system is set to 300.

Figure 4 demonstrates the results of finding novelty with static threshold setting. When novelty threshold is 1, it does not filter out any sentences. The performance gets better as the novelty threshold is higher. Figure 5 shows the results of finding novelty with dynamic threshold setting. The result reveals that the more percentage filtered, the worse the performance is. From these results, the performance will be better if we filter out fewer sentences. Therefore, we set the novelty threshold higher in the submitted runs to achieve better performance.

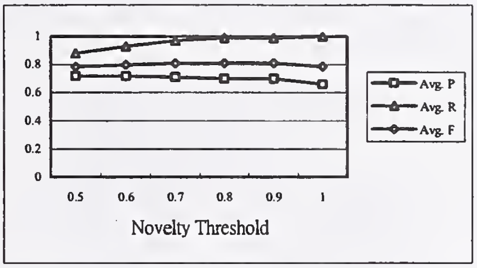


Figure 4. Experimental Results of Novelty Detection with Static Threshold Setting

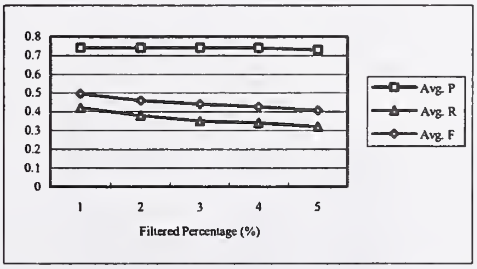


Figure 5. Experimental Results of Novelty Detection with Dynamic Threshold Setting

4 Runs Submitted

4.1 Task1 & Task3

Table 1 and 2 show the runs we submitted in task 1 and 3 of novelty detection, where the number of consulted documents is set to 300 , the dynamic relevance threshold uses Ln-1, and NTU11, NTU12, NTU13 and NTU14 uses topic description and narrative. In the novelty part of task 1, all runs use the static threshold setting where NTU11, NTU13 and NTU15 are set to 0.8; NTU12 and NTU14 are set to 0.9. In the task3, we use Ln-2, and Ln-3 functions to retrieve more relevant sentences.

Table 1. Task 1 Submitted Results

	Relevant Detection			Novelty Detection		
	Avg P	Avg R	Avg F	Avg P	Avg R	Avg F
NTU11	0.59	0.16	0.225	0.43	0.15	0.197
NTU12	0.59	0.16	0.225	0.43	0.15	0.200
NTU13	0.58	0.16	0.223	0.43	0.15	0.195
NTU14	0.58	0.16	0.223	0.42	0.15	0.197
NTU15	0.57	0.14	0.209	0.42	0.14	0.180

Table 2. Task 3 Submitted Results

	Relevant Detection			Novelty Detection		
	Avg P	Avg R	Avg F	Avg P	Avg R	Avg F
NTU31	0.56	0.20	0.266	0.39	0.19	0.217
NTU32	0.57	0.25	0.301	0.39	0.23	0.241
NTU33	0.58	0.22	0.287	0.40	0.21	0.236
NTU34	0.58	0.27	0.330	0.40	0.26	0.270
NTU35	0.57	0.22	0.287	0.39	0.21	0.240

4.2 Task2 & Task4

Tables 3 and 4 show the results of task 2 and 4 of novelty track. We use two novelty algorithms to find novelty sentences. In task 2, NTU21, NTU22 and NTU23 use static threshold; NTU24 and NTU25 use globe threshold setting. In task 4, NTU41, NTU42 and NTU43 also use static threshold; NTU44 and NTU45 use dynamic threshold.

Table 3. Task 2 Submitted Results

	Novelty Detection		
	Avg P	Avg R	Avg F
NTU21	0.71	0.98	0.812
NTU22	0.70	0.99	0.811
NTU23	0.70	0.99	0.812
NTU24	0.74	0.42	0.495
NTU25	0.74	0.42	0.501

Table 4. Task 4 Submitted Results

	Novelty Detection		
	Avg P	Avg R	Avg F
NTU41	0.67	0.98	0.785
NTU42	0.67	0.99	0.784
NTU43	0.67	0.99	0.784
NTU44	0.68	0.46	0.507
NTU45	0.68	0.47	0.509

References

Allan, J., Wade, C., and Bolivar, A.: Retrieval and Novelty Detection at the Sentence Level. In Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Toronto, Canada, July 28–August 01, 2003. ACM (2003) 314-321

Allan, J., Carbonnell, J., and Yamron, J.: Topic Detection and Tracking: Event-Based Information Organization. Kluwer (2002)

Chen, H.H., and Ku, L.W.: An NLP & IR Approach to Topic Detection. In Topic Detection and Tracking: Event-Based Information Organization, James Allan, Jaime Carbonnell, Jonathan Yamron (Editors). Kluwer (2002) 243-264

Chen, H.H., Kuo, J.J., Huang, S.J., Lin, C.J., and Wung, H.-C.: A Summarization System for Chinese News from Multiple Sources. In Journal of American Society for Information Science and Technology. (2003)

Chen, H.H., Tsai, M.F. and Hsu, M.H.: Identification of Relevant Novel Sentences Using Reference Corpus. In Proceedings of the 26th European Conference on Information Retrieval. University of Sunderland, U.K., 5th-7th April 2004. ECIR (2004)

Harman, D.: Overview of the TREC 2002 Novelty Trec. In Proceedings of the Eleventh Text REtrieval Conference. NIST Special Publication: SP 500-251, Gaithersburg, Maryland, November 19-22, 2002. TREC (2002)

Larkey, L. S. et al.: UMass at TREC2002: Cross Language and Novelty Tracks. In Proceedings of the Eleventh Text REtrieval Conference. Gaithersburg, NIST Special Publication: SP 500-251, Gaithersburg, Maryland, November 19-22, 2002. TREC (2002)

Robertson, S.E., Walker, S., and Beaulieu, M.: Okapi at TREC-7: Automatic ad hoc, Filtering, VLC and Interactive. In Proceedings of the Seventh Text REtrieval Conference, Gaithersburg, NIST Special Publication: SP 500-242, Gaithersburg, Maryland, November 9-11, 1998. TREC 7 253-264.

Tsai, M.F., and Chen, H.H.: Some Similarity Computation Methods in Novelty Detection. In Proceedings of the Eleventh Text REtrieval Conference. Gaithersburg, NIST Special Publication: SP 500-251, Gaithersburg, Maryland, November 19-22, 2002. TREC (2002)

Voorhees, E.M., Harman, D.K. (Eds.) Proceedings of the Sixth Text Retrieval Conference. NIST Special Publication: SP 500-240, Gaithersburg, Maryland, November 19-21, (1997)

Zhang, M. et al.: THU at TREC2002: Novelty, Web and Filtering. In Proceedings of the Eleventh Text REtrieval Conference, NIST Special Publication: SP 500-251, Gaithersburg, Maryland, November 19-22, 2002. TREC (2002)

QUALIFIER in TREC-12 QA Main Task

Hui Yang, Hang Cui, Mstislav Maslennikov, Long Qiu, Min-Yen Kan, Tat-Seng Chua

School of Computing, National University of Singapore
3 Science Drive 2, Singapore 117543
Email: {yangh, cuihang, maslenni, qiul, kanmy, chuats}@comp.nus.edu.sg

Abstract

This paper describes a question answering system and its various modules to solve definition, factoid and list questions defined in the TREC12 Main task. In particular, we tackle the factoid QA task by Event-based Question Answering. Each QA event comprises of elements describing different facets like *time, location, object, action* etc. By analyzing the external knowledge from pre-retrieved TREC documents, Web documents, WordNet and Ontology to discover the QA event structure, we explore the inherent associations among QA elements and then obtain the answers. There are three subsystems working parallel to handle definition, factoid, and list questions separately. We highlight the shared modules, fine-grained named entity recognition, anaphora resolution and canonicalization co-reference resolution, among the three subsystems as well.

1 Introduction

Open domain Question Answering (QA) is a complex research area formed as a distinctive combination of Information Retrieval (IR), Information Extraction (IE), and Natural Language Processing (NLP). The basic problem that QA poses is: given a question and a large text corpus, return an "answer" rather than relevant "documents". QA has received tremendous interest recently with the emergence of many commercial products, and research papers in various communities (SIGIR 2003, ACL 2003, ACMMM 2003). In TREC-11 (Voorhees 2002), we explored the use of external resources like the Web and WordNet to extract terms that are highly correlated with the query, and use them to perform linear query expansion (Yang&Chua,2002). While the technique has been found to be effective, we found that there is a need to perform structured analysis on the knowledge obtained from the Web/WordNet to further improve the performance.

This year, we model the factoid questions by Event-based Question Answering (Yang et al. 2003). Questions often refer to several aspects/elements of the events, such as Location, Time, Subject, Object, Quantity, Description and Action, etc. For most QA events, there are inherent associations among their elements. We thus perform *Event Mining* to discover and then incorporate the knowledge of event structure systematically for more effective QA.

Our system, named **QUALIFIER (QUestion Answering by LexIcal FabrIc and External Resources)**, includes modules to perform detailed question analysis, QA event construction, answer justification, fine-grained named entity recognition, anaphora resolution, canonicalization co-reference, and successive constraint relaxation.

During question parsing, detailed question classes, answer types, original query terms and NLP roles of the query terms are analyzed. We derive detailed question class ontology that corresponds to fine-grained named entities. This enables us to extract exact answer from the candidate sentences more accurately. All the questions are treated equally during the stage of detailed question analysis, and then passed to three different subsystems to handle definition, factoid and list questions separately. The original query terms can be used directly to locate potential answer candidates in the corpus. However, one major problem is that those terms do not provide sufficient evidence to retrieve the answer candidates. This is known as the semantic gap between the query space and document space. In order to bridge this gap, we use the knowledge of both the Web and lexical resources to expand the original query. The new query therefore contains terms that are related to the local context in the Web and the lexical context in WordNet. Finally, we structure the query and use it to search for answer candidates through the MG system (Witten et al. 1999). Answer candidate sentences are selected from the top returned documents and are ranked based on association rules obtained from QA Event analysis. Named entity recognition, answer justification, canonicalization resolution and answer selection are done to extract the final answer while successive constraint relaxation is used as an auto-feedback loop to boost the answer coverage.

We will describe the three subsystems one by one in the following sections. Figure 1 illustrates the flow of the main system. For factoid and list questions, they are solved similarly and definition questions are handled differently in answer extraction.

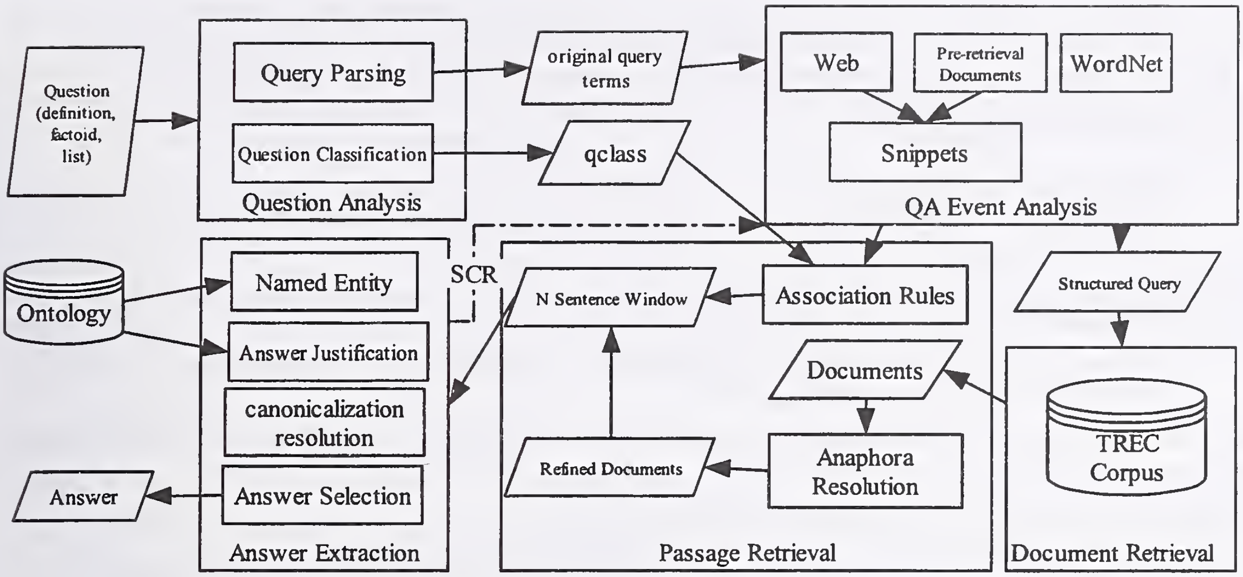


Figure 1: Overview of QUALIFIER

2 Factoid Questions

Our system performs event-based question answering for factoid questions in a few steps: external knowledge acquisition, QA event construction, query formulation, element association mining and answer processing. First, it extracts several sets of words (known elements) from the original question and employs a rule-based question classifier to identify the answer target (unknown element type). During the knowledge acquisition stage, it integrates the knowledge of the pre-retrieved TREC documents, Web, WordNet, and our manually constructed Ontology to extract additional evidences for the query. Second, it performs event construction to discover different facets or elements of events and employs the knowledge of events to perform query formulation. Third, given the newly formulated query, it employs the MG tool to search for top ranked documents in the corpus. Fourth, for the top ranked documents, it identifies the relevant passages by exploiting the associations among event elements. After performing element association mining, it computes the Answer Event Score (AES) and uses it to rank the passages from the relevant documents in the corpus. Answer justification module reinforces the confidence of the returned correct answers and filters out some unreasonable ones.

2.1 QA Event Mining

In our previous work, we modeled the world and lexical knowledge from the Web and WordNet to support effective QA. Basically, we performed the structured analysis of the external knowledge to extract the QA event structure. First, we used the original query terms in the questions to retrieve the top N_w documents by using the Web search engine and then extracted the terms that are highly correlated to the original query terms. Second, we used WordNet to adjust the term weights as well as to introduce new lexical related terms. Third, we computed the lexical, co-occurrence, and distance correlations between terms and used these as the basis to induce event elements by unsupervised semantic grouping. For example, given the question, "*What Spanish explorer discovered the Mississippi River?*", we could get the event structure as shown in Figure 2. Finally, we used this event structure to formulate boolean query to retrieve relevant documents from the QA corpus, and employed a featured-based approach to perform answer selection and extraction.

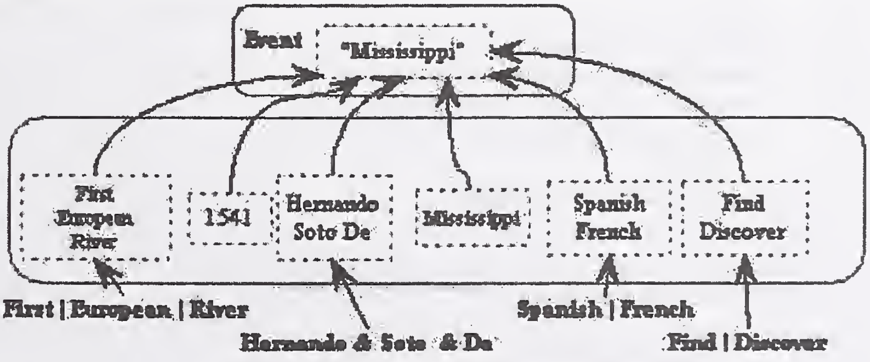


Figure 2: Example for QA Event Structure

After extracting the event structure, we employ *Event Mining* to study the relationships among the event elements. In this approach, we extract important association rules among the elements by using data mining techniques. Given a QA event E_i, we define X, Y as two sets of event elements. Event mining studies the rules of the form X → Y, where "→" means "implies", and Y is the possible answer candidate set. In order to avoid too many misleading rules, we restrict the relationships before generating the rules:

> if $X \subseteq Y$, ignore $X \rightarrow Y$.
> if cardinality(Y) > 1, ignore $X \rightarrow Y$.
> if $Y \cap \{\text{element}_{\text{original}}\} \neq \varnothing$, ignore $X \rightarrow Y$.

Also, we define *Event Mining* as follows:

> ***Event mining*** studies the association rules of the form $X \rightarrow Y$, where X, Y are QA event element sets, $X \cap Y = \varnothing$, and $Y \cap \{\text{element}_{\text{original}}\} = \varnothing$.

There are 4 major steps in the event-based QA approach and we are going to elaborate the details in the following subsections.

a) Extract the QA event from the Web and WordNet for a question.
b) Mine the event in a) to generate the useful association rules among the event elements.
c) Rank the passages in the relevant documents from the QA corpus by matching them with the association rules.
d) Extract the answer phrase from the top passages.

2.2 QA Event Generation

We explore the use of semantic grouping to structurally utilize the external knowledge extracted from the Web, WordNet, Ontology and pre-retrieved documents from TREC. The terms extracted from the relevant web snippets, WordNet, pre-retrieved TREC documents are put into a vector called $\underline{K}_q$, which is the basis for semantic grouping since it is most likely to contain the facts/terms about the QA entity or QA event. Given any two distinct terms t_i, t_j, in the $\underline{K}_q$, we compute (a) Lexical correlation $R_l(t_i, t_j)$;(b) Co-occurrence correlation $R_{co}(t_i, t_j)$; (c) Distance correlation $R_d(t_i, t_j)$. With the term association measures R_l, R_{co}, and R_d, we employ an unsupervised clustering algorithm (yang et al. 2003).to derive the semantic groups of the terms in $\underline{K}_q$, which are expected to match with the event structure. Figure 2 shows the QA event structure developed after performing knowledge modeling.

2.3 Association Rule Mining and Answer Extraction

After constructing the events from the textual data, event-mining techniques are applied to discover association rules within the QA events. We perform the rule mining in two steps: association rule generation and selection. Association rules are in the form of X → Y as we mentioned earlier. Basically, we need to find all the combinations of the event elements to form the sets X and Y. To do this, we need to assign Y to contain only one of the QA event elements. For the rest of the elements, we use them to form different X in various sizes and combinations. We then store all of the X → Y rules in an association rule bank B_i for QA event E_i. The association rule generation algorithm is given as follows.

```
Algorithm Association_Rule_Mining ($E_i$)
Input: QA event $E_i$ and event element set $S_i$
Output: Association rule bank $B_i$ of QA event $E_i$
for each element $e_j$ in $S_i$
  $Y = \{ e_j \}$
  $G_j = S_i - Y$
  for (k=1; k=cardinality($G_j$); k++)
    select k elements from $G_j$ to form a
    collection of size-k element sets $R^{(k)}$;
    for each size-k element set $r_m^{(k)}$ in $R^{(k)}$
      $X = r_m^{(k)}$
      add $X \rightarrow Y$ into rule bank $B_i$
Return $B_i = \{X \rightarrow Y | X \in \cup_k R^{(k)}\}$;
```

Usually for a QA event with n elements, the number of association rules $\aleph_{\text{rules}}$ is:

$$\aleph_{\text{rules}} = n * \sum_{k=1}^{n-1} C_k^{n-1} \tag{1}$$

The number of these rules is potentially large and hence it is necessary to prune away some rules that do not have "interesting" information. Note that not all the rules are reliable. This is because:

- Some association rules are not complete. For example, above algorithm will discover both rule X → Y and rule Z → Y, where $X \subseteq Z$. Then X → Y is not as complete as Z → Y. Thus we have to decide which one should be preserved to extract more accurate answers.
- The rules appear fewer times may not be so significant for a certain QA event. The more often a rule appears in the document collection, the more important it might be.

In order to identify the interesting association rules among the event elements, we need to measure the "usefulness" of the rules. The rules containing more information and involving more event elements are considered to be more important. We consider the typical *Support* measure in data mining (KDD) literature (Dörre et al. 1999) and the reliability of the event elements in X. Therefore, the first measure *Support (X → Y)* is given as:

$$\frac{d_w(X \wedge Y)}{\aleph_{window}} + \alpha * \aleph_{original}(X) + \beta * \aleph_{\exp anded}(X) \qquad (2)$$

where $\alpha + \beta = 1$, $\aleph_{original}(X)$ is the number of elements containing the original question terms in X, $\aleph_{\exp anded}(X)$ is the number of elements containing the expanded query terms in X, $d_w(X \wedge Y)$ is the number of passages containing both X and Y, and $\aleph_{window}$ is the total number of passages in the documents.

Another measure is *Confidence*, which helps to filter out both false information introduced by unreliable data source and the conflicting rules. *Confidence (X → Y)* as defined in data mining:

$$\frac{d_w(X \wedge Y)}{d_w(X)} \qquad (3)$$

We select the best association rules for each event element based on these two measures, which indicates the "usefulness" of the event element relationships. These association rules are combined with event element matching to rank the passages. We compute the *Answer Event Score* (AES) for each passage from the relevant documents in the QA corpus. It is defined as:

$$\frac{M_{ele} + \sum_{i=1}^{N_r} (M_i * (Support(rule_i) + Confidence(rule_i)))}{N_{ele}} \qquad (4)$$

where M_{ele} is the number of matched event elements; $Support(rule_i)$ ($Confidence(rule_i)$) is the support (confidence) for the matched rule i; M_i is set to 1 when rule i is present in this passage, otherwise it is set to 0; N_r is the total number of association rules, and N_{ele} is the total number of event elements for the question.

The good passages we have now describe the same event as the question. Hence, once we have these passages, we can either use the event element in Y as the answer or extract the answer from the passages. In order to extract the exact answers from the passages, we first apply named entity tagging on the top ranked passages and the event association rules. All the named entities whose type match with answer target are selected as answer candidates and ranked based on AES score of the passage where they extracted from, number and significance of the association rules that they match. The ranking formula for answers candidate j is defined as:

$$AES(P_j) + \sum_{i=1}^{Y_j} Support(rule_i) \qquad (5)$$

Where Y_j is the number of top association rules whose Y is matched with *j;* $Support(rule_i)$ is the support for the rule i; AES(Pj) is the AES score of the passage where answer candidate j extracted from.

2.4 Fine-grained Named Entity Recognition

With the set of the top passages obtained after document retrieval and sentence ranking, QUALIFIER performs fine-grained named entity tagging before extracting the string that matches the question class (or answer target or unknown element) as the answer. The system adopts a rule-based algorithm to detect the named entities by utilizing the lexical and semantic features such as capitalization, punctuation, context, part-of-speech tagging and phrase chunking.

The input text is going through the preprocessing stage. We split sentences and remove all characters, which potentially cannot be seen or may risk the following process (mainly, those are non-ASCII characters). In addition, we extract some simple types on this stage, like NUM_PERCENT or COD_URL. In order to avoid the problems with calculations, the program may transfer the text presentation of numbers to numerical. For example, it will convert "one hundred eleven" to 111. The output is represented in the XML format, which contains some marked named entities. We then use the shallow parser by the company Infogistics. The program performs part-of-speech tagging and extracts noun groups and verb groups.

The named entity extraction module is the core for the whole system. The heuristic rules allow creating user-defined types. Also, they support the regular expression style for features of words. Example of the possible rule:

person_title_np = listi_personWord src_, hum_Cap2+ src_, $set(HUM_PERSON/2)

This rule sets the type for each noun phrase, which contains some known person title (academic, academician etc.). The assigned type is person_title_np for this particular phrase. $set(HUM_PERSON/2) means, that the second position (positions are started from 0) should be settled to HUM_PERSON.

Interpretation of those rules starts from the lists initialization. Each document is parsed sentence by sentence. During the parsing stage, each word is represented in the form of token with several features. The output of the interpreter may involve several competing types for some named entity. We also use the following heuristics to choose the rule in such situation:

1) Longer length. We preferred to extracted longer named entities
2) Ontology. If the rule is found in some resource (e.g., confirmed list of persons), then we assigned this type of rule;
3) Handcrafted priorities, which disambiguates the named entities correctly in most of the cases.

Our NE recognizer supports 51 types of NEs, which support 2 levels of granularity. Top levels of classification are human (HUM), location (LOC), number (NUM), object (OBJ), time (TME) and code (COD). The module extracts named entities based on the set of heuristic rules and lists for the semantic categories. We added some new types of fine-grained NE into our old system. QUALIFIER detects fine-grained named entities using a two-tiered hierarchy as shown in Table 1.

Table 1: Partial List of Fine-Grained Named Entities

HUMAN:	Basic, Organization, Person
TIME:	Basic, Day, Month, Year
LOCATION:	Basic, Body, City, Continent, Country, County, Island, Lake, Mountain, Ocean, Planet, Province, River, Town
NUMBER:	Basic, Age, Area, Count, Degree, Distance, Frequency, Money, Percent, Period, Range, Size, Speed
CODE	URL, Telephone, Post code, email address, Product index
OBJECT:	Basic, Animal, Breed, Color, Currency, Entertainment, Game, Language, Music, Plant, Profession, Religion, War, Works

In last year's TREC evaluation for NUM questions, we got the NE tagging for NUMBER only 17 correct out of 29 instances, or an accuracy of 58.6%, which is the lowest rate among all the question types. In order to improve QUALIFIER's ability to recognize NUMBER, we added a range converter module into NE recognizer. We unify numbers close to each other into ranges other than the absolute surface numeric value appearing in the corpus. E.g. "5000000" should be the same range as "5.1 million".

2.5 Anaphora Resolution

In order to maximize recall of the system, which is crucial for list questions, we perform anaphora resolution for the retrieved document by MG system before we proceed to passage selection.

Firstly, we make use of Charniak's parser (Charniak, 2000) to generate the parse tree for each sentence. The Parse Tree Walker extracts two lists. One list contains all the NPs in the text and the other contains all the anaphors. Their agreement features, head-argument/head-adjunct information and all the salience factors except sentence recency are annotated. All the pleonastic pronouns are filtered out. Their antecedent is assigned as null.

For each lexical anaphor, it forms a pair with each item in a subset of the NPs (currently the implementation only considers NPs contained within three sentences proceeding the anaphor and those in the sentence where the anaphor resides). These antecedent candidate-anaphor pairs are examined by the Anaphora Binding Algorithm. For third person pronouns, similarly, the syntactic filter is applied on the candidate pairs consisting one pronoun and one NP from the set satisfying the same positional criteria. In both cases a morphological filter is applied, checking the agreement features compatibility.

For the candidates identified earlier, their salience weights are computed and they are ranked by an arbitrator accordingly. The candidate with the highest weight is proposed as the actual antecedent. In case of tie, the one closer to the anaphor is favored. Figure 3 shows the structure of the anaphora resolution process.

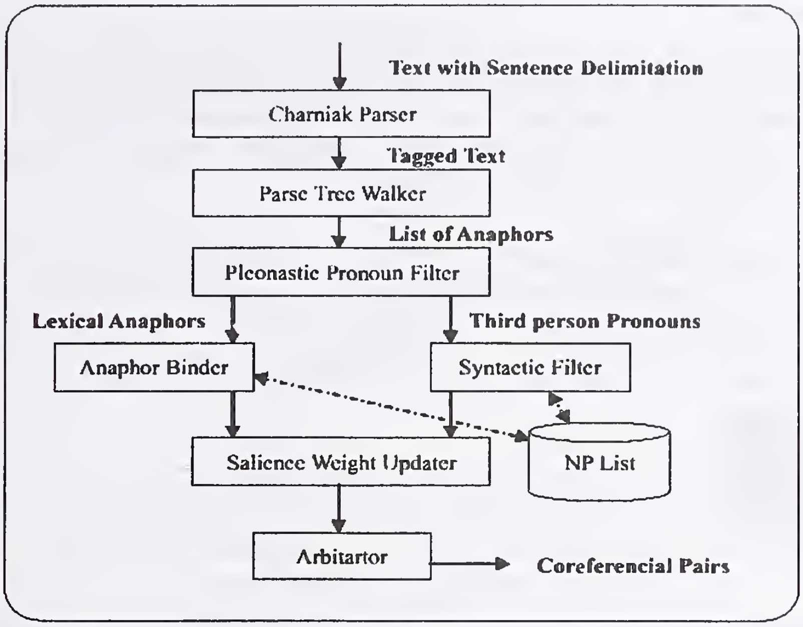


Figure 3: Anaphora Resolution

2.6 Answer Justification

Answer Justification attempts to establish lexica-semantic paths between the questions and candidate long answers, and derive logical proofs along those paths to justify whether the answer is correct. The prover uses two types of axioms. The first type is used to guide the proof and includes axioms derived from facts in the textual answers. The other type of axioms generated based on our manually constructed ontology. For example,

> q1425: What is the population of Maryland?
> Sentence: "Maryland 's population is 50,000 and growing rapidly."
> **Ontology Axiom (OA):** Maryland (c1) & population (c1, c2) -> 5000000(c2)

In this way, we could identify the wrong answer "50000", which is what the surface text shown. Only when we know certain constraint to indicate the actual range of Maryland's population, we know that the surface answer is wrong.

3 List Questions

List question answering is quite similar to what we did for factoid questions. However, we allow the answer extraction module to return multiple answers and remove duplicated answer candidates with the help of abbreviation co-reference. The exact answers are exacted based on patterns and its ranking. Each of these answers is passed to answer justification module for confirmation.

List questions are processed similarly to factoid questions. However, we allow the answer extraction module to return multiple answers and remove duplicated answer candidates with the help of abbreviation co-reference. In general, our method is more data-driven because we want it to be domain independent. However, we still utilized some heuristics to enhance the process.

From a list question, we obtain the same or less amount information comparing to factoid questions since the constraint from the list question itself will be more relax. Under this situation, possible answers for a list question could be in any number. It's hard for us to decide when the right time to stop is. The only way to abase this uncertainty is to perform exhaustive search or near exhaustive search. These patterns could mean surface text patterns, commonly used cue words. There are some examples:

- <same_type_NE>, <same_type_NE> and <same_type_NE> + verb …
- … include: <same_type_NE>, <same_type_NE>, <same_type_NE> …
- "list of …"
- "top" + number + adj-superlative
- "alphabetical list" of …
- "following list…"

4 Definition Questions

Due to the characteristics of definitional questions, i.e. informative and stress on completeness, we treat answering definitional questions as an integrated process of information retrieval and summarization. We utilize techniques from information retrieval as anti-noise mechanism and make use of summarization techniques to avoid redundancy in the results. The pipeline system can be divided into 2 modules: *sentence ranking, and sentence selection plus summary generation.* Figure 4 shows an overview of the definition question subsystem.

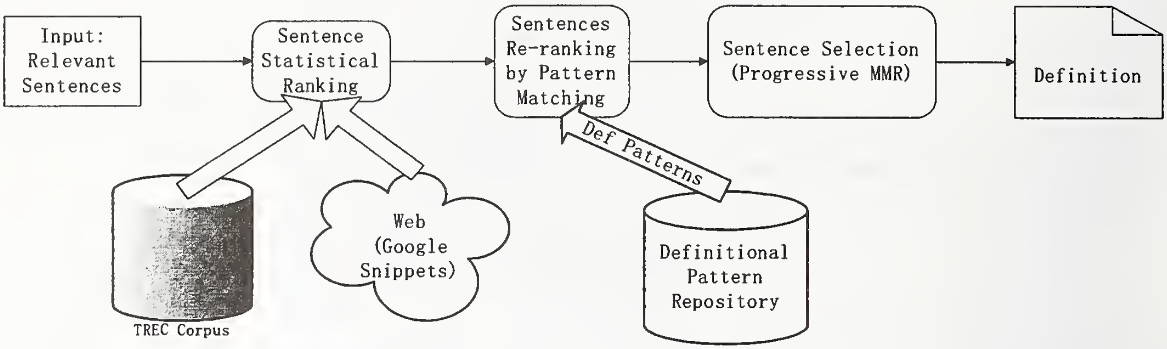

Input: Relevant Sentences
Sentence Statistical Ranking
Sentences Re-ranking by Pattern Matching
Sentence Selection (Progressive MMR)
Definition
Def Patterns
Web (Google Snippets)
Definitional Pattern Repository
TREC Corpus

Figure 4: Illustration of Definition Subsystem

Due to the heterogeneity of the news articles, many of the input documents are actually not related to the search target. We applied a document filter to the get only those documents containing all terms of the search target are labeled as "relevant". We then applied anaphora resolution to sentences from all "relevant" documents to replace appropriate pronouns with the search target. Finally, those sentences containing any part of the search target and their contextual sentences (one sentence proceeding and following them respectively) are sifted as "positive set". All other sentences in the input documents are considered as "negative set".

As for sentence ranking, we utilized evidence from two sources, namely the input documents and the Web. Basically, we use sentence frequency (the number of sentences containing the word) as the main metric to measure the importance of each word. The sentences in "negative set" are used to provide "negative examples". In other words, a word is considered important if it appears often in the sentences of "positive set" while occurs rarely in the "negative set". This can be done in a TFIDF-like fashion:

$$Weight_{Corpus}(s) = \log(1 + \sum_{w \in s} CorpusSF_{Positive}(w) \times \log(1 + \frac{\#\,Negative\ Sentences}{CorpusSF_{Negative}(w)})) \tag{6}$$

In order to account for the diversity of the news articles, we use the Web as a supplementary source. The Web evidence is derived from the snippets obtained using the queries constructed from the sentences of the "positive set" containing any part of the search target. Specifically, the expansion terms are selected by:

$$Weight_{Exp}(w) = \frac{SF(w)}{\#Total\ Sentences} \times \log(1 + \frac{Co(sch\_term, w)}{f(w) + f(sch\_term)}\ \ ) \tag{7}$$

The weight of the sentence for Web evidence can be expressed as:

$$Weight_{Web}(s) = \sum_{w} \log(1 + Web\_SF(w)) \times \log(1 + \frac{\#\,PositiveSentences}{Corpus\_SF_{Positive}(w)}) \tag{8}$$

These two weight values are linearly combined to represent the sentence weight for the search target.

$$Weight(s) = \lambda \cdot Weight_{Corpus} + (1 - \lambda) \cdot Weight_{Web} \tag{9}$$

After sentence ranking, we have a list of ordered sentences with the most relevant sentences to the search target being placed in the top of the list. We made use of summarization techniques to accomplish sentence selection because sentences from news articles are likely to contain duplicated content. We employed a variation of the Maximal Marginal Relevance (MMR) to select sentences from the list while avoiding redundancy between the summary sentences:

a) All sentences are ordered in descending order by weights.
b) Add the first sentence to the summary.
c) Examine the following sentences.
   If *Weight(stc)- Weight(next_stc)* < β *avg_sim(stc)* continue;

else Add stc to summary, where $avg\_sim(stc) = \frac{\sum_{s \in Sum} Sim(s, stc)}{\# sum}$ (10)

avg_sim(stc) is the average similarity of the sentence *stc* and the sentences already in the summary. The similarity is defined as the measure of their word overlapping as:

$$Sim(s, stc) = \sum_{w \in s \cap stc} \log(1 + \frac{\# All\_matching\_stc}{2 \times TREC\_SF_{all\_matching}(w)}) \tag{11}$$

d) Go to Step c) till the length limit of the target summary is satisfied.

In general, our method is more data-driven because we aim it to be domain independent. However, we still utilized some heuristics to enhance the process. For example, if a sentence containing "<*Target*>, a ......" or "<*Target*>, which is the ......", it is likely to be a good definitional sentence for the target. Thus in the sentence ranking and content selection processes, we employed a set of heuristic rules. In content selection process, we applied heuristics to selecting meaningful fragments of the summary sentences to construct the final answers for the search target.

5 Results

We submitted 3 runs for TREC 2003 QA main task. They are mmlnus03r1 (definition run 1 + factoid run 1 + list run 1), mmlnus03r2 (definition run 2 + factoid run 1 + list run 1) and mmlnus03r3 (definition run 3 + factoid run 2 + list run 1). The first run mmlnus03r1 emphasized more on recall (answer coverage) while the third run mmlnus03r3 more on precision (answer accuracy). The second run was a hybrid to test the balance between recall and precision.

The 2 factoid runs focus on precision and recall each. Our first run maximizes recall (answer coverage) by using anaphora resolution, abbreviation co-reference, and successive constraint relaxation to find as many answers as possible without answer justification. In the second run, however, we focused on precision (answer correctness), answer justification plays a rather important role and successive constraint relaxation keeps the recall at a satisfactory level. Table 2 gives the scores for our factoid runs.

The 3 definition runs that we submitted to TREC balance precision and recall. We empirically set the length of the summary for people and objects. Our first run maximizes recall by using full sentences. We have the same length limit for summary in run 2, while text fragments are extracted by heuristics instead of using the full sentences. In the third run (to maximize precision), fewer sentences were extracted for summary and text fragments extraction was also applied. The performance of our definition runs is given in Table 3.

Only one run of list questions is submitted. Its average precision over 37 questions is given in Table 4.

Table 5 summarizes the overall performance for the three submitted runs. It shows that nulmmlr2 gives the best performance, which uses strict constraints and focuses on answer accuracy.

Table 2: Performance over 413 Factoid Questions in TREC-12

nusmmlr1 nusmmlr2	# correct	232	Accuracy	0.562
	# unsupported	24	Precision of recognizing NIL	0.160
	# inexact	13	Recall of recognizing NIL	0.400
	# wrong	144		
nusmmlr3	# correct	225	Accuracy	0.545
	# unsupported	20	Precision of recognizing NIL	0.158
	# inexact	12	Recall of recognizing NIL	0.767
	# wrong	156		

Table 3: Performance over 50 Definition Questions in TREC-12

Average F score for definition question	nusmmlr1	0.441
	nusmmlr2	0.473
	nusmmlr3	0.432

Table 4: Performance over 37 List Questions in TREC-12

nusmmlr1 nusmmlr2 nusmmlr3	Average precision	0.568
	Average recall	0.264
	Average F1	0.317

Table 5: Overall Performance of 3 Submitted Runs in TREC-12

nusmmlr1	0.471
nusmmlr2	0.479
nusmmlr3	0.460

6 Conclusion

We develop three subsystems for this year's TREC. Definition questions answering combines techniques from information retrieval as anti-noise mechanism and make use of summarization techniques to avoid redundancy in the results. Factoid question and list questions take the advantage of Event-based QA, which employs the intuition that there exists implicit knowledge that connects an answer to a question, to extract the correct answer. Association rules are mined to get the relationship among the QA event elements. The whole system consists many modules, including detailed question analysis, QA event construction, event mining, document retrieval, passage retrieval, answer extraction, answer justification, fine-grained named entity recognition, anaphora resolution, canonicalization co-reference, and successive constraint relaxation. We also manually constructed Ontology to lever the performance of our NE and answer justification modules.

We are currently refining our approach in several directions. First, we are trying to formulate a formal proof of our QA event hypothesis. Second, we are working towards an online question answering system. Our longer-term research plan includes Interactive QA, and the handling of more difficult analysis and opinion question types.

Acknowledgement

We would like to thank Dr. Jimin Liu, Ms Jing Xiao and Mr. Chun-Keat Koh for their great efforts in the earlier development for the named entity recognition module. Also, we would like to thank our student assistants Yee-Fan Tan, Victor Goh and Shi-Yong Neo for constructing Ontology and conducting various experiments. This work was supported in part by the A*Star and NUS Joint-Lab funding.

References

E. Charniak. 2000. "A maximum-entropy-inspired parser", In Proceedings of the first 1st Conference of the North American Chapter of the Association for Computational Linguistics, (NAACL'2000).

Jochen Dörre, Peter Gerstl, Roland Seiffert.1999."Text mining: finding nuggets in mountains of textual data", In Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining (SIGKDD'1999), 398-401.

I. Witten, A. Moffat, and T. Bell, 1999, "Managing Gigabytes", Morgan Kaufmann, 1999.

Infogistics, http://www.infogistics.com/posdemo.htm

E.M.Voorhees. 2002. "Overview of the TREC 2002 Question Answering Track." In the Proceedings of the Eleventh Text REtrieval Conference (TREC'2002), 115-123.

H. Yang , T. S. Chua, 2002. "The Integration of Lexical Knowledge and External Resources for Question Answering", In the Proceedings of the Eleventh Text REtrieval Conference (TREC'2002).

H. Yang , T. S. Chua, S Wang, C Koh, 2003. "Structured Use of External Knowledge for Event-based Open Domain Question Answering", In the Proceedings of the Twenty-Sixth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'2003).

A Language Modeling Approach to Passage Question Answering

Dell Zhang[1,2]
[1] Department of Computer Science
School of Computing
S15-05-24, 3 Science Drive 2
National University of Singapore
Singapore 117543
[2] Singapore-MIT Alliance
E4-04-10, 4 Engineering Drive 3
Singapore 117576
+65-68744251
dell.z@ieee.org

Wee Sun Lee[1,2]
[1] Department of Computer Science
School of Computing
S15-05-24, 3 Science Drive 2
National University of Singapore
Singapore 117543
[2] Singapore-MIT Alliance
E4-04-10, 4 Engineering Drive 3
Singapore 117576
+65-68744526
leews@comp.nus.edu.sg

Abstract

This paper reports our efforts on developing a language modeling approach to passage question answering. In particular, we address the following two problems: (i) generalized language modeling for question classification; (ii) constrained language modeling for passage retrieval.

1 Introduction

The Text Retrieval Conference (TREC) has a Question Answering (QA) track to support large-scale evaluation for open-domain QA systems [1-4]. The TREC2003 QA track consists of two separate tasks, the main task and the passage task. We only participated in the passage task.

The passage task of a QA system is to find a small chunk of text that contains the exact-phrase answer of a given question from a large document collection. Lin et al. [5] have showed that users prefer passages over exact-phrase answers in a real-world setting because paragraph-sized chunks provide context. Furthermore, exact-phrase answers are too short to make good training data for future research, making passages a better resource.

This paper reports our efforts on developing a language modeling approach to passage question answering. In particular, we address the following two problems: (i) generalized language modeling for question classification; (ii) constrained language modeling for passage retrieval.

The rest of this paper is organized as follows. In §2, we give a brief review of the language modeling technique. In §3, we describe the architecture of our TREC2003 QA system. In §4, we describe the question classification module. In §5, we describe the passage retrieval module. In §6, we present the evaluation results. In §7, we make concluding remarks.

2 Language Modeling

The language modeling technique is originally motivated by speech recognition, and it has become widely used in many other application areas such as document classification and information retrieval. This section gives a brief review of the language modeling technique. Please be referred to [6, 7] for more detailed explanation.

The goal of language modeling, in general, is to build a language model M_L that captures the statistical regularities of natural language L. Given a word string $S = w_1 w_2 ... w_l$, M_L attempts to predict $\Pr[S \mid M_L] = \Pr_L[S]$, the occurring probability of S in L.

The most common language model is the n-gram model. Despite of its simplicity, the n-gram model works quite well in practice. Applying the chain rule of probability, we get

$$\Pr_L[S] = \Pr_L[w_1 w_2 ... w_l] = \prod_{i=1}^{l} \Pr_L[w_i \mid w_1 ... w_{i-1}].$$

The n-gram model approximates this probability by assuming that the occurrence of w_i only depends on its preceding $n-1$ words, i.e.,

$$\Pr_L[w_i \mid w_1 ... w_{i-1}] = \Pr_L[w_i \mid w_{i-n+1} ... w_{i-1}].$$

A straightforward way to estimate $\Pr_L[w_i \mid w_{i-n+1} ... w_{i-1}]$ is to use maximum likelihood estimation given by

$$\Pr_L[w_i \mid w_{i-n+1} ... w_{i-1}] = \frac{\#_L(w_{i-n+1} ... w_i)}{\#_L(w_{i-n+1} ... w_{i-1})},$$

where $\#(S)$ denotes the number of occurrences of S in the training data of L. However, maximum likelihood estimation assigns zero probabilities to the n-gram strings that were never witnessed in the training data, which are obviously untrue and cause serious problems. Therefore smoothing methods should be used to adjust maximum likelihood estimation to produce more accurate probabilities. One simple but effective smoothing method is to combine the raw model M_{La} (e.g. bigram model) with its background model M_{Lb} (e.g., unigram model) by linear interpolation:

$$\Pr_L[S] = \lambda \Pr_{La}[S] + (1-\lambda)\Pr_{Lb}[S],$$

where $0 \le \lambda \le 1$ is a weighting parameter. More powerful smoothing methods include additive smoothing (e.g. Laplace smoothing), Jelinek-Mercer smoothing, Katz smoothing, Witten-Bell smoothing, Kneser-Ney smoothing, and so on [8].

3 System Overview

The architecture of our TREC2003 QA system is shown in Figure 1. It consists of two major modules: question classification and passage retrieval.

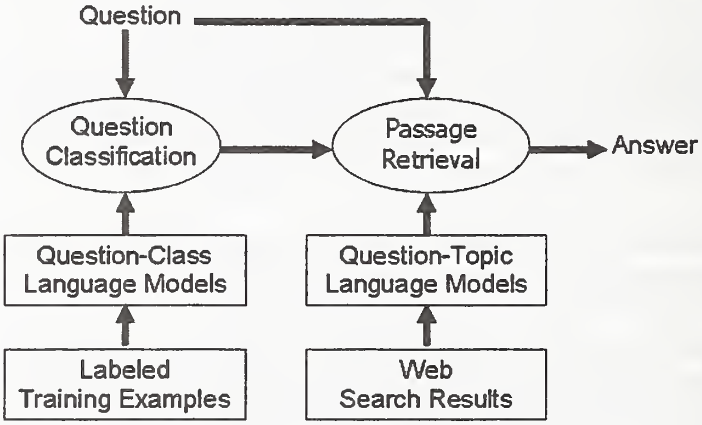

Question
Question Classification
Passage Retrieval
Answer
Question-Class Language Models
Question-Topic Language Models
Labeled Training Examples
Web Search Results

Figure 1. The architecture of our TREC 2003 QA system.

The question classification module identifies each question's preferred answer type using question-class language models, which are learned from thousands of labeled training examples. The language modeling based classification algorithm has many advantages over the popular Naive Bayes algorithm. To tackle the scarcity of training data, we build question-topic language models on generalized question structures but not specific word sequences. The generalized question structures are derived from the original questions through various lexical, syntactic and semantic generalization rules.

The passage retrieval module identifies each question's expected answer context using question-topic language models, which are learned from Web search results. Given a question, we first get a set of relevant passages from the local document collection. Then we search the Web, build a question-topic language model and augment it with a set of probabilistic constraints. Next we rank the retrieved passages using the question-topic language model. Finally, we return the highest ranked passage whose score is above a threshold as the answer. The language modeling based retrieval algorithm implicitly has the power of massive query expansion, which is helpful to overcome the lexical chasm between questions and answers.

4 Question Classification

The task of question classification could be automatically accomplished using machine learning methods [9-11]. Here we attempt to apply language modeling to question classification.

Given a question $Q = q_1 q_2 ... q_k$, it is natural to assign it to the question class which has highest posterior probability, i.e.,

$$C^* = \arg\max_C \Pr[C \mid Q].$$

The posterior probability $\Pr[C \mid Q]$ can be computed via Bayes's rule:

$$\Pr[C \mid Q] = \frac{\Pr[Q \mid C]\Pr[C]}{\Pr[Q]} \propto \Pr[Q \mid C]\Pr[C].$$

The prior probability $\Pr[C]$ can be estimated by the fraction of training questions labeled C. To estimate the probability $\Pr[Q \mid C]$, we build a question-class language model M_C for C and then get

$$\Pr[Q \mid C] = \Pr[Q \mid M_C] = \Pr_C[Q] = \Pr_C[q_1 q_2 ... q_k].$$

In our QA system, smoothed bigram models (see §2) are used to implement question-class language models.

The language modeling based classification (LMC) algorithm is very similar to the popular Naïve Bayes (NB) algorithm [12]. In fact, the LMC algorithm is a straightforward generalization of the NB algorithm: a uniram classifier with Laplace smoothing corresponds exactly to the traditional NB classifier. However, the LMC algorithm possesses many advantages over the NB algorithm, including modeling longer context with larger n and applying superior smoothing techniques in the presence of sparse data [13].

Note that the power of language modeling is often hurt by the scarcity of training data. Applying language modeling to question classification is no exception. To overcome this obstacle, we build question-topic language models on generalized question structures but not specific word sequences. For instance, a question in the form "When was *sb.* born?" always asks for a date no matter who "*sb.*" is, so if we have a DATE-class language model that can accurately predict the probability of the generalized question structure "When was <PERSON> born?", we are able to ensure correct classification of the question "When was Albert Einstein born?" even though "Albert Einstein" has never occurred in the training data.

The generalized question structures are derived from the original questions through various generalization rules, which may include:

- lexical generalization, e.g., replacing every acronym with <ACRONYM>, replacing every number with <NUMBER>;
- syntactical generalization, e.g., replacing every quoted-string with <QUOTED>, replacing every clause with <CLAUSE>;
- semantic generalization, e.g., replacing every string that is a named entity (like organization) with a tag representing its type (like <ORGANIZATION>), replacing every word that belongs to a specific semantic category (like animal) with a tag representing its hypernym (like <ANIMAL>).

The named entity recognizer is modified from a component of GATE [14] (available at http://gate.ac.uk/), and the semantic categories are defined taking advantage of WordNet (available at http://www.cogsci.princeton.edu/~wn/).

5 Passage Retrieval

Recently the language modeling technique has been introduced to information retrieval area and shown considerable success in many applications [15-19]. Here we attempt to apply language modeling to passage retrieval in QA scenario.

Given a question $Q = q_1 q_2 ... q_k$, we first get a set of relevant passages from the local document collection, using the MG software [20] (available at http://www.cs.mu.oz.au/mg/). The passages are defined as half-overlapped text windows each consisting of a fixed number (30 in our case) of words. Every passage is restricted not to cross paragraph boundary. Please be referred to [21] for a recent survey of various kinds of passages.

These passages need to be ranked according to their possibilities of containing the right answer. From the language modeling standpoint, effective ranking of passages could be achieved by constructing a question-topic language model, which represents our expectations about the answer context. The primary difficulty here is the lack of training data.

Lavrenko and Croft [15] have proposed a wise method called "relevance-based language modeling", that can build a unigram model M_R describing a topic in absence of training data. Their method is to approximate $\Pr[w \mid M_R]$ by the formula:

$$\Pr[w \mid M_R] \approx \Pr[w \mid Q] = \frac{\Pr[w, q_1, q_2, \ldots, q_k]}{\Pr[q_1, q_2, \ldots, q_k]} = \frac{\Pr[w, q_1, q_2, \ldots, q_k]}{\sum_w \Pr[w, q_1, q_2, \ldots, q_k]}.$$

To estimate the joint probability $\Pr[w, q_1, q_2, \ldots, q_k]$, we assume that there exists a set $\mathcal{M}$ of underlying source distributions from which w and $q_1, q_2, \ldots, q_k$ could have been sampled independently, then we get

$$\Pr[w, q_1, q_2, \ldots, q_k] = \sum_{M_D \in \mathcal{M}} \Pr[M_D] \left(\Pr[w \mid M_D] \prod_{i=1}^{k} \Pr[q_i \mid M_D] \right).$$

Thus the probability $\Pr[w \mid M_R]$ can be computed as

$$\Pr[w \mid M_R] = \sum_{M_D \in \mathcal{M}} \Pr[w \mid M_D] \Pr[M_D \mid q_1, q_2, \ldots, q_k].$$

Now it becomes obvious that M_R is a linear mixture of distributions from $\mathcal{M}$, where each distribution M_D is "weighted" by its posterior probability of generating the question, $\Pr[M_D \mid q_1, q_2, \ldots, q_k]$.

Since previous research work has revealed immense benefits of exploiting the Web data for QA [22, 23], we decide to construct $\mathcal{M}$ from the question's relevant Web search results. As in [23], we formulate several queries by rewriting the question Q, and submit these queries to a search engine like Google (http://www.google.com) to get search results. For each search result D, we build a smoothed unigram model (see §2) that is to be used as a source distribution $M_D \in \mathcal{M}$, so that $\Pr[w \mid M_D] = \Pr_D[w]$. To make the computation of $\Pr[w \mid M_R]$ tractable, we only use the top-N search results. This simplification is reasonable because the probability $\Pr[M \mid q_1, q_2, \ldots, q_k]$ should have near-zero values for all but the top-N search results. In practice, the strict probabilistic interpretation of $\Pr[M_D \mid q_1, q_2, \ldots, q_k]$ could be relaxed and substituted by any heuristic estimate, as long as it is non-negative and sums to 1 [16]. In our QA system, $\Pr[M_D \mid q_1, q_2, \ldots, q_k]$ is substituted by a weight of M_D whose value is set according to the precision of its corresponding query [23]. For example, the search results returned by the query "+the Louvre Museum +is located" would be weighted higher than those returned by the query "Louvre".

Furthermore, we augment the question-topic language model M_R with a set of constraints which are expressed as probabilities of various events. The constraints used in our QA system include:

- answer-type constraints, e.g., $\Pr[\overline{\mathrm{A}} \mid M_R] = 0$ that means M_R should give zero probability to passages containing no named entity of the desired answer type A;
- answer-context constraints, e.g., for a question in the form "How did sb. die?", we could force $\Pr[\text{survive} \mid M_R] = 0.0$, $\Pr[\text{wreck} \mid M_R] = 0.1$, $\Pr[\text{kill} \mid M_R] = 0.2$, $\Pr[\text{suicide} \mid M_R] = 0.2$, etc.; or we could interpolate M_R with a pre-built model $M_{die-reason}$ which is learned from question-answer pair examples on this topic.

After augmenting these constraints, M_R is adjusted to meet the requirement $\sum_w \Pr[w \mid M_R] = 1$. In this way, we are able to incorporate some prior knowledge into the question-topic language model.

What remains is to use the constructed question-topic language model M_R to rank relevant passages. For each passage P, we build a smoothed unigram model (see §2) M_P. As suggested in [16], we use the

Kullback-Leibler (KL) divergence between passage language model M_P and question-topic language model M_R to rank passages. The KL divergence (also known as relative entropy) between M_P and M_R is defined as:

$$divergence(M_P \| M_R) = \sum_w \Pr[w \mid M_P] \log \frac{\Pr[w \mid M_P]}{\Pr[w \mid M_R]}.$$

Passages whose language models have a smaller divergence with the question-topic language model are considered more relevant to the question's topic. The KL divergence yields a reasonable ranking metric, but has problems when straightforwardly used in QA scenario. Consider a passage P which is very vague (looks too much like general English), it is unlikely to contain the right answer even if $divergence(M_P \| M_R)$ is small, because it does not describe a specific topic. To avoid such trivial passages, we leverage a notion of language model clarity [17]. Given a passage language model M_P, its clarity is defined as $clarity(M_P) = divergence(M_P \| M_G)$, where M_G is the language model of general English estimated from a very large corpus. Consequently we rank the relevant passages according to the following score function:

$$\begin{aligned} score(P) &= -divergence(M_P \| M_R) + clarity(M_P) \\ &= -divergence(M_P \| M_R) + divergence(M_P \| M_G) \\ &= -\sum_w \Pr[w \mid M_P] \log \frac{\Pr[w \mid M_P]}{\Pr[w \mid M_R]} + \sum_w \Pr[w \mid M_P] \log \frac{\Pr[w \mid M_P]}{\Pr[w \mid M_G]} \\ &= \sum_w \Pr[w \mid M_P] \log \frac{\Pr[w \mid M_R]}{\Pr[w \mid M_G]}. \end{aligned}$$

That is, the degree to which M_P is similar to M_R, increased to the extent that M_P is a clear (focused) model that differs from general English. Note that adding clarity has resulted in the denominator that plays a role similar to IDF in standard information retrieval [24]. Finally, we return the highest ranked passage whose score is above a threshold as the answer. If no such answer could be found, we return 'NIL'.

Massive query expansion is an integral part of the language modeling based retrieval algorithm, because we compute the probability $\Pr[w \mid M_R]$ for every word in the language. This helps our QA system to overcome the lexical chasm between questions and answers.

6 Evaluation

The document set for evaluation is the AQUAINT collection that consists of 1,033,461 documents taken from the New York Times, the Associated Press, and the Xinhua News Agency newswires. The question set for evaluation contains 413 factoid questions that seek short, fact-based answers.

A submission for the passage task must contain exactly one answer for each factoid question. An answer is either "NIL" or an extracted passage from a document. A passage should be no longer than 250 bytes, and judged either incorrect (does not contain a correct answer), unsupported (contains a correct answer, but the document doesn't say so), or correct. Unresponsive passages (a passage that refers to an imitation or copy; a passage that contains multiple instances of the correct semantic category of the answer without actually specifying which is the answer; passages that omit necessary units; etc.) are incorrect. For a question with no correct answer in the document collection, only "NIL" answer is correct. The final score for a passage task submission is its accuracy (the fraction of answers judged correct).

The official evaluation result of our TREC2003 QA system is shown in Table 1.

#(test questions)	413
#(correct answers)	173
#(unsupported answers)	9
#(incorrect answers)	231
accuracy	173 / 413 = 0.419
precision of recognizing no answer	10 / 64 = 0.156
recall of recognizing no answer	10 / 30 = 0.333

Table 1. The evaluation result of our TREC2003 QA system.

7 Conclusion

This paper reports our efforts on developing a language modeling approach to passage question answering. We want to demonstrate and advocate that language modeling may provide a uniform framework in which QA systems can integrate evidences from multiple knowledge sources to find the right answer.

Possible future work include: extending this language modeling approach to handle definition questions and list questions; integrating textual patterns [22] into language models; building language models to exploit structured and semi-structured data, particularly HTML/XML data on the Web.

References

[1] E. M. Voorhees, "The TREC-8 Question Answering Track Report," in *Proceedings of the 8th Text Retrieval Conference (TREC)*. Gaithersburg, MD: NIST, 1999, pp. 77-82.

[2] E. M. Voorhees, "Overview of the TREC-9 Question Answering Track," in *Proceedings of the 9th Text Retrieval Conference (TREC)*. Gaithersburg, MD: NIST, 2000, pp. 71-80.

[3] E. M. Voorhees, "Overview of the TREC 2001 Question Answering Track," in *Proceedings of the 10th Text Retrieval Conference (TREC)*. Gaithersburg, MD: NIST, 2001, pp. 157-165.

[4] E. M. Voorhees, "Overview of the TREC 2002 Question Answering Track," in *Proceedings of the 11th Text Retrieval Conference (TREC)*. Gaithersburg, MD: NIST, 2002, pp. 57-68.

[5] J. Lin, D. Quan, V. Sinha, K. Bakshi, D. Huynh, B. Katz, and D. R. Karger, "The Role of Context in Question Answering Systems," presented at Conference on Human Factors and Computing Systems (CHI), Fort Lauderdale, Florida, 2003.

[6] J. Goodman, "A Bit of Progress in Language Modeling, Extended Version," Microsoft Research, Technical Report MSR-TR-2001-72, 2001.

[7] R. Rosenfeld, "Two Decades Of Statistical Language Modeling: Where Do We Go From Here?," *Proceedings of the IEEE*, vol. 88, pp. 1270-1278, 2000.

[8] S. Chen and J. Goodman, "An Empirical Study of Smoothing Techniques for Language Modeling," Computer Science Group, Harvard University, Technical Report TR-10-98, 1998.

[9] W. Li, "Question Classification Using Language Modeling," in *CIIR Technical Report*: University of Massachusetts, Amherst, 2002.

[10] X. Li and D. Roth, "Learning Question Classifiers," in *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*. Taipei, Taiwan, 2002, pp. 556-562.

[11] D. Zhang and W. S. Lee, "Question Classification using Support Vector Machines," in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. Toronto, Canada, 2003, pp. 26- 32.

[12] T. Mitchell, *Machine Learning*, international ed. Singapore: McGraw Hill, 1997.

[13] F. Peng and D. Schuurmans, "Combining Naive Bayes and n-Gram Language Models for Text Classification," in *Proceedings of the 25th European Conference on Information Retrieval Research (ECIR)*. Pisa, Italy, 2003, pp. 335-350.

[14] H. Cunningham, "GATE, a General Architecture for Text Engineering," *Computers and the Humanities*, vol. 36, pp. 223-254, 2002.

[15] V. Lavrenko and W. B. Croft, "Relevance-Based Language Models," in *Proceedings of 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. New Orleans, LA, 2001, pp. 120-127.

[16] V. Lavrenko, M. Choquette, and W. B. Croft, "Cross-Lingual Relevance Models," in *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. Tampere, Finland, 2002, pp. 175-182.

[17] V. Lavrenko, J. Allan, E. DeGuzman, D. LaFlamme, V. Pollard, and S. Thomas, "Relevance Models for Topic Detection and Tracking," in *Proceedings of the Human Language Technology Conference (HLT)*. San Diego, CA, 2002, pp. 104-110.

[18] X. Liu and W. B. Croft, "Passage Retrieval based on Language Models," in *Proceedings of the 11th ACM CIKM International Conference on Information and Knowledge Management (CIKM)*. McLean, VA, 2002, pp. 375-382.

[19] J. Jeon, V. Lavrenko, and R. Manmatha, "Automatic Image Annotation and Retrieval using Cross-Media Relevance Models," in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. Toronto, Canada, 2003, pp. 119-126.

[20] I. H. Witten, A. Moffat, and T. C. Bell, *Managing Gigabytes: Compressing and Indexing Documents and Images*, 2nd ed. San Francisco, CA: Morgan Kaufmann, 1999.

[21] M. Kaszkiel and J. Zobel, "Effective Ranking with Arbitrary Passages," *Journal of the American Society for Information Science and Technology*, vol. 52, pp. 344-364, 2001.

[22] D. Zhang and W. S. Lee, "Web based Pattern Mining and Matching Approach to Question Answering," in *Proceedings of the 11th Text Retrieval Conference (TREC)*. Gaithersburg, MD: NIST, 2002, pp. 497-504.

[23] S. Dumais, M. Banko, E. Brill, J. Lin, and A. Ng, "Web Question Answering: Is More Always Better?," in *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. Tampere, Finland, 2002, pp. 291-298.

[24] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*. New York, NY: Addison-Wesley, 1999.

Océ at TREC 2003

Pascha Iljin, Roel Brand, Samuel Driessen, Jakob Klok

Océ-Technologies B.V.
P.O. Box 101
5900 MA Venlo
The Netherlands
{pi, rkbr, sjdr, klok}@oce.nl

Abstract

This report describes the work done at Océ Research for the TREC 2003. This first participation consists of ad hoc experiments for the Robust track. We used the BM25 model and our new probabilistic model to rank documents. Knowledge Concepts' Content Enabler semantic network was used for stemming and query expansion. Our main goal was to compare the BM25 model and the probabilistic model implemented with and/or without query expansion. The developed generic probabilistic model does not use global statistics of a document collection to rank documents. The relevance of the document to a given query is calculated using term frequencies of the query terms in the document and the length of the document. Furthermore, some theoretical research has been done. We have constructed a model that uses relevance judgements of previous years. However, we did not implement it due to the time constraints.

1 Introduction

This is our first participation in the Text REtrieval Conference. We aimed to compare the models we constructed during the last two years. We decided to participate in the Robust track because it allows to evaluate IR systems given a set of topics and relevance judgements of previous years. That is exactly what we did for an internal research using the CLEF Dutch collection. Furthermore, the Robust track is oriented towards the actual practical situation in information retrieval (i.e. good results are expected for every query). Due to the time restrictions we did not manage to retrain our theoretical model for the TREC's collection of documents and queries in English.

2 Description of runs

The description of the submitted runs is presented in the table below:

Run number	Ranking model	Topic's tags used	Expansion of query terms
1	BM25	Title+Description	yes
2	BM25	Title+Description	no
3	BM25	Description	no
4	probabilistic	Title+Description	yes
5	probabilistic	Title+Description	no

The information about the query construction is presented in Section 3. The models will be described in Section 4.

3 Methods

3.1 Query

A query is constructed automatically from the *title* and *description* (in one of the experiments just the description is used, as required by the track guidelines) by splitting on non-alphanumerical characters to obtain terms. All single characters are removed afterwards. Furthermore, all remaining terms are converted to lower case. For the query expansion, the morphological collapse (dictionary based stemming) of Knowledge Concepts' Content Enabler semantic network is used to obtain root forms of query terms. The root forms are then expanded with the semantic network. The morphological variants of the root form (such as plural form, etc.) are added to the query.

Expansion of query terms

All query terms are morphologically expanded using Knowledge Concepts' Content Enabler semantic network.

Related terms and synonym expansion

Research was done on using related terms and synonyms. We found that Knowledge Concepts' Content Enabler is not good enough to create related terms and synonyms for our models. A measure of 'similarity' between two terms is needed in order to rank the proposed list of related terms and synonyms. Only terms that are very 'similar' in their meaning to a query term should be added to the expanded query.

Query consisting of *topic + description* tags

For the experiments with the queries composed of the *topic* and *description* tags, the terms from these two tags have been put together without duplicate removal. We assumed that if a term in the query is present more than one time, it is considered to be a more important term than if it occurs once.

3.2 Indexing

The index was built by splitting documents on non-alphanumerical characters. Single characters were removed from the index. Stop words were left in the index because it is very difficult to construct a universal set of stop words. If such a set is based on the frequencies within a document collection, it is highly probable that the set of stop words will not be the same for two different document collections. In case it is based on human decisions, a number of important terms from the document collection and/or query will be removed. For example, consider the terms 'new' and 'year' as stop words (they are used in this role quite often). After removing these terms from the document collection and from the queries, it becomes difficult to find a set of relevant documents for the query '*A New Year tree*'. In order to show that stop word removal is not always beneficial, consider the query '*Who said "To be or not to be?"*'. In this case *all* terms from the query could be defined as stop words. Nevertheless, the stop words should be treated different than other terms. Therefore, we weight them down. This year the following stop word lists were used:

- Search Engine World (http://www.searchengineworld.com/spy/stopwords.htm)
- Institut interfacultaire d'informatique, University of Neuchatel (http://www.unine.ch/Info/clef/)

4 Ranking models

4.1 BM25 model

The general description of the BM25 model is as follows:
Let q_i be a query term in query q
Let $q_{i,0}, q_{i,1}, \ldots, q_{i,n}$ be the expansion of q_i in which $q_{i,0} = q_i$
Let tf $(q_{i,j}, d)$ be the term frequency of expansion term $q_{i,j}$

We now calculate the document and term frequency of q_i as follows:

$$tf(q_i, d) = \sum_j tf(q_{i,j}, d) \tag{1}$$

$$\text{df}\ (q_i) = \left| \bigcup_j \text{set of documents in which } q_{i,j} \text{ occurs} \right|$$

Then for a document *d*, and query *q*, the score is calculated as:

$$\text{Rel}\ (d, q) = \sum_{q_i \in q} \frac{\log(N) - \log(df(q_i)) \cdot tf(q_i, d) \cdot (k_1 + 1)}{k_1 \cdot ((1-b) + (b \cdot ndl(d))) + tf(q_i, d)}, \tag{2}$$

in which ndl(*d*) is the length of the document *d*, divided by the average document length.

This model was used for the CLEF 2002 runs and has been described in [1]. Last year we observed that the performance of the BM25 ranking algorithm depends greatly on the choice of the values of the parameters *k1* and *b*.

The estimation of those values for the optimal performance is only possible when the document collection, the set of queries and the set of relevance judgements are all available beforehand. Hence, the Robust track for the old queries is a suitable training set.

4.2 Probabilistic model

The probabilistic model has been selected as the result of theoretical research conducted in 2002 [2]. It contains some innovations with respect to the standard probabilistic approach. The urn model (i.e. balls in an urn = terms in a document) was selected as a basis for the probabilistic model.

We calculate the degree of relevancy without using collection statistics (e.g. document frequency). The sparse data problem is commonly solved using the linear interpolation method or other smoothing techniques that are based on collection statistics. Robertson showed that “relevance of a document to a request should not depend on the other documents in the collection” in order to guarantee “optimality of ranking by the probability of relevance" [3]. Therefore, the selection of a complete document collection as a smoothing element is not strongly motivated and not even supposed to exist according to the basic principle of the probabilistic approach in information retrieval. We found experimentally that under certain distributions of terms over documents in the document collection, the linear interpolation approach will give illogical ranking results. A standard solution to the sparse data problem is to assign non-zero values for query terms that do not exist in a document. The most natural and easy way to solve the sparse data problem is to assign a constant positive value α to the terms that do not exist in the document. We named this **‘the α-method’**.

For the query without term expansion:

$$\text{Rel (d, Q)}=\prod_{q_i\in Q}[\frac{1}{2}\cdot(\frac{tf(q_i,d)}{L_d}+\alpha)], \qquad (3)$$

where L_d is the length (not normalised) of the document *d*,
α should be less than [the length of the longest document in the document collection]$^{-1}$. This guarantees coordination level ranking.

4.3 Statistical model (theoretical results)

In 2002 we aimed to implement a set of clues (that we defined) in a ‘*mathematically correct*’ model, i.e. a model without internal contradictions or violations of axioms. Examples of clues are:

- presence of terms in the document that are synonyms of the terms in the query;
- importance of a topic’s tag;
- part of speech of the query terms;
- query terms of certain document frequency;
- presence of proper nouns in the query;
- length of a document.

We found that a set of defined clues could not be entirely incorporated in the currently known information retrieval models while maintaining mathematical correctness. However, we have succeeded to construct a statistical approach that allows incorporation of these clues. For each clue, a value expressing its expected ‘*significance*’ is calculated. *Significance* values are based on relevance judgements from previous years for (document, topic) pairs.

For every clue we test whether its incorporation makes a statistically significant contribution to the overall performance of an information retrieval system.

Let us select a *clue*[1] to investigate its contribution to the improvement of the performance. The following procedure is carried out for the whole set of queries. Let us consider a query *q*:

- From *q* we determine those components that can be tested for contribution of the *clue* with respect to the total performance of an information retrieval system.

[1] Taking two or more clues simultaneously is very complex.

Let us denote by $Comp_c(q, clue)$ the c^{th} component in the query q that is tested, where $c = \overline{1, C(q, clue)}$, and $C(q, clue)$ is the total number of components from the query q that can be tested on the *clue*.

Example 1
In case the *clue* is a 'presence of query terms in a document', all query terms are components.
Example 2
In case the *clue* is a 'noun', the components from the query 'Crocodiles living in the lake' are 'crocodiles', and 'lake'.

- The following notation will be used:

$|R(J,q)|$ - the number of documents from document collection (*Dc*) that have got values *relevant* from the relevance judgements for the query *q*.

$|I(J,q)|$ - the number of documents from *Dc* that have got values *irrelevant* from the relevance judgements for the query *q*.

$|R_{Comp_c(q,clue)}|$ - the number of documents from *Dc* that have got values *relevant* from the relevance judgements for the query *q* and that contain $Comp_c(q, clue)$.

$|I_{Comp_c(q,clue)}|$ - the number of documents from *Dc* that have got values *irrelevant* from the relevance judgements for the query *q* and that contain $Comp_c(q, clue)$.

- Calculate for every component $Comp_c(q, clue)$:

$$\mathrm{R}_c(clue,q) = \frac{|R_{Comp_c(q,clue)}|}{|R(J,q)|} \tag{4}$$

$$\mathrm{I}_c(clue,q) = \frac{|I_{Comp_c(q,clue)}|}{|I(J,q)|} \tag{5}$$

The pair ($\mathrm{R}_c(clue,q)$, $\mathrm{I}_c(clue,q)$) indicates how often a component $Comp_c(q, clue)$ occurs in relevant and irrelevant documents respectively. In case $\mathrm{R}_c(clue,q) > \mathrm{I}_c(clue,q)$, the component $Comp_c(q, clue)$ occurs more often in relevant documents than in irrelevant ones.

After ($\mathrm{R}_c(clue,q)$, $\mathrm{I}_c(clue,q)$) is calculated for each component *c* of each query *q*, a set of pairs {($\mathrm{R}_1(clue)$, $\mathrm{I}_1(clue)$), ($\mathrm{R}_2(clue)$, $\mathrm{I}_2(clue)$),…, ($\mathrm{R}_t(clue)$, $\mathrm{I}_t(clue)$)} is obtained, where $t = \sum_{q=1}^{|Q|} C(q, clue)$ is the number of all components for *clue* from all $|Q|$ queries in the test collection.

In case $\sum_{\substack{i=1 \\ \{R_i(clue) > I_i(clue)\}}}^{t} 1 \quad > \quad \sum_{\substack{i=1 \\ \{R_i(clue) < I_i(clue)\}}}^{t} 1$, one can state that after incorporating the *clue*, the components of *q* appear more often in relevant documents than they appear in irrelevant ones. This statement implies that the incorporated clue is expected to improve the performance of the information retrieval system.

In order to decide if a clue may improve the performance of the system, the set of pairs {($\mathrm{R}_1(clue)$, $\mathrm{I}_1(clue)$), ($\mathrm{R}_2(clue)$, $\mathrm{I}_2(clue)$),…, ($\mathrm{R}_t(clue)$, $\mathrm{I}_t(clue)$)} should be statistically investigated. The statistical method called the Sign Test is used in order to compare two sets of pairs. It is the only method that can be used for our purpose.

The Sign Test is used to test the hypothesis that there is "no difference" between the two probability distributions (in our case, R(*clue*) and I(*clue*)). For the statistical model it tests whether the presence of the *clue* has influence on the distribution of the query components in relevant and irrelevant documents.

The theory of the Sign Test requires:
1. The pairs to be mutually independent.
2. Both R_i(*clue*) and I_i(*clue*) should have continuous probability distributions.

Because of the assumed mutual independence between queries, mutual independence between query terms, and mutual independence between terms in documents, pairs (R_1(*clue*), I_1(*clue*)) are mutually independent (point 1). A continuous distribution is defined as a distribution for which the variables may take on a continuous range of values. In the considered case, the values of both R_i(*clue*) and I_i(*clue*) take any value from the closed interval [0,1], and so their distributions are continuous (point 2). Hence, the necessary conditions for the Sign Test hold.

The hypothesis implies that given a pair of measurements (R_i(*clue*), I_i(*clue*)), both R_i(*clue*) and I_i(*clue*) are equally likely to be larger than the other. The zero hypothesis H_0: P[R_i(*clue*) > I_i(*clue*)] = P[R_i(*clue*) < I_i(*clue*)] = 0.5 is tested for every i=$\overline{1,t}$. Applying the one-sided Sign Test means that rejecting H_0, we accept the alternative hypothesis H_1: P[R_i(*clue*) > I_i(*clue*)] > 0.5. A one-sided 95% confidence interval is taken to test the H_0 hypothesis. If H_0 is rejected, the incorporation of the *clue* is expected to improve the performance of the information retrieval system.

> Remark
> Using the Sign Test described for a certain clue, we conclude whether its incorporation into an information retrieval system can improve the performance. This conclusion is based on theoretical expectations only.

Two criteria are defined to estimate the possible contribution of a clue to a system from a practical point of view.

In case there are *t* components for all the queries, $\forall i = \overline{1,t}$ calculate for *clue*
i) #(R(*clue*)) – the number of components for which R_i(*clue*) > I_i(*clue*)
ii) #(I(*clue*)) – the number of components for which R_i(*clue*) < I_i(*clue*)

According to the theoretical issues of the Sign Test, one has to ignore the statistics of the components for which R_i(*clue*) = I_i(*clue*). Thus, when a component of a certain clue is found in both relevant and irrelevant documents, and the relative frequency of R_i(*clue*) = I_i(*clue*), this is neither good nor bad. Such an observation should not influence the total statistics.

However, the other theoretical issue will not be taken into account. According to the theory of the Sign Test, when one observes more than one component with the same values of R_i(*clue*) and I_i(*clue*), all but one component should be ignored too. However, this claim cannot be valid in the area of linguistics due to the following reasons:

1. The influence of each component on the clue has to be calculated. Even in the case the same statistics are obtained for different terms, all terms will make a contribution to the performance of the system. So, every component will be an extra observation for a clue.

2. If a term is used in more than one query, it has multiple influences on the performance. For each query different statistics should be obtained. Hence, each component should be considered separately for every query.

3. In case the same component is used more than one time in a query, it is considered multiple times (according to the assumption described in 'Query consisting of *topic* + *description* tags', see Section 3.1).

To estimate the significance of a certain clue, the ratio $\dfrac{\#(R(clue))}{\#(I(clue))}$ is calculated. The larger this ratio, the higher the significance is. After calculating these ratios for all the clues, they can be ranked in a decreasing order, where the top value will correspond with the most significant clue.

- *Not all clues have the same contribution to the ranking function.*

The contribution of a certain clue depends on the level of improvement to the performance of an information retrieval system.

- *Not all clues should be implemented in the statistical model.*

A clue is implemented into a model if the ratio $\frac{\#(R(clue))}{\#(I(clue))}$ has a value higher than one. Only in this case one can expect that the selected clue can improve the performance of the system.

Experiments with the statistical model

We have done a number of experiments with the statistical model for the CLEF Dutch document collection, the set of queries and the relevance judgements for 2001 and 2002. Depending on their degree of significance, different statistics have been chosen to obtain better performance for two different sets of queries (using the same document collection). The proper choice of features and their 'gain' values lead to better results. We conclude that this model is strongly dependent on the data collection, queries and relevance judgements. Hence, the results for a set of new documents, new queries and new relevance judgements are difficult to predict. Due to time restrictions we did not retrain the model for the TREC Robust track. Therefore we did not submit the statistical model.

5 Numerical results

The following numerical results were obtained for the runs submitted by Océ at TREC 2003.

Old topics:

Run	Number of retrieved relevant documents	Average precision	R-precision	Number of topics with no relevant in top 10 (in %)	Area underneath MAP(X) vs. X curve for worst 12 topics
1 (BM25,TD,Exp)	2005 out of 4416	0.1245	0.1763	12.0	0.0117
2 (BM25,TD,noExp)	1903 out of 4416	0.1205	0.1714	14.0	0.0101
3 (BM25,D,noExp)	1570 out of 4416	0.0923	0.1470	24.0	0.0027
4 (Prob,TD,Exp)	1425 out of 4416	0.0749	0.1312	20.0	0.0041
5 (Prob,TD,noExp)	1418 out of 4416	0.0859	0.1363	20.0	0.0038

New topics:

Run	Number of retrieved relevant documents	Average precision	R-precision	Number of topics with no relevant in top 10 (in %)	Area underneath MAP(X) vs. X curve for worst 12 topics
1 (BM25,TD,Exp)	1419 out of 1658	0.3646	0.3567	10.0	0.0352
2 (BM25,TD,noExp)	1428 out of 1658	0.3379	0.3423	6.0	0.0406
3 (BM25,D,noExp)	1318 out of 1658	0.3049	0.3159	16.0	0.0134
4 (Prob,TD,Exp)	1241 out of 1658	0.2921	0.3066	12.0	0.0145
5 (Prob,TD,noExp)	1255 out of 1658	0.2846	0.3167	10.0	0.0180

All topics together:

Run	Number of retrieved relevant documents	Average precision	R-precision	Number of topics with no relevant in top 10 (in %)	Area underneath MAP(X) vs. X curve for worst 25 topics
1 (BM25,TD,Exp)	3424 out of 6074	0.2446	0.2665	11.0	0.0163
2 (BM25,TD,noExp)	3331 out of 6074	0.2292	0.2568	10.0	0.0168
3 (BM25,D,noExp)	2888 out of 6074	0.1986	0.2315	20.0	0.0055
4 (Prob,TD,Exp)	2666 out of 6074	0.1835	0.2189	16.0	0.0063
5 (Prob,TD,noExp)	2673 out of 6074	0.1852	0.2265	15.0	0.0066

6 Conclusions

We have compared the BM25 and our probabilistic model on the basis of mono-lingual runs for English. The BM25 model systematically outperforms the probabilistic one. This indicates that striving for mathematical correctness does not imply better retrieval performance. At the same time we have observed that the developed probabilistic model performs satisfactorily. Furthermore, we conclude that the query expansion using the Knowledge Concepts' Content Enabler semantic network does not improve the performance of the IR systems we constructed. The performance of the IR engine using the query consisting of the *description* tag only, is worse than using the *topic* and *description* tags.

7 References

[1] Roel Brand, Marvin Brünner: Océ at CLEF 2002. Lecture Notes on Computer Science, Springer-Verlag Heidelberg, 2003.

[2] Pascha Iljin: Modeling Document Relevancy Clues in Information Retrieval Systems. SAI, to appear in 2004.

[3] Djoerd Hiemstra: Using Language Models for Information Retrieval. Ph.D. Thesis, Centre for Telematics and Information Technology, University of Twente, 2001.

Phrases, boosting, and query expansion using external knowledge resources for genomic information retrieval

William Hersh, Ravi Teja Bhupatiraju, Susan Price
Oregon Health & Science University
{hersh,bhupatir,prices}@ohsu.edu

In our TREC Genomics Track work, we focused on domain-specific techniques in attempting to improve retrieval performance beyond a word searching baseline. One set of experiments looked at using phrases based on gene name synonyms with boosting of the canonical name of the gene. Another set assessed query expansion using external knowledge resources.

Query expansion has been a staple of the TREC ad hoc task dating back almost to the inception of TREC, showing consistent benefit when added to a wide variety of baseline techniques, e.g., [1, 2]. In the biomedical domain, however, results have been mixed. While Srinivasan obtained improved retrieval using retrieval feedback (automatic relevance feedback) in a small test collection [3], Hersh et. al. did not find improved retrieval when queries were expanded using thesaurus relationships in the Unified Medical Language System (UMLS) Metathesaurus [4]. Query expansion may be feasible in the genomics domain due to the considerable effort being devoted to creating useful cross-linkages across data sources. The most prominent example is the collection of databases maintained by the National Center for Biotechnology Information (NCBI, www.ncbi.nlm.nih.gov), a division of the National Library of Medicine (NLM, www.nlm.nih.gov) [5].

Phrases and Boosting

Our first experiments derived from a goal of defining baseline performance for the track training data. We built an IR system around the Lucene search engine with a pre-processor implemented in Python and a batch search facility implemented in Java. Shell scripts tied together these components in a way to allow experiments. The pre-processor converted the formal track queries file to a text file that contained the query terms in a Lucene format with one line per query. The search facility took this file and batched the queries into Lucene. The results were written into a text file in the trec_eval format. The report script called trec_eval with the results and the relevance file (qrels), writing the output to a report text file.

Methods

The simplest baseline approach involved taking all of the names for a gene name and turning them into a single query string of a "bag of words." Table 1 shows the gene names provided from LocusLink for topic 1 of the test data. This was transformed into the following query string:

> cyclin-dependent kinase inhibitor 1A p21 Cip1 CDKN1A P21 CIP1 SDI1 WAF1 CAP20 CDKN1 MDA-6 cyclin-dependent kinase inhibitor 1A cyclin-dependent kinase inhibitor 1A cyclin-dependent kinase inhibitor 1A DNA synthesis inhibitor CDK-interaction protein 1 wild-type p53-activated fragment 1 melanoma differentiation associated protein 6

Table 1 - Gene names for test topic 1 from LocusLink.

1	1026	Homo sapiens	OFFICIAL_GENE_NAME	cyclin-dependent kinase inhibitor 1A (p21, Cip1)
1	1026	Homo sapiens	OFFICIAL_SYMBOL	CDKN1A
1	1026	Homo sapiens	ALIAS_SYMBOL	P21
1	1026	Homo sapiens	ALIAS_SYMBOL	CIP1
1	1026	Homo sapiens	ALIAS_SYMBOL	SDI1
1	1026	Homo sapiens	ALIAS_SYMBOL	WAF1
1	1026	Homo sapiens	ALIAS_SYMBOL	CAP20
1	1026	Homo sapiens	ALIAS_SYMBOL	CDKN1
1	1026	Homo sapiens	ALIAS_SYMBOL	MDA-6
1	1026	Homo sapiens	PREFERRED_PRODUCT	cyclin-dependent kinase inhibitor 1A
1	1026	Homo sapiens	PRODUCT	cyclin-dependent kinase inhibitor 1A
1	1026	Homo sapiens	ALIAS_PROT	DNA synthesis inhibitor
1	1026	Homo sapiens	ALIAS_PROT	CDK-interaction protein 1
1	1026	Homo sapiens	ALIAS_PROT	wild-type p53-activated fragment 1
1	1026	Homo sapiens	ALIAS_PROT	melanoma differentiation associated protein 6

Because our results generated relatively low precision at various points of recall, we looked for ways to decrease the “noise” in our queries. One attempt to improve precision involved the use of phrases. Using the feature of Lucene that allows adjacency of words in a query to be designated by enclosing them in quotes (a common feature across Web search engines), we rebuilt the queries as a series of phrases. One approach involved using only the official gene name in a phrase. The search string for this approach for the gene in Table 1 was:

> cyclin-dependent kinase inhibitor 1A (p21, Cip1)

Another approach converted all of the gene names into phrases. We also discovered some additional performance improvement by using another feature of Lucene, term boosting, which allows designated terms to be assigned added weight. We empirically determined that increasing the weight of the official name phrase by 2.9 gave the best performance. The search string for this approach for the gene in Table 1 was:

> "CDKN1A"^2.9 "P21" "CIP1" "SDI1" "WAF1" "CAP20" "CDKN1" "MDA-6" "cyclin-dependent kinase inhibitor 1A" "cyclin-dependent kinase inhibitor 1A" "cyclin-dependent kinase inhibitor 1A" "DNA synthesis inhibitor" "CDK-interaction protein 1" "wild-type p53-activated fragment 1" "melanoma differentiation associated protein 6"

Results

Table 2 shows the results of these queries for the training and test data. For both data sets, the use of the official name in a phrase improved MAP modestly, while phrases of all names more than doubled it. Boosting added a small gain in MAP. One interesting finding, not only for us but also another group making their training data results public (J. Savoy, Institut interfacultaire d'informatique, Université de Neuchâtel), was that MAP decreased across the board for the test topics, although the relative performance of the different approaches was comparable. Although we have not yet analyzed why this happened, we suspect it has to do with the decision to limit the test queries to genes with three or more GeneRIFs.

Table 2 - Baseline, phrases, and boosted results for training and test data.

Topics	Run	Retrieved	Relevant	Retrieved & Relevant	Mean Average Precision
Training	Names to words	46115	335	143	0.1584
	Official name as phrase	2829	335	100	0.1998
	All names as phrases	12585	335	215	0.3256
	Official name phrase boosted	12583	335	215	0.3351
Test	Names to words	48021	566	294	0.0741
	Official name as phrase	6197	566	220	0.1372
	All names as phrases	14830	566	419	0.1725
	Official name phrase boosted	14820	566	419	0.1747

Query Expansion Using External Knowledge Resources

The bioinformatics community has produced a wealth of publicly available databases that contain various kinds of information such as:

- gene sequences
- gene clustering
- protein products
- microarray data
- apparent function
- association with diseases
- gene expression in various tissue types

Much of this information is available on the Web, often in HTML and sometimes in XML formats. The volume of data is often huge, with much of it stored in databases that can be accessed only in response to queries via Web forms whose actions are to pass the query to a database and display the results on a Web page. In addition to the large number of primary data sources, such as those maintained by NCBI, there are sites that aggregate various kinds of data. A good example is Source, which is published by a research group at Stanford University (source.stanford.edu) and compiles data from at least five different public databases [6]. The aggregation of data in Source, from multiple databases in an easily processed standardized output, led to its selection as the initial source of information for augmenting the queries.

Methods

For the query expansion experiments, we used the boosted run described above as our baseline query for expansion. Each type of information was added to the baseline query for each gene in a separate run, yielding 13 runs in addition to the baseline run. Data were extracted from Source using a collection of Perl programs. The first program automatically filled in the query form and downloaded the resulting web pages to a local file. Another program read the file for each Web page and extracted the data.

Thirteen pieces of data were collected whenever possible, but not all information was available for all genes. For the 50 genes in the test set, the number of genes for which data from a given category was available ranged from two, for the descriptions associated with accession numbers for mRNA sequences in the NCBI Reference Sequence (RefSeq) records, to 49, for the UniGene Cluster ID. Unfortunately, five of the genes were from Drosophila melanogaster (fruit fly) and had no information about them available from Source. Partial data for those queries was obtained manually from the LocusLink and

FLYBASE (flybase.bio.indiana.edu) databases. One query, PTEN, had two gene entries for the same symbol and name. Most of the data was the same for both, but differed in the UniGene Cluster ID and the tissue types in which they were expressed. Data from both was included in the expansion for that gene. Another query, Prkca in Rattus norvegicus, did not have any data in Source. Thus the data in our experiments come from 49 genes; 44 that are human, mouse, or rat, for which information is available from Source, and five fruit fly genes for which partial information was obtained manually. Queries for which a data item was not available were left unexpanded but were included in the run for that data category.

Additional Perl programs created a new query file for each of the 13 types of information and parsed the reports produced from each run in order to extract and collate the results. These files were then input into Lucene and the results passed with the qrels file to trec_eval.

Results

Our results are summarized in Table 3. None of the thirteen sources of information improved MAP of documents when added to each of the queries for which it was available. Some information categories actually caused a sizeable decrease in MAP. Direct comparisons of MAP must take into account the number of queries for which a data category was available. If data were available for only a few queries, the effect on MAP across all queries would be limited no matter how much the approach might improve it for individual queries. To mitigate this limitation, for each type of information added to the query, we also calculated how many queries were either improved or made worse by the additional information. Despite the overall decline in MAP, as many as one third of queries did see an improvement with the added information, as shown in Table 4.

Addition of identification and accession numbers had little effect on retrieval statistics. As might be expected, the categories that contained the most text words were the most likely to cause changes in retrieval statistics, both positive and negative. In general, the LocusLink summary contained the most text words, but had an almost uniformly negative effect on retrieval performance. That MAP was not the worst of all categories probably reflects the fact that it was only available for 13 of 50 queries. Addition of the top ten tissue types in which the gene is expressed had the most deleterious effect on MAP, and also had a negative effect on most of the queries for which it was available. The information from the Gene Ontology consisted primarily of words or phrases. Effects were both positive and negative, but the negative results outweighed the positive. The textual information about protein function from the SwissProt database had similarly mixed results.

Discussion

Our experiments established baseline results for the TREC Genomics Track. Designating names as phrases improved performance, especially when we used all names for the genes and boosted the weight of the official name.

Table 3 - The performance for query expansion using external resources. The baseline results consist of the best run from the phrases and boosted approach described above. Each subsequent row represents the results when a single piece of information was added to the query for each gene for which the information was available.

Fields Added to Query	Queries expanded	Retrieved	Relevant	Relevant & Retrieved	Mean Average Precision
Baseline	50	14820	566	419	.1747
Chromosome Location	44	25283	566	409	.1655
Locus Link Summary	13	25386	566	313	.1414
SwissProt Accession No.	41	14820	566	419	.1747
Protein function (from SwissProt database)	40	42092	566	317	.1268
Relationships to disease (from SwissProt database)	7	18967	566	350	.1640
Molecular functions of the gene product (from Gene Ontology)	43	45491	566	334	.1400
Biological processes mediated by the gene product (from Gene Ontology)	45	45745	566	263	.1047
Cellular components where the gene is found (from Gene Ontology)	40	41649	566	298	.1297
UniGene Cluster ID	49	14820	566	419	.1747
Top ten tissue types where gene is expressed	42	42796	566	181	.0824
UniGene Accession No.	44	14820	566	419	.1747
Accession No.s for all representative mRNA sequences in the RefSeq database	46	14820	566	419	.1747
Descriptions associated with the mRNA Accession No.s	2	15556	566	338	.1688

Query expansion using information extracted from online databases failed to improve MAP. When individual queries were examined, some benefited from some kinds of expansion. Addition of identifiers and accession numbers for genes used in the various databases had minimal effect on retrieval, suggesting that these rarely appear in the titles and abstracts of journal articles indexed in MEDLINE. Data categories that consist of larger numbers of text words had mixed results for individual queries but deleterious effects overall. The negative results are probably due to the dilution of terms in the query that match terms in the MEDLINE record. It is possible that some sort of filtering of terms added to ensure greater specificity would improve the results. But before such a filter can be designed successfully, it will probably be necessary to do a detailed failure analysis. Examination of the relevant documents that were not returned by the queries, and of relevant documents returned by the baseline query but lost during query expansion, should provide some insight into what kind of filtering might be successful. It also may be possible to identify some common features that could be exploited during query expansion. In general, terms directly related to gene or protein function appear to have the most promise based on the improvement of individual queries with the addition of data from Gene Ontology or SwissProt.

Table 4 - Number of queries showing improved or worsened mean average precision with each alteration.

Fields Added to Query	Queries Expanded	Queries Improved	Percentage of Queries Improved	Queries Worsened	Percentage of Queries Worsened
Baseline	50	N/A	N/A	N/A	N/A
Chromosome Location	44	4	9.1%	25	56.8%
Locus Link Summary	13	1	7.7%	12	92.3%
SwissProt Accession No.	41	0	0%	0	0%
Protein function (from SwissProt database)	40	9	2.2%	29	72.5%
Relationships to disease (from SwissProt database)	7	1	14.3%	6	85.7%
Molecular functions of the gene product (from Gene Ontology	43	11	25.6%	31	72.1%
Biological processes mediated by the gene product (from Gene Ontology)	45	4	8.9%	39	86.7%
Cellular components where the gene is found (from Gene Ontology)	40	6	15.0%	33	82.5%
UniGene Cluster ID	49	1	2.0%	0	0%
Top ten tissue types where gene is expressed	42	5	11.9%	33	78.6%
UniGene Accession No.	44	0	0%	0	0%
Accession No.s for all representative mRNA sequences in the RefSeq database	46	1	2.2%	1	2.2%
Descriptions associated with the mRNA Accession No.s	2	0	0%	2	100%

The data categories added to queries in these experiments are just a small subset of information that is available about genes. Addition of other types of data, from other databases, might be more successful. For example, greater exploitation of information from Gene Ontology might prove useful. Simply further expanding the queries by including terms from child and parent concepts in the hierarchy is unlikely to improve retrieval, but perhaps the concepts could be useful for filtering the terms from other sources.

One source that was not used in these experiments but might prove useful is Online Mendelian Inheritance in Man (OMIM, http://www.ncbi.nlm.nih.gov/Omim/), also available from NCBI. OMIM contains textual summaries about what is known about the role of various genes in human disease. Again, use of the information will probably need to be selective, and possibly undergo filtering. For example, a search on *BRCA1*, a gene related to breast cancer, returns a long summary that includes sections on clinical features of several types of cancers thought to be affected by mutations in this gene, inheritance, clinical management of patients with mutations in this gene, population genetics, gene mapping, molecular genetics, genotype/phenotype correlations, gene function, animal models, allelic variants, and 170 references. OMIM would probably be most useful for a more clinically focused retrieval task than the TREC 2003 Genomics Track task.

One of the features of this task was the generality of the retrieval task, i.e., find all articles about a gene. With minimal constraints on the query, the universe of possible aspects of information is quite large. It is possible that if given a more specific task, query expansion using existing knowledge about a particular aspect of a gene from online databases might make a more positive contribution to the retrieval task. Tailoring the databases used for query expansion to the type of query would be an interesting challenge and perhaps be more likely to produce successful results.

References

1. Evans DA, Lefferts RG, Greffenstette G, Handerson SK, Hersh WR, and Archbold AA. *CLARIT TREC design, experiments, and results. The First Text REtrieval Conference (TREC-1)*. 1992. Gaithersburg, MD: National Institute of Standards and Technology. 251-286.
2. Buckley C, Salton G, Allan J, and Singhal A. *Automatic query expansion using SMART: TREC 3. Overview of the Third Text REtrieval Conference (TREC-3)*. 1994. Gaithersburg, MD: National Institute of Standards and Technology. 69-80.
3. Srinivasan P, *Query expansion and MEDLINE.* Information Processing and Management, 1996. 32: 431-444.
4. Hersh W, Price S, and Donohoe L. *Assessing thesaurus-based query expansion using the UMLS Metathesaurus. Proceedings of the AMIA 2000 Annual Symposium*. 2000. Los Angeles, CA: Hanley & Belfus. 344-348.
5. Wheeler DL, Church DM, Federhen M, Lash AE, Madden TL, Pontius JU, et al., *Database resources of the National Center for Biotechnology.* Nucleic Acids Research, 2003. 31: 28-33.
6. Diehn M, Sherlock G, Binkley G, Jin H, Matese JC, Hernandez-Boussard T, et al., *SOURCE: a unified genomic resource of functional annotations, ontologies, and gene expression data.* Nucleic Acids Research, 2003. 31: 219-223.

TREC2003 Robust, HARD and QA Track Experiments using PIRCS

L. Grunfeld, K.L. Kwok, N. Dinstl and P. Deng
Computer Science Department, Queens College, CUNY
Flushing, NY 11367

1 Introduction

We participated in the Robust, HARD and part of the QA tracks in TREC2003. For Robust track, a new way of doing ad-hoc retrieval based on web assistance was introduced. For HARD track, we followed the guideline to generate clarification forms for each topic so as to experiment with user feedback and metadata.˙ In QA, we only did the factoid experiment. The approach to QA was similar to what we have used before, except that WWW searching was added as a front-end processing. These experiments are described in Sections 2, 3 and 4 respectively.

2 Robust Track

Combining the results of a number of different retrieval outcome generally improves the overall performance [1,2]. The intuitive explanation for this phenomenon is that the different retrievals are more likely to rank the same relevant documents early, than the same non-relevant documents. Consequently, combining retrieval methods that differ greatly often yields better results. Paradoxically we can obtain robust retrieval by adding the results of non-robust methods.

We start with our high performance PIRCS retrieval engine, which is considered robust since it is based on statistical methods, and combine it with retrievals for which the queries were generated based on returned web pages by the Google search engine operating on WWW data. Google queries are of Boolean type and returned results may be less stable. The addition of a single word can dramatically alter retrieval lists, and hence the queries defined by them.

For each robust task topic, our approach is to employ the 60 best-weighted words (except for common words on a stop list) contained in the top 20 web pages (returned by Google) as a reformulated query for our PIRCS engine. The rationale is that because the web is so huge and rich in content, there is a good chance that relevant pages containing the content terms of the original topic exist in the web. These pages will probably rank near the top by the Google, and may be rich in content terms related to the topic. These terms can therefore define, for our ad-hoc processing, useful alternate queries that can lead to different retrievals, and which could be useful for combining with the original retrieval list that is based on a query generated directly from the topic statement. The next section describes how we form Google queries from the description section of the original topic statement.

2.1 Generating Google Queries from Topic Statements

The Google search engine (http://www.google.com) accepts queries in a form similar to simplified Boolean expressions. It allows one to specify conjunctive clauses by having terms placed adjacent to each other, disjunctive clauses by placing the string OR between terms, and negated terms with a '-' prefixing them. Phrase matching is allowed by having words surrounded by double quotes. Un-stemmed words are used.

We employ three different strategies to create queries for Google retrieval. The queries are formed using only the Description section of a topic. Since previous experience has shown that retrievals combine better if they are dissimilar, our aim is to make the queries as different as possible. The three query formation strategies (identified by the names qds, qdp and qdt) are described below:

qds queries:

This simplest approach just employs sequentially the first six content words from a topic in a logical AND fashion. Caution is needed to avoid using too many words; otherwise nothing is retrieved. As an example consider the original Query 378 and the generated Google query G378:

Q378 - Identify documents that discuss opposition to the introduction of the euro, the European currency.

G378 - opposition introduction euro, European currency.

This method works fairly well except when queries are long. Consider the following:

Q610 - Find claims made by U.S. small businesses regarding the adverse impact on their businesses of raising the minimum wage.

G610 - claims U.S. small businesses adverse impact

The generated query G610 does not include important terms like: “raising”, “minimum”, “wage”, and the returned pages are not satisfactory.

qdp queries:

This and the following qdt method attempt to create Google queries by identifying important words in the topic based on Dekang Lin's MINIPAR parser [3] available at http://www.cs.ualberta.ca/~lindek/minipar.htm. MINIPAR is a general-purpose parser for an English sentence, identifies phrases, and generates a dependency structure where each word modifies at most one head word. Our strategy is to select the six best nouns based on the following order of priority: nouns appearing in phrases, nouns that designate a person, country, location, corporation, language or title, followed by other nouns. As an example, Q610 above generates the following:

G610 - minimum wage U.S. businesses impact claims

where “minimum”, “wage” come first since it is part of a phrase, followed by “U.S.”, a country, then followed by other nouns. The exclusion of verbs and adjectives sometimes harms performance. Consider Query 644:

Q644 - Identify documents that discuss exotic species of animals that are imported into the U.S. or animals that are imported into the U.S. or U.K.

G644 - U.S. U.K. species animals

The resultant G644 misses out the important verb “imported” and adjective “exotic”. Another example is:

Q362 - Identify incidents of human smuggling.

G362 - incidents smuggling

which misses the important adjective “human”.

qdt queries:

This strategy first selects the phrases identified by MINIPAR. If there are none, other phrases defined by patterns (N1 gov N2), (N1 N2) (N2 N3) (if the 3-word phrase N1 N2 N3 are defined), and (A gov N) are then used to select words in this order. If query is < 6 words, nouns and verbs are added (AND'ed) until query has 6 words. For example, Q362 becomes:

G362 - "human smuggling" incidents

where the quotes tell Google that “human smuggling” needs to be adjacent. Another example is:

Q643 - What harm have power dams in the Pacific northwest caused to salmon fisheries?

G643 - "Pacific northwest" "salmon fisheries" harm dams

which includes most of the content terms. A problem with this method is that some queries become too specific and no web pages are returned.

As an example, we include in the following the output of Q643 after analyzed by MINIPAR:

```
>(
E1    (()      fin C    *      )
1     (What    ~ N      E1     whn     (gov fin))
2     (harm    ~ N      3      s       (gov have))
3     (have    ~ V      E1     i       (gov fin))
E3    (()      harm N   3      subj    (gov have)      (antecedent 2))
E0    (()      fin C    3      fc      (gov have))
4     (power   ~ A      5      mod     (gov dam))
5     (dams    dam N    10     s       (gov cause))
6     (in      ~ Prep   5      mod     (gov dam))
7     (the     ~ Det    9      det     (gov northwest))
8     (Pacific ~ N 9 nn (gov northwest)(atts (sem (+location))))
9     (northwest        ~ N    6       pcomp-n(gov in))
10    (caused  cause V  E0     i       (gov fin))
E4    (()      what N   10     obj     (gov cause)     (antecedent 1))
E5    (()      dam N    10     subj    (gov cause)     (antecedent 5))
11    (to      ~ Prep   10     mod     (gov cause))
12    (salmon ~ N       13 nn  (gov fishery)(atts (sem (+gname +male))))
13    (fisheries        fishery N      11      pcomp-n(gov to))
)
>
```

2.2 Generating PIRCS Queries from Retrieved Web Pages

Each of the Google queries in the previous sub-section was used to retrieve the top 20 web pages. From these html tags, non-text items and some common words are removed. A query is then created using the 60 best-weighted words. Weight of a word is defined as sum (over all web pages in which it occurs) of its frequency divided by html text length. Exceptionally long pages are skipped. The alternate query for our target retrieval is composed of these 60 words normalized by the least weight. For example, G643 in the previous qds section leads to the following alternative query for our PIRCS retrieval:

SALMON 10 RIVER 9 DAMS 8 WATER 6 COLUMBIA 6 FISH 5
POWER 5 DAM 4 NORTHWEST 3 FEDERAL 3 SNAKE 3
BASIN 2 SPECIES 2 OREGON 2 .. + 46 other single terms

Note that this alternative query includes important geographical information such as “Columbia River”, “Snake River”, “Columbia Basin” and “Oregon” that are absent in the original topic description section. This query has good performance.

2.3 Retrieval based on Data Fusion

The final robust retrieval submission is based on combination of retrieval lists: using our normal query qd (description) or qa (all sections) obtained from a topic statement, and from alternative queries as discussed in the previous sub-section. All these query types undergo retrieval using our PIRCS engine on the given collection. Experiments have been performed using the description section of a topic only (pircRBd?, where ?=1 means normal PIRCS retrieval with PRF (pseudo-relevance feedback), ?=2 means web-assisted retrieval using combination strategy (i), while ?=3 means combination strategy (ii). The combination strategies are: (i) (qd 0.4) ⊕ (qds 0.2) ⊕ (qdp 0.2) ⊕ (qdt 0.2), and (ii) (qd 0.5) ⊕ (qdt 0.5). The symbol ⊕ is used to denote ranked list combination, and each retrieval list is weighted by the given factors. Experiments using all sections were also submitted pircRBa?: ?=1 means normal PIRCS retrieval, and ?=2 means (qa 0.3) ⊕ (pircRBd1 0.7). Note that pircRBa2 not only make use of web-assistance, but also combine description with all-section query results.

2.4 Robust Track Results and Discussions

Results of our submissions are shown in Table 1: split into 50 old topics, 50 new topics and all 100 merged. The 50 old topics may be considered as training topics since their relevant answers are known from previous TREC experiments, and the 50 new as testing set. Evaluation measures shown are the standard ones used in TREC: Rel.Ret = total relevant items in the 1000 retrieved documents, MAP = mean average precision, R-Pre = average precision value at the exact number of available relevant documents for each query, and Pnn = average precision at nn documents retrieved, where nn = 10, 20 or 30. Two new measures for Robust track are: number of topics without relevant documents at 10 retrieved '# no-relv-@10' and the 'area' measure which is a weighted sum of the precision for the worst-25% of the topics. An immediate observation is that all effectiveness values are much lower for the old topics than for the new, showing that the 50 old topics are much more difficult for retrieval with this TREC-8 collection of documents. In particular, the 'area' values are less than .01 for old 'description' queries, while they vary from .05 to .08 for new queries.

In Table 1, we also show results of our PIRCS retrieval with PRF (pircRBd1 for description query and pircRBa1 for all-section queries) as basis for comparison. Of the two web-assisted description runs pircRBd2 and pircRBd3, the former has better performance. The latter makes use of qdt queries only and is not sufficiently robust, and more combination of retrievals appears useful. Using the 100 query results, one sees that our method of web-assisted retrieval brings substantial improvements for the Robust track measures: reducing the '# no-relv-@10' from 16 in pircRBd1(100) to 8 for pircRBd2(100), while the 'area' value increases from .0122 to .0219, an 80% boost. There are also smaller improvements in the other measures such as MAP, P10, etc. Similar improvements are also observed for the all-section queries.

Table 2 shows percentage improvements of certain measures of the web-assisted runs (for both description and all section queries) compared to their respective basis runs and separated into old and new queries. For the training set (Old-50), the web-assisted retrievals have double-digit percentage improvements compared to basis PIRCS retrieval for the description queries. For the testing (New-50) set, only pircRBd2 has slight improvements. We might have over-trained and the strategy does not carry over to testing set well, or that it is difficult to attain increases for the better performing queries of the New-50. For the long queries, however, except for slight decrease of 1% in two measures, other measures show good improvements over the Basis run in both old training and new testing sets.

Run ID	Rel.Ret	MAP	R.Pre	P10	P20	P30	# no-relv-@10	area
Total No. of Relevant Documents: old50 set=4416, new50 set=1658, all100=6074								
Query size: description section only								
pircRBd1 (old)	2216=50%	.1526	.1887	.3220	.2810	.2393	14/50=28%	.0045
pircRBd1 (new)	*1534=93%*	*.4022*	*.3963*	*.5200*	*.4230*	*.3500*	*2/50=4%*	*.0804*
pircRBd1 (100)	**3750=62%**	**.2774**	**.2925**	**.4210**	**.3520**	**.2947**	**16/100=16%**	**.0122**
Web-assisted runs: (qd 0.4) ⊕ (qds 0.2) ⊕ (qdp 0.2) ⊕ (qdt 0.2)								
pircRBd2 (old)	2377=54%	.1772	.2148	.3820	.3240	.2787	6/50=12%	.0091
pircRBd2 (new)	*1565=94%*	*.4029*	*.3845*	*.5320*	*.4170*	*.3467*	*2/50=4%*	*.0819*
pircRBd2 (100)	**3942=65%**	**.2900**	**.2996**	**.4570**	**.3705**	**.3127**	**8/100=8%**	**.0219**
Web-assisted runs: (qd 0.5) ⊕ (qdt 0.5)								
pircRBd3 (old)	2287=52%	.1754	.2106	.3760	.3250	.2727	7/50=14%	.0065
pircRBd3 (new)	*1444=87%*	*.3878*	*.3781*	*.5220*	*.4080*	*.3273*	*2/50=4%*	*.0540*
pircRBd3 (100)	3731=61%	.2816	.2944	.4490	.3665	.3000	9/100=9%	.0165
Query size: all sections of topic								
pircRBa1 (old)	2562=58%	.1796	.2282	.3640	.3230	.2867	6/50=12%	.0136
PircRBa1 (new)	*1494=90%*	*.4405*	*.4150*	*.5440*	*.4550*	*.3800*	*3/50=6%*	*.0716*
pircRBa1 (100)	**4056=67%**	**.3101**	**.3216**	**.4540**	**.3890**	**.3333**	**9/100=9%**	**.0203**
Web-assisted runs: (qa 0.3) ⊕ (pircRBd1 0.7)								
pircRBa2 (old)	2641=60%	.1854	.2234	.4000	.3340	.2907	5/50=10%	.0135
pircRBa2 (new)	*1575=95%*	*.4369*	*.4159*	*.5760*	*.4520*	*.3760*	*1/50=2%*	*.1062*
pircRBa2 (100)	**4217=69%**	**.3111**	**.3197**	**.4880**	**.3930**	**.3333**	**6/100=6%**	**.0290**

Table 1: Robust Retrieval - Summary for All Submitted Runs -- Lenient Evaluation

	Old-50 training set			New-50 testing set		
	MAP %imp	P10 %imp	area %imp	MAP %imp	P10 %imp	area %imp
Query size: description section only						
pircRBd1	.1526 *	.3220 *	.0045 *	.4022 *	.5200 *	.0804 *
Web-assisted runs						
pircRBd2	.1772 +16	.3820 +19	.0091 +102	.4029 +0	.5320 +2	.0819 +2
pircRBd3	.1754 +15	.3760 +17	.0065 +44	.3878 -4	.5220 +0	.0540 -33
Query size: all sections of topic						
pircRBa1	.1796 *	.3640 *	.0136 *	.4405 *	.5440 *	.0716 *
Web-assisted runs						
pircRBa2	.1854 +3	.4000 +10	.0135 -1	.4369 -1	.5760 +6	.1062 +48

Table 2: Comparing Web-Assisted to Basis Retrieval – Training and Testing Sets

Run ID	Best	Median AP (>/=/<)	Worst	MAP	% no-relv-@10	area	Worst25% MAP
pircRBd1	2	64/3/33	1	0.2774	16%	0.0122	0.0310
pircRBd2	**1**	**74/2/24**	**0**	**0.2900**	**8%**	**0.0219**	**0.0478**
pircRBd3	4	73/1/26	2	0.2816	9%	0.0165	0.0418
pircRBa1	11	79/0/21	0	0.3101	9%	0.0203	0.0467
pircRBa2	**7**	**86/2/12**	**0**	**0.3111**	**6%**	**0.0290**	**0.0622**

Table 3: Comparing PIRCS 100-Topic Results with Median

Compared to all submissions, our results perform very favorably. Table 3 shows the comparison with median AP values. For example, the web-assisted pircRBd2 average precision has 74 topics better than median, 2 equal and 24 worse. One of the 74 has best average precision and none has worst. PircRBa2 is even better.

2 HARD Track

'HARD' (High Accuracy Retrieval from Documents) is a new 2003 extension to previous ad-hoc retrieval experiments. Its purpose is to study the effects of user feedback and metadata on retrieval effectiveness. After a first round of retrieval by a search engine ('Basis Retrieval'), the system is allowed to solicit user feedback by creating a 'Clarification Form' concerning the topic. Users are allowed three minutes time per topic to answer questions presented in the form. Afterwards, the system is able to make use of the form data, as well as further on-topic metadata that is provided, in order to improve on the Basis Retrieval.

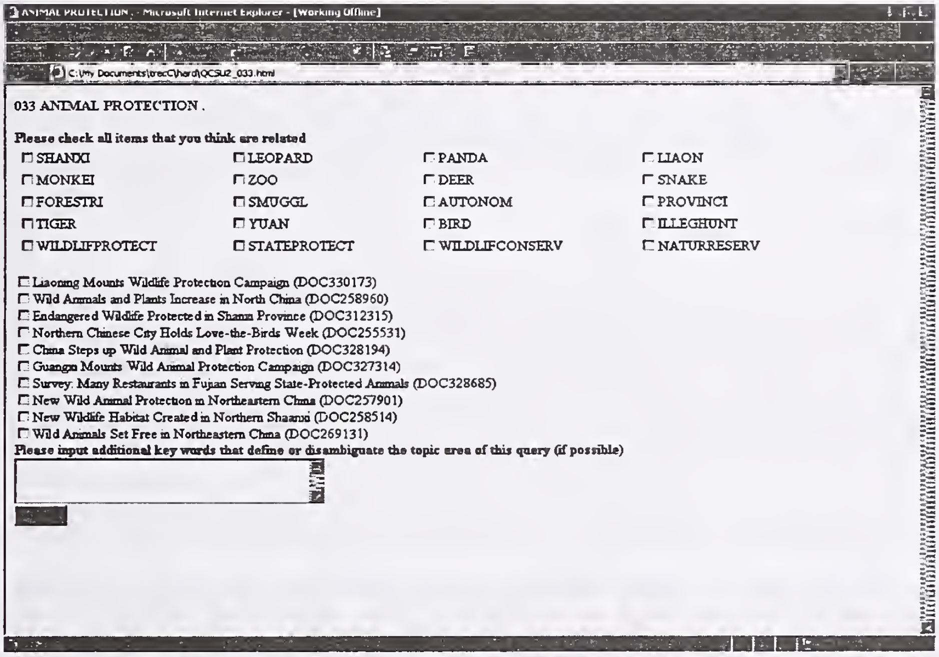
2.1 Basis Retrieval

We employ our standard PIRCS ad-hoc processing and retrieval to provide first-round results for the user. This involves an initial retrieval plus a pseudo-relevance feedback (PRF) processing using 20 top documents and 60 best terms (20d60t). This 2-stage retrieval is called pircHDBt1, and is our 'Basis Retrieval' results in this HARD track environment. This involves only the title section of a topic as query. Another basis retrieval using both title and description sections as query is also submitted and denoted as: pircHDBtd1. We have also captured the above ad-hoc processing using only the first stage retrieval. Their retrieval results: pircHDBt0 and pircHDBtd0 are not submitted. An alternative is to use the first stage retrieval as basis retrieval. This saves second stage retrieval time, provides data faster for the user, but the basis would generally not be as high since PRF usually brings higher average precision values.

2.2 Clarification Form Design

After the Basis Retrieval of the previous section, three clarification forms are generated automatically and denoted as: C1 (submitted name QCSU1), C2 (QCSU2) and C3 (QCSU3). The general layout of our clarification form consists of three sections for users to make relevance judgment: i) candidate *related terms*, and ii) candidate *related document* titles or first sentences; iii) user *keyword input*. Each related term or document is associated with a radio button for clicking 'yes ', i.e. relevant. C1 makes use of WordNet [4] to obtain related terms to a query and display them in the clarification form. Clarification form C1 also does not display any related documents, and hence it does not rely on any retrieval and therefore less costly. C2 needs retrieval processing to define top documents and best terms for display. C2 makes use of the title section of a topic only for the initial retrieval. C3 is similar to C2, but uses both title and description sections of a topic for retrieval. The keyword input section is a scrollable. Since a user has only 3 minutes to complete a form, he or she would not have much time left for keyword input even though the scrollable window allows for the space.

For C1, we employ the title section of a topic and define each consecutive two-word as a phrase. Each phrase is passed to WordNet to pick up synonyms, and then the single words. The synonyms obtained are displayed in the 'related term' section of C1 for the user to judge. Some topics may fail to pick up synonyms; for these the related term section may be blank or the query words themselves. Users moreover can type in any words they deem important for a topic in the keyword input section of the clarification form. An example is Topic Hard-044 "Amusement Park Safety". The phrase "amusement park" gets Wordnet to return the following synonyms: "amusement park, funfair, pleasure ground". The last two phrases would not be obtained if words were used individually. Thus, the single words "amusement" gets synonyms "entertainment, amusement", and "park" gets "park, parkland, commons, common, green, ballpark, Mungo park, parking lot, car park, parking area". Since a user is present to judge these terms, presence of noise terms is tolerable.


ANIMAL PROTECTION . - Microsoft Internet Explorer - [Working Offline]
C:\My Documents\trecC\hard\QCSU2_033.html

033 ANIMAL PROTECTION .

Please check all items that you think are related

□ SHANXI	□ LEOPARD	□ PANDA	□ LIAON
□ MONKEI	□ ZOO	□ DEER	□ SNAKE
□ FORESTRI	□ SMUGGL	□ AUTONOM	□ PROVINCI
□ TIGER	□ YUAN	□ BIRD	□ ILLEGHUNT
□ WILDLIFPROTECT	□ STATEPROTECT	□ WILDLIFCONSERV	□ NATURRESERV

□ Liaoning Mounts Wildlife Protection Campaign (DOC330173)
□ Wild Animals and Plants Increase in North China (DOC258960)
□ Endangered Wildlife Protected in Shanxi Province (DOC312315)
□ Northern Chinese City Holds Love-the-Birds Week (DOC255531)
□ China Steps up Wild Animal and Plant Protection (DOC328194)
□ Guangxi Mounts Wild Animal Protection Campaign (DOC327314)
□ Survey: Many Restaurants in Fujian Serving State-Protected Animals (DOC328685)
□ New Wild Animal Protection in Northeastern China (DOC257901)
□ New Wildlife Habitat Created in Northern Shaanxi (DOC258514)
□ Wild Animals Set Free in Northeastern China (DOC269131)

Please input additional key words that define or disambiguate the topic area of this query (if possible)

Fig.1: C2 Clarification Form for Query Hard-033

For C2 (title) and C3 (title + description), the related document section comes from the title or first sentence of the 10 top-ranked documents of a 2-stage ad-hoc retrieval. (Initially our design was to use the top documents from an initial retrieval. However, after the conference we discovered that an un-intended mix-up of files actually led to the use of the 2nd stage results instead). 20 top-ranked terms are also displayed in the related term section. An undesirable situation arises because some of the feedback terms are actually stems not regular words. Porter's stemming algorithm has been employed, and the process is irreversible. Some other terms may be a combination of two stems into one string, which are the result of our adjacent two-word phrase indexing. These may be useful as indexing terms but not suitable for user browsing. We hope to remedy these situations in future enhancements. The user can click on whichever term or document they think is useful for a topic. It is possible that, for some difficult queries, none of the suggested terms or documents is related. However, the keyword input section is available for the user to type in additional words so as not to get frustrated. We believe that a user can complete the clarification form – click through 30 items and input some key words -- in three minutes time. An example of a C2 form is shown in Fig.1.

2.3 Final Retrieval – Document Level

The system has to decide on how to make use of the clarification form data to do further processing. Our strategy is to employ the 'user-clicked' related terms and the 'keywords' typed by the user to expand on the original query (either the title section or the title + description of each topic). Typed words can have typographic errors. We use Google's spell-check facility to remedy the situation. Afterwards, these keywords have to be stemmed to be compatible with other existing index terms. Repeated mention of the same term is kept to provide higher weight. Each expanded query is used for a fresh full 2-stage retrieval. However, during pseudo-relevance feedback (except for C1 form), the 'user-clicked' documents are guaranteed to be among the 20 feedback documents used. Term expansion is still kept at 60. These procedures provide the submissions pircHDC1t1, pircHDC2t1 and pircHDC3td1. It is to be noted that because of time constraints on the part of the assessors, 4 topics in C3 (036, 048, 053, 105) were not filled in. Results of these queries default back to those of the Basis Retrieval.

For clarification forms C2 and C3, we have two further submissions: pircHDC2t2 and pircHDC3td2. In many queries, the related document section receives very few or no 'clicks'. We used a threshold of less than 3. This suggests evidence that the Basis Retrieval results are not good (assuming the user can do correct judgment using the title or first sentence of the retrieved document), and may imply that the topic is a difficult one. For these topics (16 for C2: 36, 59, 87, 115, 117, 124, 154, 177, 180, 186, 187, 220, 226, 228, 231, 235 and 13 for C3: 59, 87, 117, 154, 177, 194, 203, 215, 217, 220, 228, 231, 235), we disable the 2nd stage retrieval during final retrieval and used the initial retrieval results instead. The idea is that quite often for difficult topics, 2nd stage retrieval can lead to worse results compared to initial retrieval.

2.4 Final Retrieval – Phrase Level

After the clarification data has been filled in, additional information in the form of metadata such as: purpose of retrieval, document genre wanted, user familiarity with topic, granularity of result, sample of relevant texts are also released concerning each topic. We only focus on the granularity metadata which can have values: document, passage, sentence and any. There are 16 (18-2 removed) queries that have the requirement of granularity = passage or sentence. For these, each of their retrieval lists is passed to our PIRCS-QA system (see also Section 3) for further processing to try to isolate a small text extent as answer for the topic. Our QA system can be summarized as follows [5]:

1) returning n top-ranked subdocuments from PIRCS retrieval using a query with stemming and stop-word removal;
2) scoring and returning top-ranked sentences from the subdocuments with respect to the general context of the question keywords using a set of eleven heuristics -- both raw and stemmed words were taken into account;
3) analyzing specific properties of the question to obtain its expected answer types, and assigning one of four functional modules that use keywords, meta-keywords and patterns to detect possible answers and add bonus weights to top-ranked sentences for selection purposes;
4) extracting answer strings of required size from top candidate sentences based on the previous question analysis with rules and heuristics for entity definition or identification.

Instead of evaluating sentences we evaluate paragraphs. They are detected by the </p> tag or blank line or an indented line. Since more than one result was submitted and the order is important, we increased the retrieval score bonus. A document-offset bug which was present in our QA track this year was also fixed. The run-id's with the phrase processing are identified with a 'p' at the end like: pircHDC2tp.

2.5 HARD Track Results and Discussion

2.5.1 Results of Document-Level Evaluation

Table 4 below shows results of our submitted and some of the un-submitted runs. Each column has "%imp" denoting "% improvement" from the Basis Retrieval. Relevance judgment is of type 'hard' where partially relevant documents are excluded as relevant. 'Soft' type judgment results that include partially relevant documents are shown in Table 5. The following TREC measures are tabulated: Rel.Ret (number of relevant documents within the top-ranked 1000), MAP (mean average precision over the 48 queries), Pnn (average precision at nn retrieved documents where nn = 10, 20 and 30), and RPre (average precision at rr retrieved where rr = the exact number of relevant documents for each query).

Concerning the short, title only query (average 2.8 terms) results, Table 4 shows that our submitted Basis Retrieval (pircHDBt1) is substantially better than the un-submitted first stage retrieval (pircHDBt0) without PRF, and provides a much higher basis to compare with the final retrievals using clarification or metadata. The queries using Wordnet-based clarification forms (C1) for expansion have average 11.0 terms, and the final retrieval pircHDC1t1 improves over basis 3 to 11% in various measures except for P30 with a decrease of 1%. Using forms C2, the queries have average 12.6 terms and the pircHDC2t1 result improve over basis from 7 to 15% except for Rel.Ret with a decrease of 2%. Looking at MAP values, pircHDC1t1 employing the less costly C1 forms leads to slightly higher performance compared to pircHDC2t1. However, pircHDC2t1 has better effectiveness in low-recall high-precision retrieval region, achieving double-digit improvements for P10-30 over the basis; pircHDC1t1 has more erratic performance: from 10% increase in P10 to 1% decrease in P30. C1 Wordnet only suggests synonyms of the different senses of a word and for 21 queries did not suggest new word. C2 always have some suggested terms that may be related, not necessarily synonyms. It seems that the three minutes spent by a 'user' can bring out significant precision improvements (>5%) over the basis retrieval.

Since there are two sources (related terms and keyword input) of user feedback to augment the original title query, we investigate to see which source is more useful and whether both are necessary. The un-submitted runs ~pircHDC1t1term and ~pircHDC1t1key show results using either the clicked related terms or the typed keywords only. Each leads to an average of 4.8 and 9.2 query terms respectively. Similarly for C2 runs ~pirHDC2t1term (7.5 terms) and ~pircHDC2t2key (8.3 terms). Thus, related terms from Wordnet provide on average only 4.8-2.8=2 relevant words while PRF provides 7.5-2.8=4.7. This reinforces previous experience that general purpose thesauri often miss the query words or may not contain the right sense of query terms (for a user to click). In both C1 and C2 cases, use of only one source of feedback data performs worse than using both, and often worse than the basis values. When both sources are used, the original title, input keywords and clicked related terms often overlap. This is equivalent to user weighting some good terms higher, and may contribute to better results. Comparing pircHDC1t1term with pircHDC1t1key rows, the former is uniformly worse, leading us to conclude that Wordnet supplied terms are less useful than typed keywords. On the other hand, comparing pircHDC2t1term and pircHDC2t1key, the former has better results except for Rel.Ret, leading us to believe that PRF supplied terms are more useful than typed keywords. Different forms have different keywords typed, probably by different users.

Our submitted run pircHDC2t2 that disables PRF when user clicks fewer than 3 relevant documents, was in error due to a use of wrong files. The corrected run is shown as ~pircHDC2t2. It is a bit worse compared to pircHDC2t1 and the procedure is not effective. Of the 15 queries affected, only 5 is better to ignore PRF processing or little change. The additional user-typed keywords may make a query better. It is also qualitatively similar in the case of using title+description queries (pircHDC3td2).

Overall, results for the longer title+description queries (9.2 terms average) improves slightly over the title only run: e.g. the basis run pircHDBtd1 MAP value of .3277 is better by 2% compared to pircHDBt1 title basis value of .3219. The run with clarification form C3, pircHDC3td1 with average of 18 query terms, improves over the basis pircHDBtd1 between 8 to 15% in all measures except for a 1% decrease in Rel.Ret. As in title runs, the effect of using either the clicked related terms (un-submitted ~pircHDC3td1term, average 15.5 terms) or the typed keywords (~pircHDC3td1key, average 14.3 terms) is to depress performance compared to using both. Also, clicked terms are preferred over user input keywords as in the title only results. This run provides the best overall MAP value of 0.3604 and R-Pre of 0.3875 for all submitted runs.

Our official title run pircHDC2t1 compares favorably with other submitted runs, with 37 queries (1 best) above median, 10 below median (1 worst) and 1 equal to median average precision. Table 5 provides results of soft evaluation, i.e. when partially relevant documents are also treated as relevant. Behavior is similar to that of hard evaluation in Table 4.

Run-ID	**Rel.Ret %imp**	**MAP %imp**	**R-Prec %imp**	**P10 %imp**	**P20 %imp**	**P30 %imp**
hard criteria evaluation (48 queries; total 5123 relevant documents)						
Query size = title						
No clarification form used (average query size 2.8 terms)						
~pircHDBt0	3482 -11	.2170 -33	.2558 -26	.3500 -21	.3094 -27	.2917 -29
pircHDBt1	3893 *	.3219 *	.3460 *	.4417 *	.4229 *	.4132 *
Clarification form with Wordnet (average query size 11.0 terms)						
pircHDC1t1	3999 +3	.3583 +11	.3740 +8	.4854 +10	.4344 +3	.4111 -1
Using clicked Wordnet terms only (average query size 4.8 terms)						
~pircHDC1t1term	3766 -3	.2995 -7	.3206 -7	.4292 -3	.3792 -10	.3625 -12
Using input keywords only (average query size 9.2 terms)						
~pircHDC1t1key	3802 -2	.3197 -1	.3437 -1	.4604 +4	.4083 -3	.3958 -4
Clarification form with PRF data (average query size 12.6 terms)						
pircHDC2t1	3812 -2	.3536 +10	.3717 +7	.5083 +15	.4802 +14	.4535 +10
Using clicked PRF terms only (average query size 7.5 terms)						
~pircHDC2t1term	3507 -10	.3021 -6	.3242 -6	.5042 +14	.4635 +10	.4236 +3
Using input keywords only (average query size 8.3 terms)						
~pircHDC2t1key	3564 -8	.2900 -10	.3079 -11	.4437 +0	.4083 -3	.3785 -8
pircHDC2t2	3589 -8	.3048 -5	.3191 -8	.4542 +3	.4354 +3	.4125 -0
~pircHDC2t2	3791 -3	.3469 +8	.3648 +5	.5063 +15	.4812 +14	.4535 +10
Query size = title + description						
No clarification form used (average query size 9.2 terms)						
~pircHDBtd0	3606 -9	.2719 -17	.3075 -8	.4417 -9	.3875 -7	.3424 -9
pircHDBtd1	3958 *	.3277 *	.3360 *	.4875 *	.4167 *	.3743 *
Clarification form with PRF data (average query size 18.0 terms)						
pircHDC3td1	3915 -1	.3589 +10	.3813 +13	.5271 +8	.4802 +15	.4306 +15
Using clicked PRF terms only (average query size 15.5 terms)						
~pircHDC3td1--term	3766 -5	.3388 +3	.3574 +6	.4875 +0	.4323 +4	.4049 +9
Using input keywords only (average query size 14.3 terms)						
~pircHDC3td1--key	3699 -7	.3033 -7	.3169 -6	.4604 -6	.3937 -6	.3632 -3
pircHDC3td2	3901 -2	.3604 +10	.3875 +15	.5146 +6	.4740 +14	.4236 +13

Table 4: 'HARD' Retrieval with Hard Evaluation (~ denotes un-submitted data)

Run-ID	Rel.Ret %imp	MAP %imp	R-Prec %imp	P10 %imp	P20 %imp	P30 %imp
soft criteria evaluation (48 queries; total 7576 relevant documents)						
Query size = title						
~pircHDBt0	4938 -8	.2548 -30	.2893 -25	.4460 -17	.4140 -21	.3833 -26
pircHDBt1	5372 *	.3650 *	.3857 *	.5396 *	.5260 *	.5167 *
pircHDC1t1	5533 +3	.4069 +11	.4250 +10	.5979 +11	.5469 +4	.5299 +3
pircHDC1t1term	5251 -2	.3449 -6	.3703 -4	.5375 +0	.4958 -6	.4764 -8
pircHDC1t1key	5202 -3	.3585 -2	.3882 +1	.5646 +5	.5094 -3	.4937 -4
pircHDC2t1	5215 -3	.3986 +9	.4242 +10	.6500 +20	.6104 +16	.5799 +12
pircHDC2t2term	4726 -12	.3188 -13	.3512 -9	.6021 +12	.5479 +4	.5035 -3
pircHDC2t2key	4857 -10	.3191 -13	.3501 -9	.5438 +1	.5208 -1	.4868 -6
pircHDC2t2	4891 -9	.3314 -9	.3454 -10	.5583 +3	.5323 +1	.5062 -2
~pircHDC2t2	5201 -3	.3902 +7	.4156 +8	.6479 +20	.6094 +16	.5771 +12
Query size = title + description						
~pircHDBtd0	5069 -7	.3037 -17	.3387 -14	.5600 -5	.4900 -9	.4473 -7
pircHDBtd1	5430 *	.3656 *	.3932 *	.5875 *	.5365 *	.4826 *
pircHDC3td1	5470 +1	.3934 +8	.4131 +5	.6271 +7	.5896 +10	.5403 +12
PircHDC3td1term	5203 -4	.3667 +0	.3926 -0	.5771 -2	.5281 -2	.4979 +3
PircHDC3td1key	5118 -6	.3363 -8	.3532 -10	.5667 -4	.5010 -7	.4708 –2
pircHDC3td2	5445 +0	.3937 +8	.4133 +5	.6167 +5	.5865 +9	.5319 +10

Table 5: 'HARD' Retrieval with Soft Evaluation (~ denotes un-submitted data)

2.5.2 Results of Phrase-Level Evaluation

Table 6 shows results of our passage retrieval. The official evaluation measures are P10, R-Pre and F-measure at 30 documents retrieved (F30). As discussed before, run-id's that end with 'p' undergo special passage processing and return a passage list, i.e. document id with a text extent. Other runs without 'p' return document id lists only. We may consider them as document id with a text extent equal to the whole document length, and each document id contributes one retrieval result only. Errors were also discovered for the pircHDC2tp and pircHDC3tdp runs: there were 18 queries out of 42 that went through our QA

Run-ID	P10	%imp	R-Prec	%imp	R30	%imp	P30	%imp	F(30)	%imp
hard criteria evaluation (42 queries)										
Query size = title										
pircHDBt1	.2809	*	.1810	*	.2359	*	.2491	*	.1491	*
pircHDC1t1	.3152	+12	.2335	+29	.2724	+15	.2369	-5	.1479	-1
pircHDC1tp	.3770	+34	.3195	+77	.1839	-22	.3081	+24	.1403	-6
pircHDC2t1	.3209	+14	.2145	+19	.2501	+6	.2766	+11	.1549	+4
pircHDC2tp	.3754	+34	.2508	+39	.1426	-40	.3191	+28	.1269	-15
~pircHDC2tp	.3829	+36	.2595	+43	.1762	-25	.3336	+34	.1423	-5
Query size = title+description										
pircHDBtd1	.3186	*	.1699	*	.2404	*	.2316	*	.1280	*
pircHDC3td1	.3359	+5	.2141	+26	.2922	+22	.2374	+3	.1452	+13
pircHDC3tdp	.3353	+5	.2555	+50	.1746	-27	.2772	+20	.1283	+0
~ pircHDC3tdp	.3438	+8	.2575	+52	.1812	-25	.2797	+21	.1294	+1

Table 6: 'HARD' Passage Retrieval with Hard Evaluation (~ denotes un-submitted data)

processing. The rest were supposed to be the document-level retrieval from the basis pircHDC1t1 and pircHDC3td1 respectively; however we erroneously used the first stage retrievals pircHDC2t0 and pircHDC3td0 instead. The corrected runs are shown in Table 6 as ~pircHDC2tp and ~pircHDC3tdp.

It is seen that for all passage runs with run-id ‘p’, precision values are high, but recall and F(30) values are low compared to the basis. One reason precision values are high is that precision calculation favors shorter passages than whole documents. One reason recall values are low is that relevant documents always cover all relevant materials, especially if it has multiple relevant passages. Comparing pircHDC1t1 (which has Wordnet form C1 feedback) with the basis pircHDBt1, we see P10 improves but F(30) decreases by 1% due to individual precision and recall values at 30 retrieved. Looking at pircHDC1tp where the retrieved list are passage-level, precision values improve over basis probably also because relevant passages are promoted earlier. However, recall values at 30 retrieved are low leading to decreases in F(30).

3 Question-Answering Track

The Internet is a great storehouse of information and facts. Millions search it daily and use it to solve their information needs. It is not surprising therefore that a number of participants in the QA track make use of it as a source of knowledge. This year we join this trend and extend our QA system to use results of the Google search engine.

We extract answers from Google retrievals in two different ways. For certain question types we developed reliable patterns, which identify the answer term. For other questions we use the most frequent word from the snippets returned by Google. For questions that require named entity recognition we make use of Minipar’s NE capability.

We submitted three runs for the passage track. pircsQA1 is virtually identical to our QA system for the 2001 QA track. It uses the top 100 documents retrieved by our PIRCS search engine. It combines probabilistic IR methods with search pattern recognition to select the highest-ranking sentence.

PIRCS does not always return documents with the answer and sometimes the document with the correct answer is ranked very low. To remedy this situation we have merged the original query submitted to PIRCS with possible answer extracted from a Google answer snippets. pircsQA2 uses the top 100 documents created by this retrieval. pircsA3 makes use of a different strategy to utilize suggested answers from Google: extra bonus is added to sentences that contain them.

The official results are quite low, due to a bug in the system, which caused the document offsets to be calculated incorrectly. Here we report our own unofficial evaluation, for the 382 queries which had answers in the document collection.

	Score	% improve
pircsQA1	0.249	
pircsQA2	0.264	6.32%
pircsQA3	0.338	35.79%

Table 7: Unofficial results for 382 queries with answers.

pircsQA1 performed worse than in 2001, which indicates that the queries are getting harder. pircsQA2 shows that results can be improved by enhancing the query submitted to the front end search engine, but not by much. The greatest improvement comes from searching the test collection for answers found in the Web. This makes the task somewhat unrealistic, sometimes the answer is available, but we don’t tell the user, because it is not in the document collection. Our systems performance can be improved by developing improved patterns.

4 Conclusions

A method of exploiting the WWW to improve ad-hoc retrieval from a target collection was introduced for the Robust Track. This involves forming Boolean-type Google queries from TREC description queries to perform web retrieval, defining alternate queries from returned web pages, and data fusion to define final results. This approach was successful and has improved the worst-query measures substantially from 30% to 80%.

For the HARD Track, clarification forms for each query were designed to solicit relevant term and document information from a user. One type makes use of Wordnet to suggest synonyms to query terms and asks the user to 'click' the relevant ones for query expansion purposes. In addition, users can type in more keywords. Data for this form does not rely on a retrieval and is less costly. A second type of form makes use of retrieval results of the original query, with the top retrieved documents and top related terms presented to the user for feedback. In both cases, results seem to indicate that two inputs, user keywords and 'clicked' terms, are necessary to get improvements compared to results not using these forms. Although document-level results show that forms that rely on a retrieval has a slight edge over the Wordnet forms, passage-level results indicate otherwise.

Acknowledgment

This work was partially supported the Space and Naval Warfare Systems Center San Diego, under grant No. N66001-1-8912, and by a U.S. Govt. DST/ATP contract 2003*H532600*000.

References

[1] Vogt, C.C. and Cottrell, G.W. (1999) Fusion via a linear combination of scores. Information Retrieval, 1(3):151-173.

[2] Kwok, K.L., Grunfeld, L. & Chan, M. (2000) TREC-8 ad-hoc, query and filtering experiments using PIRCS. In: Information Technology: The Eighth Text REtrieval Conference (TREC-8), E.M. Voorhees & D.K. Harman, eds. NIST Special Publication 500-246, US GPO: Washington, DC. pp.217-227.

[3] Lin, D. (1994) PRINCIPAR – an efficient, broad-coverage, principle-based parser. *Proc of COLING-94*. pp.482-488

[4] Miller, G. (1990) Wordnet: an online lexical database. *International Journal of Lexicography*. 3(4)

[5] Kwok, K.L., Grunfeld, L. Dinstl, N & Chan, M. (2001) TREC 2001 Question-answering, web and cross language track experiments using PIRCS. In: Information Technology: The Tenth Text Retrieval Conference, TREC 2001. E.M. Voorhees & D.K. Harman, eds. NIST Special Publication 500-250, US GPO: Washington, DC. pp.452-456.

Partitioning a Graph of Sequences, Structures and Abstracts for Information Retrieval[1]

Aynur Dayanik
Computer Science
Rutgers University
aynur@cs.rutgers.edu

Craig G. Nevill-Manning
Google, Inc.
craignm@google.com

Rose Oughtred
Chemistry and Chemical Biology
Rutgers University
rose@rcsb.rutgers.edu

Abstract

In this paper, we consider the problem of finding the MEDLINE articles that describe functions of particular genes. We describe our experiments using the mg system and the partitioning of a graph of biological sequences, structures and abstracts. We participated in the primary task of the TREC 2003 Genomics Track.

1 Introduction

Computational biology deals with a wide range of entities, including DNA sequences, protein sequences, protein structures, gene functions and academic publications. Databases of these entities include explicit links between them. For example, MEDLINE abstracts often reference sequences and structures that are relevant to the publication. Sequences are related to the structures of the proteins that they encode. Sequences are also related to homologous sequences in other organisms. In this paper, we explore how these relationships can be used to enhance retrieval of relevant MEDLINE abstracts. Our intuition is this: two abstracts may not share a significant number of common terms, but if they are both connected to many common sequences and structures, then a researcher interested in one abstract should be alerted to the existence of the other.

An assumption of our work is that recall is more important than precision. In the context of a professional investigation, researchers are willing to spend more time evaluating possible relevant literature than, say, the average web searcher is willing to spend on evaluating pages returned from a casual search. In other words, the cost of a false negative is much higher than the cost of a false positive.

To test this idea, we create a graph of biological entities, where edges are defined by the explicit links between them. We then partition the graph to find clusters of topologically related entities, including abstracts. Finally, these clusters are used to adjust the ranking of abstracts returned by a simple text retrieval engine.

[1] This material is based upon work supported by the National Science Foundation under Grant No. 9986085.

The paper is organized as follows. The next section describes the primary task of the Genomics track. Section 3 describes the biological databases we used. In section 4, we describe the construction of the graph from the databases, and then present our graph partitioning approach in section 5. In section 6, we describe the details of the official runs. In section 7, we discuss our results. Finally, we summarize and discuss possible directions for our future work.

2 Primary Task

The primary task of the genomics track was an ad hoc information retrieval problem: For a given gene X, find all MEDLINE articles that focus on the basic biology of the gene X or its protein products. Basic biology includes isolation, structure, genetics and function of genes/proteins in normal and disease states [9]. The relevance judgements were obtained from the LocusLink database [11]. A portion of a sample LocusLink entry is shown in Table 1. A LocusLink entry contains references to MEDLINE database as well as brief descriptions of gene functions extracted from the MEDLINE articles. These references are called GeneRIF (Gene References Into Function). In the example shown in Table 1, the unique PubMed identifier 12482586 and the description attached to it define a GeneRIF in the LocusLink database for the gene *EIF4E* of the organism *Homo sapiens*. GeneRIFs were used as query relevants – *qrels* in the TREC terminology.

LOCUSID: 1977
ORGANISM: Homo sapiens
OFFICIAL_SYMBOL: EIF4E
OFFICIAL_GENE_NAME: eukaryotic translation initiation factor 4E
ALIAS_SYMBOL: EIF-4E
...
GRIF: 12482586—eIF4E is associated with 4E-BP3 in the cell nucleus and cytoplasm
GRIF: 11959093—Mutations in the S4-H2 loop of eIF4E which increase the affinity for m7GTP
...

Table 1: A sample LocusLink entry

The provided MEDLINE collection consisted of 525,938 MEDLINE abstracts, indexed between April 1, 2002 and April 1, 2003. The training and test queries were obtained from LocusLink entries, and consisted of 50 queries each. Each query was specific to a gene, and chosen from the LocusLink database with gene identifiers such as *official gene name, official symbol, alias symbol, organism, etc.* For an overview of the track, see [7].

3 Data Sources

We used several genomics resources in addition to the provided MEDLINE collection. In this section, we will briefly describe the data sources we used to construct a graph.

MEDLINE: MEDLINE is a digital collection of life science literature consisting of over twelve million abstracts together with some additional information associated with each abstract such as manually assigned MeSH terms and chemical names. Moreover, there are links from MEDLINE abstracts to the sequences and structures that the article discuss.

GenBank: GenBank is the NIH genetic sequence database, an annotated collection of all publicly available DNA sequences. Bibliographic references to MEDLINE articles are included for all published sequences.

Swiss-Prot: The Swiss Protein Database (Swiss-Prot) is a curated protein sequence database [4]. The Swiss-Prot entries are cross-referenced to several other databases, including MEDLINE, PROSITE and the PDB.

PDB and Ligands: The Protein Data Bank (PDB) contains 3–D structural data of biological macromolecules (proteins and nucleic acids) [5]. The PDB entries also contain references to small molecules, known as ligands. We used PDB structures as well as ligands to connect PDB structures based on their binding patterns. The PDB entries are also cross-referenced to the primary citations in MEDLINE and other databases including ENZYME and Swiss-Prot.

SCOP: The SCOP database provides a hierarchical classification of all proteins whose structure is known, including all entries in the PDB [10]. We represented the leaves of the SCOP hierarchy in our graph to be able to connect to the PDB structures.

PROSITE: PROSITE is a database of protein families and domains [6]. It consists of biologically significant sites, patterns and profiles. PROSITE provides cross-references to Swiss-Prot, PDB and ENZYME databases.

ENZYME: ENZYME is a repository of information relative to the nomenclature of enzymes [3]. ENZYME provides explicit links to Swiss-Prot and the PDB.

4 Constructing the Graph

We construct a weighted undirected graph where nodes correspond to entries from the databases listed in Section 3, including MEDLINE abstracts, DNA and protein sequences from GenBank and SwissProt, structures from PDB, patterns from PROSITE, classifications from SCOP and ENZYME, and chemical names from MEDLINE abstracts. Edges correspond to explicit links between entries encoded in the databses, e.g. the sequence annotations in MEDLINE abstracts.

Some nodes in the graph have high degree. Papers that describe genomic sequencing, for example, often reference all the sequences produced by the project. In these cases, the individual edges are not highly significant – the relationship between sequences from the same organism is not strong. So we assign these edges low weight. We assign weights to edges as follows. Let (u, v) be an edge in the graph. Then, the weight of the edge (u, v), $w(u, v)$, is computed as

$$w(u,v) = \min\left(\frac{1}{n(u,v)}, \frac{1}{n(v,u)}\right) \quad (1)$$

where $n(u, v)$ is defined as the number of edges between u and all other vertices of same type as vertex v, and $n(v, u)$ is defined as the number of edges between v and all other vertices of same type as vertex u.

As an example, consider a relation between the article A_1 and the sequence S_1. Let us say A_1 is related to 10 sequences in total, and S_1 is one of them. Also, assume that S_1 is related to 4 articles in total. Therefore, we will have an edge (u, v) corresponding to the relation between the article A_1 and the sequence S_1 in our graph with weight $1/10$ since $\min(1/10, 1/4) = 1/10$. Note that, for this particular example, when we compute the weight, we take into account only article-sequence relationships. In general, we consider the same type of relationships to compute edge-weights. We think that normalization is a fair method for assigning edge-weights because some objects are related to too many other objects of the same type and some only to a few.

5 Graph Partitioning

The objective of graph partitioning is to partition vertices of a graph in k equal subsets such that the total weight of edges connecting different subsets is minimized, thereby each subset is highly similar. The graph partitioning problem is NP-complete, but good heuristics exist. We employed a partitioning approach based on multilevel recursive bisection [8]. First, the size of the graph is reduced by collapsing nodes and edges to a few thousand nodes. Then the smaller graph is partitioned into two parts. Partitioning is repeated by uncoarsening each part one level up until all k subsets are obtained. We used publicly available graph partitioning software, METIS [1].

6 Run Descriptions

We employed the mg system as our retrieval engine [2, 12]. We submitted two official runs to the Genomics track. The first run was done using only the mg system while the second run was obtained by reordering the retrieved abstracts by mg using the clusters of abstracts defined by the graph partitions.

Run using mg: We indexed the following sections from the MEDLINE abstracts: title, abstract, MeSH terms and chemical names. We slightly modified mg to be able to tokenize biological terms properly. It currently forms words as a sequence of letters, digits and the following special characters: (,), [,], ', -, ',' and /. Note that these special characters cannot be the first or last character of the word. It is clear that these special characters appear in gene names and synonyms. For example, 1,25-dihydroxyvitamin, dead/h and cyclin-dependent are now treated as single indexing terms.

mg performed case-folding but not stemming. Query parsing was done identically to document parsing. We formed queries from the gene names and synonyms. We eliminated duplicate words and stopwords from the queries. The mg system includes support for ranked queries, where similarity is evaluated using the cosine measure. We issued ranked queries.

Run using mg but ordering results by clusters: The graph is disconnected, with one large graph of about 500,000 nodes, and 224,440 smaller graphs, the largest of which has 1,500 nodes. The smaller graphs are considered single clusters. The large graph is partitioned to produce 5000 clusters of about 100 nodes each.

In this run, we first obtained the search results using mg, and then grouped them by clusters. We assigned each group the highest mg score in that cluster. We ordered the groups first by group scores, and then in each group, by the mg scores. The intuitive idea of reranking mg search results is that if it can identify some qrels at the top ranked results, we can push more qrels to the top results by using the additional information about their relatedness, i.e., being in the same cluster.

7 Results

This section reports our results obtained using the mg system and the clusters. The orginal mg achieved a mean average precision (MAP) of 0.2759 whereas the modified mg achieved a MAP of 0.3054 on the training data. Since the modified version increased the performance in terms of MAP, all the results reported for the test data were obtained by the modified version of mg. While mg achieved a MAP of 0.3054 on the training data, ranking mg search results by clusters achieved a MAP of 0.3191. However, the mean average precisions for mg and using clusters are 0.1652 and 16.36, respectively, on the test data.

Figure 1 compares mg for the top 1000 retrieved articles to the best and median systems using the test data. In general, our performance is close to that of the median systems. We tried to understand the possible reasons for this low performance, and noticed that retrieval is quite sensitive to query formulations. For example, for test topic 7, the mg system identified *only* one qrel as relevant out of four known qrels for this topic. In Figure 1, mg exhibits very low performance for topic 7 compared to the other participating systems. The reason is that the query for this topic includes "syndecan 4", however, three qrels contain only "syndecan-4", and only one qrel contains both "syndecan" and "syndecan-4". We indexed "syndecan-4" as a single word for those three qrels, and therefore mg cannot locate them when the query is "syndecan 4". In fact, when we change the query to include only "syndecan-4" (even omitting gene symbols), mg identified all four qrels as the $4th$, $5th$, $7th$ and $8th$ documents. Another example is that "1,25-dihydroxyvitamin" appears in MEDLINE abstracts, whereas test topic 12 expresses it with an additional space, i.e. as"1,25- dihydroxyvitamin", thereby breaking it up into two words. Therefore, different variations of gene names and symbols are an important issue to be considered when preparing queries.

Figure 1 also compares mg and ranking using clusters for the top 100 retrieved articles using the test data. As can be seen from the figure, ranking by clusters significantly improved the MAP for eight queries, but significantly decreased for the other eight queries. However, upon manual inspection, we find out that many qrels fall into same clusters. We believe that clusters can be useful for researchers by itself or in conjunction with a retrieval method to support browsing similar entities.

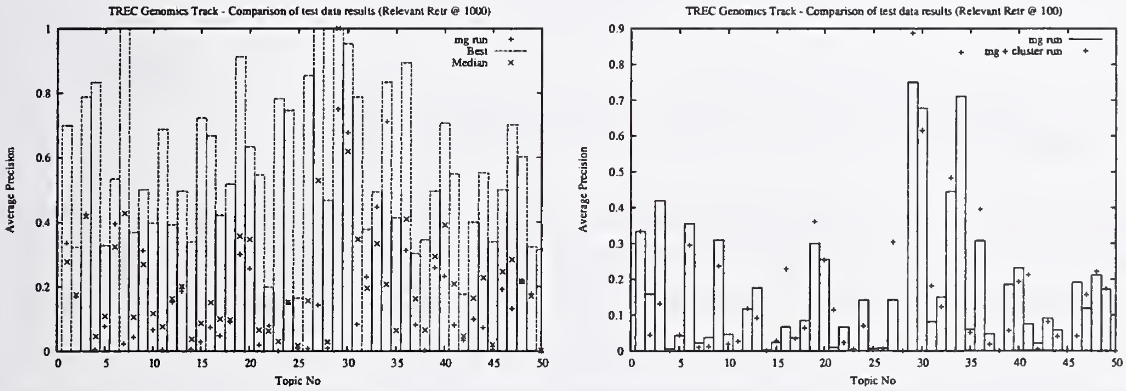


Figure 1: Comparison of mg for the top 1000 retrieved articles to the best and median systems using the test data (left), and Comparison of mg and clustering results for the top 100 retrieved articles using the test data (right)

We analyzed one cluster to assess its quality. Figure 2 shows the qrels and their corresponding clusters for test topic 3. Test topic 3 have thirteen qrels, and eleven of them fall into same cluster (the other two that fall into two other separate clusters are PMID 12167712 and PMID 12186496). We picked the cluster containing these eleven qrels. Figure 3 and 4 present the Chemical names, PDB and GenBank entries in this cluster. These figures aslo show the descriptive words extracted automatically from the MEDLINE abstracts in the cluster based on the word frequencies in the cluster and in the entire corpus of MEDLINE articles.

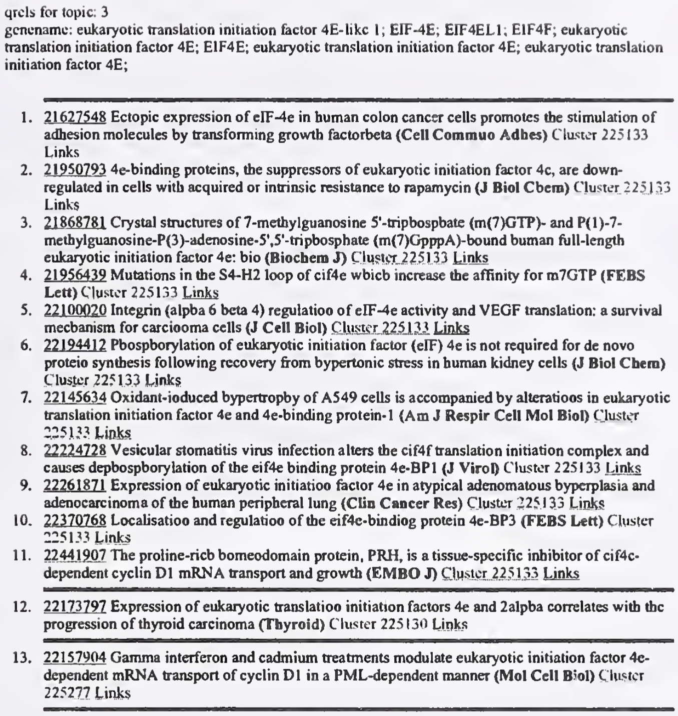
Our scientific domain expert carefully analyzed this cluster and found it to be highly relevant to the gene of interest, eukaryotic initiation factor 4e (eif4e). The descriptive words are meaningful with respect to the eif4e gene and to the biological pathway within which the eif4e protein is involved. The chemical names are also relevant to the mechanism of eif4e and all of the associated PDB macromolecular structures contain the eif4e protein. In addition, approximately 60% of the Swiss-Prot and GenBank sequences in this cluster are relevant to one another in that they all play a role in the biological process of protein synthesis.

Let us summarize the expert's analysis of this cluster:

Quality of cluster: Very high
Descriptive words are meaningful
Relevancy: Very good
Chemicals: All relevant
PDB structures: All relevant
GenBank: 8 out of 13 relevant
Swiss-Prot: 3 out of 5 directly relevant

8 Conclusions

Our primary goal was to demonstrate to the Information Retrieval and Bioinformatics communities experiments that involved the open-source mg system and graph partitioning for an information retrieval problem in genomics. Our results are close to the median performance of the participating systems in the Genomics track. Even though we observed some high-quality clusters, we did not obtain a significant improvement over mg alone using our clusters for retrieval purposes. In the future, we plan to carry out more experiments in order to better understand the quality of the clusters. Moreover, we want to try other retrieval methods together with our clusters in order to determine how the initial retrieval method affects the performance of the retrieval using clusters. We also want to investigate the effects of creating clusters during the retrieval process from the neighborhood graphs of retrieved abstracts.



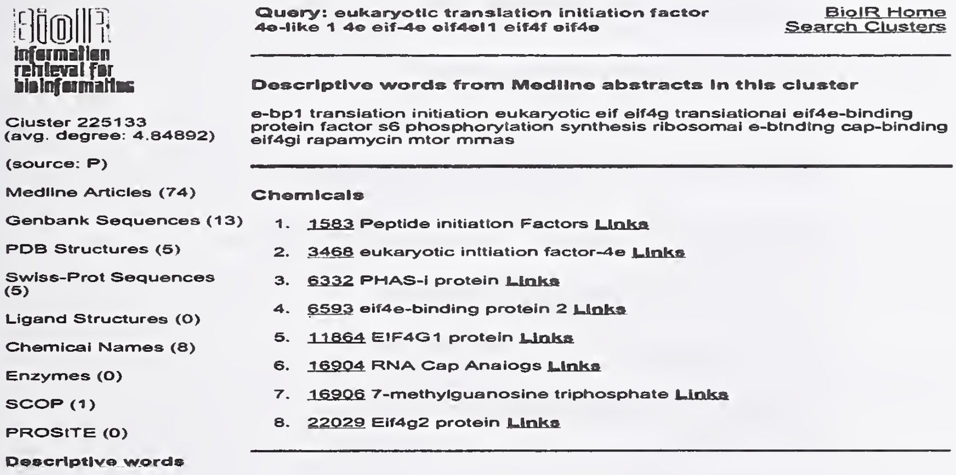
qrels for topic: 3
genename: eukaryotic translation initiation factor 4E-likc 1; EIF-4E; EIF4EL1; EIF4F; eukaryotic translation initiation factor 4E; EIF4E; eukaryotic translation initiation factor 4E; eukaryotic translation initiation factor 4E;

1. 21627548 Ectopic expression of eIF-4e in human colon cancer cells promotes the stimulation of adhesion molecules by transforming growth factorbeta **(Cell Commuo Adbes)** Cluster 225133 Links
2. 21950793 4e-binding proteins, the suppressors of eukaryotic initiation factor 4c, are down-regulated in cells with acquired or intrinsic resistance to rapamycin **(J Biol Cbem)** Cluster 225133 Links
3. 21868781 Crystal structures of 7-methylguanosine 5'-tripbospbate (m(7)GTP)- and P(1)-7-methylguanosine-P(3)-adenosine-5',5'-tripbosphate (m(7)GpppA)-bound buman full-length eukaryotic initiation factor 4e: bio **(Biochem J)** Cluster 225133 Links
4. 21956439 Mutations in the S4-H2 loop of cif4e wbicb increase the affinity for m7GTP **(FEBS Lett)** Cluster 225133 Links
5. 22100020 Integrin (alpba 6 beta 4) regulatioo of eIF-4e activity and VEGF translation: a survival mecbanism for carciooma cells **(J Cell Biol)** Cluster 225133 Links
6. 22194412 Pbospborylation of eukaryotic initiation factor (eIF) 4e is not required for de novo proteio synthesis following recovery from bypertonic stress in human kidney cells **(J Biol Chem)** Cluster 225133 Links
7. 22145634 Oxidant-ioduced bypertropby of A549 cells is accompanied by alteratioos in eukaryotic translation initiation factor 4e and 4e-binding protein-1 **(Am J Respir Cell Mol Biol)** Cluster 225133 Links
8. 22224728 Vesicular stomatitis virus infection alters the cif4f translation initiation complex and causes depbospborylation of the eif4e binding protein 4e-BP1 **(J Virol)** Cluster 225133 Links
9. 22261871 Expression of eukaryotic initiatioo factor 4e in atypical adenomatous byperplasia and adenocarcinoma of the human peripheral lung **(Clin Cancer Res)** Cluster 225133 Links
10. 22370768 Localisatioo and regulatioo of the eif4e-bindiog protein 4e-BP3 **(FEBS Lett)** Cluster 225133 Links
11. 22441907 The proline-ricb bomeodomain protein, PRH, is a tissue-specific inhibitor of cif4c-dependent cyclin D1 mRNA transport and growth **(EMBO J)** Cluster 225133 Links

12. 22173797 Expression of eukaryotic translatioo initiation factors 4e and 2alpba correlates with thc progression of thyroid carcinoma **(Thyroid)** Cluster 225130 Links

13. 22157904 Gamma interferon and cadmium treatments modulate eukaryotic initiation factor 4e-dependent mRNA transport of cyclin D1 in a PML-dependent manner **(Mol Cell Biol)** Cluster 225277 Links

Figure 2: Test topic 3 qrels and their clusters – A screen snapshot from our BioIR system



Query: eukaryotic translation initiation factor 4e-like 1 4e eif-4e eif4el1 eif4f eif4e

Cluster 225133
(avg. degree: 4.84892)

(source: P)

Medline Articles (74)

Genbank Sequences (13)

PDB Structures (5)

Swiss-Prot Sequences (5)

Ligand Structures (0)

Chemicai Names (8)

Enzymes (0)

SCOP (1)

PROSiTE (0)

Descriptive words

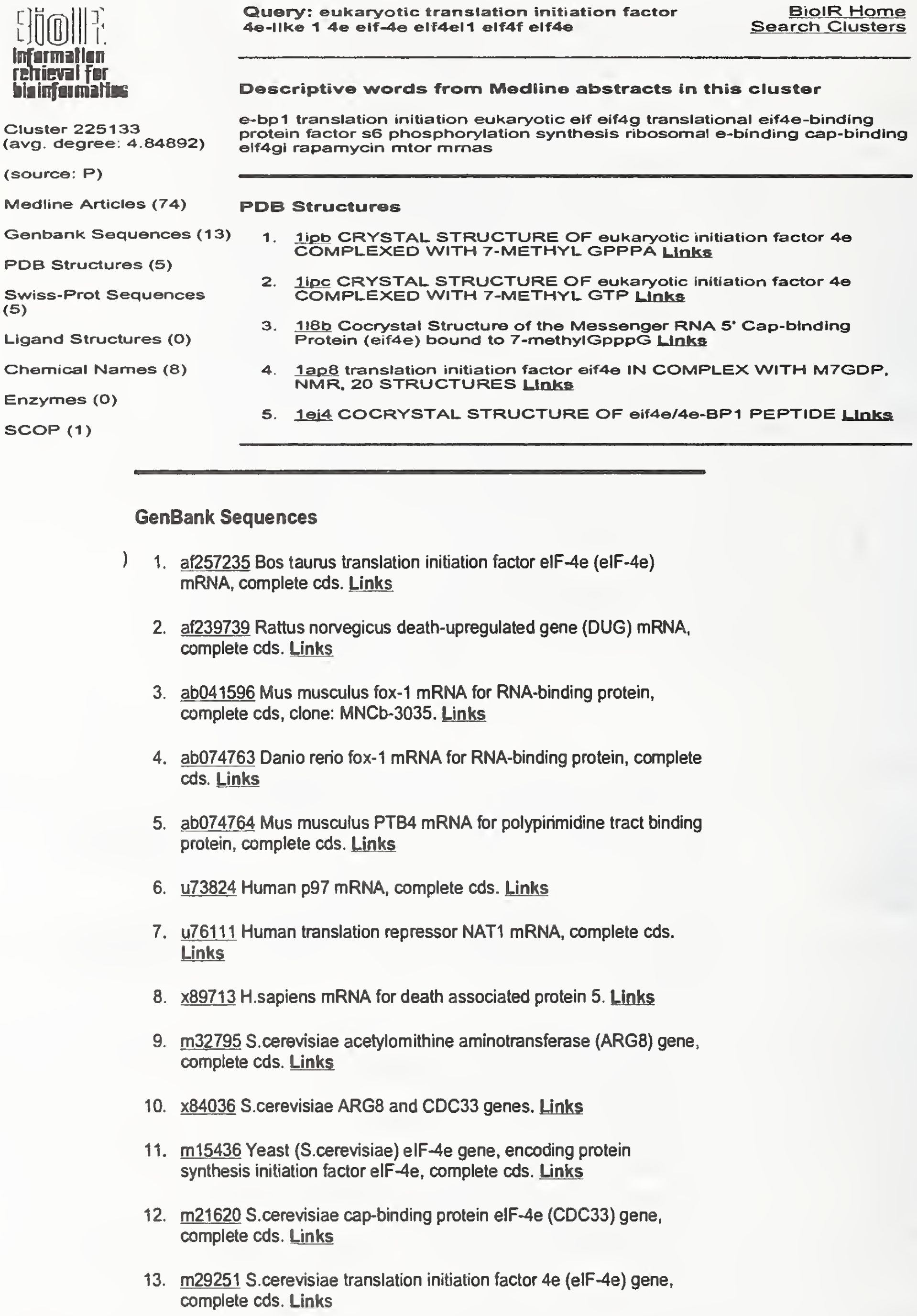
Descriptive words from Medline abstracts in this cluster

e-bp1 transiation initiation eukaryotic eif eif4g transiational eif4e-binding protein factor s6 phosphorylation synthesis ribosomai e-binding cap-binding eif4gi rapamycin mtor mmas

Chemicals

1. 1583 Peptide initiation Factors Links
2. 3468 eukaryotic inttiation factor-4e Links
3. 6332 PHAS-i protein Links
4. 6593 eif4e-binding protein 2 Links
5. 11864 EIF4G1 protein Links
6. 16904 RNA Cap Anaiogs Links
7. 16906 7-methylguanosine triphosphate Links
8. 22029 Eif4g2 protein Links

Figure 3: Chemical names assigned to the cluster having eleven qrels out of thirteen qrels for test topic 3 – A screen snapshot from our BioIR system



BioIR
Information retrieval for bioinformatics

Query: eukaryotic translation initiation factor 4e-like 1 4e eif-4e eif4el1 eif4f eif4e

BioIR Home
Search Clusters

Cluster 225133
(avg. degree: 4.84892)

(source: P)

Medline Articles (74)

Genbank Sequences (13)

PDB Structures (5)

Swiss-Prot Sequences (5)

Ligand Structures (0)

Chemical Names (8)

Enzymes (0)

SCOP (1)

Descriptive words from Medline abstracts in this cluster

e-bp1 translation initiation eukaryotic eif eif4g translational eif4e-binding protein factor s6 phosphorylation synthesis ribosomal e-binding cap-binding eif4gi rapamycin mtor mrnas

PDB Structures

1. 1ipb CRYSTAL STRUCTURE OF eukaryotic initiation factor 4e COMPLEXED WITH 7-METHYL GPPPA Links
2. 1ipc CRYSTAL STRUCTURE OF eukaryotic initiation factor 4e COMPLEXED WITH 7-METHYL GTP Links
3. 1l8b Cocrystal Structure of the Messenger RNA 5' Cap-binding Protein (eif4e) bound to 7-methylGpppG Links
4. 1ap8 translation initiation factor eif4e IN COMPLEX WITH M7GDP, NMR, 20 STRUCTURES Links
5. 1ej4 COCRYSTAL STRUCTURE OF eif4e/4e-BP1 PEPTIDE Links

GenBank Sequences

1. af257235 Bos taurus translation initiation factor eIF-4e (eIF-4e) mRNA, complete cds. Links
2. af239739 Rattus norvegicus death-upregulated gene (DUG) mRNA, complete cds. Links
3. ab041596 Mus musculus fox-1 mRNA for RNA-binding protein, complete cds, clone: MNCb-3035. Links
4. ab074763 Danio rerio fox-1 mRNA for RNA-binding protein, complete cds. Links
5. ab074764 Mus musculus PTB4 mRNA for polypirimidine tract binding protein, complete cds. Links
6. u73824 Human p97 mRNA, complete cds. Links
7. u76111 Human translation repressor NAT1 mRNA, complete cds. Links
8. x89713 H.sapiens mRNA for death associated protein 5. Links
9. m32795 S.cerevisiae acetylomithine aminotransferase (ARG8) gene, complete cds. Links
10. x84036 S.cerevisiae ARG8 and CDC33 genes. Links
11. m15436 Yeast (S.cerevisiae) eIF-4e gene, encoding protein synthesis initiation factor eIF-4e, complete cds. Links
12. m21620 S.cerevisiae cap-binding protein eIF-4e (CDC33) gene, complete cds. Links
13. m29251 S.cerevisiae translation initiation factor 4e (eIF-4e) gene, complete cds. Links

Figure 4: PDB and GenBank entries in the cluster having eleven qrels out of thirteen qrels for test topic 3 – A screen snapshot from our BioIR system

References

[1] METIS: Family of Multilevel Partitioning Algorithms. http://www-users.cs.umn.edu/~karypis/metis/.

[2] MG software. http://www.cs.mu.oz.au/mg/.

[3] A. Bairoch. The ENZYME database in 2000. *Nucleic Acids Res*, 28:304–305, 2000.

[4] A. Bairoch and R. Apweiler. The SWISS–PROT protein sequence database and its supplement TrEMBL in 2000. *Nucleic Acids Res*, 28:45–48, 2000.

[5] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28:235–242, 2000.

[6] L. Falquet, M. Pagni, P. Bucher, N. Hulo, C. J Sigrist, K. Hofmann, and A. Bairoch. The PROSITE database, its status in 2002. *Nucleic Acids Res*, 30:235–238, 2002.

[7] William Hersh and Ravi Teja Bhupatiraju. TREC Genomics Track Overview. In *Proceedings of the Twelfth Text Retrieval Conference, TREC-12*, Gaithersburg, MD, 2003.

[8] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392, 1998.

[9] J. A. Mitchell, A. R. Aronson, J. G. Mork, L. C. Folk, S.M. Humphrey, and J.M. Ward. Gene indexing: Characterization and analysis of NLM's GeneRIFs. In *Proceedings of the AMIA Annual Symposium*, pages 460–464, 2003.

[10] A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, 247:536–540, 1995.

[11] K.D. Pruitt and D. R. Maglott. RefSeq and LocusLink: NCBI gene-centered resources. *Nucleic Acids Res*, 29(1):137–140, 2001.

[12] I. H. Witten, A. Moffat, and T. C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images.* Morgan Kaufmann, San Francisco, CA, 2 edition, 1999.

Rutgers' HARD and Web Interactive Track Experiences at TREC 2003

N.J. Belkin, D. Kelly, H.-J. Lee, Y.-L. Li, G. Muresan, M.-C. Tang, X.-J. Yuan, X.-M. Zhang
School of Communication, Information and Library Studies, Rutgers University, New Brunswick, NJ 08901
[belkin, diane, hyukjinl, lynnlee, muresan, muhchyun, xjyuan, xzhang]@scils.rutgers.edu

1 Introduction

This year, members of our group, the Information Interaction Laboratory at Rutgers, SCILS, participated in the HARD track, and in the Interactive Sub-track of the Web track. Since there were no points of commonality between the two separate investigations, we describe and present the results and conclusions for each separately.

2 The HARD Track

2.1 Introduction and hypotheses

The goal of our work in the HARD track was to test techniques for using knowledge about various aspects of the information seeker's context to improve IR system performance. We were particularly concerned with such knowledge which could be gained through implicit sources of evidence, rather than explicit questioning of the information seeker. We therefore did not submit any clarification form[1], preferring to rely on the categories of supplied metadata concerning the user which we believed could, at least in principle, be inferred from user behavior, either in the past or during the current information seeking episode. To this end, based on the training data supplied and our previous research, we attempted to test the following hypotheses:

H1: People who are familiar with a topic will want to see documents which are detailed and terminologically specific; people who are unfamiliar with a topic will want to see general and relatively simple documents. This we operationalized by promoting the value of documents which scored toward the unreadable end of readability scales for people highly familiar with the topic, and by promoting the value of documents which scored toward the easily readable end of the scales for people unfamiliar with the topic.

H2: Different document genres can be identified by their vocabularies. This we operationalized by constructing language models for all the retrieved documents for each training topic and for just the completely relevant documents for each topic. We then identified words which occurred with greater than expected probability, based on the entire topic language model, in the relevant documents, for all topics which had the same genre. These words were considered to be indicators of the genre. We added the words associated with a particular genre to queries for topics which requested that genre.

H3: Certain document sources will be relevant, or not, to different desired genres. This we operationalized by promoting documents from certain sources to the top of the retrieved list for topics with some genres, by removing documents from some sources entirely from the retrieved list for topics with some genres, and by demoting the value of documents from some sources in the retrieved list for topics with some genres.

H4: If there are texts which the information searcher has identified as relevant to the topic, using them as the basis for automatic query expansion will improve retrieval performance. This was operationalized by choosing terms for query expansion from the relevant texts, based on a combined ranking formula.

H5: If the desired granularity of the retrieval result is passage, then the retrieved documents should be ranked on the basis of their best passage, rather than on the document as a whole. This was operationalized by using the InQuery best passage ranking function.

Our official submission was with queries constructed on the basis of hypotheses 2, 4 and 5.

Our basic IR system was InQuery, version 3.2, obtained from the Center for Intelligent Information Retrieval, University of Massachusetts (http://ciir.cs.umass.edu) using its default indexing, query processing and retrieval algorithms. The queries for our baseline run were constructed using both title and description fields from the topics, and were just the weighted sum of the stemmed, non-stoplist words from the title and description fields. These queries were then used as the basis for our experimental runs, with them, or their results, modified according to the metadata, as described in section 2.2, below.

2.2 How metadata about the searcher was used

The experimental condition of the HARD track was for each site to submit at least one baseline run for the set of 50 (eventually 48) topics, using only the title and (optionally) description fields for query construction. The results of the

[1] See Allan, this volume, for detailed information about the goals and conditions of the HARD track.

baseline run(s) were compared with the results from one or more experimental runs, which made use of the searcher metadata that was supplied, and of a clarification form submitted to the searcher, asking for whatever information each site thought would be useful in improving search results. We used only the supplied metadata, for the reasons stated in section 2.1, and especially because we were interested in how to make initial queries better, rather than in how to conduct a dialogue with a searcher. There were five categories of searcher metadata for each topic (not all topics had values for all five): Purpose, Genre, Familiarity, Granularity and Related text(s), which were intended to represent aspects of the searcher's context which might be useful in tailoring retrieval to the individual, and the individual situation. We made the assumption that at least some of these categories would be available to the IR system prior to (or in conjunction with) the specific search session, either through explicit or implicit evidence. Therefore, for us the HARD track experimental condition was designed to test whether knowledge of these contextual characteristics, and our specific ways of using that knowledge, would result in better retrieval performance than a good IR system without such knowledge.

We understood that there would be, in general, two ways in which to take account of the metadata. One would be to modify the initial query from the (presumed) searcher, before submitting it for search; the other would be to search with the initial query, and then to modify (i.e. re-rank) the results before showing them to the searcher. We used both of these techniques in taking account of the different types of metadata.

Knowledge of the purpose of a search (i.e. the searcher's general goal) has long been understood to be important for human search intermediaries in tailoring a search to the specific user (cf. Belkin, 1984). Whether such knowledge could be used effectively in a direct end-user IR system is still an open question. Unfortunately, we were unable to investigate this issue in this experiment. One reason for this is that the training data that were supplied in the HARD track did not have sufficient variety on this characteristic for us to investigate different hypotheses about how to take account of it; another is that the types of purpose that were identified did not immediately suggest how they could be used.

Desired genre for the results of a search has also been identified as potentially significant in improving search performance (e.g. Rauber & Müller-Kögler, 2001). In this case, we had two hypotheses. One was general: that the genre of a document could be identified by its vocabulary. This hypothesis we operationalized in the following way. For the training data, we constructed a language model[2] based on the top 100 documents retrieved by our basic query for each topic, and a language model based on all of the documents which were evaluated as both topically relevant, and satisfying all of the metadata conditions with respect to that topic. We then identified those words which appeared with a significantly higher probability in relevant documents than in all retrieved documents, for each topic associated with each specific genre. We also identified those words which were significant in the relevant documents, but had a low probability of being generated by the language model of the retrieved documents. Using these two lists, and given the nature of the metadata, we were able to identify some words which seemed to be indicative of the genre class, Overview. These words: *one, two, three, year, last, more, total, average, historically, spanning, surveyed, trends*; were added to the baseline queries for all topics which specified Genre as Overview, using the InQuery "or" operator.

The second hypothesis for genre was based on specifics of the HARD collection. The HARD database consists of the AP Newswire, the New York Times, the Xinghua newspaper (in translation), the Federal Register and the Congressional Record. We noted that documents satisfying the Genre category of Administrative were almost certainly to be found in the Federal Register or the Congressional Record. For such topics, we therefore submitted the basic query, and increased the value on which the document rank was based (the Retrieval Status Value – RSV) for all Congressional Record and Federal Register documents as follows:

$$new\ RSV = 1 + original\ RSV \qquad (1)$$

This had the effect of placing all CR and FR documents at the top of the retrieved list, in their original order with respect to one another. We also noted that the Genre category Reaction would almost certainly never be satisfied by a document from the Federal Register collection, and was most likely to be satisfied by documents from news databases. We therefore deleted all Federal Register documents from the results lists for topics with Genre = Reaction, and demoted the value of Congressional Record documents according to the following formula:

$$new\ RSV = original\ RSV - 0.5(original\ RSV) \qquad (2)$$

Familiarity with a topic has been identified as having a significant impact on relevance assessments and on how interactive IR searches are conducted (e.g. Kelly & Cool, 2002), and it is easy to imagine various ways in which familiarity would impact understanding and usefulness of a document to a person. We hypothesized that people familiar with a topic would not only be able to read and understand technical and detailed documents on the topic, but that they also would prefer those to more general documents on the topic. On the other hand, people who are unfamiliar

[2] Using the language modeling toolkit at http://mi.eng.cam.ac.uk/~prc14/toolkit.html

with a topic might prefer more general documents, and might not be able to comprehend technical ones. Failing any better ideas, we decided to use readability as a measure of technicality/generality; the less readable, the more technical, the more readable, the more general. Although there was insufficient variety on this characteristic in the training data for our hypothesis to be tested on it, we did compute the readability of a systematic random sample of the HARD collection. This led us to an additional hypothesis: that some documents are too simple to read or too unreadable to be of use to anyone searching in this collection. We therefore implemented the following procedure for taking account of familiarity.

The readability of each of the top 1200 documents retrieved by a query to the collection was computed, using three widely used measures. The measures were Fog index, Flesch reading ease score, and Flesch-Kincaid grade level score, computed using algorithms implemented in the PERL programming language by Kim Ryan in 2000[3]. All documents which had all three readability scores at or below, or at or above extreme outlier values for the collection as a whole (as estimated by our sample of the collection) were discarded from the results. Then, for all topics which had a readability level of 4 (meaning very familiar), the RSV was increased for documents which had a readability score greater than (meaning less readable) or equal to 3 standard deviations above the mean as follows:

$$\textit{new RSV} = \textit{original RSV} + 0.2(\textit{original RSV}) \tag{3}$$

For all topics which had a familiarity level of 1 (meaning no familiarity), the RSV for documents which had a readability score less than (meaning very readable) or equal to 3 standard deviations below the mean were promoted according to equation (3).

Granularity of response was a category of metadata to which we paid relatively little attention, primarily because we did not have the capability for effective passage and sentence-level retrieval. However, we made the assumptions that documents with highly relevant passages might have those passages near the beginning of the document, or that such passages would be easy to spot in the document. Then we addressed the Granularity category of Passage by submitting the queries for all such topics using InQuery's passage-level ranking of retrieval results rather than whole-document-based ranking, with a passage length of 200 words, approximating a paragraph.

Finally, we used the Related Text metadata as the basis for query expansion (QE) of the baseline queries for all topics which specified related texts. We did not use these texts for query term re-weighting, and we simply added the QE terms to the basic weighted sum query. The terms added to a query were determined by using three different QE term-ranking measures on the set of relevant texts, combining the rankings according to the median rank, and then selecting the top 10. We decided on this method based on results reported by Carpineto, Romano & Giannini (2002), which suggest that using different QE ranking techniques and then combining them leads to better retrieval performance than using any single QE ranking technique. We ranked according to the following three formulae:

rank = t , with ties being resolved according to DF, lowest DF value first;

rank = $[(t/R) - (t/DF)] / t/DF$;

rank = $(t/R) \times \log[(t/R)/(t/DF)]$;

in which t represents number of occurrences of the term in the relevant documents; R represents total number of term tokens (i.e. the number of *different* words) in the relevant documents; and DF represents total number of documents in the collection with the term..

We planned to apply the different techniques for taking account of the various metadata types in sequence, combining them all into one single query modification plus results re-ranking as follows:

Baseline query + relevant text QE + Overview words + passage-level ranking = results list 1

Results list 1 + Administrative re-ranking + Reaction re-ranking + Familiarity re-ranking = final result list

Unfortunately, for a variety of reasons, we were able to complete this process only as far as results list 1 in time for the official submission. This is the basis for the results reported below.

2.3 HARD results

Our baseline results were rather good, and substantially above the median of the experimental results for all systems. This is likely to be a result of our using both title and description for our queries; it seems likely that most other sites used title only, or title plus some form of pseudo-relevance feedback or other query expansion technique. Of more interest, of course, are our experimental results.

[3] Available online: http://aspn.activestate.com/ASPN/CodeDoc/Lingua-EN-Fathom/Fathom.html#SYNOPSIS

With respect to experimental results from all sites participating in the HARD track, Rutgers did quite well. Figure 1 indicates, for each topic, the amount above or below the median value of the Rutgers results for both R-precision and average precision.

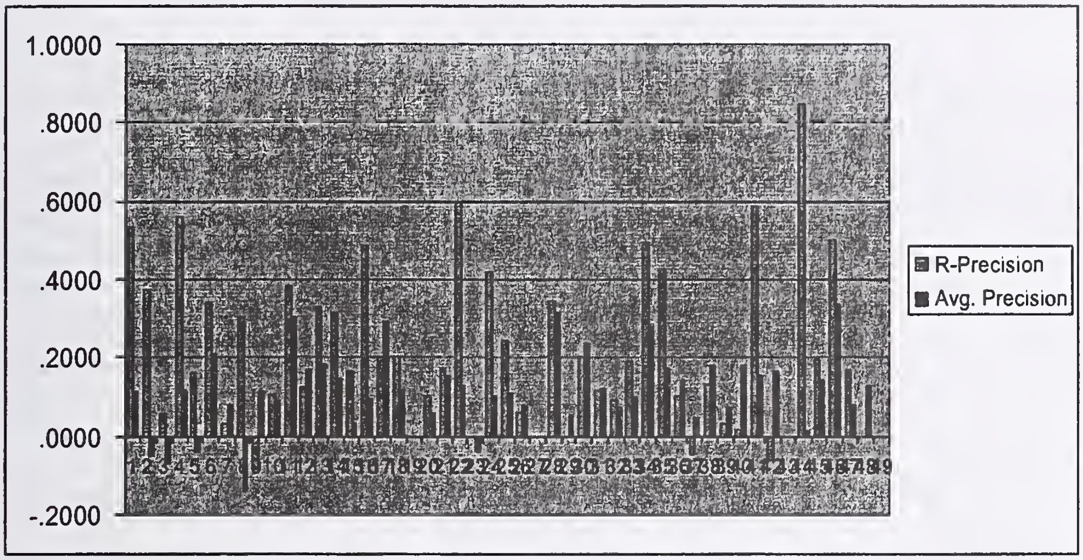


Figure 1. Difference between median values and Rutgers results for R-precision and Average Precision

Table 1 shows roughly the same data, indicating how many times the Rutgers results were best, above the median, at the median (M), and below the median for three performance measures (Rutgers was not worst for any topics).

Measure	Best	Above M	At M	Below M
Rel. Ret. @ 10	10	23	13	2
R-precision	4	32	8	4
Average Precision	3	39	3	3

Table 1. Rutgers' results compared to all results for experimental run.

Unfortunately, this comparison to everyone else does not really tell the full story. In fact, since the goal of the HARD track is to use metadata to improve over the baseline, it is much more important to look at that comparison. Here, things do not look so good. In fact, as table 2 indicates, performance on almost all measures was slightly lower for our experimental run (called Rutmeta) than for our baseline run (called rutbase2), when summarizing over all topics. Although the differences are clearly not significant, they are somewhat disheartening.

Run	Precision @ 10	R-precision	Avg. Precision	Rel. Ret.
rutbase2	0.4750	0.3451	0.3186	3736
Rutmeta	0.4750	0.3308	0.3019	3728

Table 2. Mean values of performance measures for baseline and experimental Rutgers runs.

Fortunately, this again does not tell the whole tale. If the results are compared on a topic-by-topic basis, and cumulated as in table 3, then we see that for three out of the four measures, the baseline did better than the experimental run a few times, but for average precision, the experimental run did better on 26 out of the 48 topics, and was equal for three.[4]

[4] For four topics, there was no metadata used at all, so these are not counted.

	Rel. Ret. @ 10		R-Precision		Avg. Precision		Rel. Ret	
	Rutmeta	rutbase2	Rutmeta	rutbase2	Rutmeta	rutbase2	Rutmeta	rutbase2
Better	11	15	16	19	26	17	12	17

Table 3. Topic-by-topic comparison of performance between baseline and experimental runs.

Although we do not have results which can conclusively indicate what effect each of our different techniques had on performance, we can look at some aspects of this issue. The various topics had different combinations of metadata that we used in our official experimental run, so that there are instances of each technique used separately, and the techniques used in various combinations. Table 4 indicates the effect of the different techniques by displaying for how many of the four evaluation measures using the particular metadata technique, or combination of metadata techniques, the technique did better, the same as, or worse than just the baseline. Entries in the 3 leftmost data columns indicate some advantage to having used the metadata; the fourth and fifth data columns indicate no real difference between metadata and baseline, and the sixth and seventh data columns indicate a distinct disadvantage to using the metadata. These data suggest that there was some overall advantage to enhancing the baseline query by using relevant text query expansion in combination with overview query expansion, and, if we disregard the "no difference" values, that using metadata had an overall advantage of better performance on 19 topics compared to 15 topics with better performance in the baseline condition. We still need to figure out how it happened that a topic to whose query we thought we had done nothing turned out to perform worse in the experimental condition than in the baseline.

Metadata	3 or 4 >	2 > 2 =	2 > 1 = 1 <	3 or 4 =	2 > 2 <	2 < 2 =	3 or 4 <	Total
None (3)							1!	4
QE only	4	1	1	2	4		6	18
Passage only	1							1
Overview only				2			1	3
QE + P			1	1		1	3	6
QE + O	5		3				1	9
P + O				1	1			2
QE + P + O	1	1	1				2	5
TOTALS	11	2	6	6	5	1	14	48

Each column indicates the number of topics for which using metadata resulted in the specified number of evaluation measures being better, equal to, or worse than the baseline .
> means Meta better than baseline; = means Meta same as baseline < means Meta worse than baseline
QE is query expansion; Passage (P) is ranking by best passage; Overview (O) is adding "overview vocabulary" to queries.

Table 4. Effect of application of different metadata information to baseline queries.

We also did several runs in which we tested the effect of applying only one category of metadata at a time to the baseline run. The results are displayed in Table 5, where it is easy to see that using Overview query expansion and our version of Passage retrieval had no effect. However, both Genre using the source, and Query expansion had positive effects on performance. Although the differences in performance levels are typically not great for these two, the number of topics positively affected by these two treatments was substantially greater on several of the measures.

2.4 Discussion and conclusions on the HARD results

Although the average performance of our official run using metadata is somewhat lower than our baseline run, more detailed analysis suggests that we did indeed gain some advantage from using the metadata to modify the baseline queries, in some respects. In particular, performance as measured by average precision was improved for well over half the topics, and there appears to be some advantage to the relevance feedback-like query expansion techniques. The language model-based genre technique did not work well, however. Of course, the ways in which we used the metadata to modify rankings and queries were quite ad hoc, and without real theoretical justification, which could go some way toward explaining negative results. We are still not in a position to evaluate properly the effects of each of the techniques which we have proposed on retrieval performance, nor of their complete combination, nor are we able to respond with any level of confidence to our initial hypotheses. We intend to perform further studies in which we compare all of the different techniques, and vary their parameters, in order to address this problem.

Metadata	Overview		Genre (Passage)		Query Expansion		Genre (Source)	
Number of topics	20		14		38		12	
Condition	base	meta	base	meta	base	meta	base	meta
Rel. Ret.	3736	3732	3736	3645	3736	3715	1062	1046
Number better*	1	1	3	7	15	11	3	4
Avg. Prec.	0.3186	0.3196	0.3186	0.3041	0.3186	0.3187	0.2538	0.2666
Number better	3	1	9	4	11	22	2	4
Prec. @ 10	0.4750	0.4667	0.4750	0.4646	0.4750	0.4938	0.4250	0.4917
Number better	2	0	5	5	11	11	0	3
R-Prec.	0.3451	0.3458	0.3451	0.3284	0.3451	0.3475	0.2945	0.3178
Number better	1	1	9	2	11	17	2	3

*Number of topics for which the condition had better results. When the two add to less than the total topics, all others were equal.

Table 5. Performance of single metadata treatments compared to baseline.

3 The Web Interactive Track

3.1 Introduction and hypotheses

This year the interactive TREC experiment was set up as part of the Web track and was built around the topic distillation task: finding a list of key resources for a particular topic, concentrating solely on websites as resources[5]. In the interactive sub-track, the searchers' task was to construct such a resource list for each of a set of broad topics, through interaction with an information access system[6]. The purpose of the experiment was to investigate whether the human capacity to interpret and summarize can beat machine algorithms at the topic distillation task. Apart from the direct comparisons of results, the observation of the human searchers' behavior could potentially offer clues to improving topic distillation algorithms.

We investigated the role that **the layout of search results** plays in supporting human searchers executing topic distillation tasks. Success was measured in terms of accuracy and precision, operationalized as coverage and overlap, so the searcher was expected to find documents that provide information on as many distinct aspects of the assigned topic as possible, with as little overlap between them as possible. Our hypothesis was that using the structure of the domain and of the document corpus in order to organize the search output, would help identify aspects of the search topic in different sub-domains of the document collection, would reduce the searchers' cognitive load and would produce better results than the classic hit list. We tested this hypothesis by using two user interfaces for the Panoptic search engine, one with a simple list output, and the second with documents clustered based on common URL elements.

The experimental (or hierarchic) interface, depicted in Figure 2 and described in Box 1, grouped the search results based on commonality of URL parts (sub-domain and path) and displayed them in a one level tree. The groups of hits were ranked based on the Panoptic rank of their top document; the Panoptic ranks were also used to sort hits within each group. The structured layout determined us to take two design decisions that go against common Web search engine result arrangements. Firstly, we reckoned that "More results" or "Next page" would be either ambiguous or confusing, so we did not provide such functionality. Instead, the sets of search results contained 30 hits, which was considered sufficient for the topic distillation task: if no relevant document can be found in the top 30 hits, then a query formulation is probably more appropriate than a request for more hits. Secondly, also in order to avoid confusion, the actual ranks were not displayed in the hierarchic output, but the subjects were explained the ranking scheme.

The baseline (or linear) interface was almost identical, the only difference being the layout of the 30 hits: they were displayed in a list, with the ranking provided by Panoptic. For consistency, the ranks were not displayed, but the subjects were told that documents at the top of the list were more likely to be relevant.

We used the neutral version of Panoptic, so that the subjects' task would not be supported by a topic distillation algorithm; judging the relevance of retrieved documents and the completion of the topic distillation task was entirely based on the subjects' effort.

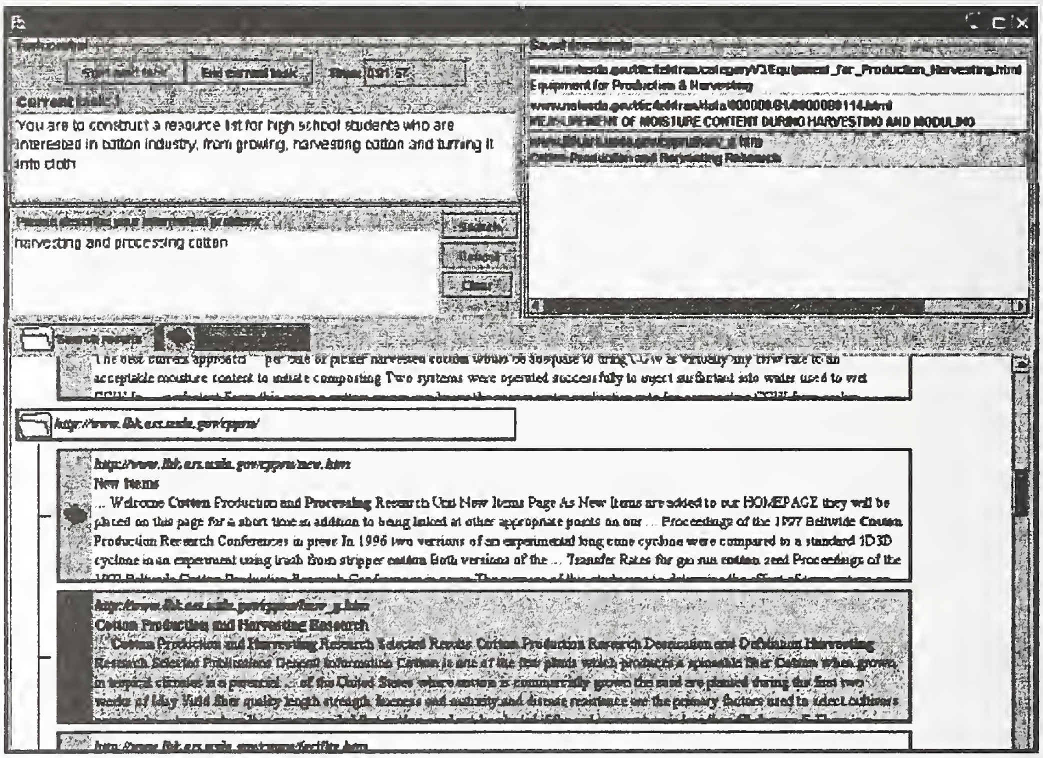
Apart from the measures of coverage and overlap, provided by NIST based on the assessors' relevance judgments, we planned to use a set of objective measures that indicate search effort such as time required to complete the task, number

[5] http://es.cmis.csiro.au/TRECWeb/guidelines_2003.html

[6] http://www.ted.cmis.csiro.au/TRECInt/guidelines.html

of iterations (or queries submitted), number of documents seen[7], selected[8] and viewed[9], number of documents saved during the interaction and number of documents kept[10]. We also prepared questionnaires in order to measure subjective measures of success such as user satisfaction and perception of success, and to investigate the correlation between success and measures such as familiarity or expertise with the topic, search expertise etc.

We were also interested in continuing previous years' investigation by looking at the effect that the query formulation panel and the instructions provided to the subject have on the syntax, length and specificity of the queries submitted. As time and resource constraints did not allow us to build another two user interfaces and run more subjects, we have no rigorously tested results. However, observations of the subjects and comparisons to last year's experiment allowed us to draw some anecdotal conclusions.



You are to construct a resource list for high school students who are interested in cotton industry, from growing, harvesting cotton and turning it into cloth

harvesting and processing cotton

www.nalusda.gov/ttic/tektran/category/V3Equipment_for_Production_Harvesting.html
Equipment for Production & Harvesting
www.nalusda.gov/ttic/tektran/data/000008/14/0000081114.html
MEASUREMENT OF MOISTURE CONTENT DURING HARVESTING AND MODULING
www.lbk.ars.usda.gov/cppru/harv_g.htm
Cotton Production and Harvesting Research

The best current approach ... per bale of picker harvested cotton would be adequate to bring CGW at virtually any flow rate to an acceptable moisture content to initiate composting Two systems were operated successfully to inject surfactant into water used to wet

http://www.lbk.ars.usda.gov/cppru/

http://www.lbk.ars.usda.gov/cppru/new.htm
New Items
... Welcome Cotton Production and Processing Research Unit New Items Page As New Items are added to our HOMEPAGE they will be placed on this page for a short time in addition to being linked at other appropriate points on our ... Proceedings of the 1997 Beltwide Cotton Production Research Conferences in press In 1996 two versions of an experimental long cone cyclone were compared to a standard 1D3D cyclone in an experiment using trash from stripper cotton Both versions of the ... Transfer Rates for gin run cotton seed Proceedings of the

http://www.lbk.ars.usda.gov/cppru/harv_g.htm
Cotton Production and Harvesting Research
... Cotton Production and Harvesting Research Selected Results Cotton Production Research Dessication and Defoliation Harvesting Research Selected Publications General Information Cotton is one of the few plants which produces a spinnable fiber Cotton when grown in tropical climates is a perennial ... of the United States where cotton is commercially grown the seed are planted during the first two weeks of May Yield fiber quality length strength fineness and maturity and disease resistance are the primary factors used to select cultivars

http://www.lbk.ars.usda.gov/cppru/facility.htm

Figure 2: The experimental user interface.

[7] Document surrogates seen while scrolling through the search results.

[8] Documents selected from the set of search results output by Panoptic. These are a subset of the set of documents seen.

[9] Documents either selected from the hit set, or obtained by following links in the Document Viewer, or by editing the URL and loading the specified webpage, if available in the .gov collection.

[10] Saved documents could be unsaved if the subject found better documents.

Box 1. Description of the user interfaces:

The **Task Panel** allows the subject to start a task (which opens a log file), displays the text of the task, including the topic, keeps track of the time, and allows the subject to end the task (which closes the log file).
The **Query Panel** encourages the subject to describe the information problem and provides sufficient space for several sentences.
The **Search Results Panel** displays the output from the Panoptic search engine, each hit being represented by a URL, a document title, and a summary. The subject can scroll and select documents for viewing in the Document Viewer. For improved usability, color coding is used to mark the currently selected document (visible by switching to the Document Viewer by clicking on the appropriate tab), the already saved documents and the already viewed documents.
The **Document Viewer** displays the full text of the selected document, allows the user to follow hyperlinks, to specify a URL and to request the loading of the specified document; it also allows the user to save the current document or to go back to a previously displayed document.
The **Saved Documents Panel** displays the URL and title of the saved documents. If the user clicks on one of the items in the panel, the corresponding document is displayed in a new window, above the Document Viewer, for comparison with the current document, so that the user can decide if there is overlap between the documents and which document is better. A saved documents can be unsaved if the user finds a better one as replacement, or reviews its relevance in view of the information retrieved.

3.2 The Interactive experiment

We had 16 subjects, volunteers mostly recruited from among Library and Information Science students. Eight were female and four male, and the ages were evenly distributed in the range 18-47. They all displayed a high level of experience with computers (6.44, 0.96), with WWW browsers (6.31, 1.01), with search engines (6.25, 0.93), and displayed a high level of confidence in being able to find information (6.00, 1.10)[11]. While this gave us confidence that the subject would easily learn and adapt to our user interfaces, it also made impossible any comparison between people with different levels of expertise. The experimental design was established by NIST, so the reader is referred to the relevant webpage[12], which also details the topics. Each subject conducted eight searches, four on the baseline and four on the experimental system. The order of systems and topics was rotated as described in the experimental design to minimize the effect of learning and tiredness on the result.

3.3 Data analysis and results

3.3.1 Objective measures

Each set of documents saved by each subject, while searching on each of the eight topics, was judged by two NIST assessors and given two scores by each: coverage (of the different aspects of the topic), ranging from 1 (very good) to 5 (very bad) and overlap (between saved documents), ranging from 1 (none) to 5 (way too much). Although there were significant differences in the reviewers' judgments, the conclusions drawn from comparing the linear and the hierarchic system were consistent. Even though a t-test failed to find a statistically significant difference, the data summarized in Table 6 indicates a tendency of linear display to be more conducive to better coverage and of hierarchal display to be more conducive to less overlap.

		Linear	**Hierarchy**
Reviewer 1	Coverage	2.67(1.72)	2.92(1.64)
	Overlap	2.59(1.23)	2.34(.96)
Reviewer 2	Coverage	2.58(1.73)	2.69(1.77)
	Overlap	2.44(.99)	2.25(1.05)

Table 6: Search results judged by expert reviewers

[11] The values in parentheses represent mean values and standard deviation on a 7-point Likert scale.
[12] http://www.ted.cmis.csiro.au/TRECInt/guidelines.html

A possible explanation of this result is that users of the baseline interface have no structure to support their exploration of the search results and therefore have to scan a larger number of documents to be satisfied with what they find. While more time-consuming and more cognitively demanding, this process has the potential to give a better coverage of a topic. On the other hand, the users of the hierarchic system have the option to direct their browsing at different sub-domains of the collection; once the user gets familiar with this kind of output, it is expected that the user would do more analysis, deciding what areas to explore, and less browsing, the result being less "direct interaction" and less overlap between content of saved documents.

Scanning all the documents in a collection has the potential for complete coverage of a topic, but is obviously not feasible; recall needs to be balanced by precision or effort. The slight increase in coverage shown by the linear system needs to be considered in the context of effort, measured in terms of time taken to search, number of iterations, and number of documents seen, selected and viewed. These measures are compared in Table 7.

Interaction Measures	Linear	Hierarchy
Iterations	4.19(2.85)	3.61(1.96)
Time (seconds)	618.53 (204.85)	575.80(117.81)
Number seen	42.78(22.12)	41.91(21.95)
Number viewed	11.81(4.52)	11.50 (5.02)
Number selected	10.64 (4.11)	10.25(4.35)
Number of ever saved	6.14 (3.08)	5.78(2.98)
Number of final saved	6.00 (3.00)	5.63(2.97)
Ratio of viewed to seen	.312 (.18)	.306(.19)
Ratio of selected to seen	10.64 (4.11)	10.25 (4.35)

Table 7: Interaction measures by display modes

Even if the difference is not statistically significant, the data in this table indicates a tendency that appears to confirm our hypothesis and expectations: the hierarchic system is conducive to less interaction.

Based on previous years' experiments, which indicated a negative correlation between user satisfaction with a system and the amount of interaction (Belkin et al, 2003a), the results from our objective measures would predict that the users would prefer the hierarchic system. Other experimental results have indicated that users like to have control over the interaction (Koenemann & Belkin, 1996); this provides another reason for us to expect the hierarchic system to be favored by users, as it allows the searcher more navigational control. Let us see if our subjective measures confirm our expectations.

3.3.2 Subjective measures

3.3.2.1 Direct comparison

The exit questionnaires provide a direct comparison between the two systems: the subjects were asked which system they found easier to learn, easier to use, which system they felt supported the task better, and which system they liked more overall. The results are as shown in Table 8:

	Linear	Hierarchical	No difference
Easier to learn to use	4	1	11
Easier to use	3	8	5
Support your tasks better	3	9	4
Like the best overall	2	10	4

Table 8: Direct system comparison (frequencies)

The results show that most of the subjects (11) perceive no difference between the linear and the hierarchic system with respect to which one is easier to learn to use. On the other three questions most of the subjects (8, 9, 10 respectively) preferred the hierarchical system. A Chi-Square test indicates that the skewness of the distribution of subject perception is statistically significant in terms of ease to learn (?^{2} (2, N=16) = 9.875, p<.01) and overall preference (?^{2} (2, N=16) = 6.500, p<.05) and not quite significant in the other cases. We can conclude that the systems are perceived as similar in ease to learn and that people prefer the hierarchic output. Apart from the layout of the display, the two systems were identical, which explains that the subjects found no real difference in learning to use them and in using them. As expected, they clearly preferred the hierarchic display.

3.3.2.2 Indirect comparison

An indirect comparison between systems was provided by answers to questionnaires administered after a subject finished using a system. The questions focused on the searchers' perception of the system with regard to ease to learn, ease to use, understanding of how to use the system, and usefulness in helping accomplish the search tasks. The subjects answered by assigning scores on a 1 – 7 Likert scale, and these scores obtained by the systems were compared by a t-test. No statistical difference was observed overall (t(30) = -.048, p>.05), or in terms of ease to learn (t(30) = -.425, p >.05), ease to use (t(30) = -.116, p>.05), clarity of the conceptual model (t(30) = .227, p>.05) or usefulness for the search task (t(30) = -.374, p>.05).

Another indirect comparison between systems was provided by answers to questionnaires administered after each of the eight searches. The questions focused on the subjects' perception of the task completion and the quality level that was achieved on each task:

- "Do you think the resource list you just constructed focuses on the topic well?"
- "Do you think the resource list you just constructed provides a good coverage of the topic?"
- "Do you think the resource list you just constructed will be helpful for those people who are interested in this topic?"

	Linear	Hierarchy
Compiled list helpful to others	5.09(1.57)	4.88(1.59)
Compiled list focuses on the topic	4.95(1.58)	4.95(1.59)
Have enough time to do search	4.75(1.62)	4.38(2.01)

Table 9: Subjects' perception of search results

The results in Table 9 indicate that the sets of documents saved with the linear system tend to be slightly better, which correlates with the slightly better coverage observed in the objective measures. However, a t-test (t(126) =.324, p>.05) shows no significant.

Table 10 shows the correlations between a few task-related factors and the subjects' subjective search performance (as measured by a scale constructed from their responses to the post-search questions on the extent to which the compiled list is helpful to others, covers the topic, and focuses on the topic). Data on these factors were collected both before and after each search. The results show that all of them are highly correlated with the subjects' subjective search performance. This suggests that these factors may be critical to impact the subjects' search performance

		Correlation with subjective search performance
Before the search	Familiarity with the topic	.420**
	Expertise on the topic	.423**
	Perceived amount of available information on topic	.263**
After the search	How easy the task was perceived to be	.760**
	Enough time for this task	.660**

**Correlation is significant at the 0.01 level (2-tailed).

Table 10: Correlation of subjective assessment of search performance with task-related factors

3.4 Discussion of the interactive results

3.4.1 Query formulation

Last year we investigated two query-formulation modes. In the former, the experimenter and the text displayed in the user interface encouraged users to submit keywords. In the latter experimental mode, the subjects were specifically asked to use sentences to describe their information need and were provided sufficient space to do so. The experimenters' insistence and their demonstration of describing information problems in sentences, combined with the parenthetical statement "(the more you say, the better the results are likely to be)" had effect on the subjects' behavior: they did follow the instructions and did write sentences. These sentences proved to provide longer queries and fewer iterations, and to generate more satisfaction with the search outcome (Belkin et al., 2003b). This year, the Query Panel of our user interface was nearly identical to that from the second mode of last year and provided the same amount of space, suitable for writing several sentences. The difference was that the parenthetical statement was removed from the Query Panel, which was reduced to "Describe your information problem" and the subjects were not specifically asked to write sentences. The result: no subjects generated any sentences. Very familiar with Web searching, the subjects seemed to enter "Google mode": they ignored the instruction from the screen and typed instead keywords, as they are used to. Consequently, the query length distribution (mean 3.04 (st.dev. 1.25) including stopwords and 2.72 (0.87) without stopwords) was surprisingly low compared to the expectations created by last year's experiment. Another explanation for this behavior may be related to the fact that, unlike last year, the topic descriptions were rather "naively" constructed, in order to be appropriate for the automatic tasks of the Web track. The essential topic keywords were present in the topic description, so most users copied and pasted the keywords into the query box, rather than having to generate them based on a problem and context description.

3.4.2 User comments

In the exit interviews, many subjects praised the capacity of the hierarchic organization to separate the different sub-domains of the collection and therefore different aspects of the topic at hand. The structured output saved them from having to mentally organize the hits and judge the overlap between their content; this was perceived as saving both time and cognitive effort. Such comments confirm our intuition that a structured display should support a structure-based task such as topic distillation.

Another feature mentioned often was the Saved Results panel, which helped users keep track of documents saved and allowed them to do side-by-side comparison between the currently examined document and already saved documents in order to compare their quality and the degree of content overlap.

Some comments indicated the need to improve the usability of the interfaces and the clarity of the underlying conceptual model. Despite the pre-experiment tutorial, some subjects did not notice the difference between the linear and the hierarchic display, as the indentation of the hierarchy was not seen as significant; others thought that the order of the hits in the display was random, rather than based on some probability of relevance score.

There were several complaints concerning the experiment settings: the constraint to limit the search to .gov documents invalidated many hyper-links, which frustrated most subjects; the time limit (10 minutes) put pressure on searchers and potentially generated un-natural behavior; thinking aloud impaired some users' ability to concentrate on the test.

3.5 Conclusions on the interactive experiment

Although it does not produce better coverage than the linear interface, the hierarchic interface seems to be conducive to less effort for the searcher: fewer iterations, shorter search sessions, fewer documents seen, selected and viewed. With regards to subjective measures, users perceived the hierarchic one as easier to use and better at supporting the topic distillation task. These results were not statistically significant. What was statistically significant is that the subjects perceived the two systems equally easy to learn and that they prefer the hierarchic display.

One advantage of the structured output, as suggested by the objective measures and highlighted by the users' comments, is the support for investigating different sub-domains of a document collection and consequently different aspects of a topic. The searcher does not need to make a cognitive effort to separate the search results into sub-domain, so the layout makes the interaction easier and more pleasant and more accurately supports the searcher's judgment on task completion.

This correlates with results obtained by CSIRO at TREC 2002: although the motivation to use a hierarchic organization was somewhat different, the structure imposed on the search output improved the retrieval performance in the case of complicated tasks, when relevant information needs to be gathered from various parts of the document collection (Craswell et al, 2002).

One direction in which we intend to continue our investigation is in displaying more than one levels of the hierarchic structure of webpages. While experiments with Cha-Cha[13] have shown promise, we are interested in whether combining navigation by browsing the hierarchic structure and following links to other parts of the hierarchy would help or confuse users.

4 Acknowledgements

Many thanks to Colleen Cool for her work on the HARD study, to Michael Coviello for his contributions to the interactive experiment, and to our volunteer subjects. The research reported here was supported in part by a Rutgers Research Council Grant to X.-M. Zhang, and by NSF grant No. 9911942. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation (NSF).

[13] http://cha-cha.berkeley.edu/

5 References

Allan, J (this volume) Overview of the TREC 2003 HARD track.

Belkin, N.J. (1984) Cognitive models and information transfer. *Social Science Information Studies*, vol. 4, nos. 2&3: 111-129.

Belkin, N.J., Cool, C., Kelly, D., Kim, G., Kim, J.-Y., Lee, H.-J., Muresan, G., Tang, M.-C., & Yuan, X.-J. (2003a). Interaction and query length in interactive retrieval. In D. Harman & E. Voorhees (Eds.), *The Eleventh Text Retrieval Conference (TREC2002)* (pp. 539-548). Washington, DC.: GPO.

Belkin, N.J., Cool, C., Kelly, D., Kim, G., Kim, J.-Y. , Lee, H.-J., Muresan, G., Tang, M.-C., Yuan, X.-J. (2003b) Query Length in Interactive Information Retrieval. In *Proceedings of SIGIR 2003* (pp.205-212). New York: ACM.

Carpineto, C., Romano, G. & Giannini, V. (2002) Improving retrieval feedback with multiple term-ranking function combination. *ACM Transactions on Information Systems*, v. 20 no. 3: 259-290.

Chen, M., Hearst, M., Hong, J., Lin, J. (1999) Cha-Cha: A System for Organizing Intranet Search Results, in Proceedings of the 2nd USENIX Symposium on Internet Technologies and SYSTEMS *(USITS)*, Boulder, CO, October 11-14, 1999.

Craswell, N., Hawking, D., Thom, J., Upstill, T., Wilkinson, R., Wu, M. (2002) TREC11 Web and Interactive Tracks at CSIRO. In E. Voorhees & D. Harman (Eds.) *Proceedings of TREC2002* (pp. 197-206). Washington, DC: GPO.

Kelly, D. & Cool, C. (2002) Effects of topic familiarity on information search behavior. In *Proceedings of the Second ACM/IEEE-CS Joint Conference on Digital Libraries – JCDL 2002* (pp. 74-75). New York: ACM.

J. Koenemann, J. & Belkin, N.J. (1996) A Case for Interaction: A Study of Interactive Information Retrieval Behavior and Effectiveness. In *Proceedings of CHI '96* (pp. 205-212) New-York: ACM..

Rauber, A. & Müller-Kögler, A. (2001) Integrating automatic genre analysis into digital libraries. In *Proceedings of the First ACM/IEEE-CS Joint Conference on Digital Libraries - JCDL 2001* (pp. 1-10). New York: ACM.

Combining First and Second Order Features in the TREC 2003 Robust Track

Endre Boros, Paul B. Kantor and David J. Neu

November 18, 2003

Abstract

This year at TREC 2003 we participated in the robust track and investigated the use of very simple retrieval rules based on convex combinations of similarity measures based on first and second order features.

1 Introduction

In the robust track, systems attempt to retrieve documents relevant to 100 different information needs, using only the text which is provided in a short descriptive passage known as a topic. The systems submit a list of up to 1000 documents which they attempt to rank by their relevance to the information need.

A generally accepted tenet in information retrieval is that the more topic terms that appear in a document, the more likely that document is to be relevant. It is also widely agreed that the co-occurrence of topic terms is also a good indication of relevance.

We investigated the use of very simple retrieval rules based on convex combinations of similarity measures based on first and second order features, where first order features were terms in the topic and second order features were features designed to capture information about term co-occurrence.

2 Approach

The topics in this year's robust track consisted of title, description and narrative sections. Participants were required to submit at least one run which only utilized the description section. All runs we submitted only utilized the description section.

As mentioned in §1 our retrieval rule is based on two different types of features. First-order features are simply the non-stopword terms appearing in the topic description and the first-order topic feature vector for a topic or document is a Boolean vector in which the i^{th} component is 1 if the text contains the i^{th} first order feature and 0 otherwise. The SMART stopword list was utilized.

Second-order features are term pairs which occur within w terms of each other in the topic description prior to the removal of stopwords, and the second-order feature vector for a topic is a vector in which the i^{th} component is the minimum distance between the pair of terms which comprise the i^{th} second order feature in the topic description.

As an example, of second-order feature construction, consider the string, "The focus of the next conference is Boolean functions.". The terms "the", "of", "next" and "is" are stopwords, so the list of non-stopword terms is [_, "focus", _, _, _, "conference", _, "Boolean", "functions"] . The distance between the non-empty term pairs is shown below:

Pair	Distance
conference, focus	4
boolean, focus	6
focus, functions	7
boolean, conference	2
conference, functions	3
boolean, functions	1

So using $w = 3$, the list of the second-order features would be: [("boolean", "conference"), ("conference", "functions"), ("boolean", "functions")].

As can be seen, we have decided to utilize a purely Boolean model which only captures whether a term, or term pair appears in a document or not, thereby ignoring all term frequency information.

For each document d, the score for a given topic is

$$\sigma(d, w, \lambda) = \lambda\phi(d) + (1 - \lambda)\psi(d, w)$$

where the first order similarity measure, ϕ is the cosine of the angle between the first order topic feature vector and the first order document feature vector, and the second-order similarity measure ψ is the cosine of the angle between the second order topic feature vector and the second order document feature vector. That is, the score, σ is the convex combination (weighted average) of the first-order and second-order similarity measures. We submitted five runs with $\lambda \in \{0.0, 0.25, 0.5, 0.75, 1.0\}$, corresponding to different weightings of the first and second order similarity measures. In all submitted runs, $w = 3$ was used.

3 Results

Analysis our performance showed that our scores did not meet expectations. We did attain the median number of relevant retrieved documents at 10, in about one-quarter of the topics, and exceeded it in about 5 percent of the topics, for all our runs. A more detailed comparison between our performance and the median performance is provided below.

λ	Measure	$\geq$ Median	$>$ Median
0.0	Rel. Ret. @ 10	23	3
0.0	Avg. Precision	5	5
0.25	Rel. Ret. @ 10	25	3
0.25	Avg. Precision	4	4
0.5	Rel. Ret. @ 10	23	5
0.5	Avg. Precision	5	5
0.75	Rel. Ret. @ 10	27	5
0.75	Avg. Precision	4	4
1.0	Rel. Ret. @ 10	25	8
1.0	Avg. Precision	4	3

The following two table demonstrated that there was substantial overlap in the topics that performed above the median for the number of relevant documents retrieved at 10 and average precision measures, thereby providing some evidence that λ need not be selected on a per topic basis.

λ	Topics in which Rel. Ret. @ 10 Exceeded the Median
0.0, 0.25	303, 608, 618
0.5, 0.75	303, 347, 379, 608, 618
1.0	303, 330, 347, 379, 409, 612 618, 628

λ	Topics in which Avg. Precision Exceeded the Median
0.0	303, 416, 608, 618, 627
0.25	303, 608, 618, 627
0.5	303, 389, 608, 618, 627
0.75	303, 389, 608, 618
1.0	379, 608, 618

Finally, an analysis of the detailed results indicates that performance on the new topics was notably better than on the old topics and that performance measures improved slightly as λ increased. This later observation indicates that the use of co-occurrence information weakened rather than then improved our performance, which was contrary to expectations.

4 Conclusion

Even for such a simple model, our robust track runs performed below expectations. However, the fact that performance on all measures increased slightly with λ seems, to indicate that the method could be improved by tuning λ. In addition, we suspect that utilization of a purely Boolean model and using a relatively small value of w may have negatively impacted performance.

Future research will involve investigation of the impact of varying w as well as the incorporation of term frequency information into our model.

References

[1] William B. Frakes and Ricardo Baeza-Yates, editors. *Information Retrieval: Data Structures and Algorithms.* Prentice-Hall PTR, 1992.

[2] Elke Mittendorf, Bojidar Mateev, and Peter Schauble. Using the co-occurrence of words for retrieval weighting. *Information Retrieval*, 3(3):243–251, 2000.

[3] C.J. van Rijsbergen. A theoretical basis for use of co-occurrence data in information retrieval. *Journal of Documentation*, 33(2):106–119, 1977.

Knowledge-Based Access to the Bio-Medical Literature

Ontologically-Grounded Experiments for the TREC 2003 Genomics Track

Richard Tong
Tarragon Consulting Corporation
1563 Solano Avenue, #350
Berkeley, CA 94707
rtong@tgncorp.com

John Quackenbush
The Institute for Genomic Research
9712 Medical Center Drive
Rockville, MD 20850
johnq@tigr.org

Mark Snuffin
DataNaut, Inc.
11 Dudley Court, Suite 100
Bethesda, MD 20814
snuffin@datanaut.com

Abstract

The Tarragon Consulting team participated in the primary task of the TREC 2003 Genomics Track. We used a combination of knowledge-engineering and corpus analysis to construct semantic models of the interactions between genes/proteins and other biological entities in the organism, and then used automatic methods to convert these models into evidential queries that could be executed by the K2 search engine from Verity, Inc. The primary goal of our participation in the Genomics Track was to establish a performance baseline using ontologically-grounded techniques that are scalable and implementable using current commercial retrieval technology. The results from both our official submissions and subsequent experiments demonstrate that good performance can be achieved using our approach.

Overall Approach

In our approach we focused on "function" as opposed to the other aspects of the basic biology of the gene and its protein products. That is, we interpreted the primary task to be one that requires us to identify the ways in which the gene/protein is involved in the organism's behavior, as opposed to one that simply requires us to identify that some property of the gene/protein is being discussed.

The framework for constructing our semantic models is an ontology that makes a set of core distinctions between: (a) the gene/protein subsystem; (b) the organism; (c) the interactions of the gene/protein subsystem with the organism; and, (d) the documents that report on the biological entities and processes. (See Figure 1 at the end of the paper for a high-level, and much simplified, view of the ontology as a UML static structure.)

We then interpret function to be synonymous with interaction, and thus make the retrieval task one of finding documents that use language that describe, or report on, these interactions.

We used the PubMed corpus and the GeneRIFs from the training set to construct a "lexicon of interaction" (LoI) which was hand-edited for consistency and validity, and then used as input to the automatic, evidential query generation process.

To run the experiments, we created a minimal XML variant of the PubMed records (see Figure 2) and indexed them using Verity's K2 search engine.[1] We used the output of the evidential query generation process as input to the K2 engine, and ran the fifty queries (i.e., one for each gene in the test set) against the indexed collection. The results from the K2 engine were then converted into the standard TREC format for submission to NIST.

Official Submissions

Our experimental strategy prior to the official submissions explored two main issues: (a) detection and recognition of genes/proteins; and, (b) effective ways of exploiting the LoI. Based on our experiments with the training data, we settled on a single strategy for name detection and recognition, and on two strategies for exploiting the LoI. Our official runs (*tgnBaseline* and *tgnVariant1*) reflect this two-fold strategy.

In both our official submissions, we modeled gene/protein names by focusing on the symbols (both the `OFFICIAL_SYMBOL` and the `ALIAS_SYMBOL`) and then creating a set of regular expression variants based on treating all punctuation as optional and also allowing for potential whitespace or punctuation when the symbol character sequence changes from

[1] See: http//www.verity.com/ for basic information about the K2 family of products.

alphabetic characters to numerical character, and vice versa.

So for example, in Topic 2 ("E2F transcription factor 1"), the symbols get mapped to the regular expressions:

```
E2F1      =>    E_2_F_1
RBP3      =>    RBP_3
E2F-1     =>    E_2_F_1
RBBP3     =>    RBBP_3
```

where "_" denotes an optional single space or punctuation character. Note that here, as with many other symbol sets, the transformation produces equivalent regular expressions. We remove any such duplicates to create a final set of expressions for each gene.

How best to exploit the gene/protein names in our models, is more problematic. Based on our experiments with the training data, we decided to use all of the `OFFICIAL_GENE_NAME`, the `PREFERRED_PRODUCT`, the `PRODUCT` and the `ALIAS_PROT` (if they are different) as part of the model. In many cases, however, we hand-edited the names to remove "annotation" or to extract alternates. So, for example in Topic 7, the official name:

```
syndecan 4 (amphiglycan, ryudocna)
```

becomes a three-fold set of names:

```
syndecan 4
amphiglycan
ryudocna
```

Once the name data is processed into our standard form, we automatically generate a sequence of K2 topic fragments such as:

```
tgn_geneName_2 <Or>
* 1.00 <Many><Phrase>
** 'E2F'
** 'transcription'
** 'factor'
** '1'
* 0.90 <Many><Phrase>
** 'retinoblastoma'
** 'associated'
** 'protein'
** '1'
```

which defines the gene name specification for Topic 2 and where notation like `<Or>` denotes an operator in the Verity Query Language (VQL).

The function component of our model leverages the LoI by creating three sub-modules that capture verbs, verb phrases, and general vocabulary that are related to function. As noted earlier, this initial LoI was created using a mix of corpus analysis techniques and knowledge engineering methodologies, and was developed to give us a basis for exploring a more formal linguistic analysis derived from our semantic models.

The LoI contains verbs such as "upregulate" and "phosphorlyate", verb phrases such as "localize in" and "mechanism for", as well a small set of mostly nouns, such as "pathway" and "antagonist", that relate to biological entities typically involved with functional behavior. In VQL this part of the model becomes (somewhat simplified for presentation purposes):

```
tgn_function_lexicon <Accrue>
* 0.60 tgn_function_vs
* 0.80 tgn_function_vps
* 0.20 tgn_domain_lex
```

Finally, we modeled the species constraint as a test for the presence of the corresponding MeSH keyword. So, for example, to be a candidate for retrieval for a Homo Sapiens gene, we check to see if the keyword "human" appears anywhere in the MeSH tags. In VQL this test becomes:

```
"Human" <In> /zonespec = "MH"
```

The overall query model for a gene topic is then essentially just a conjunct of the gene name and symbols, the function model, and the species test. In VQL this becomes (again somewhat simplified):

```
tgn_trecgenQuery_1 <And>
* 1.00 _isHuman
* 1.00 <Sum>
** 0.80 <And>
*** 1.00 tgn_geneModel_1
*** 1.00 tgn_function_lexicon
** 0.20 _queryProximity_1
```

where we also show a component of the model (here `_queryProximity_1`) that tests for the proximity of the gene model (here `tgn_geneModel_1`) and the function model (here `tgn_function_lexicon`). In training we found that this improved our overall scores, as measured using the `trec_eval` scoring program.

Using this core model and the two variant function models, our official scores for the two submissions (i.e., *tgnBaseline* and *tgnVariant1*) are shown in Table 1. Note that these gave just about the same overall performance, with *tgnBaseline* doing slightly better on Average Precision, and *tgnVariant1* doing slightly better on R-Precision.

Both runs, though, were better than the overall median scores reported (i.e., 0.2117 for Average Precision), with 37 and 36 individual topics (respectively) getting Average Precision scores greater than the median individual topic score.

Failure Analysis

Our preliminary failure analysis of the official runs showed that there were two main causes of poor performance (relative to the median published scores).

First of all, we had some significant recall failures. Overall we only retrieved 463 of the possible 566 relevant documents, and while in most cases we missed just one or two, we did have more serious failures. For example, for Topic 7 we missed all 7 relevant documents, and for Topic 37 we missed 37 of the 61 relevant documents.

We analyzed each failure and identified whether it was due to either: (a) a failure to detect the gene/protein; (b) a failure to identify the "function" being discussed; or, (c) a failure of the species test. In a few cases there were multiple causes of the failure. Overall though, of the 103 non-retrieved document, we attributed 81 failures to name recognition, 16 to function, and 13 to the species test.[2]

The second major performance failure was ranking failure. That is, our inability to get the relevant documents sufficiently high in the retrieval ranking. At this point in our investigation, we are less concerned with this issue since we believe, as do other groups[3], that the GeneRIFs we have been using as the "ground truth" in this exercise significantly under-represent the amount of "RIF-able" material in the collection.

Alternate Experiments

Given the failure analysis, we experimented with a number of alternate name detection/recognition algorithms. These were all variants of what we might call "token n-gram" methods in which, instead of attempting to match the exact name as a phrase (e.g., "ETF transcription factor 1"), we explored various forms of sub-string matching that involve both ordered and un-ordered matching of tokens in the name.

The alternate run labeled *ptVar01* in Table 1 uses a combination of ordered bi-grams and un-ordered k-grams (where k+1 is the number of tokens in the name). Note that we now find 523 of the 566 relevant documents and also retrieve many more documents than before—30,605 as compared to 7,758. Both Average Precision and R-Precision are better than either of the official submissions, although not by very much.

The other alternate run reported here, labeled *ptVar02* in Table 1, is the first is a series aimed at seeing if exploiting document structure can improve performance. This run simply adds an additional test to the model used in *ptVar01* to check if the full name, or one of the symbol variants, of the gene/protein appears in the title of the document (i.e., in the `<TI>` field), increasing the retrieval score of the document if it does.

Note that this simple extension produces a significant jump in performance over the *ptVar01* model—a 13.13% increase in the Average Precision, and a 19.66% increase in the R-Precision.

This is a surprising result given the fact that the documents are all abstracts, and suggests to us that the selection of the set of GeneRIFs used as the ground truth was heavily influenced by the titles of the original documents.

Ground Truth and Other Observations

A key element of the approach we adopted for he TREC 20003 Genomics main task was to model the concept of "function" directly. Yet as the results presented at the TREC meeting show, it was not necessary to model function in order to do well on the task. In fact, non of the top three best performing systems used any explicit representation of function.

This suggests to us that, in the context of MedLine abstracts, almost any mention of the gene/protein is likely to be relevant to function under the very broad definition we were working with (i.e., "MEDLINE references that focus on the basic biology of the gene or its protein products from the designated organism. Basic biology includes isolation, structure, genetics and function of genes/proteins in normal and disease states.").

Coupled with the fact that the GeneRIFs we have relied on for test and evaluation obviously under-report the relevant material, we think it is safe to say that we cannot draw too many significant conclusions from this initial venture into the genomics arena.

Nevertheless, this was definitely a worthwhile exercise and helped us validate our basic approach. At the same time, it made us aware of the many special issues associated with this field (e.g., name variation).

We look forward to TREC 2004 in which we can address a problem that is better motivated by the needs of practicing biologists, and for which we have a more defensible evaluation framework.

[2] Of these 13 species failures, our analysis suggests that at least 11 are due to incorrectly labeled GeneRIFs with respect to species.

[3] For example, the initial analysis reported by Bill Hersh, Sarah Corley, Ravi Teja Bhupatiraju in "Relevance Analysis for Primary Task of TREC Genomics Track" distributed via the TREC Genomics mailing list on 2003.10.13 shows that over 40% of the documents they retrieved were "RIF-able" but not labeled by the ground truth.

Figure and Tables

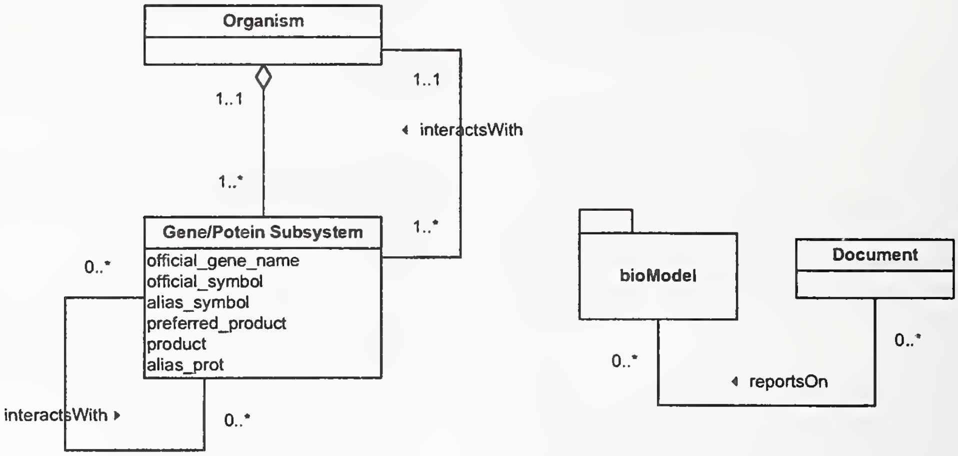


Figure 1: Simplified Ontology

```
<PubmedArticle>
<PMID>11727758</PMID>
<DCOM>20020520</DCOM>
<TI>
Opiates promote T cell apoptosis through JNK and caspase pathway.
</TI>
<AB>
Opiate addicts are prone to recurrent infections. In the present study we
evaluated the molecular mechanism of opiate-induced T cell apoptosis. Both
morphine and DAGO ([D-Ala2,N-Me-Phe4,Gly5-ol]enkephalin) enhanced T cell
apoptosis. Morphine as well as DAGO activated c-Jun NH2-terminal kinase (JNK) in T
cells. Moreover, opiates increased the expression of ATF-2. a specific substrate
for JNK and P38 mitogen activated kinases (MAPK). Furthermore, opiates attenuated
extracellular signal related kinase (ERK) in T cells. Both morphine and DAGO
cleaved pro-caspases 8, 9, and 10 and generated caspases 8, 9 and 10 (active
products). Morphine as well as DAGO also cleaved poly-(ADP-ribose) polymerase
(PARP) into 116 and 85 kD proteins indicating the activation of caspase-3. These
results suggest that opiate-induced T cell apoptosis may be mediated through the
JNKcascade and activation of caspases 8 and 3.
</AB>
<MH>Apoptosis/drug effects/physiology</MH>
<MH>Caspases/*metabolism</MH>
<MH>Enkephalin, Ala(2)-MePhe(4)-Gly(5)-/toxicity</MH>
<MH>Enzyme Activation/drug effects</MH>
<MH>Human</MH>
<MH>In Vitro</MH>
<MH>Jurkat Cells</MH>
<MH>Mitogen-Activated Protein Kinases/*metabolism</MH>
<MH>Morphine/toxicity</MH>
<MH>Narcotics/*toxicity</MH>
<MH>Support, U.S. Gov't, P.H.S.</MH>
<MH>T-Lymphocytes/*cytology/*drug effects/enzymology/immunology</MH>
</PubmedArticle>
```

Figure 2: Example `<PubmedArticle/>` XML Format

Table 1: Summary Results for Official Submissions and Selected Alternate Experiments

	tgnBaseline	*tgnVariant1*	*ptVar01*	*ptVar02*
# Docs Retrieved	7758	7758	30605	30634
# Docs Relevant	566	566	566	566
# Docs Rel_ret	463	463	523	532
Interpolated R-P:				
at 0.00	0.5721	0.5606	0.5799	0.5492
at 0.10	0.4975	0.4922	0.5398	0.5284
at 0.20	0.4179	0.4192	0.4952	0.5081
at 0.30	0.3707	0.3577	0.4222	0.4533
at 0.40	0.3298	0.3193	0.3348	0.4062
at 0.50	0.3150	0.3086	0.3060	0.3875
at 0.60	0.2622	0.2616	0.2519	0.3326
at 0.70	0.2067	0.2083	0.2041	0.2559
at 0.80	0.1614	0.1637	0.1646	0.2125
at 0.90	0.1220	0.1214	0.1095	0.1393
at 1.00	0.0992	0.0998	0.0922	0.1156
Average Precision	**0.2837**	**0.2791**	**0.2917**	**0.3300**
Precision:				
at 5 docs	0.2640	0.2720	0.3000	0.3120
at 10 docs	0.2180	0.2220	0.2280	0.2420
at 15 docs	0.2000	0.1973	0.1933	0.2253
at 20 docs	0.1760	0.1780	0.1760	0.1990
at 30 docs	0.1473	0.1480	0.1493	0.1713
at 100 docs	0.0752	0.0758	0.0754	0.0818
at 200 docs	0.0426	0.0422	0.0449	0.0472
at 500 docs	0.0184	0.0184	0.0205	0.0207
at 1000 docs	0.0093	0.0093	0.0105	0.0106
R-Precision	**0.2850**	**0.2852**	**0.2858**	**0.3420**

THUIR in TREC2003: HARD Experiments

[1]Liang Ma, [1]Wei Tan, [1]Qunxiu Chen, [1]Shaoping Ma
[2]Shuicai Shi, [2]Shibin Xiao, [2]Hongwei Wang, [2]Hongjun Wang
[1]State Key Lab of Intelligent Tech. & Sys., CS&T Dept, Tsinghua University, Beijing 100084,China
[2]TRS Open Software Laboratory, Beijing Information Technology Institute, Beijing, China

maliang00@mails.tsinghua.edu.cn

Abstract

In this paper, we describe ideas and related experiments of Tsinghua University IR group in TREC-12 HARD Track. In this track, we focus on an automatic delivering mechanism, which combine the existing IR methods and can provide a quick retrieval solution for the practical environment. The final official evaluation show the old ways perform not well, but we think the experiment data will be helpful in evaluating the new ideas developed by other teams.

1. Brief Introduction

As a new evaluation track, HARD is designed to find an effective way to locate the search focus precisely from the data coming from the user, including his/her additional information (such as the he/his background) and an interactive input, so as to provide better retrieval result to the original query request.

In this track, the key issue is to find the real search focus. There can be two ways to finish it: manually and automatic way. Though the former usually can provide a satisfied performance, we think the automatic way is more useful in practical use and thus try to devote us in this way.

In following sections, we introduce what we did in our research work and give the final evaluation result. Some further research work done after final TREC submit is also listed.

2. Construct Baseline Run

We get our baseline run (only with document) using the initial query by a BM25 TF*IDF scoring schema. It is a popular method that is fast and practical. The special treatment is only used for initial query: For each topic, the query is constructed simply by the task description (The detail restriction for none-relevant document are ignored). For the search items, different weights are set according to their location(such as description field) and importance in the task description. Also, there is no positive training documents are used to refine the query, because usually the training resource is unlikely to be provided for various immediate search requirements in Web IR.

3. Focus Probe

In focus probing, we try to find the search focus of the user input. In this period, there are two missions we did:

3.1 Finding potential search items

In our Clarification Form, all the potential search issues to be confirmed by user are listed with checkbox, together with a text field to fill if he/she finds there are something we missed. These search terms are presented as some keywords or phrases instead of long statements extracted from web pages. Some existing technologies, such as complex passage analysis or do self-learning from related training resources, seems to be good ideas but time-consuming. Here we choose a fast mechanism to extract them automatically. They are got from two ways follow:

(1) The kernel words/phrases in topic description. We parse the description and get presentive words/phrase set from each fields, then all the set are combined and those words/phrase existing in multi set are thought as kernel words/phrases.

(2) Terms with high statistical weight in top-100 ranked documents in search result. But only the terms in the title and the first paragraph(not the whole passage) are calculated, for there should be more focus-related words in these two sections. To keep the search deviation under control, we limit the potential search terms up to 10 issues.

Compared to other methods, our idea is efficient in finding the potential search terms, also it doesn't require any training resource, therefore it is feasible when applied in a practical use, but the accuracy of this method has not been proved to be very satisfied.

3.2 Locate the Desired Focus

We locate the desired focus from:

(1) **Returned Clarification Form from LDC.** Since the returned Clarification Form has been processed by search user, all the words/phrases in selected checkbox and the content filled in the additional text field are thought as desired search focus.

(2) **The *searchitem* filed in metadata.** Only this field in metadata is used to provide short search terms, other fields are all ignored.

4. Refine the Query

Based on the initial queries used in the baseline run, we improve them using the desired focus newly located. But different update styles are used according to how the focuses are located.

(1) Focus from returned Clarification Form

The words/phrases in each selected checkbox and the filled content in return CF are thought as one search focus. Based on the kernel terms in initial query and the current search item, a sub-query is constructed for a specific search focus. Then the initial query is divided into several queries for different search focus.

(2) Focus from *searchitem* field

All the search terms in the **searchitem** field are simply added to the initial query as new weighted terms. They are merged using Rocchio-like feedback mechanism.

From the above improvement, we construct the search query for the final run.

5. Refine the Query

5.1 Return type detection

For various topics, the user want to receive different search result: document, passage and sentence. We decide the return type by following rules:

- We return document if topic require so.
- For passage and sentence, we usually return the result based on paragraph. For passage, usually there are one or two paragraphs are included. For sentence, it is nearly impossible to present an efficient result in such rough retrieval, therefore a paragraph will be more meaningful.
- If any type is welcomed, we analyze the topic description and decide the result should be passage or document. For example, if there are some words ' the document should....'in the description, then we think a document should be returned.

5.2 Paragraph level indexing

For test corpus, we built an index based on the document level. Since the paragraph will also be returned, we create a new index based on the paragraph. Each paragraph in the document is taken as a single passage and indexed. For some short paragraphs, we merge them to the neighbor paragraphs until the length of this paragraph to be indexed is large than average paragraph length of the corpus.

5.3 Result merging for final submission

All the improved queries are submitted in the document index or paragraph index according to their return type. For topics that return passage and sentence, we also do the retrieval work in the document index. Before getting the final result, we do the following work to the scored list:

- Merge by sub-query: For the topics which have sub-queries presenting different search focus, the final retrieval result is the combination of all of sub-queries, and the scored item is ranked as their order in baseline run(for passage and sentence, they use the order of their host document).
- Document detection for passage and sentence: we return paragraph when topic require a passage or sentence. To keep out of the noise paragraphs, for a retrieved relevant paragraph, only its host document is also ranked as the topic-relevant that can we set it to the returned final result.

6. Final Submission and Evaluation

We finally submitted three runs which expand query using different data source (but with same weighting/scoring parameters for query). For each run, the detail parameters and its evaluation result(also include the baseline run) are listed in table 1 and table 2.

Table 1. Parameters Setup in Each Run

Run	Fields used in task description	Use Clarification Form	Use Metadata	Merge Result
Baseline Run	Title Description Narrative			
TUCSHARD1		Yes	Yes	Yes
TUCSHARD2		Yes	No	Yes
TUCSHARD3		Yes	Yes	No

Table 2. The TREC Evaluation Result for Each Run

Run	Evaluation			
	Passage level		Document level	
	R-Precision	F-score at 100 passage	R-Precison	
			Hard-rel	Soft-rel
Baseline Run	0.1235	0.1294	0.1960	0.2560
TUCSHARD1	0.1868	0.1396	0.2148	0.2818
TUCSHARD2	0.1655	0.1296	0.2012	0.2627
TUCSHARD3	0.1868	0.1396	0.2138	0.2711

7. Conclusions

The evaluation result tell us the clarification form, in lifting the query precision, work better than the metadata. Some of our work later on constructing clarification form using certain cluster algorithm provided us more satisfied result. Also we noticed the result merge seems an effective tool, especially in small amount of documents returned.

Reference

1. Robertson, S. E., and Walker, S. Okapi/Keenbow at TREC-8. In Proceedings of the *TREC-8*. Maryland. 1999.
2. S.E. Robertson, H. Zaragoza, M. Taylor, Microsoft Research Ltd. HARD Track Overview in TREC 2003 High Accuracy Retrieval from Documents. In Notebook of TREC-2003.
3. T. Morton. Using Coreference in Question Answering. In Proceedings of the TREC-8. Maryland. 1999.
4. Y. Ogawa, H. Mano, M. Narita, S. Honma. Structuring and Expanding Queries in the Probabilistic Model. In Proceedings of the TREC-9. Maryland. 2000.
5. R. B. Allen, P. Obry and M. Littman. An interface for navigating clustered document sets returned by queries. In Proceedings of the ACM Conference on Organizational Computing Systems, 1993

THUIR at TREC 2003: Novelty, Robust and Web*

Min Zhang, Chuan Lin, Yiqun Liu, Le Zhao, Shaoping Ma
State Key Lab of Intelligent Tech. & Sys., CST Dept, Tsinghua University, Beijing 100084, China
z-m@tsinghua.edu.cn

This is the second time that Tsinghua University Information Retrieval Group (THUIR) participates in TREC. In this year, we took part in four tracks: novelty, robust, web and HARD, describing in following sections, respectively. A new IR system named TMiner has been built on which all experiments have been performed. In the system, Primary Feature Model (PFM)[1] has been proposed and combined with BM2500 term weighting [2], which led to encouraging results. Word-pair searching has also been performed and improves system precision. Both approaches are described in robust experiments (section 2), and they were also used in web track experiments.

1. Novelty track

Our research on this year's novelty track mainly focused on four aspects: (1) unsupervised relevance judgment; (2) efficient sentence redundancy computing; (3) supervised sentence classification; (4) supervised redundancy threshold learning.

1.1. Unsupervised relevance judgment

The work of finding relevant information is useful for task1 and task3.· Since words mismatch problem is dominant in sentence comparison, three kinds of approaches have been carried out in unsupervised relevance judgment to solve the problem as following.

(1) Query Expansion (QE) using WordNet synonymy and hyponomy, and Dr Lin Dekang's dictionary of term dependency [3];

(2) QE with query terms' local co-occurrence words in a window of W according to supposed relevant document set (without initial search, named *LCE* in our previous study[4]). In task3, the co-occurrence information was got in given relevant sentences in top five documents;

(3) Pseudo relevance feedback. After initial retrieval, top M terms in top ranked N_1 sentences and not in last N_2 sentences were added to the query. In task3, the top documents were defined as given relevant sentences.

Approach (1) and (2) had been already described in our last year's novelty track report [4], and were to make further observation in this year's tasks. Therefore we only give some more information about approach (3) as follows.

Local feedback strategies are based on expanding the query with terms correlated to known query terms. Generally, the expanding n terms are extracted from the top m document (In our novelty track experiments, each sentence is taken as an individual document, m=2) after initial search. The n terms are chosen based on the similarity sim(q, k_v) of them[5]. The value of sim(q, k_v) is calculated as:

$$sim(q,k_v) = \vec{q} \cdot \vec{k_v} = \sum_{k_u \in Q} w_{u,q} \times c_{u,v}$$

Where $w_{u,q}$ is indexed query weight. $c_{u,v}$ is calculated as follows in our experiments:

* Supported by the Chinese National Key Foundation Research & Development Plan (Grant G1998030509), Natural Science Foundation No.60223004, and National 863 High Technology Project No. 2001AA114082.

i. $c_{u,v} = \sum_{d_j \in D_l} f_{s_u,j} \times f_{s_v,j}$,

where $f_{s_u,j}$ and $f_{s_v,j}$ are frequencies that term s_u and s_v occurred in the whole document, respectively.

ii. $c_{u,v} = \sum_{d_j \in D_l} f'_{s_u,j} \times f'_{s_v,j}$

where $f'_{s_u,j}$ and $f'_{s_v,j}$ are frequencies that term s_u and s_v occurred within the distance of *W* words to the observing query term, respectively. In our experiments, the window *W* is set to 3.

iii. $c_{u,v} = \sum_{k_i \in V(s_u)} \sum_{k_j \in V(s_v)} \frac{1}{r(k_i, k_j)}$

Where $r(k_i, k_j)$ is the distance between terms k_i and k_j in the same sentence. If k_i and k_j are in different documents, $r(k_i, k_j) = \infty$.

There are quite a few terms that occur in a great deal of sentences frequently. They may be in both relevant and irrelevant sentences. We assume that the last *k* sentences in the initial retrieval are not relevant (*k*=25 in all experiments), therefore terms in these sentences are useless and should not be expanded.

Table 1 shows experimental results with above approaches. All QE were performed on short queries. It is shown that all QE approaches improve the system performance. The more terms expanded, the better results were got. Among all QE approaches we observed, local co-occurrence within 10 words' window with Mutual Information weighting performs best, which made 14.5% improvement. And QE with Dr Lin Dekang's proximity and dependency dictionary are also greatly helpful. Unfortunately, our official run THUIRnv0312 used the one with almost the least improvement.

Table1 experimental results of QE with short query in task 1

	QE approaches	P	R	F	Improvement (vs short)
No QE	Short query	0.633	0.637	0.552	--
	Long query	**0.593**	**0.805**	**0.622**	**+12.7%**
WordNet	synset	0.625	0.665	0.564	+2.17%
	Hyponomy	0.618	0.68	0.569	+3.08%
	Synset + hyponomy	0.616	0.695	0.575	+4.17%
Dr Lin dekang's Dictionary	**proximity**	**0.58**	**0.831**	**0.608**	**+10.1%**
	dependency	**0.59**	**0.827**	**0.616**	**+11.6%**
Local Cooccurrence Expansion	MI,win = 1	0.557	0.866	0.613	+11.1%
	MI,win=1,sim>0.1	0.632	0.638	0.552	0%
	MI,win = 10	**0.536**	**0.955**	**0.632**	**+14.5%**
Pseudo Feedback	Top 10 terms	0.593	0.716	0.584	+5.8%
	Top 15 terms	0.59	0.732	0.587	+6.34%
	Top 100 terms	0.589	0.744	0.594	+7.61%
Combine (Official run)	***Syn+mi_win1_sim>0.1***	***0.624***	***0.665***	***0.564***	*+2.17%*

1.2. Efficient sentence redundancy computing

On sentence redundancy elimination, rather than general similarity computing, we used unsymmetrical sentence overlap. That is the same as last year. Our experimental results show it makes trivial improvement comparing to the symmetrical measure of similarity.

Besides, the subtopic-based redundancy elimination has been proposed. Generally in a topic, several key-points are concerned, while only one point can be described in each result sentence. Therefore, a natural idea is to divide the topic into subtopics, and then inter-topic documents, inborn, take novel information. Therefore only documents in the same subtopic (cluster) should be used to calculate redundancy. Considering that clusters will help prevent the elimination of inter-cluster documents, if clusters are neatly defined, a better recall will be assured.

Three subtopic clustering approaches have been proposed: one is topic-oriented, and another two are document-oriented.

For topic-oriented clustering approach, <title>, <description>, and each sentence in "narrative" were taken as subtopic individually. Documents were clustered to subtopics according to their distance to each subtopic description. (Shown as subtopic I in following)

For document-oriented clustering, subtopic clusters are built by two ways:

(1) Clustering with syntax analysis: To "event" type topic, take date (year and month) as the feature; to "opinion" type topic, the opinion holder is taken as the feature. Both features were extracted using rules automatically. (Shown as subtopic II in following)

(2) Automatic results clustering by sentence vector distance using KNN-like approach. The sentences vectors are extremely sparse so that only a small number of clusters were formed. Therefore, the method hardly improves (though not deteriorating) the results. (Shown as subtopic III in following)

Table 2 shows the effects of different subtopic-based redundancy elimination approaches. All official runs we submitted in task1 were using overlap-based redundancy computing, and the elimination thresholds were all set to 0.55 in task1.

In task2, mistakes has been made during upload the official runs results because of the coming of submit time deadline. Then all official runs are all wrong in task2, which we can not resume and the THUIRnv0221 and THUIRnv0222 are the same ones. The correct ones we tend to submit are also listed in Table2 (un-official runs).

Table 2 Results of subtopic-based redundancy elimination (task2)

Task		P	R	F	
1	Baseline, short query, overlap threshold = 0.55	0.49	0.58	0.453	THUIRnv0313
	Subtopic I, threshold = 0.55	**0.46**	**0.74**	**0.505**	**THUIRnv0315**
	Short query, Subtopic II, threshold = 0.8	0.47	0.62	0.457	un-official run
	Short query, Subtopic III, threshold = 0.8	0.48	0.62	0.458	un-official run
2	Subtopic II, threshold = 0.8	0.708	0.983	0.812	un-official run
	Subtopic III, 4 clusters, threshold = 0.8	**0.713**	**0.982**	**0.815**	**un-official run**
	Baseline, tuned threshold = 0.8	**0.714**	**0.980**	**0.815**	**un-official run**
	Unknown wrong submission	0.68	0.72	0.687	THUIRnv0323
	Unknown wrong submission	0.69	0.69	0.679	THUIRnv0321 & THUIRnv0322

Besides above approaches, redundancy computing based on triple overlap has been proposed and studied. Firstly, extract syntax triples of each relevant sentence. Secondly, only those triples of V or N are kept for further computation. Thirdly, compute overlap of each sentence pair by triples' overlap. At last, give a threshold and eliminate sentences with high overlap score. Following table3 shows the effects of the approach. The approach was used in task4, and the threshold for official runs is learnt by the given five documents. Following Table3 shows the effects of the approach.

Table 3 Results of triple-overlap-based elimination

Elimination threshold	P	R	P*R	F	
0.8	0.716	0.789	0.563	0.734	un-official run
0.85	*0.70*	*0.88*	*0.614*	*0.765*	*THUIRnv0342*
0.9	**0.684**	**0.939**	**0.641**	**0.777**	**un-official run**
0.95	0.652	0.982	0.641	0.771	un-official run
Different threshold for Each topic	0.720	0.780	-	0.728	*THUIRnv0341*
Different threshold for Each topic (tuned parameters)	**0.728**	**0.954**	**0.697**	**0.819**	**un-officia! run**

1.3. Supervised sentence classification

In task3, we taken the problem of finding relevant information as finding a suitable binary sentence classification using provided relevant sentences in top five documents. A SVM classifier has been used. Features were extracted according to key words in training sentences (sentences given in first five documents). The basic assumption is that the features of relevant sentences are different from that of irrelevant sentences. In terms of positive examples and negative examples, SVM finds a hyper-plane in the feature space, which is chosen to maximize the margin of the training positive and negative points [6]. In the given first 5 documents of each topic, relevant sentences are used as positive examples, and the remaining ones are negative examples. Proportion of positive and negative examples has been balanced in our approach by giving a weighting of 150:1. A linear SVM has been trained. We used a SVM package (version 2.4, by Chih-Wei Hsu, etc. [7]) to create the classifier.

It shows that this supervised sentence classification does helpful on finding relevance, which can be seen in the following Table 4.

Table 4 Effects of supervised sentence classification using SVM

	P	R	F	
Baseline (using long query)	0.43	0.73	0.479	THUIRnv0334
Linear SVM	**0.42**	**0.82**	**0.493**	**THUIRnv0331**

1.4. Supervised redundancy threshold learning

As we known, redundancy threshold setting is one of the important issues in finding new information. For unsupervised task (task1 and task2), according to the high similarity between documents and topics, a fixed threshold has been set. While For task3 and task4, supervised learning has been performed. We trained the threshold by two ways: one is to tune a fixed threshold to all topics by the given known documents, and another one is to tune a different parameter for each topic. It seems that the elimination threshold trained by top five documents, which was set to fixed 0.8 in task4, works well to the remaining ones.documents. Further analysis and observation need to be done.

1.5. Submitted official runs

Task	Approach description	P	R	F	
1rel	Short query+feedback	0.61	0.73	0.593	THUIRnv0311
	Synset + LCE(MI, win = 1, sim>0.1)	0.62	0.67	0.564	THUIRnv0312
	Baseline, short query	0.49	0.58	0.453	THUIRnv0313
	cut results when sim<0.95*top_sim	0.63	0.63	0.548	THUIRnv0314
	Subtopic I	0.46	0.74	0.505	THUIRnv0315
1new	Short query+feedback, overlap, threshold=0.55	0.47	0.66	0.481	THUIRnv0311
	Synset + LCE(MI, win = 1, sim>0.1)	0.48	0.61	0.459	THUIRnv0312
	Baseline, short query, overlap threshold = 0.55	0.49	0.58	0.453	THUIRnv0313
	cut results when sim<0.95*top_sim	0.49	0.57	0.451	THUIRnv0314
	Subtopic I, overlap threshold = 0.55	0.46	0.74	0.505	THUIRnv0315
2new	Unknown wrong submission	0.69	0.69	0.679	THUIRnv0321 & THUIRnv0322
	Unknown wrong submission	0.68	0.72	0.687	THUIRnv0323
3rel	SVM classification	0.57	0.87	0.627	THUIRnv0331
	Long query + feedback	0.56	0.89	0.623	THUIRnv0332
	Long query + QE with LCE (window=10)	0.54	0.92	0.621	THUIRnv0333
	Long query	0.58	0.81	0.610	THUIRnv0334
3new	SVM classification, overlap, threshold=0.8	0.42	0.82	0.493	THUIRnv0331
	Long query + feedback， threshold=0.85	0.40	0.86	0.489	THUIRnv0332
	Long query + LCE (win=10), threshold=0.8	0.39	0.88	0.491	THUIRnv0333
	Long query, topic-different threshold	0.43	0.73	0.479	THUIRnv0335
4new	Triple overlap with topic different threshold	0.72	0.78	0.728	THUIRnv0341
	Triple overlap with fixed threshold 0.85	0.70	0.88	0.765	THUIRnv0342
	Overlap, threshold = 0.8	0.68	0.99	0.791	THUIRnv0343
	Overlap + QE (MI, win=1,sim>0.8)	0.68	0.98	0.790	THUIRnv0344
	Overlap + event/opinion clustering, threshold0.8	0.68	0.96	0.780	THUIRnv0345

2. Robust track

In this year's robust track, our basic idea is: "bad" topics are not topics that only seldom documents can be returned after retrieval, but the ones that return too many irrelevant documents. Therefore the key point of our work is to improve the system precision to these topics, namely, to perform a cagey and strict judgment of relevance.

We implemented two novel approaches in our TMiner system, namely Primary Feature Model and Word Pair Search, aiming to enhance system performance in terms of precision. They are used in both robust and web tracks.

Besides, query quality is also an important factor to system performance. We eliminate the negative description of the topic in <narr> field automatically, hence a long query without negative information was generated. Effect of using other fields of topics was also observed, shown in Figure1.

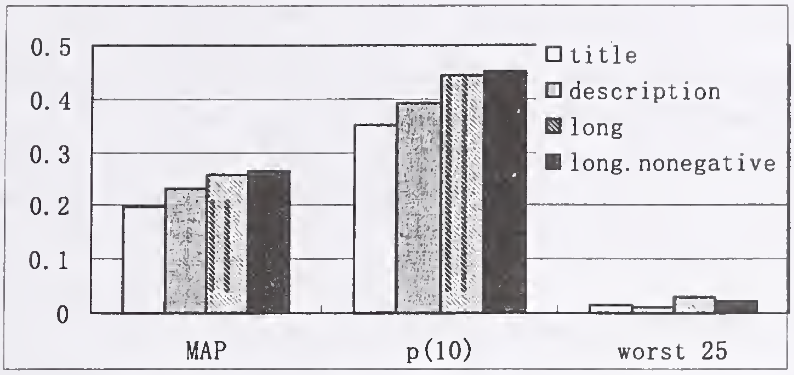


Figure 1 Effects of using different topic field to generate queries

2.1. Primary Feature Model

As we known, terms appear in the title, heading or other emphasized fields in the document are more generally important to the reader than the other body text. They represented the notion of the authors on the main content of the document. We take these special fields as Primary Feature Fields (*PFF*). Terms in the primary feature fields in the entire collection construct a Primary Feature Space (*PFS*). This space is not of uniform distribution. Therefore, terms with higher density in the space should be more significant as features. Generally, the density of the feature dimension can be represented by the term frequency in the primary feature space:

$$D_i = \sum_{k=1}^{N} tf_{ik}$$

Where tf_{ik} is occurrence frequency of term i in *PFF* of document k, and N is the total number of documents in the collection.

Since documents are built by different authors with different backgrounds and writing habits, the description of the field may not be canonical. To reduce the influence of the information leak caused by different author, it's better to limit the same terms from one page contributing to the feature density only once. i.e.

$$tf_{ik} = \begin{cases} 1, & \textit{if term } i \textit{ occurs in the primary field of doc } k \\ 0, & \textit{otherwise} \end{cases}$$

Therefore the density of the feature dimension is be simplified to:

$$D_i = \sum_{k=1}^{N} tf_{ik} = n_i$$

Where n_i is the number of documents which contains term i in its primary feature space in the collection. Then the term weight in the primary feature space can be represented by:

$$w^{(2)} = \log(n+1)$$

Where tf_i and n_i are simplified as tf and n respectively.

By doing so, in a documents collection, the weight of a term in the query is composed of the weights in *PFS* and in body text. Therefore the new similarity scoring function can be presented as:

$$\sum_{T \in Q} [\lambda w_{BT}^{(1)} + (1-\lambda) w_{PF}^{(2)}] \frac{(k_1+1)tf\ (k_3+1)qtf}{(K+tf\ )(k_3+qtf\ )}$$

where $w_{BT}^{(1)}$ is the Robertson/Sparck Jones weight for a term according to body text, $w_{PF}^{(2)}$ is the weight in terms of primary feature field, λ is the impact factor of the body text. When $\lambda = 1$, the scoring function is same as traditional Robertson/Sparck Jones probabilistic model.

In this year's robust task, the <headline> of the documents were used as Primary Feature Field. The effects are shown in Table 5 (λ=0.1). Trivial improvement is got on the performance of worst topics.

2.2. Word Pair Search

The basic idea of word pair is that if adjacent words in the query are also adjacent in the document, then the document would be more likely to be relevant. In our word pair implementation, word pairs in a query are treated as additional terms for the query; and *DF*, *TF* information are calculated and added to the original query term weights by a weighted summation.

$$W = \sum_{t \in query\ terms} W^{(1)} \frac{(k_1+1)\, tf_t\, (k_3+1)\, qtf_t}{(K+tf_t)\,(k_3+qtf_t)} + \lambda \sum_{wp \in query\ wordpairs} W_2$$

$$where \quad W^{(1)} = \log\frac{N-df_t+0.5}{df_t+0.5}, \qquad W_2 = W_2^{(1)} \frac{(k_1+1)\, tf_{wp}\, (k_3+1)\, qtf_{wp}}{(K+tf_{wp})\,(k_3+qtf_{wp})}, \qquad W_2^{(1)} = \log\frac{N-df_{wp}+0.5}{df_{wp}+0.5}$$

Note: a weight λ is multiplied to the word pair part of the total weights to constrain the impact of the word pair weight on the final document results.

DF of word pair (e.g. word pair *AB*) can be estimated in different ways:

a. *DF* is specified by a constant the user provides. This estimation method is called wp_1 in our experiments.

b. *DF* of *AB* can be estimated with the number of documents that contain both the two terms: *A* and *B*. We call it wp_2 method.

c. *DF* of *AB* can be the exact document frequency of the pair *AB*. The wp_3.

Effects of Word Pair Search are shown in Table 6. It is encouraging that although Word Pair Search makes much improvement in terms of average system performance, but it enhances the average performances of "bad" topics obviously. (In the table 6, λ of title and description query were set to 0.2 and 0.3 respectively).

Table 5 Effects on using Primary Feature Model

	MAP	p(10)	worst 25 topics
long	0.2571	0.444	0.0282
long.pfs	0.2597	0.446	0.0289
long.non-neg	0.2667	0.452	0.0235
long.no-neg.pfs	0.2676	0.453	0.0252
description	0.2307	0.392	0.0114
description.pfs	0.2339	0.394	0.0118

Table 6 Effects of Word Pair Search

	MAP	p(10)	worst 25 topics
Title	0.199	0.351	0.0142
Title.wp_3	0.205	0.356	0.0153
	+3.1%	**+1.4%**	**+7.8%**
Desc	0.231	0.392	0.0114
Desc.wp_3	0.240	0.406	0.0132
	+3.8%	**+3.6%**	**+15.8%**

2.3. Submitted Official Runs

		MAP (%)			p(10) (%)			worst 1/4 topics (%)		
		old	new	all	old	new	all	old	new	all
1	**THUIRr0301**	**13.73**	**38.21**	**25.97**	**36.00**	**53.20**	**44.60**	**2.03**	**5.70**	**2.89**
2	THUIRr0302	14.45	38.88	26.67	36.20	54.20	45.20	1.61	5.65	2.35
3	THUIRr0303	13.31	38.00	25.71	35.00	53.00	44.00	1.99	5.61	2.84
4	THUIRr0304	12.37	37.06	24.72	31.20	50.40	40.80	1.52	4.41	1.95
5	THUIRr0305	11.66	37.02	24.34	31.20	50.80	41.00	0.94	4.75	1.48

1: long query retrieval, using BM2500+PFS weighting

2: long query without negative information, using BM2500+PFS weighting

3: long query baseline retrieval

4: combine short query retrieval and retrieval on description field

5: retrieval on description field with Word Pair Search and BM2500 + PFS weighting

3. Web track

3.1. Introduction

In this year's TREC Web Track research, we participated in both Known-Item Search and Topic Distillation Tasks.

In Topic Distillation task, only entry pages of key sites should be returned as results. Therefore, two kinds of approaches have been studied: "Top-to-Bottom" retrieval, and "Bottom-to-Top" one. The key point in both approaches is: how to give a clear and precise definition of entry pages and use it to get an entry page list. According to our definition, we developed an entry page locating algorithm concerning the following characteristics of pages: URL information, document length, in-site out-link rate and in-link number. With the algorithm, four entry page lists have been built according to different threshold (strict or relaxed metric) and number of entry pages returned from one site.

On term weighting, a Primary Feature Space (PFS) model has been proposed. In PFS model, as has been described in section 2, a DF-related weighting is performed on words in Primary Feature Field (PFF), which makes improvement compared with traditional IDF-related weighting. In-link anchor text, title and keywords of in-site-out-link pages, and bold text are proved to be good PFF in our experiments.

Besides, searching with word pair has been added to scoring, which improves the accuracy of result ranking, as has been described in section 2.

A novel site uniting method is also proposed to confirm that any key resource we found is not part of a larger site also principally devoted to the topic.

In Page Finding task, other than in-link anchor text used in TD task, the out-link anchor text are used to decide the webpage candidate set. Then rank the candidates according to the term weighting on full text and return top results. It shows that the out-link anchor text is much more effective than the full text of the page. And it is different to the general idea that the out-link anchor text describes the page the link point to more than the page itself.

Abbreviations were extracted from the corpus automatically, and were expanded in topics.

3.2. Definition and locating algorithm of entry pages

In this year's topic distillation task, we are concentrating solely on websites (represented by their entry pages) as key resources. Before retrieval, it is necessary for us to find out what an entry page is and how to find it. In our dictionary, an entry page is the main entry point of a web site and a bridge between pages inside and outside the web site; it should have the following characteristics:

(a) An entry page can connect (directly or indirectly) to all pages in the web site it belongs to.

(b) When pages outside the web site want to visit pages inside the site, they mainly visit the entry page at first.

According to the definition, we may see that entry pages should have some special features in their URLs, link structures, etc. In our research, we divide entry pages into two types: topic-related and non-topic-related. For topic-related entry pages, their URLs always contain one or more query words. For example, “www.drugabuse.gov/DrugPages/Marijuana.html” is an entry page of its sub-site for the topic “Where can I locate information on the pros and cons of legalizing marijuana”. For non-topic-related entry pages, they always have the following features:

(1) url: root, subroot, named as "index.htm" etc, or named as the topic words of the sub-site;

(2) length: an entry page is usually not too long;

(3) in-site-out-link: an entry page should have enough in-site-out-link;

(4) in-link: an entry page should have enough in-link (especially links from pages in different site/server).

In this way, about 15% pages have been selected to form the key entry pages collection, which cover about 68% useful key resources for 50 testing queries.

3.3. Topic distillation with entry page lists

With the entry page locating algorithm, four entry page lists have been built according to different threshold (strict or relaxed metric) and number of entry pages returned from one site.

List Name	Description
Base list	Strict threshold
Unite list	Loose threshold, but only one Entry Page per site
Parse list	Strict threshold, and only one Entry Page per site
Root list	Non-file pages only, and one Entry Page per site

1) **Top-down topic distillation**

In the top-down algorithm, we restrict our data set on selected entry pages. With an ideal entry page locating algorithm, entry page retrieval is obviously a clever choice for topic distillation task, because it discard pages that are not possibly key resources. But in practical world, any locating algorithm brings in noises and may discard some entry pages which will perhaps be key site representatives.

2) **Bottom-up topic distillation**

In the bottom-up algorithm, we try to locate key sites instead of key pages. Entry page of one site is returned as result. We first retrieve on full text, and then use the following algorithm to re-rank the results:

```
For each item in the Entry Page list {
        For each page N an Entry Page A links to {
            weight (A) += weight (N)
        }
}
Re-rank the result list;
```

3) **Comparison with ordinary full text-retrieval**

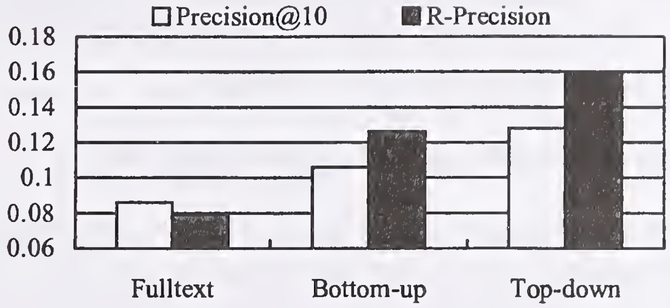


Figure 2 bottom-up and top-down comparison

According to experiment results in Figure 2, both bottom-up and top-down perform much better than full-text retrieval. It shows the entry page locating algorithm is conceive and reliable. And the top-down method is perhaps more useful for this kind of topic distillation.

3.4. Using document structures

The effect of using HTML document structure is studied in our experiments. It is found that in-link anchor information is useful for known-item search. We also tested several new parts of HTML document in topic distillation experiments. They are: in-site out-link anchors, in-site out-link page titles and in-site out-link page keywords.

For a certain page A, in-site out-link anchors are anchors describing links from A to other pages in the site where A is. So the anchors are located on A; that is quite different from frequently-used in-link anchors which are on pages linking to A. However, in-site out-link title / keyword are on pages A points to, although in the entry page finding process, we can consider them as descriptions of A.

Using of in-site out-link is particularly useful for topic distillation task, shown in Figure 3, the experiment is on the entry page data set. Full text retrieval experiment is shown in Figure 4.

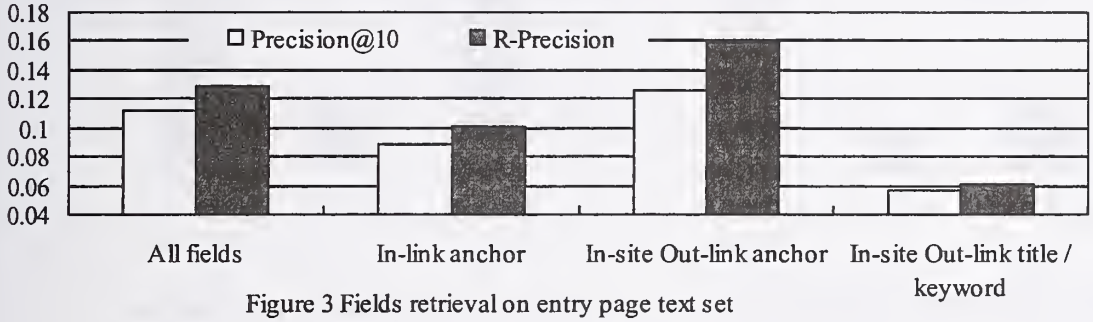


Figure 3 Fields retrieval on entry page text set

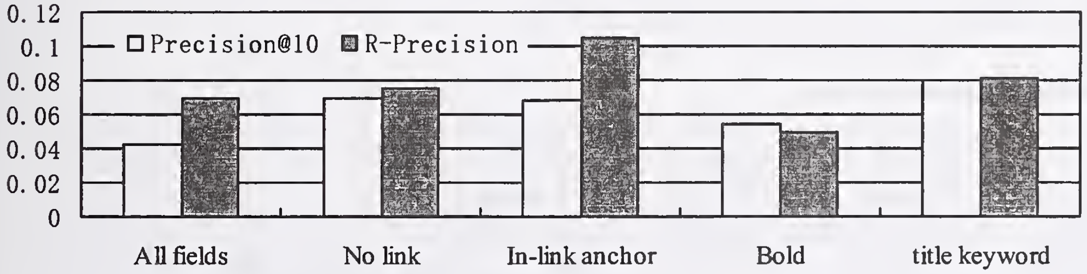


Figure 4 Fields retrieval on full text set

Then we can conclude that:

To the whole collection, in-link anchor performs better. To the entry pages text set: 1) in-link anchor no longer works; 2) full text is reliable; 3) in-site out-link anchor leads to astonishing results; 4) in-site title / keywords may be too short to improve retrieval results.

3.5. Topic distillation with different weighting schemes

We experimented with the following weighting schemes in the topic distillation task: Primary feature space model and BM2500 model. In the chart below, category axis represents the rate BM2500/PFS. It shows that with a broad parameter scale, we get improvement using PFS weighting scheme combined with BM2500 weighting.

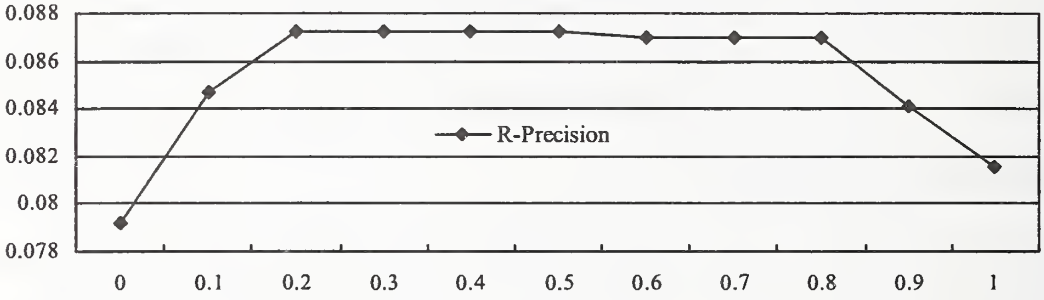
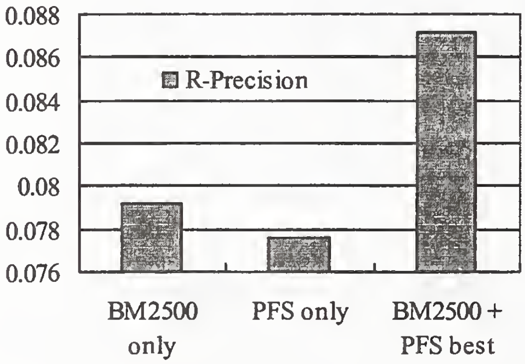


Figure 5 PFS + BM2500 retrieval

We compared the two weighting schemes in the following charts. The first experiment is performed on full text data set; while the second is on the entry page set. It shows that PFS method only performs poor, but it can stably improve performance if we combines it with the BM2500 weighting scheme.



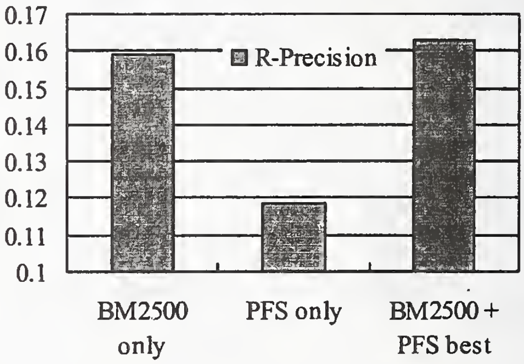


Figure 6 weight schmes on full text set

Figure 7 weight schmes on entry page set

3.6. Runs submitted and Evaluation Results

Topic distillation runs description:

Runs	Data Set	Weighting	Field Selection	R-pre
THUIRtd0301	Parse list	BM2500 only	Full text bold	0.1036
THUIRtd0302	Unite list	BM2500 only	Full text bold	0.0994
THUIRtd0303	Base list	PFS only	Entry Page in-site out-link title / keyword	0.0786

THUIRtd0304	Root list	PFS only	Entry Page in-site out-link anchor	0.0692
THUIRtd0305	Full text	BM2500+ PFS	Full text all fields	0.1262
Unofficial run1	Base list	BM2500 only	Entry page all fields	**0.1293**
Unofficial run2	Base list	BM2500+ PFS	Entry Page in-site out-link anchor	**0.1629**

Known item search runs description and results:

Runs	Description	MRR
THUIRpf0301	Full text retrieval +NPrank+ THUIRtd0305 (BM2500)	0.561
THUIRpf0302	Full Text retrieval + THUIRtd0305 (BM2500)	0.45
THUIRpf0303	THUIRpf0304 + NPrank	0.496
THUIRpf0304	Retrieve on anchor text, using abbr., BM2500	0.463
THUIRpf0305	Retrieve on anchor text, not using abbr., BM2500	0.466

Reference

[1] Min Zhang, Ruihua Song, and Shaoping Ma, DF or IDF? On the Use of HTML Primary Feature Fields for Web IR, the 12th World Wide Web conference, (www2003), poster, May, Hungarian, 2003.

[2] Robertson, S. E., and Walker, S., Okapi/Keenbow at TREC-8. In TREC-8.

[3] http://www.cs.ualberta.ca/~lindek/downloads.htm

[4] Min Zhang et al, Expansion-Based Technologies in Finding Relevant and New Information: THU TREC2002 Novelty Track Experiments, in TREC-2002.

[5] R.Attar and A. S. Fraenkel. Local feedback in full-text retrieval systems, Journal of the ACM, 24(3): 397-417, 1977

[6] Marti A. Hearst, Trends and controversies: Support vector machine. IEEE Intelligent Systems, 13(4): 18-28, 1998.

[7] http://www.csie.ntu.edu.tw/~cjlin/libsvm/index.html

Document Structure with IRTools

Gregory B. Newby[*]
Arctic Region Supercomputing Center
University of Alaska Fairbanks

Abstract

The IRTools software toolkit was modified for 2003 to utilize a MySQL database for the inverted index. Indexing was for each occurrence of each term in the collection, with HTML structure, location offset, paragraph, and subdocument weight considered. This structure enables some more sophisticated queries than a "bag of words" approach. *Post hoc* results from the TREC 2002 Named Page Web task are presented, in which a staged fall through approach to topic processing yielded good results, with exact precision of 0.49. The paper also provides an overview of IRTools and its interactive interface, as well as an invitation for IR researchers to get involved with the GridIR standards formation process.

Introduction

This year, the IRTools software toolkit was not quite ready in time for the TREC 2003 Web submission. Instead, this paper describes a set of runs on the 2002 Named Page Web track completed in October and November 2003. The paper should be interesting to TREC participants because it describes a rather different, and considerably more flexible, approach to information retrieval (IR) than described in the author's prior TREC entries (Newby, 2002).

IRTools is a software toolkit intended for IR research. Development was partially funded by the NSF, and the software is freely downloadable at http://sourceforge.net/projects/irtools. The goal of IRTools, scheduled for official release in 2004, is to operate as a programmer's toolkit for IR experimentation. It encompasses several major IR models (the vector space model or VSM, Boolean retrieval, and variations on latent semantic indexing or LSI). It enables both interactive use via a Web-based front end, and batch-oriented retrieval for TREC-like experiments.

IRTools is one of several systems being adopted as a reference system for Grid Information Retrieval (GIR, see http://www.gridir.org), a working group under the Global Grid Forum (http://www.gridforum.org), which the author co-chairs. GIR-WG has presented requirements and architecture documents (Gamiel et al. 2003; Nassar et al. 2003), and members of the working group are developing reference implementation systems as both proof-of-concept and early models for operational systems. GridIR is similar to WAIS (Kahle et al., 1992), in that multiple retrieval collections are federated in *ad hoc* ways to provide merged results. GridIR operates in standards-based environment such as Web services (http://www.w3c.org/2002/ws) and the Open Grid Services Architecture and other

[*] 910 Yukon Drive, Fairbanks AK 99775. newby@arsc.edu or http://www.arsc.edu/~newby . The research described here was partially funded by National Science Foundation grant #0352029.

Grid standards (Foster, 2003). These standards offer infrastructure for end-to-end security, event notification, and other capabilities.

In this paper, some of the back-end of IRTools is described. *Post hoc* results from the 2002 Named Page Web track results are presented. Future research is described.

Data Structure and Back End

Background

Similarly to most long-time TREC participants, the author has gone through many different variations in the code base for IRTools and predecessor systems. Fundamentally, though, these IR systems have several main components and purposes in common:

1. Document metadata, in which documents are assigned document ID numbers (docids). How many terms per document? What TREC document number (docnum), URL or other label is associated with a document?
2. Term metadata, in which terms are assigned term ID numbers (termids). How frequently does the term occur in a collection?
3. Inverted index, in which lists of docids for each termid are stored for quick lookup. How frequently does term i occur in document j, and at what locations in the document?
4. Sequential index, in which representations of documents are stored for relevance feedback, query expansion, context extracts, etc. What terms occur near term i in document j?

One of the largest fundamental technical challenges for nearly all IR systems is to quickly determine a set of candidate docids, given a list of termids as query. Set building occurs when the individual lists of docids from inverted index entries are merged (or sorted and merged). Once the sets are built, ranking of results can occur. This general approach is taken regardless of whether a Boolean AND or a Boolean OR is used, as well as for relevance feedback or other forms of query expansion.

We can consider the problem of information retrieval in terms of matrices of term-document relations. Table 1 shows a small set of documents and their term frequencies:

Table 1: Term by Document Matrix

	Doc 1	Doc 2	Doc 3	Doc 4	Doc 5
Term 1	2	0	1	0	1
Term 2	0	1	0	0	2
Term 3	0	3	1	0	2
Term 4	0	0	0	3	0

In Table 1, most terms do not occur in most documents, and most documents do not have most terms. This results in many cells with zero entries. Such a sparse matrix may be more efficiently represented as a list of postings in an inverted file, as shown in Table 2.

Table 2: Postings in an Inverted Index

Term 1	Doc 1=2	Doc3=1	Doc 5=1
Term 2	Doc 2=1	Doc 5=2	
Term 3	Doc 2=3	Doc 5=20	
Term 4	Doc 2=3	Doc 3=1	Doc 5=2

The advantage of the method shown in Table 2 over Table 1 is that significant space savings results by not storing the zero cells (well over 99% of cells in large IR test collections). Furthermore, multi-way sort and merge algorithms (see Knuth, 1998) enable stepping through the list of postings for each query term without requiring that the entire inverted index, or even a complete row of postings, be in main memory.

The benefit of the inverted index is not without a price, however: in Table 1, it is a simple matter to see what terms occur in a particular document by reading down the columns, and document statistics such as average term frequency are easily computed on the fly. With an inverted index, the other structures mentioned earlier (or something similar) are required for computing term and document weights and for query expansion.

In practice, of course, there is considerable variety in exactly what is needed by a particular IR system for effective ranking. By post-processing the inverted index, for example, it might be possible to rank entries by document weight, such that early entries are more likely to be associated with highly ranked documents.

Postings in IRTools

In past years, IRTools and earlier systems have used a variety of file structures to store the inverted index and other data about an IR test collection. The primary desire left unfulfilled by these file structures is to consider document qualities beyond the "bag of words" level. The bag of words, which is one of the fundamental (often implicit) approaches used in IR literature, looks at term occurrences in documents but not at where those terms occur. Furthermore, the bag of words model does not take document structure into account – for example, HTML documents have title tags, meta tags, paragraph tags, and so forth which might be important for computing the weight of a term in a document. Moreover, term position within documents is the fundamental element for phrase matching, or adjacency/nearness measures. Alternate structures, such as PAT arrays (Gonnet et al., 1992), may be employed for this, but for current purposes we would like to see whether the inverted index might be modified to add these capabilities.

By taking document structure and term position into account, new types of queries are enabled. "Term 1, near term 2, both in a TITLE tag." "Term 1 and Term 2 in the same paragraph tag will be weighted twice as much as when they are not in the same paragraph." "Term 1 and Term 2 in the same document, but without Term 3 as a table heading."

Two challenges were encountered in implementing this level of analysis. First, the model needed to change from a bag of words, in which a posting in the inverted index is made for each term in each document, to a model where information needs to be stored for each *occurrence* of a term in each document. Secondly, in addition to fast search methods at the term level (i.e., the rows in Table 2 above), fast search on other qualities are also required, such as on the paragraph, subdocument, and offset location in a document. These goals seemed to fit well with what database management systems are good for, so MySQL was chosen for the TREC 2003 implementation of the inverted index.

MySQL, like PostgreSQL, is free and open source, and therefore suitable for use with IRTools. Both have similar capabilities and characteristics, but the availability of a C++ API for MySQL was a deciding factor for its choice. MySQL's MyISAM or INNODB table styles utilize either the Berkeley DB or similar approaches to storage on disk, in B-trees and related file structures. (We note here that IRTools has utilized Berkeley DB tables directly through their C++ API for several years.) Table 3 shows the table structure for the inverted index. The term and document data remained in Berkeley DB tables managed by IRTools directly, and will not be further elaborated on here.

Table 3: Inverted Index Table Structure in MySQL

Name	DocID	Offset	TermID	TagListID	WhichPara	WeightInSubdoc
Type	uint	usmall	uint	usmall	utiny	ufloat

The size and range of unsigned integers (uints) is 4 bytes, from 0 to 4GB, unsigned smalls (usmall) is 2 bytes (0 to 64K), and tiny integers (utiny) from 0 to 255. This nets 17 bytes per posting – that is, per term occurrence in a document, plus overhead and indexing. As shown in Table 4, an index was built on each of these database columns as well as combinations of columns, which more than doubled both insertion time and the database size on disk, but allowed many queries to run without requiring linear searches through the postings.

In the postings, Offset is simply the word number in the document, with any term offset over 64K being skipped. WhichPara is simply the paragraph number (with some simple rules for "what is a paragraph" in HTML, implemented in the LibWWW parser), with any paragraphs over 255 being mapped to 255. WeightInSubdoc is simply a traditional document weight that is incremented for additional occurrences. The ability to uniquely weigh a term occurrence within a document is powerful, but further research is needed to determine what sort of weighting scores to implement.

TagListID is the most interesting column. In the collection, a list of all HTML tag sequences was kept. For example, well-formed HTML should start (after a DOCTYPE) with an HTML tag, a HEAD tag, and perhaps a TITLE or META tag. So, the first title term might occur in a tag sequence such as: HTML, HEAD, TITLE. In the inverted index, a unique TagListID was assigned to each tag sequence. By normalizing the HTML data and forcing them to be indexed as well formed, this enables searches for terms in title tags, as well as much more specific searches (e.g., terms in italics, in table columns, in table rows, in the body section of an HTML document). In practice, it was found that just under 64K unique TagListIDs were needed for the Web02 collection.

Table 4: Inverted Index Table Creation

```
CREATE TABLE `inv0web02` (
 `docid` int(10) unsigned NOT NULL default '0',
 `offset` smallint(5) unsigned NOT NULL default '0',
 `termid` int(10) unsigned NOT NULL default '0',
 `taglistid` smallint(5) unsigned NOT NULL default '0',
 `whichpara` tinyint(3) unsigned NOT NULL default '0',
 `weight_in_subdoc` float unsigned NOT NULL default '0',
 PRIMARY KEY  (`docid`,`offset`),
  KEY `termid_index` (`termid`),
 KEY `whichpara_index` (`whichpara`),
 KEY `taglistid_index` (`taglistid`),
 KEY `weight_index` (`weight_in_subdoc`),
 KEY `docid_index` (`docid`),
 KEY `offset_index` (`offset`),
 KEY `termid_docid_whichpara_offset` (`termid`,`docid`,`whichpara`,`offset`),
 KEY `termid_docid_whichpara_offset_weight`
(`termid`,`docid`,`whichpara`,`offset`,`weight_in_subdoc`),
 KEY `termid_docid_taglist_weight` (`termid`,`docid`,`taglistid`,`weight_in_subdoc`),
 KEY `termid_docid_taglistid_offset_weight`
(`termid`,`docid`,`taglistid`,`offset`,`weight_in_subdoc`),
 KEY `termid_docid` (`termid`,`docid`)
) TYPE=MyISAM
```

Indexing the Web02 Collection

IRTools was used to index the TREC Web02 test collection of 1.2M HTML documents (about 20GB). Other than the MySQL database described above, the main innovation this year was to add a complete HTML parser. LibWWW from the World Wide Web consortium was chosen. LibWWW has proven to be fast and reasonably efficient, but poorly documented and rife with memory leaks that occur in ongoing use (such as the multi-day indexing process of Web02). Term and document information was stored in Berkeley DB files, as in prior years, and the sequential index was dropped, because it can be efficiently generated by selecting all postings for a DocID from the MySQL database.

Statistics for the Web02 collection are presented in Table 5. Terms with more than 20 occurrences in a document were arbitrarily capped at 20 (although term counts continued to accrue, data concerning the 21st term occurrence and beyond were omitted). All terms were considered, without use of a stoplist, truncation or stemming.

Table 5: Web02 Build Statistics

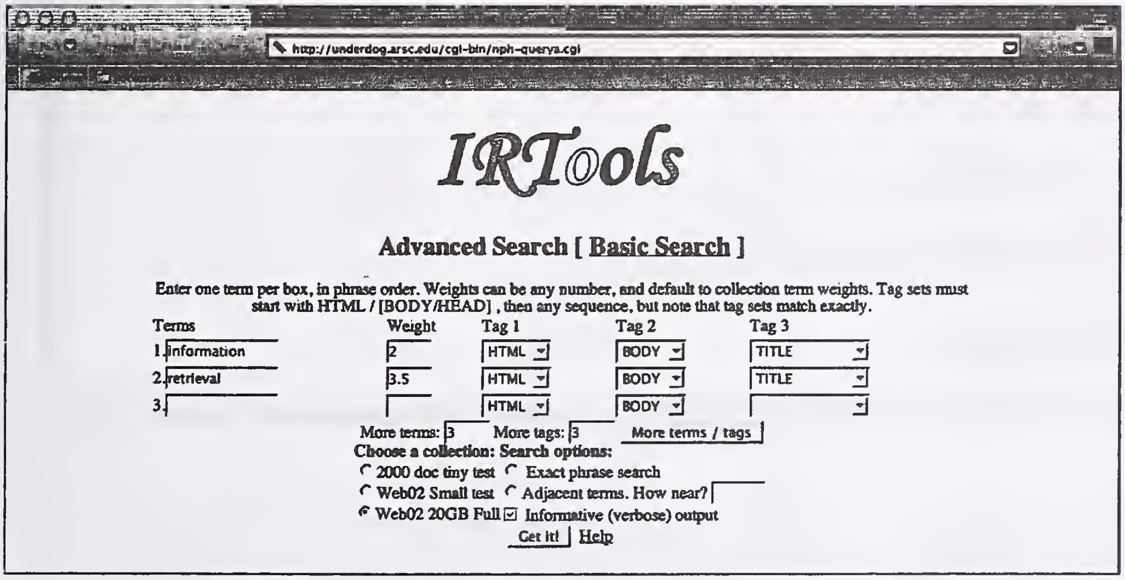
System	Dell 4600
	Dual Xeon processor 2.8Ghz
	16GB RAM
	960GB RAID-5 disk
Processing time	5 days
MySQL database size (inverted index)	About 80GB
Number of inverted rows (postings)	About 468,000,000
Size of other file structures	About 500MB
Total index size	About 81GB

We note that 468 million rows is, indeed, a large database table. At a ratio of 4:1, the size of the database compared to input data is not nearly as favorable as for other IR systems. Furthermore, the random access nature of inserts (combined with numerous indexes) resulted in insertions which were very much disk bound. While indexing, CPU utilization was often below 10%, while awaiting disk I/O. Generally this behavior is consistent with other indexes for IR, and no slower than the Berkeley DB or other B-tree disk methods. The difference was that keeping track of virtually every term occurrence resulted in a far larger database.

Query Flexibility

With the use of the MySQL database and indices on all columns, IRTools is suitable for both batch-oriented TREC topics, as well as a variety of on-demand queries. Context-sensitive document extracts are simply a matter of retrieving a range of database rows for a particular DocID. So is a title for display. Figures 1 and 2 illustrate some of the query flexibility and output options provided by IRTools through a Web-based front-end.

Figure 1: Advanced IRTools Query Form



http://underdog.arsc.edu/cgi-bin/nph-querya.cgi

IRTools

Advanced Search [Basic Search]

Enter one term per box, in phrase order. Weights can be any number, and default to collection term weights. Tag sets must start with HTML / [BODY/HEAD] , then any sequence, but note that tag sets match exactly.

Terms	Weight	Tag 1	Tag 2	Tag 3
1. information	2	HTML	BODY	TITLE
2. retrieval	3.5	HTML	BODY	TITLE
3.		HTML	BODY	

More terms: 3 More tags: 3 More terms / tags

Choose a collection: Search options:

○ 2000 doc tiny test ○ Exact phrase search

○ Web02 Small test ○ Adjacent terms. How near?

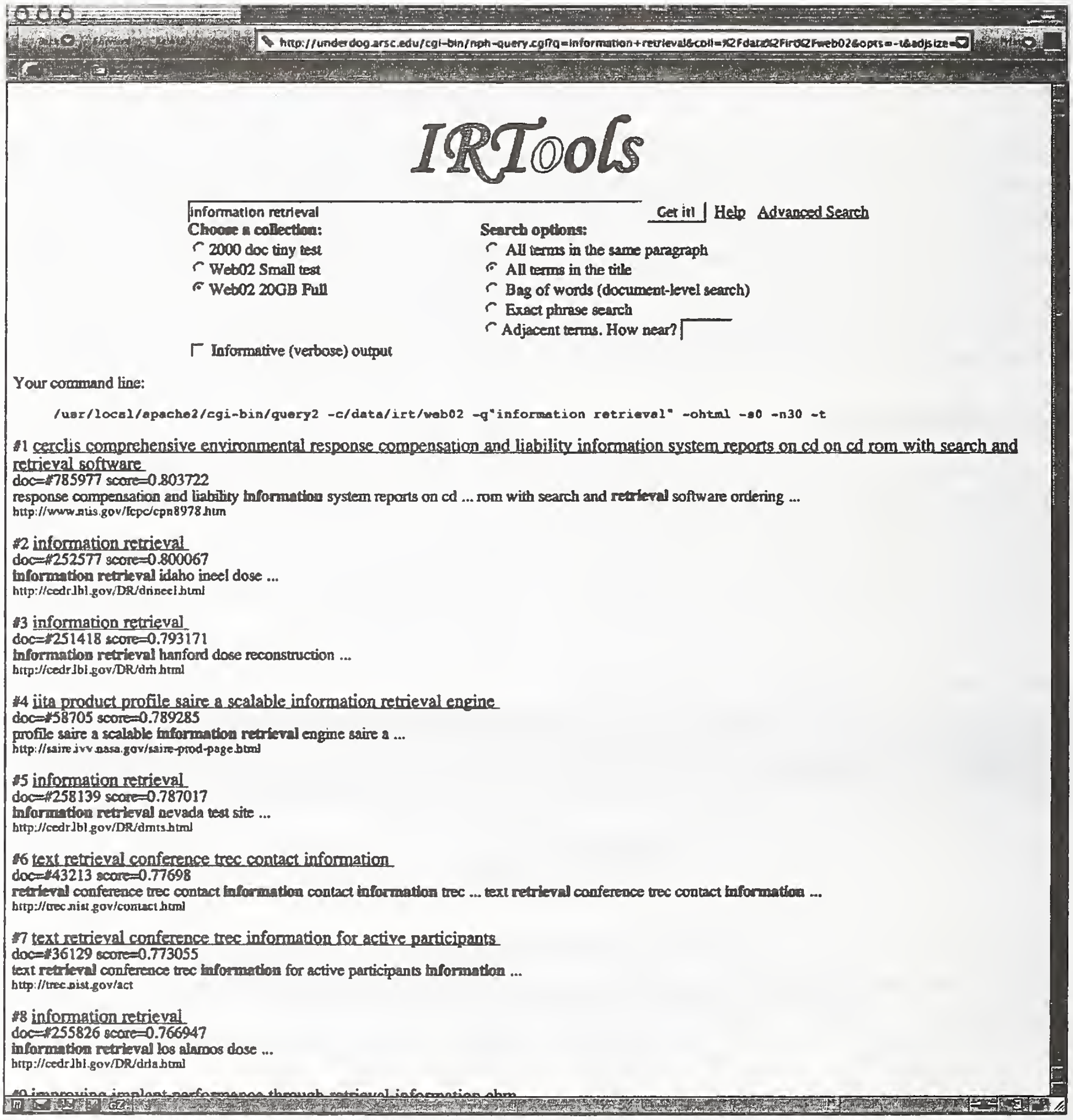
● Web02 20GB Full ☑ Informative (verbose) output

Get it! Help

Tasks and Outcomes

The revisions to IRTools discussed here were not quite ready in time for the TREC 2003 Web track deadline (results from running TREC 2003 tasks on last year's system were submitted instead). Here, we present results using relevance judgments (qrels) from the TREC 2002 Named Page finding task of the Web track. In that task, the goal was to identify particular pages or Web sites based on a description of their name.

Figure 2: Basic IRTools Query with Output



http://underdog.arsc.edu/cgi-bin/nph-query.cgi?q=information+retrieval&coll=%2Fdata%2Firt%2Fweb02&opts=-t&adjsize=

IRTools

information retrieval Get it! | Help Advanced Search

Choose a collection:
○ 2000 doc tiny test
○ Web02 Small test
● Web02 20GB Full

Search options:
○ All terms in the same paragraph
● All terms in the title
○ Bag of words (document-level search)
○ Exact phrase search
○ Adjacent terms. How near?

□ Informative (verbose) output

Your command line:

```
    /usr/local/apache2/cgi-bin/query2 -c/data/irt/web02 -q"information retrieval" -ohtml -s0 -n30 -t
```

#1 cerclis comprehensive environmental response compensation and liability information system reports on cd on cd rom with search and retrieval software
doc=#785977 score=0.803722
response compensation and liability **information** system reports on cd ... rom with search and **retrieval** software ordering ...
http://www.ntis.gov/fcpc/cpn8978.htm

#2 information retrieval
doc=#252577 score=0.800067
information retrieval idaho ineel dose ...
http://cedr.lbl.gov/DR/drineel.html

#3 information retrieval
doc=#251418 score=0.793171
information retrieval hanford dose reconstruction ...
http://cedr.lbl.gov/DR/drh.html

#4 iita product profile saire a scalable information retrieval engine
doc=#58705 score=0.789285
profile saire a scalable **information retrieval** engine saire a ...
http://saire.ivv.nasa.gov/saire-prod-page.html

#5 information retrieval
doc=#258139 score=0.787017
information retrieval nevada test site ...
http://cedr.lbl.gov/DR/drnts.html

#6 text retrieval conference trec contact information
doc=#43213 score=0.77698
retrieval conference trec contact **information** contact **information** trec ... text **retrieval** conference trec contact **information** ...
http://trec.nist.gov/contact.html

#7 text retrieval conference trec information for active participants
doc=#36129 score=0.773055
text **retrieval** conference trec **information** for active participants **information** ...
http://trec.nist.gov/act

#8 information retrieval
doc=#255826 score=0.766947
information retrieval los alamos dose ...
http://cedr.lbl.gov/DR/drla.html

#9 improving implant performance through retrieval information ohm

By its nature, this is a task that desires relatively small response sets for high precision. Only a few relevant pages were identified for most topics. Four runs, plus a combined run, were evaluated in two different scenarios.

Run 1 searched only the HTML title tag for query terms. Weighting for this and all other runs favored the exact query phrase, but was otherwise Lnu.Ltc using the VSM.
Run 2 looked for the exact query phrase within the same paragraph (where paragraph is defined as any block-level set of text, including tags such as p, table, and ul).
Run 3 ranked documents containing all query terms with offsets plus or minus 10 from one another. (The actual implementation was that each query term had to be within 10 terms from the first query term.)
Run 4 was a "bag of words" approach, in which any document with all query terms was considered for ranking.
The fifth set was the combination of all prior sets. These results are summarized in Tables 6 and 7.

It was speculated that the four runs could operate in a "fall through" manner: if run 1 yielded no results for a particular topic, run 2 would ensue. Similarly, run 3 would only occur for a topic if run 2 yielded no results. The bag of words approach in run 4 was, essentially, a move of desperation. Thus, the "Combined" column of Table 6 is the result of all 150 topics in which one or more of the runs were completed, only the last of which produced results. This is the fall through scenario.

The other scenario is to simply use each method alone on all 150 topics. At the outset, we presumed that Run 1 would provide the highest precision, while Run 4 would find additional relevant documents but at the expense of far lower recall.

Table 6: Web02 Named Page Task Results Summary for Fall Through Queries

Recall	Precision				
	Run 1	Run 2	Run 3	Run 4	Combined
		(If Run 1 fails)	(If Run 2 fails)	(If Run 3 fails)	(Complete set)
0	0.5392	0.2461	0.1447	0.4048	0.2839
0.1	0.5392	0.2461	0.1447	0.4048	0.2839
0.2	0.5392	0.2461	0.1447	0.4048	0.2839
0.3	0.5392	0.2461	0.1447	0.4048	0.2839
0.4	0.5392	0.2461	0.1447	0.4048	0.2839
0.5	0.5392	0.2461	0.1447	0.4048	0.2839
0.6	0.451	0.2333	0.1447	0.3095	0.2483
0.7	0.451	0.2333	0.1447	0.3095	0.2483
0.8	0.451	0.2333	0.1447	0.3095	0.2483
0.9	0.451	0.2333	0.1447	0.3095	0.2483
1	0.451	0.2333	0.1447	0.3095	0.2483
# of Queries	34	13	77	21	145
Retrieved	142	104	541	75	862
Relevant	39	17	82	27	165
Relevant retrieved	20	6	22	11	59
Exact precision	0.49	0.23	0.1	0.33	0.24

Table 6 shows that relatively high precision was achieved in Run 1 (title only), but at the expense of many queries for which no results were submitted. 34 topics resulted in 142 documents identified as potentially relevant, 20 of which actually were. Most topics only produced a few documents, with only 64 ("work/life center map"), 88 ("export import bank") and 137 ("endangered species picture book") in the double digits with 31, 31, and 14 documents each, but none was relevant. Those topics are in contrast to topics that hit exactly, with one to three documents found, all of which were relevant.

Of the 116 (150 – 34) topics submitted to Run 2 (exact phrase), only 13 resulted in documents being presented, and with lower scores overall than Run 1. As would be expected, exact phrase is not necessarily indicative of a named page – and in this case, the named pages with exact phrase were more likely to be title pages.

Another 77 of the remaining 103 topics resulted in "hits" for Run 3, the adjacency search. Numerically, this was the richest set, with 22 new relevant documents found out of 82 possible for those topics. But the 519 non-relevant documents resulted in low overall scores, and a very low exact precision. The low variety in precision scores across all runs are because most topics had very small response sets, due to the specificity of the IRTools query.

Only 26 topics remained for Run 4, and 21 of these resulted in some documents being found. Failure to produce any hits on the other 5 is mostly attributable to parsing issues (i.e., "e-coli" versus "ecoli", and the infamous "u.s." versus "us" versus "u.s"). The bottom line here is that a fall through approach seems reasonable: start with more specific topics, and then try less specific queries before giving up. Adjusting the adjacency search to a smaller window, or requiring term ordering as in the topic, could help.

For comparison, runs of all 150 topics were made with each of Run 1 through Run 4. Results for Run 1, of course, matched those for the fall through method. For Runs 2, 3, and 4, results dropped off rapidly due to increasing numbers of non-relevant documents being added to the response set. Table 7 summarizes these results.

Table 7: Web02 Named Page Task Results Summary for Independent Runs

	Run 2	Run 3	Run 4
# of Queries	150	150	150
Queries with results	33	123	145
Retrieved	403	1548	1593
Relevant	38	137	165
Relevant retrieved	17	50	47
Exact precision	0.21	0.1	0.1

We see in Table 7 that the high specificity of Run 2, with an exact phrase search, did not fare much worse than in the fall through case, where Run 2 only occurred if Run 1 failed. But for Runs 3 and 4, a disproportionately large number of candidates were submitted for a relatively small number of relevant documents. These boosted the Relevant Retrieved scores, but were not especially successful methods otherwise.

Future Directions

IRTools has many facets. It is hoped that other information scientists will consider utilizing parts of it for their own research, perhaps even contributing new components. For the immediate future, the main research topic of interest is how to allocate term weights in subdocuments. Must these be computed after the initial indexing, when term characteristics for the whole collection are known? What about incorporating collection-level statistics such as page rank? What adjustments to traditional tf, idf and pivot scores are needed?

From a development perspective, the biggest effort will go to a reference implementation for GridIR. TREC participants are urged to get involved with GridIR. Apart from being of inherent interest to many of us, GridIR is a ripe platform for experimentation on query results merging, information filtering, and different document types. More information, including standards documents, is online at www.gridir.org.

References

Foster, Ian. 2003. "The Grid: Computing without bounds." Scientific American April. Online: www.sciam.com

Gamiel, Kevin; Karimi, Sousan; Newby, Gregory; Nassar, Nassib. 2003. "Grid information retrieval requirements." Online: www.gridir.org/wg_docs.html

Gonnet, Gaston H.; Baeza-Yates, Ricardo; Snider, Tim. 1992. "New indices for text: PAT trees and PAT arrays." In: Information Retrieval: Data Structures and Algorithms. Upper Saddle River, New Jersey: Prentice-Hall.

Kahle, B.; Morris, H.; Davis, F.; Tiene, K.; Hart, C.; Palmer, R. 1992. "Wide Area Information Servers: An executive information system for unstructured files." Electronic Networking: Research, Applications and Policy 1(2): 59-68.

Knuth, Donald. 1998. The Art of Computer Programming Vol. 3: Sorting and Searching. New York: Addison-Wesley.

Nassar, Nassib; Newby, Gregory; Gamiel, Kevin; Dovey, Matthew; Morris, Jeremiah. 2003. "Grid information retrieval architecture." Online: www.gridir.org/wg_docs.html

Newby, Gregory B. 2002. "Progress in General-Purpose IR Software." In Voorhees, Ellen (Ed.). TREC 2002 Proceedings. Gaithersburg, Maryland: NIST.

Question Answering
By Pattern Matching, Web-Proofing, Semantic Form Proofing

Min Wu, Xiaoyu Zheng, Michelle Duan, Ting Liu and Tomek Strzalkowski
ILS Institute, Computer Science Department
SUNY Albany

Abstract

In this paper, we introduce the University at Albany's question answering system, ILQUA. It is developed on the following methods: pattern matching over annotated text, web-proofing and semantic form proofing. These methods are currently used in other QA systems, however, we revised them to work together in our QA system.

1. Introduction

This is our first time to participate in QA TREC with three runs and we focus on how to build a QA system in a short time (around half a year) with available methods and get a reasonable performance. Our system evaluation results show that it is possible.

2. Overview

2.1 Architecture

The system components are: question analysis, document retrieval and annotation, pattern matching, web-proofing and semantic form proofing. See Figure 1.

2.2 Question Analysis

There are two main tasks in question analysis: question categorization and query expansion.

We classify questions according to their answer target. The answer targets of TREC questions usually fall into several categories such as person, location, date, quantity, manner, works, organization and so on. These categories also have subcategories. For example, location contains the subcategories nation, city, mountain, lake, river, etc. So our question is categorized according to the main category and subcategory. It is easy to classify "who", "where" and "when" questions. For "what" and "how" questions, we need the help of dictionary tools like WordNet to identify what is asked. This type is usually identified by the noun or adjective following the question word, e.g., "What country…" or "How long …", etc.

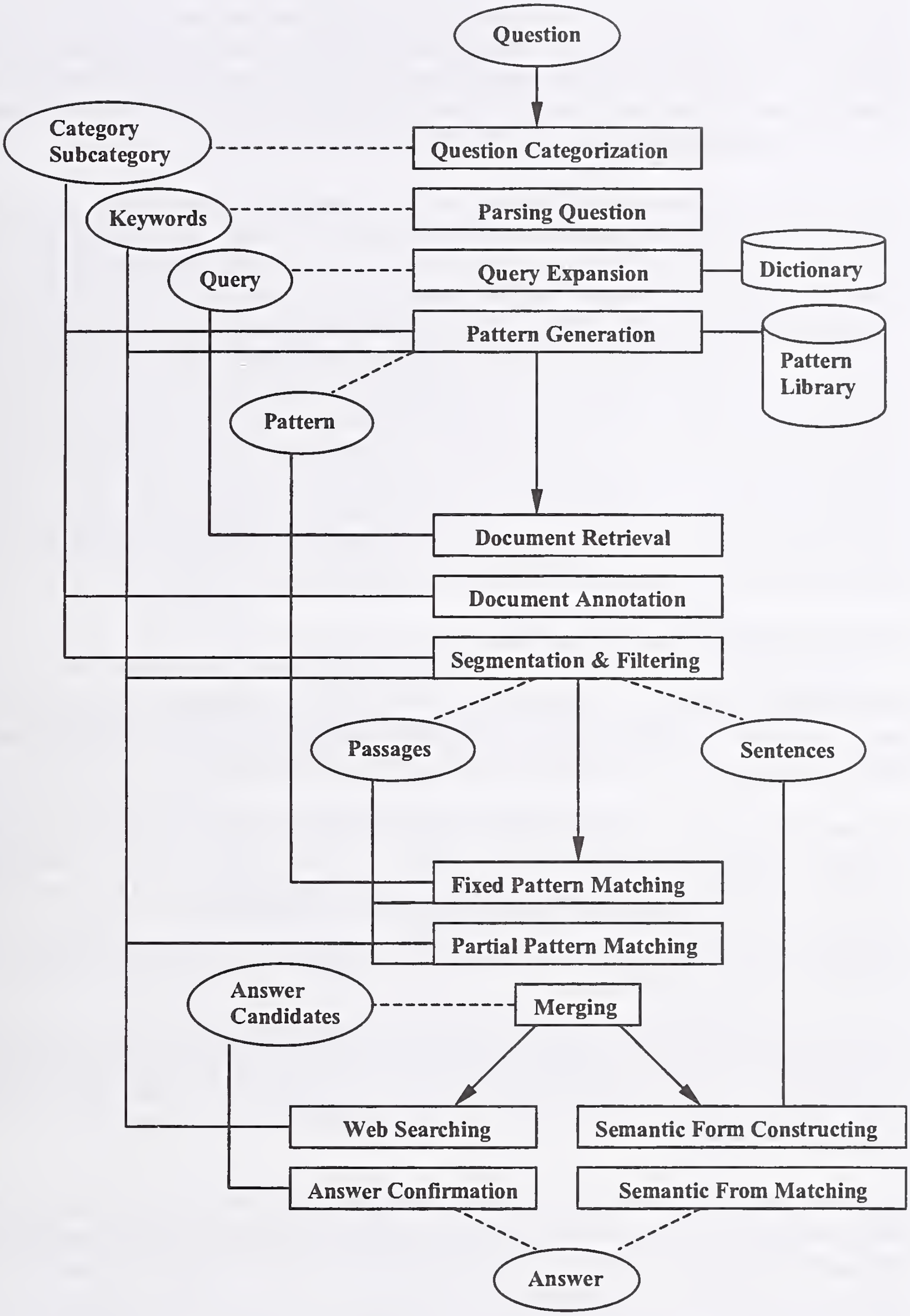


Figure 1. ILQUA System Architecture
(Note: Solid line shows input or tool, dashed line shows output, arrow line shows system processing flow)

Once a question falls into some category, the possible answer patterns can be retrieved from our pattern library. These patterns are general to the category and replacing keywords is necessary to get the specific patterns to the question. However, for some definition questions such as "What is golden parachute", it is difficult to find the answer target. We classify these questions as "No-Pattern" questions.

The first step in the answer finding process is information retrieval (IR), which fetches a number of "relevant" documents from the database. Query modification is usually necessary to increase the recall of IR. It involves both deleting and adding terms to the initial query, which is obtained from the user question through the usual stemming and stopping process. For some questions, we delete some common terms that are not helpful in retrieval. For example, in the question such as "What country is Aswan High Dam located in?", the term "country" is not of much use in retrieving relevant documents. So it is necessary to delete it from our query in order to increase the recall. However, for some nouns and verbs, it may be helpful to get their synonyms and related forms, for example, it is often necessary to find noun and adjective forms for verbs. These keywords and their expansions are kept for future use. WordNet and other dictionaries are used to finish these tasks. Finally, the query is constructed as a Boolean formula that can be processed by the IR component, in our case the UMass' Inquery system.

2.3 Document Retrieval & Annotation

The IR system we use to pre-fetch documents is the Inquery system developed at Univ. of Mass. at Amherst. ILQUA selects the top 50 documents from the file list returned by Inquery and passes those documents to the annotation component.

The annotation tool we use is BBN's Identifinder system. It annotates documents according to the answer target of the question. The annotated entity types include ANIMAL, DISEASE, FAC, GAME, EVENT, GPE, LANGUAGE, LAW, LOCATION, NATIONALITY, PERSON, PLANT, PRODUCT, SUBSTANCE, WORK_OF_ART, CARDINAL, MONEY, ORDINAL, PERCENT, QUANTITY, DATE and TIME. We use most of these types in our system.

The annotated documents are filtered according to the keywords. We cut documents into passages and keep those passages that contain the annotated answer target and keywords.

2.4 Pattern Matching

Our pattern matching component consists of two parts, fixed pattern matching and partial pattern matching. For questions with a simple answer pattern, the answer candidates can be found by fixed pattern matching. As for those with complex answer patterns, we try to locate answer candidates via partial pattern matching.

Fixed pattern matching scans each passage and does pattern matching. Patterns are organized in a list according to their scores. Once a pattern is matched, the answer is extracted from the text and put into the primary answer list along with the score of the pattern which has been used to extract it. After all of the passages are processed, the system merges what appear to be the same answer in the answer list and calculates a final score for each distinct answer. Figure 2 shows an example of how this fixed pattern matching works. It is very effective for questions such as "When was Adolf Hitler born?", "What does ACLU stand for?", "When was the first camera invented", etc.

Question: *When was MTV started?*
Major Target: Date
Keywords: NP MTV VP started
Query: query: #passage350(MTV,#syn(start,started,starts,starting))
Pattern: ("started" can be replaced by any of it's synonym)

```
started[ ]+?MTV[ ]+?in[ ]+?<Date>([^<>]+?)<\/Date>
MTV[ ]+?started[ ]+?in[ ]+?<Date>([^<>]+?)<\/Date>
Started[ ]+?MTV[^<>]*?<Date>([^<>]+?)<\/Date
MTV[^<>]*?started[^<>]*?<Date>([^<>]+?)<\/Date>
started[^<>]*?<Date>([^<>]+?)<\/Date>[^<>]*?MTV
<Date>([^<>]+?)<\/Date>[^<>]*?started[^<>]*?MTV
<Date>([^<>]+?)<\/Date>[^<>]*?MTV[^<>]*?started
```

Matched Passage:
MTV's parent company, Viacom, also owns Nickelodeon and VH1. Nickelodeon is the most popular of the outlets (with MTV second and VH1 third) and celebrates its 20th anniversary on April 1. *MTV started in 1981 and VH1 in 1985.* VH1's older-demographic audience consists of ``graduates of MTV who still love music," said Freston.
Matched Pattern:

```
MTV[ ]+?started[ ]+?in[ ]+?<Date>([^<>]+?)<\/Date>
```

Figure 2. Example of Fixed Pattern Matching

The idea of partial pattern matching is based on the assumption that the answer is usually surrounded by keywords and their synonyms. Let E_k ($1 \leq k \leq$ m) denote the kth named entity in the annotated passage, T_i denotes the ith query keyword ($1 \leq i \leq$ n), W_i ($0 \leq i \leq 1$) denotes the weight of T_i and D_i denotes the distance from T_i to E_k. If T_i or its synonyms occur in the passage, D_i is equal to the count of words between T_i and E_k, otherwise, D_i is equal to the maximum average length of the passages *Max*. The matching score of E_k is:

$$S_k = \frac{Max - \left(\sum_{i=1}^{n} D_i \times W_i\right) * \frac{1}{n}}{Max}$$

If the matching score, S_k, is above a threshold, the named entity E_k is extracted and added to the secondary answer list. Finally, the secondary answer list is merged and sorted. Figure3 and Figure4 show a simple example of how partial matching works.

ILQUA selects the top 5 answer candidates from the two answer lists and passes them to the web-proofing. It is important that pattern matching gets the correct answer included in the top 5 ranked answers because the final selection is based on relative likelihood among the 5 candidates.

For definition questions, it is difficult to decide their answer targets and answer patterns in advance. Instead, after the relevant passages are processed, sentences containing a possible answer are passed to the semantic proofing component.

Question: *Who created the literary character Phineas Fogg*

Major Target: Person

Question verbs: create created creates creating

Question verb nouns: creator

Question verb synonyms:

make made makes making produce produced produces producing

Question noun tokens: literary character Phineas Fogg

Answer:

Jules Verne's Phileas *Fogg* *made* *literary* history when he traveled ``around the world in 80 days" in 1873.

Score: 0.5017

Figure 3. Example of Partial Matching

Question: *What band did the music for the 1970's film "Saturday Night Fever"*

Major Target: Organization

Minor Target: Band

Question noun tokens: band music 1970 film Saturday Night Fever

Answer:

First to appear will be ``*Saturday* *Night* *Fever*," already boasting a $14 million advance and the benefit of a surefire title and score audiences know from the 1977 *film* that featured John Travolta and *music* by the ***Bee Gees***.

Score: 0.5612

Figure 4. Example of Partial Matching

2.5 Web-Proofing

Web-proofing is used to select an answer from the answer candidate list. As the most widely used search engine, Google is chosen to search the Web. The query is submitted to Google and the number of occurrences of each answer candidate is calculated. The assumption here is that the correct answer will have more occurrences in the list returned by Google than other answer candidates. This simple method works for factoid questions. For list questions, if any of the answer candidates occur in the top documents returned by Google, we will add it to the final answer list. This web-proofing method needs to be improved to deal properly with cases where the correct answer is not among the candidates being proofed or when a slight nuance in the question requires us to find a less popular answer.

2.6 Semantic Form Proofing

In our system, the semantic form proofing is only applied to definition questions. We try to find the answer from the sentence list returned (without a match) by the pattern matching step. The system then builds semantic representation for both the question and the selected sentences. The semantic representation is in the form of an acyclic directed graph where each link is assigned a weight. The system then attempts to match the semantic form of the question to the semantic forms of the candidate sentences. For each matching, it assigns a score. The answer is chosen from the highest scoring sentence.

To build up semantic forms, the head driven parser from M. Collins is adopted. The parser provides the head of phase information for each sentence and semantic form (or say semantic tree) is built up based on this phase head information. Figure 5 shows an example.

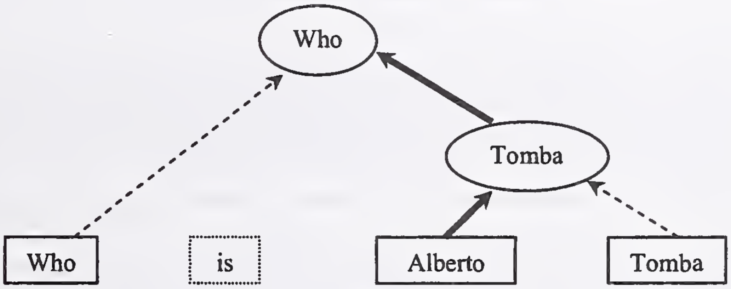


Figure 5. Semantic Form

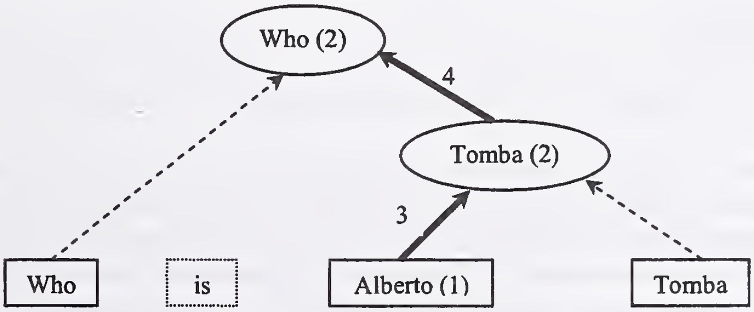


Figure 6. Weighted Semantic Form

The weighted semantic form is built upon semantic form. As we can see from the example that semantic form is an acyclic directed graph and each word is assigned a value to indicate its importance. Since the head of a phase is likely to be more important than other words in the phase, its value is its in-degree plus one (to avoid 0-value links). In Figure 6, the value of "Tomba" is 2. Also a weight is assigned to each link. The value of a link is the sum of the two nodes' weights in that link. In figure 6, the weight of link between "Who" and "Tomba" is 4.

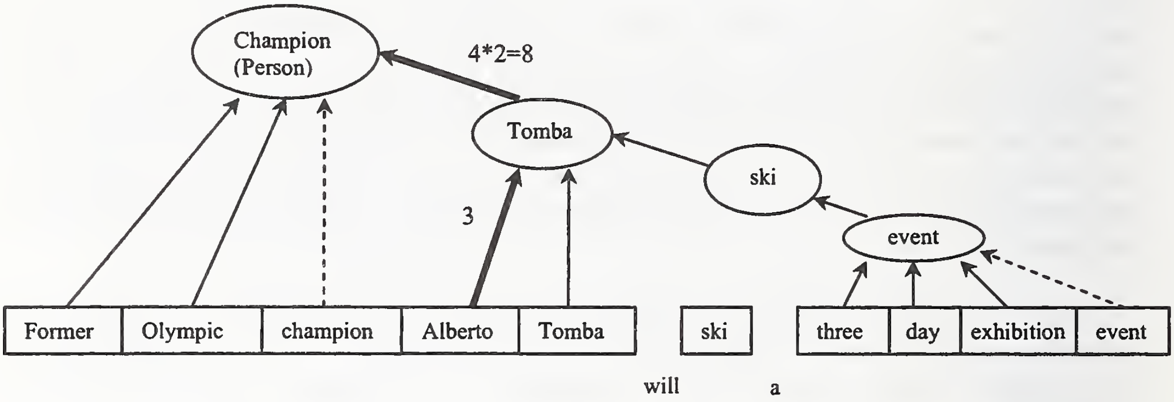


Figure 7. Answer Semantic Form

Similarly, the weighted semantic forms of sentences containing answer candidate are built. However, for some important links, their weight is doubled. In Figure 7, the weight of the link between "Champion" and "Tomba" is doubled. After the assignment of weights to each link, we compare links in sentences containing answer candidate with the links in the question. Once matched links are found, the sum of weights of matched links is the matching score of the answer candidate. In Figure 7, the matching score of word "champion" is 8+3=11. If the score is above some threshold, the answer candidate is extracted and put in the answer list. The answer candidate with the highest matching score will be chosen as the final answer.

3. Experiment

We submitted three runs for evaluation. The evaluation results are shown in Table 1.

We try to assign different weights to keywords in each run. In run AlbanyI2, all of the keywords receive the same weight. In runs AlbanyI3 and AlbanyI4, keywords containing digits and proper nouns receive smaller weights. The evaluation results show that the weighting method is more useful to list questions because the accuracy of list questions in Albany03I4 is better than Albany03I2. However, for factoid questions, the weighting method produces a reduction in performance.

	Albany03I2	Albany03I3	Albany03I4	Median of 54 Runs
Factoid (Accuracy)	0.24	0.206	0.228	0.177
List (Average F)	0.085	0.075	0.096	0.069
Definition (Average F)	0.146	0.133	0.134	0.192

Table 1. Evaluation Result

4. Discussion & Future Work

Since ILQUA has been built in a short period of time as a semester project for a small group of CS students, some of our ideas and implementations are not yet mature. The system needs to be improved in many ways, and we hope this will happen by the next year TREC meeting.

Actually, from the analysis of our experiment results, pattern matching can retrieve the correct answer in the top 5 answer candidates for nearly 45 percent of the factoid questions, and in the top 10 answer candidates for 50 percent of factoid questions. Some techniques need to be applied to increase the rank of correct answers while decreasing the rank of incorrect answers. For partial matching, different weighting methods should be applied to different types of questions.

A more robust and efficient web-proofing method is necessary in our future development, especially a component to deal adequately with junk web content. It is also very difficult for semantic form proofing to handle definition questions whose answer needs to be inferred from the context. Improving ILQUA in this aspect is also in our future plan.

5. Acknowledgements

Thanks to all the other students in our CSI660(Spring 2003) class. Especially to Zhenyu Dai who help us set up Inquery system.

6. Reference

John M. Prager, Jennifer Chu-Carroll, Eric W. Brown, Krzysztof Czuba. Advances in Open Domain Question Answering, Kwwer Academic Publisher, 2004. Question Answering by Predictive Annotation
Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Michael Junk, Chin-Yew Lin, 2000. TREC-9 Proceedings. Question Answering in Webclopedia. http://trec.nist.gov/pubs/trec9/t9_proceedings.html
Deepak Ravichandran, Eduard Hovy. ACL 2002. Learning Surface Text Patterns for a Question Answering System.
http://www.isi.edu/natural-language/projects/webclopedia/pubs/02ACL-patterns.pdf
Martin M. Soubbotin, Sergei M. Soubbotin. TREC-10 Proceedings. 2002. Patterns of Potential Answer Expressions as Clues to the Right Answers.http://trec.nist.gov/pubs/trec10/t10_proceedings.html
Sanda Harabagiu, Dan Moldovan. 2000. TREC-9 Proceedings. FALCON: Boosting Knowledge for Answer Engines. http://trec.nist.gov/pubs/trec9/t9_proceedings.html

The University of Amsterdam at the TREC 2003 Question Answering Track

Valentin Jijkoun Gilad Mishne Christof Monz*[*]
Maarten de Rijke Stefan Schlobach Oren Tsur[†]
Language & Inference Technology Group
University of Amsterdam
`http://lit.science.uva.nl/`

Abstract: We describe our participation in the TREC 2003 Question Answering track. We explain the ideas underlying our approaches to the task, report on our results, provide an error analysis, and give a summary of our findings so far.

1 Introduction

The aim for our participation in the Question Answering track at TREC 2003 was to experiment with a new multi-stream architecture, in which we implemented 6 separate subsystems that each try to answer questions in different ways. We also wanted to experiment with a dedicated biography question module that is currently in development.

Our experiments exploited the home-grown FlexIR document retrieval system [9]. The main goal underlying FlexIR's design is to facilitate flexible experimentation with a wide variety of retrieval components and techniques; we used FlexIR's implementations of the `Lnu.ltc` weighting scheme, various language models, as well as the Okapi scheme.

Current Question Answering (QA) systems, as reflected by the TREC QA track participants, can be divided into two categories: *knowledge-intensive* systems, that make use of various linguistic tools for the question answering process, and *redundancy-based* systems, that rely on very high volumes of data (in many cases, the Web) and take a more shallow approach to text analysis. Until last year, we were focused on the first approach, concentrating our QA efforts exclusively on *Tequesta* [10, 11]. This approach may be successful for some types of questions, but for others more shallow approaches seem more beneficial. This year we expanded our QA work and implemented a *multi-stream* approach. While maintaining Tequesta as one of the approaches, we developed additional systems that compete which each other to find the correct answer. These systems, or "streams," employ a range of redundancy-based and knowledge-intensive techniques. We took part in the main QA task and in the passage QA task. For our participation in the main task we employed our new multi-stream architecture; for the passage task we relied on the Tequesta stream only.

The rest of this paper is organized as follows. In two (largely self-contained) sections we describe our work for the main task and the passage task. Finally, we summarize our findings in a concluding section.

2 The Main Task

2.1 System Description

We now describe the approach we adopted for the main QA task; we devote separate subsections to factoid questions on the one hand, and list questions and definition questions on the other. The system consists of 6 separate QA streams and a final answer selection module that combines the results of all streams and produces the final answers. An important benefit of this architecture is easy modification, maintenance, and testing of the dif-

[*] Now at the Institute for Advanced Computer Studies, University of Maryland, 3161 A.V. Williams Building, College Park, MD 20742, USA. Email: `christof@umiacs.umd.edu`.

[†] Now at Bar-Ilan University, Ramat Gan, Israel. Email: `tsuror@cs.biu.ac.il`.

ferent subsystems as well as easy integration of multiple sources of information. Evaluation of the contribution of each stream to the entire QA process becomes a relatively simple task too. We now describe the streams.

Table Lookup. This stream uses specialized knowledge bases constructed by preprocessing the collection, similar in spirit to [4]. The stream exploits the fact that certain types of information (such as country capitals, abbreviations, and names of political leaders) tend to occur in a small number of fixed patterns. When a question type indicates that the question might potentially have an answer in these tables, a lookup is performed in the appropriate table and answers found are assigned high confidence.

We hand-crafted a small number of regular expressions for extracting information about the categories listed in Table 1. For instance, the "Location" category concerns geographic information of the following type "Amu Darya, river, Turkmenistan, XIE19990811.0277," where the first field indicates a location, the second its type, the third a country or region in which it is located, and the fourth the identifier for the document from which it was extracted. "Geography" contains similar information, but without the type; "Leaders" has information of the following kind "Dutch, Foreign Minister, Jozias van Aartsen, XIE19991027.0270", and "Roles" generalizes this to also include other roles besides government-related ones.

Table 1: Facts extracted from the AQUAINT corpus.

Category	# Facts	Category	# Facts
Abbreviations	31737	Birthdates	9156
Capitals	1273	Currencies	231
Dates	9331	Deathdates	1510
Geography	70363	Height	15603
Inhabitants	2025	Languages	853
Leaders	18073	Locations	1348
Manners of death	857	Organizations	98758
Roles	396558		

When a question is classified as possibly having an answer in a table, we first identify the question keywords that will be used in the table search. Next, a line matching all of the words in the order they appeared in the question is searched; if no line matches, we look again for a line containing all words, this time in any word order. If there is still no match, we start removing words from the list of words to match; the order of removal is based on the frequency of words in the language (i.e., common words are removed first) and part-of-speech tags (e.g., superlatives like *fastest*, *largest* are removed last). We do this until some threshold is reached (percentage of lookup words out of total keywords in the question). When a matching line is found, we return the text in the column that is declared to contain the information required as the answer.

Pattern Matching. This stream exploits the fact that in some cases, the contextual format of an answer to a question can be back-generated from the question itself. For example, an answer to a question such as *2257. What is the richest country in the world?* will possibly match the pattern `<Capitalized-Words>(,| is) the richest country in the world`. In these cases, the position of the answer within the context is also known when generating the context pattern; in the given example, it would be the capitalized word or words (and indeed, in document XIE19980302.0146, this pattern matches against "... Although the United States is the richest country in the world, 20 percent of its population ...").

The Pattern Matching stream consists of three stages: *Generation*, *Document Prefetch* and *Matching*. In the Generation stage, the question is analyzed and possible answer patterns are generated. For questions like *2347. Where is Mount Olympus?* the question type and focus (both provided by the question classifier) are sufficient for generating a number of answer patterns. For other questions (e.g., *2375. What date did Thomas Jefferson die?*) we also use a set of manually created rules based on part-of-speech tags of the question words and a dictionary of word forms, in order to rewrite the question into declarative forms (e.g., `Thomas Jefferson (died|dies) (on|in) <answer>`). In the Prefetch stage, for each generated pattern a query containing words from it is formed, and documents are retrieved from the collection using the query. In the final stage, the patterns are matched against the retrieved documents, and answers are extracted from the matches.

Two variations of this stream were implemented, *Web Pattern Matching* and *Collection Pattern Matching*. For the first variation the text collection was the Web, and for the second, the local AQUAINT corpus. For the prefetch stage we used the top-ranking documents from Google (for the Web variation) and all matching documents retrieved using a boolean query to our document retrieval engine FlexIR against the AQUAINT corpus.

Ngram Mining. This stream, similar in spirit to [2, 3], constructs a weighted list of queries for each question using a shallow reformulation process, similar to the Pattern Match stream. The queries are then sent to a large document collection; we implemented two variations for this stream, *Web Ngram Mining* and *Collection Ngram Mining*, using the Web and the local AQUAINT corpus, respectively. For Web searches, we used Google, and for local searches FlexIR, with the `Lnu.ltc` weighting scheme. Then, we looked at word ngrams in the relevant retrieved document paragraphs (for the Web we used the snippets provided by Google, and for the collection we used a window of 200 bytes around the query). The ngrams were ranked according to the weight of the query that generated them, their frequency in the paragraphs, their NE type, the proximity to the query keywords and more parameters, and the top-ranking ngrams were considered answer candidates. To find justification for the answer in the local corpus, we constructed a query with keywords from the question and the answer, and considered the top-ranking document for this query to be the justification, this time using an Okapi model as this tends to do well on early high precision in our experience.

Tequesta. As mentioned before, this is a stream that implements a linguistically informed approach to QA. We defer a discussion of this stream to Section 3 where we describe our strategy for the passage task.

Many components are shared by all streams, including a locally developed named entity tagger and the following:

Question Classifier. An incoming question is first analyzed for its type (e.g., `date-of-birth`), expected answer type (e.g., `location`) and focus (the *core* of the question, used e.g., for answer pattern generation). Currently our system recognizes 37 question types. The question analysis is based on surface and part-of-speech patterns. We also use hierarchical relations in WordNet to identify semantic classes of question focus words (e.g., this allows us to assign the type `person-ident` to the question *1943. What is the name of Ling Ling's mate?*).

Web Ranking. The answer candidates produced by the streams have different confidence levels, generated by stream-specific parameters and measuring methods. To compare these levels, a uniform way of ranking the candidates was required. To this end we implemented a search engine hit count module, similar to [6].

Answer Selection. Each of our streams produces a pool of answer candidates, with normalized confidence scores. After filtering the candidates to remove obvious noise, we create a joint pool of answers, adjusting each candidate's score by a factor that reflects the past performance of its stream on questions of the same type. We tried different ways of assigning these stream/question-type weights: manually (i.e., based on human intuition about how good different streams perform on different questions) and automatic (using Machine Learning to find weights that optimize the performance of the system on a training set of questions) [5]. In the joint pool of answer candidates we identify identical or similar (small edit distance) answers, merge and add their confidence scores. Finally, a candidate with the highest score is returned.

List and Definition Questions

Because of time constraints, we were unable to implement a proper module for handling list questions. All list questions were automatically rewritten into factoids using rule-based transformations (e.g., *2097. Which countries were visited by first lady Hillary Clinton?* was transformed to *Which country was visited by first lady Hillary Clinton?*) and fed to our multi-stream QA system. The top *N* candidate answers to this factoid question were submitted as answers to the original list question. We experimented with different values of *N* (10 and 20 in our official runs) and with different numbers of retrieved documents used during answer selection (both for collection- and web-based QA streams).

In contrast to list questions, we did invest a serious effort in developing a component for handling definition questions. More precisely, we piggybacked on ongoing inhouse activities aimed at developing a QA system for handling "biography oriented" definitions on the web [13]. The main steps in our handling of definition questions are Question Analysis (very similar to the analysis carried out for factoids), Answer Retrieval (always from external resources), Answer Filtering, and Answer Justification (very similar to the justification performed for externally found answers to factoid questions).

For concept definition questions we followed a

WordNet-based strategy as discussed in the literature [12]. Given a question that asks for a definition of a concept, we simply consult WordNet. As our primary strategy for handling person definition questions, we also consulted an external resource. The main resource used is `biography.com`. However, in many cases no biography could be found in this resource. In such cases we backed off to using Google, with queries obtained by combining the name of the person in question with varying subsets of a predefined set of hand-crafted features (including "born", "graduated", "suffered", etc.) For questions asking for definitions of organizations the latter was the strategy used (with a set of "organization features").

As a final fallback option for each type of definition question, if the use of the strategies mentioned earlier returned no satisfactory results, we simply submitted `<question term> is a` to Google and mined the snippets returned. This method worked surprisingly well for questions like *2385. What is the Kama Sutra?*.

Given a set of candidate answer snippets, we performed two more steps before carrying out the final answer justification step: we separated junk snippets from valuable snippets and we identified snippets whose content is very similar. We addressed the first step by analyzing the distances between query terms submitted to the search engine and the sets of features, and by means of shallow syntactic aspects of the different features such as sentence boundaries. To address the second step we developed a snippet similarity metric based on edit distance, stemming, stopword removal, and keyword overlap.

2.2 Runs

We submitted 3 runs. These runs used the exact same strategies and settings for definition questions. They did differ in their settings for factoids and list questions. Here's a brief description:

UAmsT03M1 For factoids, the answer selection module used automatically learned stream/question weights; answers coming only from external sources (streams based on Web) were justified against the AQUAINT collection using the Okapi model. For each list question the top 10 answers to its factoid counterpart were submitted.

UAmsT03M2 For factoids, the weights for answer selection were learned automatically; external answers were discarded. For list questions the number of collection and web documents used for answer mining was increased, and the top 20 answers were submitted for each question.

UAmsT03M3 Manually assigned weights were used for answer selection; external answers were discarded. The number of documents for answering list questions was as in UAmsT03M2, but only top 10 answers were submitted.

Our three runs allowed us to compare the impact of justification, and the impact of using manually assigned versus learned weights for our answer selection. For the list questions we wanted to evaluate the effect of *using more data* and of *giving more answers* on the final performance.

2.3 Results

Table 2 gives the detailed results of our system for the 413 factoid questions: accuracy and the number of correct (R), unsupported (U), inexact (X) and wrong (W) answers.

Table 2: Results for the QA track (factoid questions).

Run identifier	Accuracy	R	U	X	W
UAmsT03M1	0.136	56	22	32	303
UAmsT03M2	0.145	60	20	26	307
UAmsT03M3	0.128	53	24	30	306

2.4 Error Analysis

Analyzing errors made by a QA system is a complex task. In [7], such an analysis is based on examination of the outputs of every module in the process separately, and attributing the error to the first malfunctioning module. In many cases an error in one of the earlier stages of the QA pipeline (for example, the question classification module) does indeed cause cascaded errors later. But when diagnosing a system with multiple independent approaches such as ours, this does not necessarily hold; we found incorrectly classified questions which were still answered correctly due to the redundancy-based modules, and many other counter-examples to the "cascaded errors" assumption. Therefore, we chose to examine the incorrect answers produced by the system, and associate each of them with a *main error type*, the dominant reason for producing this incorrect answer.

Table 3 shows the most common error types for run `UAmsT03M1`; for each incorrectly answered question (counting inexact or unsupported answers as incorrect), we examined the candidate answer list produced by our system (the list contains less than 10 candidates on average). If the correct answer was in this list, we classified the error types of the candidates that received a higher rank than it; otherwise, we classified the top 3 ranking candidates. Since we examined more than one answer per question, there may be multiple error types for a specific question.

Table 3: Frequent Error Types in UAmsT03M1.

Error Type	Frequency
Answer Selection	134 (38%)
Named-Entity	78 (22%)
Question Classification	67 (19%)
Justification	58 (16%)
Unit Error	53 (15%)

A brief explanation of main error types follows:

- *Answer Selection* errors describe an incorrect answer with the correct named entity or concept type which typically appears in relevant documents. An example is the answer `George Bush` for *2391. What president created social security?* – the answer type is correct, and is very frequent in relevant documents (both in the local collection and on the web).
- *Named-Entity* errors result from an incorrect classification of a phrase as a named entity which matches the expected answer type. An example is the answer `Springsteen` for *2001. What rock band sang "A Whole Lotta Love"?*.
- *Question Classification* errors are cascaded errors originating from an incorrect question type assigned to the question at an early stage of the QA pipeline, or failure to assign any question type to it.
- *Justification* errors are correct answers which were obtained using external resources, and were not projected correctly to the local corpus.
- *Unit Errors* are answers of the correct named entity type, but incorrect granularity (i.e. state instead of city) or out of range (according to world-knowledge). An example is the answer `about 2 billion dollar` for *2302. How much did the first Barbie cost?* (referring to profits rather than costs).

While our analysis revealed many technical issues that need to be addressed – such as over-tiling of ngrams, resulting in inexact answers (`Colombia country South America` instead of `Colombia`) – most of the errors stem from the shallow answer-selection techniques used by our system. We currently use mostly frequency counts and proximity measures to select the answer candidates; this works for questions which have a large amount of relevant documents, but for other questions deeper analysis is required. An alternative approach, still relying on redundancy methods, is to expand the number of retrieved documents using query expansion methods (both for the local corpus and the web) – an approach which is also almost not used in our system. Our main conclusion is that while we continue to see redundancy-based methods as our basic strategy for QA, shallow NLP and reasoning methods should be selectively used throughout the process, especially when the number of retrieved documents is low.

A few more remarks are worth making. First, although the run with the automatically learned weights for answer selection from multiple streams (`UAmsT03M2`) outperformed the run with manually assigned weights (`UAmsT03M3`), our subsequent experiments revealed that whereas a small difference exists, it is not statistically significant. However, both runs improve significantly over a baseline system with equal weights to all streams.

We also evaluated the contribution of different streams to the performance of the system on the factoids (using unofficial answer patterns). Table 4 gives the results (the number of "correct" answers, i.e., those that match the patterns) for the whole system, for separate streams and for the system with one of the streams turned off. As expected, each of the six streams answered some questions correctly and more interestingly, each stream contributed to the overall performance of the system. The two "worst" performing streams (predictably, collection-based pattern matching and ngram mining) brought one more answer each either at the top rank or in the top 5. Surprisingly, the "winner" among the streams is equivocal: while *Table Lookup* allows the system to answer 15 questions more, *Web Ngrams* accounts for more (35 vs. 19) unique correct

Table 4: Contribution of different streams.

Configuration	# correct	# correct in top 5
All streams	98	165
Collection ngrams	39	42
Without collection ngrams	98	164
Web ngrams	65	115
Without Web ngrams	89	130
Collection patterns	39	39
Without collection patterns	97	165
Web patterns	51	59
Without Web patterns	94	163
Table lookup	71	77
Without table lookup	83	146
Tequesta	63	102
Without Tequesta	91	140

answer candidates in the top 5.

Table 5 gives the combined results for the 3 QA tasks (accuracy for factoids, F score for list and definition questions) and the final scores of our runs. The results for the list questions suggest that using more retrieved documents for answer extraction and submitting more answer candidates hurts performance: the increase in recall does not compensate for the drop in precision.

Table 5: Results for the QA track.

Run identifier	A (Fact)	F (List)	F (Def)	Overall
UAmsT03M1	0.136	0.054	0.315	0.160
UAmsT03M2	0.145	0.042	0.308	0.160
UAmsT03M3	0.128	0.035	0.292	0.146

Turning to definition questions now, recall that there is *no* difference between the three runs listed in Table 5 as far as definition questions are concerned, despite the different scores in the table. The differences are due to inconsistencies in the judgments provided by NIST. Table 6 provides a breakdown of the scores for the different types of definition questions; the highest scores are obtained for person definitions, which reflects the fact that those are the type of definition questions in which we put most work. As an aside, in our submission we found *no* answer for 19 of the 50 definition questions. If we compute the F score not over all 50 question but only over questions with a positive F score, we obtain an average of 0.527. In post-submission experiments we changed the subsets of features we use in the queries sent to Google as well as the number of queries/subsets we use. The snippets-similarity threshold was also tuned in order to filter more snippets. This resulted in a reduction of unanswered definition questions to 6 instead of 19. Using our own (unofficial) assessment, this yielded an F score of 0.688. Those changes also reflected in the average answer length. After the parameters tuning the average length was half of the average TREC submission answer, improving precision and contributing to the F score.

Table 6: Breakdown of F scores for definition questions.

Run identifier	Concept	Person	Org.	Overall
UAmsT03M1	0.150	0.392	0.268	0.315

2.5 Conclusions for the Main Task

Our general conclusion on answering factoid questions is that our new multi-stream approach helped answer considerably more questions than our "old" single-stream Tequesta system. This year's questions seem *much* harder than those of previous years. A preliminary error analysis shows that retrieval, named entity recognition, and answer selection all require further attention. Our main conclusion on answering definition questions is that external dictionary-like resources are crucial, but a feature-based approach offers an effective strategy if such resources are absent or too sparse. Following an analysis of our TREC results, we investigated the use of trainable text classifiers as a pre-processing stage instead of the features vectors, treating the web as a 'noisy' external knowledge source, and using the text classifier to filter out the noise. Initial results show that using text classifiers greatly improves the F score and the coherence of answers.

3 The Passage Task

The aim of the passage task was to return an excerpt from a document rather than an exact answer. Excerpts had to be unmodified snippets from a document in the AQUAINT collection, and were not allowed to be longer than 250 characters. For the passage task only the factoid questions from the main task were used, i.e., list and definition questions were not included.

3.1 System Description

For the passage task, we used a modification of the Tequesta question answering system, which has remained largely unchanged since TREC 2002 [10, 11]. We dropped the exact answer output feature, and included some of the context surrounding the answer identified by Tequesta. We added the use of minimal span weighting for identifying documents that are likely to contain an answer to a given question. We used minimal matching spans as the snippets in which to find the exact answer.

Minimal span weighting takes the positions of matching terms into account, but does so in a more flexible way than passage-based retrieval; see [8] for details. Intuitively, a minimal matching span is the smallest text excerpt from a document that contains all terms which occur in the query and the document. More formally, given a query q and a document d, the function $\text{term\_at\_pos}_d(p)$ returns the term occurring at position p in d. A *matching span* (ms) is a set of positions that contains at least one position of each matching term, i.e. $\bigcup_{p \in \text{ms}} \text{term\_at\_pos}_d(p) = q \cap d$.

Then, given a matching span ms, let b_d (the beginning of the excerpt) be the minimal value in ms, i.e., $b_d = \min(\text{ms})$, and e_d (the end of the excerpt) be the maximal value in ms, i.e., $e_d = \max(\text{ms})$. A matching span ms is a *minimal matching span* (mms) if there is no other matching span ms$'$ with $b'_d = \min(\text{ms}')$, $e'_d = \max(\text{ms}')$, such that $b_d \neq b'_d$ or $e_d \neq e'_d$, and $b_d \leq b'_d \leq e'_d \leq e_d$.

Minimal span weighting depends on three factors.

1. *document similarity*: The document similarity is computed using the Lnu.ltc weighting scheme Buckley et al. [1] for the whole document. Similarity scores are normalized with respect to the maximal similarity score for a query.
2. *span size ratio*: The span size ratio is the number of unique matching terms in the span over the total number of tokens in the span.
3. *matching term ratio*: The matching term ratio is the number of unique matching terms over the number of unique terms in the query, after stop word removal.

The msw score is the sum of two weighted components: the normalized original retrieval status value (RSV), which measures *global similarity* and the spanning factor which measures *local similarity*. Given a query q, the original retrieval status values are normalized with respect to the highest retrieval status value for that query:

$$\text{RSV}_n(q,d) = \frac{\text{RSV}(q,d)}{\max_d \text{RSV}(q,d)}.$$

The spanning factor is the product of two components: the span size ratio, which is weighted by α, and the matching term ratio, which is weighted by β. Global and local similarity are weighted by λ. The optimal values of the three parameters λ, α, and β were found to be $\lambda = 0.4$, $\alpha = 1/8$, and $\beta = 1$ by empirical means. Parameter estimation was done using the TREC-9 data collection only, but it proved to be the best parameter setting for all collections.

The final retrieval status value (RSV') based on minimal span weighting is defined as follows, where $|\cdot|$ is the number of elements in a set: If $|q \cap d| > 1$ (that is, if the document and the query have more than one term in common), then

$$\text{RSV}'(q,d) = \lambda \cdot \text{RSV}_n(q,d) + (1-\lambda) \cdot \left(\frac{|q \cap d|}{1 + \max(mms) - \min(mms)} \right)^{\alpha} \cdot \left(\frac{|q \cap d|}{|q|} \right)^{\beta}$$

If $|q \cap d| = 1$ then $\text{RSV}'(q,d) = \text{RSV}_n(q,d)$.

Given a minimal matching span, the document analysis component of Tequesta tries to identify a phrase which is of the appropriate type. All phrases that are of the appropriate type are considered candidate answers. Tequesta selects answers by considering the frequency of a candidate answer and relying on linking a candidate answer to the question by proximity. Hence, all candidate answers are weighted equally. But there is one exception. If the question is of type `what-np`, candidate answers that are in a WordNet hypernym relationship with the question focus receive a higher weight than candidate answers that are identified by means of the fallback strategy.

Once an answer has been selected, the corresponding minimal matching span from which the answer has been extracted is returned as the answer passage, trimmed down to 250 characters if necessary.

3.2 Runs, Results and Conclusion for the Passage Task

We submitted one run to the passage task, run id `UAmsT03P1`. The results are shown in Table 7. (R) stands for passages that contained a correct and exact answer, (U) for passages that contained the correct answer, but were not supported by the corresponding document, and (W) stands for wrong answers. The passage track does not make a distinction between exact and inexact (X) answers, as in the main task. Here, an inexact answer is simply judged wrong (W).

Table 7: Results for the QA passage track

Run identifier	Accuracy	R	U	W
UAmsT03P1	0.111	46	6	361

The results were quite disappointing. At this point we are not sure what caused this rather bad performance. Before submitting this year's run to the passage track, we conducted some experiments on the question sets from previous TRECs, and these results were substantially better. Therefore, one explanation could be that this year's question set was much harder than the previous ones, but a more detailed error analysis remains to be done.

4 Conclusions

We have described our participation in the TREC 2003 Question Answering Track. This year, our work was largely motivated by our move to a new, multi-stream architecture. Although a further and more detailed analysis of the performance of the system remains to be done, our preliminary results show that different approaches to the QA process do produce answers to different question types. Our combined use of external resources and hand-crafted feature sets proved to be a successful approach for answering definition questions.

Acknowledgments

Thank you to Börkur Sigurbjörnsson for useful suggestions and discussion. We thank Raffaella Bernardi for help on the question classifier and table extraction components, and Karin Müller and Detlef Prescher for help with our named entity recognizer. Many thanks to Henry Chinaski for sanity checks and inspiration.

Valentin Jijkoun, Gilad Mishne, and Stefan Schlobach were supported by the Netherlands Organization for Scientific Research (NWO) under project number 220-80-001. Christof Monz was supported by NWO under project numbers 612-13-001 and 220-80-001. Maarten de Rijke was supported by NWO under project numbers 612-13-001, 365-20-005, 612.069.006, 612.000.106, 220-80-001, 612.000.207, and 612.066.302.

References

[1] C. Buckley, A. Singhal, and M. Mitra. New retrieval approaches using SMART: TREC 4. In *The Fourth Text REtrieval Conference (TREC-4)*, 1996.

[2] C. Clarke, G. Cormack, and T. Lynam. Exploiting redundancy in question answering. In D. H. Kraft, W. B. Croft, D. J. Harper, and J. Zobel, editors, *Proceedings of SIGIR 2001*, pages 358–365, 2001.

[3] S. Dumais, M. Banko, E. Brill, J. Lin, and A. Ng. Web question answering: Is more always better? In P. Bennett, S. Dumais, and E. Horvitz, editors, *Proceedings of SIGIR 2002*, pages 291–298, 2002.

[4] M. Fleischman, E. H. Hovy, and A. Echihabi. Offline strategies for online question answering: Answering questions before they are asked. In *Proceedings of ACL 2003*.

[5] V. Jijkoun and M. de Rijke. Answer selection in a multi-stream open domain question answering system. In *Proceedings of ECIR'04*, 2004.

[6] B. Magnini, M. Negri, R. Prevete, and H. Tanev. Is it the right answer? exploiting web redundancy for answer validation. In *Proceedings of ACL 2002*, pages 425–432, 2002.

[7] D. Moldovan, M. Pasca, S. Harabagiu, and M. Surdeanu. Performance issues and error analysis in an open-domain question answering system. *ACM Trans. Inf. Syst.*, 21(2): 133–154, 2003.

[8] C. Monz. *From Document Retrieval to Question Answering*. PhD thesis, University of Amsterdam, 2003.

[9] C. Monz and M. de Rijke. Shallow morphological analysis in monolingual information retrieval for Dutch, German and Italian. In C. Peters, M. Braschler, J. Gonzalo, and M. Kluck, editors, *Proceedings CLEF 2001*, 2002.

[10] C. Monz and M. de Rijke. Tequesta: The University of Amsterdam's textual question answering system. In Voorhees and Harman [14], pages 519–528.

[11] C. Monz, J. Kamps, and M. de Rijke. The University of Amsterdam at TREC 2002. In E. M. Voorhees and L. P. Buckland, editors, *The Eleventh Text REtrieval Conference (TREC 2002)*, pages 603–614, 2003.

[12] J. Prager, J. Chu-Carroll, and K. Czuba. Use of Wordnet hypernyms for answering what-is questions. In Voorhees and Harman [14], pages 250–257.

[13] O. Tsur. Definitional question answering using trainable classifiers. M.Sc thesis, University of Amsterdam, 2003.

[14] E. M. Voorhees and D. K. Harman, editors. *The Tenth Text REtrieval Conference (TREC 2001)*, 2002.

Approaches to Robust and Web Retrieval

Jaap Kamps Christof Monz* Maarten de Rijke Börkur Sigurbjörnsson
Language & Inference Technology Group
University of Amsterdam
http://lit.science.uva.nl/

Abstract: We describe our participation in the TREC 2003 Robust and Web tracks. For the Robust track, we experimented with the impact of stemming and feedback on the worst scoring topics. Our main finding is the effectiveness of stemming on poorly performing topics, which sheds new light on the role of morphological normalization in information retrieval. For both the home/named page finding and topic distillation tasks of the Web track, we experimented with different document representations and retrieval models. Our main finding is effectiveness of the anchor text index for both tasks, suggesting that compact document representations are a fruitful strategy for scaling-up retrieval systems.

1 Introduction

This year, our aim for the Web track was to experiment with different document representations and retrieval models for the home/named page finding and topic distillation tasks. The Robust track was new in 2003; our aim here was to investigate the impact of blind feedback and stemming on poorly performing topics.

For both tracks, our experiments exploited the home-grown FlexIR document retrieval system [9]. The main goal underlying FlexIR's design is to facilitate flexible experimentation with a wide variety of retrieval components and techniques. FlexIR is implemented in Perl, and supports many types of pre-processing, scoring, indexing, and term-weighting methods.

The rest of this paper is organized as follows. In two (largely self-contained) sections we describe our work for the Robust and Web tracks. Finally, we summarize our findings in a concluding section.

2 Robust Track

After describing the experimental setup for this track, we discuss our runs investigating the impact of blind feedback and stemming on the poorly performing topics.

System Description

All Robust track runs use the FlexIR information retrieval system. We employ a number of techniques:

Tokenization We remove punctuation marks, apply case-folding, and map marked characters into the unmarked tokens. We either index the words themselves, or the stems of the words. We use the Snowball stemming algorithm [13]. Snowball is a small string processing language designed for creating stemming algorithms for use in information retrieval

Retrieval model We use a multinominal language model with Jelinek-Mercer smoothing [4]. For all robust track runs, we use a uniform query term importance weight of 0.15.

Blind feedback Term weights are recomputed by using the standard Rocchio method [12], where we consider the top 10 documents to be relevant and doc-

*Present address: Institute for Advanced Computer Studies, University of Maryland, 3161 A.V. Williams Building, College Park, MD 20742, USA. Email: christof@umiacs.umd.edu.

uments ranked 501–1000 to be non-relevant. We allow at most 20 terms to be added to the original query.

Runs

We conduct two sets of experiments using (1) only the description field of the topics (D-topics), or (2) both the title and description fields (TD-topics). Using the resulting queries, we constructed the following four runs:

Words Language model run on a word-based index. This runs serves as the baseline for our stemming and feedback experiments.

Words+feedback Language model run on a word-based index, using Rocchio blind feedback.

Stems Language model run on the Snowball stemmed index.

Stems+feedback Language model run on the Snowball stemmed index, using Rocchio blind feedback.

Results

Table 1 gives the results of the runs over all 100 robust topics (best scores in boldface). The second column shows the mean average precision, the third the precision at 10 documents, the fourth the percentage of topics with no relevant document in the top 10; the fifth shows the area underneath the MAP(X) versus X curve for the worst 25 topics.

Table 1: Results for the Robust track (D top and TD bottom).

Run identifier	MAP	Prec.10	NoTop10	MAP(X)
Words	0.2065	0.3530	15.0%	0.0076
Words+feedback	0.1970	0.3420	17.0%	0.0059
Stems	**0.2319**	**0.3960**	**14.0%**	**0.0126**
Stems+feedback	0.2068	0.3570	16.0%	0.0098
Words	0.2324	0.4050	9.0%	0.0216
Words+feedback	**0.2452**	0.4110	13.0%	0.0210
Stems	0.2450	**0.4150**	**6.0%**	0.0256
Stems+feedback	0.2373	0.4040	14.0%	**0.0273**

The results of blind feedback are mixed at best. On the one hand feedback helps the overall score for the runs using TD-topics, with a best precision at 10 and a best score for mean average precision. On the other hand feedback hurts the performance on the worst scoring topics. For the runs using D-topics, feedback deteriorates scoring on all measures.

We can regard the T-field of the topics as a "gold standard" experiment on query expansion. If we compare the score of runs using TD-topics with the scores of runs using D-topics, we see an improvement on all measures and runs. In particular the improvement on the weak-scoring topic measures is substantial.

The results for Snowball stemming are positive overall. Stemming helps both the overall performance, with a best score for precision at 10, as well as the performance of the worst scoring topics, with a best score for the percentage of topics with a top 10 relevant document. For runs using D-topics, stemming gives the best score for all measures. The use of both stemming and feedback gives the best score for the area under the MAP(X) curve for the runs using TD-topics, but does not promote performance on the other measures.

We also break down the score over the 50 old topics (in Table 2) and the 50 new topics (in Table 3). Note that the area underneath MAP(X) versus X curve (in the last column) is now calculated for the worst 12 topics. For both the old and new topics, the effectiveness of feedback and stemming is comparable to the effectiveness on all topics. There is, however, a striking difference in the performance between the two types of topics: the new topics

Table 2: Results for the old topics (D top and TD bottom).

Run identifier	MAP	Prec.10	NoTop10	MAP(X)
Words	0.1066	0.2640	**14.0%**	0.0064
Words+feedback	0.0969	0.2460	20.0%	0.0039
Stems	**0.1164**	**0.3020**	18.0%	**0.0108**
Stems+feedback	0.1065	0.2640	18.0%	0.0085
Words	0.1349	0.3180	12.0%	0.0142
Words+feedback	**0.1377**	0.3200	16.0%	0.0143
Stems	0.1327	**0.3300**	**6.0%**	0.0185
Stems+feedback	0.1361	**0.3300**	16.0%	**0.0204**

Table 3: Results for the new topics (D top and TD bottom).

Run identifier	MAP	Prec.10	NoTop10	MAP(X)
Words	0.3064	0.4420	16.0%	0.0142
Words+feedback	0.2971	0.4380	14.0%	0.0105
Stems	**0.3475**	**0.4900**	**10.0%**	**0.0294**
Stems+feedback	0.3071	0.4500	14.0%	0.0216
Words	0.3300	0.4920	**6.0%**	0.0433
Words+feedback	0.3528	**0.5020**	10.0%	0.0368
Stems	**0.3572**	0.5000	**6.0%**	**0.0551**
Stems+feedback	0.3386	0.4780	12.0%	0.0478

give a much higher mean average precision score. This is an obvious consequence of the way the old topics were selected for inclusion in this year's Robust track. As a result, the worst topic measures are dominated by the old topics.

3 Web Track

After describing our experimental setup for this track, we discuss our runs for the home/named page finding task (known-item search), followed by the runs for the topic distillation task (key resource search).

System Description

All Web track runs use the FlexIR information retrieval system. We employ a number of techniques:

Document representation We create indexes for (1) the full documents, (2) the text in the title tags, (3) the anchor texts pointing toward the document. For the anchor texts index, we unfold relative links and normalize URLs, and do not index repeated occurrences of the same anchor text [10].

Tokenization We remove HTML-tags, punctuation marks, apply case-folding, and map marked characters into the unmarked tokens. We either index the free-text without further processing, or use the Snowball stemming algorithm [13].

Retrieval model We use three retrieval models. First, a statistical language model [4] with a uniform query term importance weight of either 0.35 or 0.70. Second, the Okapi weighting scheme [11] with tuning parameters $k = 1.5$ and $b = 0.8$. Third, the `Lnu.ltc` weighting scheme [1] with *slope* at 0.1 or 0.2; the pivot was set to the average number of unique words per document.

Combination We use the standard combination methods such as CombSUM and CombMAX [3], or weighted fusion [14]. We combine either full length runs, or limit the combination to the top *n* results. Unless indicated otherwise, we normalize the scores before combining them.

Minimal span weighting We calculate a minimally matching span for each document. Intuitively, a minimal matching span is the smallest text excerpt from a document that contains all terms which occur in the query and the document. Minimal span weighting depends on three factors (for details, see [2, 5, 8]).

1. *document similarity*: The document similarity is computed for the whole document, i.e., positional information is not taken into account. Similarity scores are normalized with respect to the maximal similarity score for a query.
2. *span size ratio*: The span size ratio is the number of unique matching terms in the span over the total number of tokens in the span.
3. *matching term ratio*: The matching term ratio is the number of unique matching terms over the number of unique terms in the query, after stop word removal.

In two separate sections, we will now address our runs and results for the home/named page finding task, and the topic distillation task.

3.1 Home/Named Page Finding Task

Runs

We submitted the following five official runs for the home/named page finding task:

UAmsT03WnOWS CombSUM of top 1000 of Okapi on word-based and stemmed full document indexes.

UAmsT03WnLM Language model run ($\lambda = 0.70$) on word-based full document index.

UAmsT03WnLn3 CombMAX on the top 25 of `Lnu.ltc` runs (*slope* = 0.2) on the three stemmed indexes: full documents, titles, and anchor texts.

UAmsT03WnLM3 Weighted fusion of language model runs ($\lambda = 0.70$) on the three word-based indexes: 0.7 full documents, 0.2 titles, and 0.1 anchor texts.

UAmsT03WnMSW Minimal span weighting based on the `Lnu.ltc` run (*slope* = 0.1) on the stemmed full document index.

Results

The results of the official runs for the home/named page finding task are shown in Table 4 (best scores in boldface). The second column gives the mean reciprocal rank, the third the number and percentage of topics with a relevant document in the top 10, the fourth the number and percentage of topics for which no relevant document is found (in the top 50). The language model run combining the non-stemmed documents, titles, and anchors scores best with an average reciprocal rank of 0.5185. The `Lnu.ltc` weighted combination of the three stemmed indexes scores second best.

Table 4: Results for home/named page finding.

Run identifier	MRR	Top 10	not found
`UAmsT03WnOWS`	0.3833	178 (59.3%)	70 (23.3%)
`UAmsT03WnLM`	0.3592	170 (56.7%)	81 (27.0%)
`UAmsT03WnLn3`	0.4982	**218** (72.7%)	**38** (12.7%)
`UAmsT03WnLM3`	**0.5185**	214 (71.3%)	46 (15.3%)
`UAmsT03WnMSW`	0.4073	189 (63.0%)	64 (21.3%)

Table 5 shows the mean average precision of the base runs used in combinations for our official runs. All `Lnu.ltc` runs use a slope of 0.2, and all language model runs use a uniform term weight of 0.70. Here, we retrieve up to 1,000 documents per topic, leading to slightly higher MRRs than the official runs using a maximum of 50 documents. We see an interesting difference between the three retrieval models: where the `Lnu.ltc` and Okapi models score best on the full document representation, the language model runs on the anchor text index score more than 20% better than the runs on the full document index. In fact, our best score on a single index is on the language model run on the non-stemmed anchor text index. There is no clear benefit of the use of a stemming algorithm on the mean reciprocal ranks: stemming improves the score for four out of the nine comparative runs.

Table 5: MRR for home/named page finding base runs.

Index type		`Lnu.ltc`	Okapi	LM
Documents	Words	0.3750	0.3795	0.3604
	Stems	0.3697	0.3833	0.3616
Titles	Words	0.2339	0.3421	0.3536
	Stems	0.3655	0.3334	0.3487
Anchors	Words	0.3068	0.3593	**0.4436**
	Stems	0.2934	0.3379	0.4278

There is another interesting difference between the retrieval models, which has to do with combination. The combination of Okapi runs on the document stems and words, `UAmsT03WnOWS`, does not improve over document stems run. The combination of the three stemmed `Lnu.ltc` runs, run `UAmsT03WnLn3`, does improve 34.8% over the best scoring stemmed runs. The combination of the three non-stemmed language model runs, `UAmsT03WnLM3`, improves 16.9% over the best scoring base runs. Finally, the run using the matching-span weighting uses a `Lnu.ltc` full document base run with a different slope of 0.1 scoring a MRR of 0.2742. The resulting run, `UAmsT03WnMSW`, improves no less than 48.5% over the underlying base run.

Table 6: Results for home page topics.

Run identifier	MRR	Top 10	not found
`UAmsT03WnOWS`	0.2567	67 (44.7%)	55 (36.7%)
`UAmsT03WnLM`	0.2462	64 (42.7%)	60 (40.0%)
`UAmsT03WnLn3`	0.4105	97 (64.7%)	**26** (17.3%)
`UAmsT03WnLM3`	**0.4402**	**101** (67.3%)	33 (22.0%)
`UAmsT03WnMSW`	0.2708	73 (48.7%)	53 (35.3%)

We also break down the score over the 150 home page topics (in Table 6) and the 150 named page topics (in Table 7). Here we see a much better performance on the named page topics. This is perhaps unexpected because named page finding is conceived to be a more difficult task than home page finding. The simple explanation is that we decided not to apply special home page finding strategies. Although techniques like slash-counts or URL priors are effective for home page finding [7], they seem to hurt the named page topics considerably. Even without a particular home page bias, home pages can be retrieved with reasonable effectiveness, as is witnessed by our results for the home page topics in Table 6.

Table 7: Results for named page topics.

Run identifier	MRR	Top 10	not found
`UAmsT03WnOWS`	0.5098	111 (74.0%)	15 (10.0%)
`UAmsT03WnLM`	0.4721	106 (70.7%)	21 (14.0%)
`UAmsT03WnLn3`	0.5859	**121** (80.7%)	12 (8.0%)
`UAmsT03WnLM3`	**0.5969**	113 (75.3%)	13 (8.7%)
`UAmsT03WnMSW`	0.5438	116 (77.3%)	**11** (7.3%)

3.2 Topic Distillation Task

Runs

We submitted the following five official runs for the topic distillation task:

UAmsT03WtOk3 Weighted fusion of Okapi runs on the three stemmed indexes: 0.7 full documents, 0.2 titles; and 0.1 anchor texts.

UAmsT03WtLM3 Weighted fusion of language model runs on the three stemmed indexes: 0.7 full documents ($\lambda = 0.35$), 0.2 titles ($\lambda = 0.7$), and 0.1 anchor texts ($\lambda = 0.7$). We combine the probabilities without normalization.

UAmsT03WtOkI Weighted fusion of 0.9 Okapi run on the stemmed full document index with 0.1 of a link topology measure. We applied the realized indegree on the top 10 documents [10]. This is a variant of HITS [6] where we consider the fraction of inlinks that is in the local set—roughly a `tf·idf` measure for link topology.

UAmsT03WtLMI Weighted fusion of 0.9 language model run ($\lambda = 0.35$) on the stemmed full document index with 0.1 of the realized indegree of the top 10 documents.

UAmsT03WtOkC Weighted fusion of 0.8 Okapi run on the stemmed full document index with 0.2 of a URL-based reranking. The reranking was done by clustering the found pages by their base URLs, and to only return the page with the lowest slash-count per cluster.

Results

The results of the official runs for the topic distillation task are shown in Table 8 (best scores in boldface). The second column shows the mean average precision, the third to fifth columns show the precision at 10, 20, and 30 documents, respectively. The best score is obtained by `UAmsT03WtOk3`, the fusion of Okapi runs on the three stemmed indexes. The second best score is obtained by `UAmsT03WtOkC`, a URL-based clustering of the Okapi full documents run. Before discussing the results of our experiments, we first evaluate the results of the runs used to create our official runs.

Table 8: Results for topic distillation.

Run identifier	MAP	Prec. at 10, 20, 30		
UAmsT03WtOk3	**0.1344**	**0.0980**	**0.0810**	**0.0787**
UAmsT03WtLM3	0.1019	0.0840	0.0630	0.0533
UAmsT03WtOkI	0.0862	0.0760	0.0660	0.0567
UAmsT03WtLMI	0.0412	0.0280	0.0260	0.0267
UAmsT03WtOkC	0.1127	0.0860	0.0650	0.0540

Table 9 shows the results of the base runs used in combination for our official runs. All these runs use the Snowball stemming algorithm [13]. We see a remarkable divergence between the scoring for Okapi and the language model. The Okapi model performs comparable on all the three indexes, documents, titles, and anchors. The language model performs poorly on the document and title indexes, but excels for the anchor text index. The combination of the three Okapi runs, `UAmsT03WtOk3`, improves significantly over the best underlying run (MAP +38.4%, Precision at 10 +25.6%). The combination of language model runs, `UAmsT03WtLM3`, uses far from optimal relative weights and, as a result, does not improve over the anchor text run. The runs using the hyperlink graph topology do not result in significant improvement. The Okapi run `UAmsT03WtOkI` slightly improves its precision at 10 over the document run; whereas the language model run `UAmsT03WtLMI` slightly decreases its precision at 10 over the document run. Finally, the Okapi run clustering per base URL, `UAmsT03WtOkC`, does improve over the Okapi document run (MAP +25.1%, Precision at 10 +16.2%).

Table 9: Results for topic distillation stemmed base runs.

Run type	MAP	Prec. at 10, 20, 30		
Doc. Okapi	0.0901	0.0740	0.0580	**0.0527**
Title Okapi	0.0870	0.0780	**0.0590**	0.0453
Anchor Okapi	0.0971	0.0780	0.0560	0.0493
Doc. LM (0.35)	0.0386	0.0300	0.0320	0.0293
Title LM (0.70)	0.0434	0.0480	0.0360	0.0293
Anchor LM (0.70)	**0.1068**	**0.0860**	0.0560	0.0473

4 Conclusions

In this paper we have described our participation in the TREC 2003 Robust and Web tracks.

For the Robust track, we experimented with the impact of stemming and feedback on the worst scoring topics. Our results suggest that blind feedback can help overall performance but does not increase the effectiveness on the lowest scoring topics. Our results also suggest that applying a stemming algorithm does benefit both the overall performance, as well as the performance of the worst scoring topics. This result sheds some new light on the role of morphological normalization in information retrieval.

For the Web track, we saw very similar results for both the home/named page finding task and the topic distillation task. Using the hyperlinks in the collection for creating an anchor text index turns out to be very effective. Also, the use of HTML-structure in the documents to elicit their titles turns out to be effective. Combining these alternative document representations with a standard document index led to our best scores for both tasks.

A further general observation is the effectiveness of compact document representations, such as indexing only document titles, or only anchor texts pointing toward documents. These compact document representations result in performance that meets or exceeds the performance of a massive full document text index. This result suggests that it is feasible to create effective retrieval indexes for even larger web collections, provided that the appropriate document representation is chosen.

Acknowledgments Jaap Kamps was supported by NWO under project numbers 400-20-036 and 612.066.302. Christof Monz was supported by NWO under project numbers 612-13-001 and 220-80-001. Maarten de Rijke was supported by NWO under project numbers 612-13-001, 365-20-005, 612.069.006, 612.000.106, 220-80-001, 612.000.207, and 612.066.302.

References

[1] C. Buckley, A. Singhal, and M. Mitra. New retrieval approaches using SMART: TREC 4. In D. Harman, editor, *The Fourth Text REtrieval Conference (TREC-4)*, pages 25–48. National Institute for Standards and Technology. NIST Special Publication 500-236, 1996.

[2] C. Clarke, G. Cormack, and T. Lynam. Exploiting redundancy in question answering. In D. H. Kraft, W. B. Croft, D. J. Harper, and J. Zobel, editors, *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 358–365. ACM Press, New York NY, USA, 2001.

[3] E. Fox and J. Shaw. Combination of multiple searches. In D. Harman, editor, *The Second Text REtrieval Conference (TREC-2)*, pages 243–252. National Institute for Standards and Technology. NIST Special Publication 500-215, 1994.

[4] D. Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, Center for Telematics and Information Technology, University of Twente, 2001.

[5] V. Jijkoun, G. Mishne, C. Monz, M. de Rijke, S. Schlobach, and O. Tsur. The University of Amsterdam at the TREC 2003 question answering track. In *The Twelfth Text REtrieval Conference (TREC 2003)*. National Institute for Standards and Technology, 2004.

[6] J. M. Kleinberg. Authoritative structures in a hyperlinked environment. *Journal of the ACM*, 46:604–632, 1999.

[7] W. Kraaij, T. Westerveld, and D. Hiemstra. The importance of prior probabilities for entry page search. In K. Järvelin, M. Beaulieu, R. Baeza-Yates, and S. H. Myaeng, editors, *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 27–34. ACM Press, New York NY, USA, 2002.

[8] C. Monz. *From Document Retrieval to Question Answering*. ILLC dissertation series 2003-04, University of Amsterdam, 2003.

[9] C. Monz and M. de Rijke. Shallow morphological analysis in monolingual information retrieval for Dutch, German and Italian. In C. Peters, M. Braschler, J. Gonzalo, and M. Kluck, editors, *Evaluation of Cross-Language Information Retrieval Systems, CLEF 2001*, volume 2406 of *Lecture Notes in Computer Science*, pages 262–277. Springer, 2002.

[10] C. Monz, J. Kamps, and M. de Rijke. The University of Amsterdam at TREC 2002. In E. M. Voorhees and L. P. Buckland, editors, *The Eleventh Text REtrieval Conference (TREC 2002)*, pages 603–614. National Institute for Standards and Technology. NIST Special Publication 500-251, 2003.

[11] S. Robertson, S. Walker, and M. Beaulieu. Experimentation as a way of life: Okapi at TREC. *Information Processing & Management*, 36:95–108, 2000.

[12] J. Rocchio, Jr. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART Retrieval System: Experiments in Automatic Document Processing*, Prentice-Hall Series in Automatic Computation, chapter 14, pages 313–323. Prentice-Hall, Englewood Cliffs NJ, 1971.

[13] Snowball. Stemming algorithms for use in information retrieval, 2003. `http://www.snowball.tartarus.org/`.

[14] C. C. Vogt and G. W. Cottrell. Predicting the performance of linearly combined IR systems. In W. B. Croft, A. Moffat, C. J. van Rijsbergen, R. Wilkinson, and J. Zobel, editors, *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 190–196. ACM Press, New York NY, USA, 1998.

Bangor at TREC 2003: Q&A and Genomics Tracks

Terence Clifton Alex Colquhoun William Teahan
{terence, alex, wjt}@informatics.bangor.ac.uk

This paper describes the participation of the School of Informatics, University of Wales, Bangor at TREC'2003 in the Q&A and Genomics Tracks. The paper is organized into three parts as follows. The first part provides a brief overview of the logic-based framework for Knowledgeable Agents that is currently being developed at Bangor. This was adopted as the basis for implementations used for both Tracks. The second part describes the Q&A system that was developed based on the framework, and the final part describes some experiments that were conducted within the Genomics Track at specifying context using GeneRIFs (for a Q&A system being developed for the BioMedical domain).

"Knowing About" Knowledge: A Framework for Knowledgeable Agents

A.1 Introduction

We are in the process of designing and developing a novel logic-based framework for implementing knowledgeable agents that will become the core component for our multi-agent information retrieval systems.

In Teahan (2003), we describe a framework for designing and implementing knowledgeable agents and Knowledge Grids. The framework is based on three types of knowledge relations: *Knows*, *KnowsAbout*, and *KnowledgeableAbout*. These are used to define what an agent *knows*, what it *knows about*, and whether an agent has been judged to be *knowledgeable* by other agents. In Teahan (2003) and in the second part of this paper, we describe how a Knowledge Grid could be implemented (based on the framework) which has "knowledge" based on the three defined knowledge relations. Essentially, the architecture is based on using knowledgeable agents as a middle layer between the user and the information resources. A key aspect of the design is the use of information extraction coupled with compression-based language modelling technology (Teahan & Harper, 2003) and the use of a conversational agent that the user asks questions of and receives answers from the system.

In this architecture, there are three types of objects: users, knowledgeable agents and information resources. The users do not interface directly with the information resources. Instead, they must go through a knowledgeable agent who effectively acts as a knowledge broker in determining which of the information resources are likely to contain an answer to the user's questions. Notice that knowledgeable agents may need to go though other knowledgeable agents in the hunt to find the most relevant answer to the user's questions.

A.2 A framework for knowledgeable agents

This section outlines the logic-based framework that we wish to use as the basis of knowledge within the Knowledge Grid architecture. We wish to stress that the framework as described below is still in its developmental stage, and its final form, we envisage, will be somewhat different based on the experiences we garner from future research.

We feel that the traditional propositional truth-based approach that epistemic logic-based multi-agent systems take, which are usually formulated as normal modal logics using the semantics of Kripke (Wooldridge, 2002), is not sufficiently expressive enough for our purposes. Instead, we would like to adopt some of the capabilities of Question/Answering systems within our inference capabilities. A problem with the propositional truth-based approach is that although we can state what an agent may know *per se*, it does not help us find out whether an agent *knows* an *answer* to a *question*, and just as importantly, *what answers* an agent *knows* to a *question*. Neither does it help us find out what an agent knows *about* (where knowing *about* a topic implies that you know *something* about the topic, but it does not imply that you know *everything* about the topic).

We feel that there are three necessary conditions for an agent to be "knowledgeable". The key condition, which we refer to as the *Knowledge Test*, is the following: "*An agent is judged to be knowledgeable by other (external) knowledgeable agents*". This states that judges are used to adjudicate on whether an agent is knowledgeable or not (analogously to the Turing Test in Artificial Intelligence). The judges are agents – either human or computer-based – that must also be "knowledgeable". Like the Turing Test, it is assumed that a question and answering testing process is used before making the judgment. The second condition is a logical consequence of the first condition: "*Other agents must have the ability to learn about and/or be informed of what the agent knows about.*" Simply stated, if other agents don't know about what the agent knows about, then they can't make a judgment in the first place. The third condition states: "*The agent must know: what it knows about, and what it doesn't know about.*" This again relies on the judging process used for the Knowledge Test: it would seem a natural response for a knowledgeable agent to answer "*Sorry, I only know about X and not Y*" to something it doesn't know about.

We have devised the following logic-based framework based on these conditions. We define three logical relationships – *Knows*, *KnowsAbout* and *KnowledgeableAbout*.

A.2.1 The *Knows* relation

We define *Knows*, a 5-tuple relation, as follows:
Knows (agent, context, question, answer, relevance).

This is explained as follows: The specified *agent* believes that an *answer* to a *question* for a specified *context* has the specified *relevance* (this is a real number in the range 0 to 1.0 with 1.0 indicating absolute belief that the answer is relevant to the question). A *representation* of the context, question and answer is provided by the specified *context, question* and *answer* which can be arbitrary text passages or strings or some other representation (depending on the implementation). Note that it is possible to ask the same question but in different contexts. The following example is provided as further explanation.

Example 1: *Knows (A, "Domain: Geography", "Where is Bangor?", "North Wales"*, 1.0).

In this example, the agent believes she knows that the answer to the question "*Where is Bangor*?" is "*North Wales*". She assigns a relevance ranking of 1.0 (in other words, she believes that the answer is certainly correct). The context in this case is the domain of geography.

An agent may believe many answers are relevant to a particular question and context. As a shorthand notation, we write this in the following manner:
Knows (agent, context, question): answer$_1$, r_1; *answer*$_2$, r_2; ...

We can also assign an agent's list of answers to a variable. For example, K_A = *Knows (A, "Domain: Geography", "Where is Bangor?")*. Similarly, we can assign to a variable all that an agent knows on a particular context: K_B = *Knows (B, "Domain: Geography")*.

A.2.2 The *KnowsAbout* relation

We define *KnowsAbout*, a 5-tuple relation as follows:
KnowsAbout (agent, topic, context, knows, relevance).

This is explained as follows: The *agent* believes that the list of questions and answers denoted by *knows* are related to the *topic* given the *context* and have the specified *relevance*. Intuitively, the agent believes that she *knows about* the topic given a certain context because she knows the answers to the specified questions. Topics and contexts can be arbitrary text passages or strings as above for the *Knows* relation. Note that the agent may know about the same topic but in different contexts.

Example 2: *KnowsAbout (B, "Bangor", "Domain: General Knowledge"*, K_A, 0.9).

In this example, agent *B* has some general knowledge about the topic "*Bangor*". What he knows are the same answers that agent *A* knows to the question "*Where is Bangor*?" in a geographical context. He assigns a weighting of 0.9 to his belief in the relevance of agent *A*'s answers.

It seems reasonable to assume that if an agent knows the answer to something, then it knows *about* that something. This is written as follows:
∀*agent, context, question, relevance*
K_i = *Knows (agent, context, question, answer, relevance)* ⇒ *KnowsAbout (agent, question, context*, K_i, *relevance)*.

In this case, the agent is inferred to know *about* each question because she knows the answers to them. So for example, from K_A above, we can infer that agent *A* knows about the following: *KnowsAbout (A, "Where is Bangor?", "Domain: Geography"*, K_A, *relevance)*.

By default, the same relevance from the *Knows* relation can be adopted for the *KnowsAbout* relation, although this can be overridden at a latter time.

It also seems reasonable to assume that if an agent knows the context of a given question and answer, then it knows *about* that context. This is written as follows:
∀*agent, context, question*
K_i = *Knows (agent, context, question, answer, relevance)* ⇒ *KnowsAbout (agent, context, context*, K_i, *relevance)*.

In this case, the topic that the agent knows about is the context itself.

A.2.3 The *KnowledgeableAbout* relation

We define *KnowledgeableAbout*, a 6-tuple relation as follows:
KnowledeagbleAbout (knowledgeable-agent, testing-agent, agent, topic, context, relevance).

This is explained as follows: The *knowledgeable-agent* believes that the *agent* is knowledgeable about the *topic* given the *context* with the specified *relevance* because that agent knows about the same things as the *testing-agent* knows about. Effectively, an external agent, which is designated as being knowledgeable, uses test questions to determine if a person or agent knows about some topic. The knowledgeable agent delegates the testing agent to perform the test – this may be a "virtual" agent that is provided with sufficient knowledge necessary in relation to the test. The testing agent may in fact be provided with a subset of the knowledge known by the knowledgeable agent. Alternatively, other possibilities are having the knowledgeable agent spawn a testing agent to perform the test, or having the knowledgeable agent designate an independent testing agent to perform the test. Notice that the series of test questions have themselves now become a form of knowledge.

A.3 Questions, contexts and topics

Note that in our definitions above of *Knows*, *KnowsAbout* and *KnowledgeableAbout*, we do not explicitly state how the questions are represented or how they are to be matched with each other, and similarly for the topics and contexts. If questions, contexts and topics consist of text strings, an inference system may simply impose the *strictest* requirement that the strings match exactly. Alternatively, a less strict matching system may be employed. For example, if contexts are specified as a *set of labels* that specify the context's relevant domains then the conditions for contexts to match may simply be that there exists at least one label common to both sets of domains. For example, the context for the question "*How do you cook pumpkin pie*?" may be the set of domain labels "*Domain: Cooking, Recreation*". Another agent may know about the answer to the same question, but in the slightly different context, "*Domain: Cooking, Hobbies*", although the inference system may infer the contexts match because of the common label "*Cooking*" present in both.

We have deliberately left open the specific representation of the questions, contexts and topics to the designer of the knowledge system. We feel that different representations are required in different applications depending on the nature of the knowledge that needs to be specified and/or manipulated. For example, in a knowledge-based information retrieval system, the context could be used to specify the *information purpose* of the agent that produced each document, and then this can be matched against the *information need* of the user based on the user's question and previous questions.

A.4 References

W. Teahan and D. Harper. 2003. "Using Compression-Based Language Models for Text Categorization" In *Language Modelling for Information Retrieval,* Chapter 7, pp 141-165. Kluwer Academic Publishers.

W. Teahan. 2003. "Knowing About Knowledge: Towards a Framework for Knowledgeable Agents and Knowledge Grids". Artificial Intelligence and Intelligent Agents Tech Report AIIA03.2, School of Informatics, University of Wales, Bangor.

Wooldridge, M. 2002. *An Introduction to MultiAgent systems.* Wiley, New York.

QITEKAT

Question Inference Tools Employing Knowledgeable Agent Technologies

Abstract

We present the QITEKAT Question-Answering system based on the conceptual theory of *Knowing About Knowledge*, which adopts an agent-based approach to extract information from suitable corpora. The components of the QITEKAT system entered by the School of Informatics, University of Wales, Bangor, in the 2003 Text Retrieval Conference are described in detail. We describe PPM compression techniques for Named Entity classification; distributed agent technologies for developing a Knowledgeable Framework and Knowledge Grid; and a Search Engine corroboration system for generating confidence estimates for Question Answering. We present favourable results for certain question types in the TREC Question Answering Track, and discuss future directions for the QITEKAT architecture.

B.1. Introduction

In developing the QITEKAT system, we were aiming to take a first step on the TREC road, providing a foundation for future development within the School of Informatics, at the University of Wales Bangor, for knowledge representation, extraction and language processing techniques. Agent technologies and techniques are a popular tool in modern computer science, and have been applied to a number of problems, including previous TREC question and answering tracks (Chu-Carroll et al, 2002). As a secondary goal, we were aiming to use the TREC Question Answering track as a benchmark to evaluate a developing framework for Knowledgeable Agents and Knowledge Grids (Cannataro and Talia, 2003), based on the concepts of 'Knowing About Knowledge' (Teahan, 2003).

The short development time period (7 weeks) meant that many of the core components of the QITEKAT system were based on standard information extraction and question/answering techniques, although we were able to incorporate a number of interesting features, particularly in Named Entity tagging and relevance ranking.

This report firstly describes the main components of the UWB QITEKAT Question Answering System (Section 2). Section 3 presents results obtained from various experiments on past and current TREC Q&A data, and is followed by a brief analysis of the performance of the system (Section 4). The report concludes with a discussion of possible future enhancements (Section 5).

B.2. System Description

The QITEKAT system was designed not only to offer a practical implementation for the theoretical concepts of 'Knowing About Knowledge', which are explained in greater detail at the start of this paper, but to offer a foundation for the future development of information extraction and question answering techniques to enhance the system for future use, either through the TREC forum, or for practical applications. This need for extensibility, and to be able to swap out various sections of the system as new techniques were developed, leant itself to the use of an object-oriented development platform. We decided that the Java language would offer the greatest flexibility for future development.

The knowledge framework proposed by Teahan (Teahan, 2003), which is used as the basis for the extraction of knowledge relations from suitable source documents essentially relies on a reverse approach to standard Q&A techniques. Rather than using the question text to retrieve a subset of documents from the test collection, which are then analysed to find an answer, the QITEKAT system was designed to parse the entire collection, forming a number of question/answer relations before any actual questions are posed.

The TREC 2003 Q&A Track uses the AQUAINT document collection as its source corpus, which consists of over 1 million documents, totalling 375 million words. Quite obviously, performing any kind of extensive parsing or analysis of this size of document collection would be computationally intensive, and not best suited to the Java language.

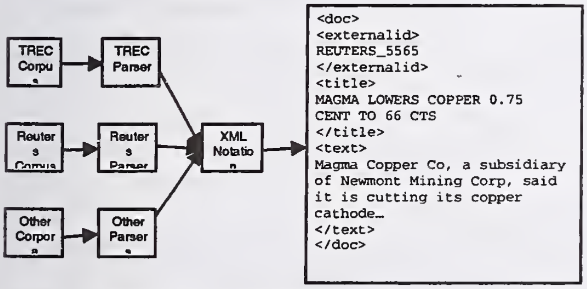
With these considerations in mind we adopted a 2-level modular approach to the system development, using the Java language to facilitate extensibility, and C where speed was of the essence, integrated using Java native methods. The system was developed based around three main stages:

- document normalisation and storage;
- knowledgeable agents;
- question analysis and answer ranking.

Figure B.2 shows the component make up, and how each of the individual modules interacts with the rest of the system, and a more detailed explanation of each of the key components follows.

B.2.1 XML Document System

Although the TREC Q&A track was our main target during the development of the QITEKAT system, it was important that we consider its application to other areas and document sources. With this in mind we developed a rudimentary XML notation to normalise any source documents, and store them in a consistent fashion for analysis by the Knowledgeable Agents of the system.



```
<doc>
<externalid>
REUTERS_5565
</externalid>
<title>
MAGMA LOWERS COPPER 0.75
CENT TO 66 CTS
</title>
<text>
Magma Copper Co, a subsidiary
of Newmont Mining Corp, said
it is cutting its copper
cathode…
</text>
</doc>
```

Figure B.1 – Simple XML Notation

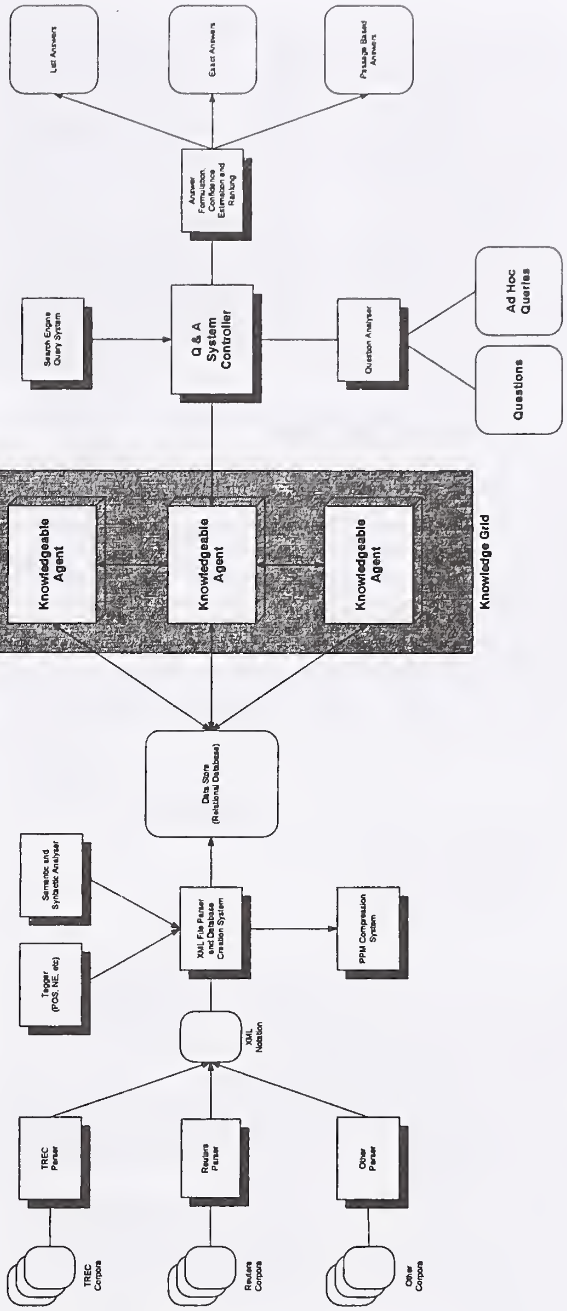


Figure B.2 – System Design

This means that the addition of a new corpora or alternative source of information could be handled using a simple Java based API, and plugged into the system as details of the data source become available.

B.2.2 Speech Tagger

The speech tagger forms a major portion of the QITEKAT development, as the part of speech and named entity tags are used as the basis for extracting knowledge relations from the AQUAINT documents.

The system is loosely based upon the Fastus system (Hobbs et al, 1996), employing the architecture of a cascading finite state automata in order to achieve usable levels of performance. Each stage of the system was developed as a switchable module, so it could be invoked as required, depending on the document structure it is being used to parse.

The system handles each document as a complete entity, separating it into sentences, and then words, before passing it on to first the POS tagger, and subsequently the NE tagger. Again, XML is used extensively to provide run-time modification of the rules and constructs used to tag the portions of a sentence.

POS Tagger and Phrase Chunker

The Part of Speech (POS) tagger is a 2-phase tagger, adopting ideas proposed by Brill (Brill, 1992). It uses a frequency count, extrapolated from a pre-tagged version of the Brown corpus to assign preliminary part of speech tags to each word in a sentence, using the Penn Treebank tagset. These pre-tagged words are then re-examined by a transformation based tagger. The rules for this tagger were developed through automatic examination of the Brown corpus, with some minor manual modification.

Once tagging of individual parts of speech is complete, the sentences are passed on to the phrase chunking module, which adopts a three-stage approach. A POS is tagged with one of three standard types

- Inside a chunk;
- Outside a chunk;
- Boundary of a chunk.

These are based solely on the POS tag assigned to the particular word. This phase is succeeded by a transformation based chunk, again based on rules generated from the Brown Corpus.

Once the final chunking is complete, each phrase chunk is examined to determine its content, and is labelled accordingly (Verb, Proper Noun, Noun, Punctuation, Other).

XML is used to store the transformation and frequency rules for this portion of the speech tagging system, offering a look-ahead/behind matching system on three entity types:

- Words;
- POS Tags;
- Chunk Tags.

Figure B.3 shows an example of an XML rule description for transforming a noun tag (NN) to a verb tag (VB). We can see that the rule specifies that in order for the transformation to take place the POS tag *TO* must be found in the position before the tag being examined. Multiple conditions can be applied for each rule. The validity of the new tag is checked using the original tag frequency information from the Brown corpus, to ensure that the new tag is a suitable option for the current word. Transformations are applied in frequency order and can be cascaded to apply multiple transformations to the same entity.

```
<rule>
<initialtag>NN</initialtag>
<newtag>VB</newtag>
<condition>POS</condition>
<operator1>TO</operator1>
<operator2>-1</operator2>
</rule>
```

Figure B.3 – XML Based Tag Transformation Rule

NE Tagger

Once phrase chunking and identification is complete the system is aware of the phrases in a document that correspond to Proper Noun phrases, and are therefore candidates for Named Entity Tagging. Each of the phrase chunks is passed to the NE tagger, which applies a cascading series of modules to determine the type of Named Entity that the chunk refers to: Currencies; Dates; Times; Locations; Professions; Relations; Measures; Organisations; Names (Pre/Post Honours).

Each of these types is defined by a series of rules, again stored as XML for easy modification, which rely on a combination of direct matching, designator matching and sure-fire context rules

Direct matching

Certain named entity types fall into this category, in particular dates and times, which follow a series of standard word patterns. Regular expression matching is used to identify matches, which are then tagged accordingly. For example the regular expression below can be used to match the initial portion of a date such as 23rd October.

```
((0?[1-9]|[1|2][\\d]|3[0|1])(st|nd|rd|th)?)
```

Designator Matching

This method is adopted to determine such NEs as organisations and persons, and relies on common pre and post entity word matches. For example if we have the NE *British Gas Plc*, we can match the Plc designator, and tag the phrase as an organisation. Other such designators that the QITEKAT system relies upon are:

Mr	Sr	Corp
Dr	Ltd	Jr

Sure-fire context rules

Certain sentence constructs are used to determine the type of Named Entity for a specific phrase, where the surrounding context unambiguously denotes a specific type. As an example, take the partial sentence:

```
Shares in XYZ rose 54% on the days trading…
```

The context *Shares in ???* implies that ??? is an organisation, and can be used as a suitable sure-fire rule to tag that particular unknown NE.

A small number of these sure-fire rules were manually created (again stored as XML constructs) in order to tag these particular sentence contexts.

PROFESSION of ???	PERSON
RELATION of ???	PERSON
??? province	LOCATION

Partial Matching

Natural language, and particular the construction of news articles, which are the basis for the TREC Question & Answering Track Corpora, often rely heavily human memory and implicit definition. For example, in an article about a particular person, they may be referred to by their full name only once, early in the article, yet will be referred to again on numerous occasions throughout the text. This may be by some abbreviation of their name, say their surname only, or some other means such as anaphora (*He said...*). It is important for a successful Named Entity tagging system to be able to handle this *cross-reference* in a particular document in order to correctly tag the unknown NEs present.

The QITEKAT system employs a simple partial matching algorithm to solve these problems, and cross-tag equivalent entities. This works by extracting all known NEs from a particular document (which have been identified previously, either by designator matching or some other means) and creating partial orders of each. These partial orders are then compared to the remaining unknown NEs in the document, and should a match occur, the new NE is tagged with the equivalent type.

As an example, take a document that discusses the work of `Dr. Bill Teahan`. This phrase would be correctly identified as a PERSON, by matching of the Dr. designator. Partial orderings of this phrase would then be constructed (retaining word order to ensure correct cross-matching):

```
Dr. Bill, Dr. Teahan, Bill Teahan
```

Should these phrase constructs occur elsewhere in this same document, they would be tagged according to the original phrase (i.e. As a PERSON type).

PPM-Based Language Modelling

The final stage of the QITEKAT speech tagging system focuses on labelling all remaining unknown NE phrases, and adopts a compression-based language modelling system to achieve this goal. Much research has been carried out into the use of PPM compression systems for the text classification (Teahan and Harper, 2003), whether it be to identify languages, determine authorship or otherwise.

We have adopted a PPM based compression system to deal with unknown NE classification, by training PPM models on various known data sets corresponding to the available NE types in the QITEKAT system (PERSONS, ORGANISATIONS, etc). Given a suitably large data set of known phrases of each type, we have been able to train compression models for each. These models are then used in turn to compress unknown phrases from the document set. The model providing the best compression level (i.e. the shortest code length) is thus assumed to be the most appropriate type for the unidentified phrase.

In initial tests on 200 Reuters news articles, this compression system was able to produce very favourable results, when applied as the final stage in the QITEKAT tagging process.

Number of unknown NEs	141
Number of NEs correctly identified	132
Number of NEs incorrectly identified	9

B.2.3 Knowledgeable Agents

The theory of Knowledgeable Agents proposed in Teahan, 2003, and outlined at the start of this paper is used as the basis for the main document processing component of QITEKAT. Each agent is capable of running autonomously and analysing a given series of XML documents to generate *Knows* and *KnowsAbout* relations, which it then stores for the purpose of question-answering.

B.2.3.1 Regular Expressions

In order to extract *Knows* relations from the AQUAINT corpora, regular expressions were developed manually to pattern match sentence construction for common question types. These expressions were developed using the TREC 2001 question text, and focus on the *Who* and *When* question types only, due to time constraints.

It was important to make the best use of the tagged documents, and to ensure that regular expressions used by the system were not too specific as to require multiple expressions for a single question construct. This led us to develop a dynamic substitution system, whereby a generic RE was populated at run-time using the tagged contents of the sentence it was being applied to.

Again all rules are stored in an XML file, to enable rapid updating and maintenance of the rule base, and a typical entry looks as follows. The file denotes a basic regular expression format, suitable substitution types, an allowable answer type, and a question format for the particular relation

- When did OBJECT1 die?
- Who was OBJECT1?

```
<questionpack>
<domain>PEOPLE</domain>
<answer>DATE</answer>
<object1>PERSON</object1>
<object2>NONE</object2>
<object3>NONE</object3>
<regexp>
(OBJECT1)\sdied\s((on|in|around)\s(ANSWER)
</regexp>
<format>When did OBJECT1 die?</format>
</questionpack>
```

Figure B.4 – XML Based Regular Expression Rule

By using the NEs already tagged in this sentence, the system creates a number of regular expressions, substituting suitable NE types into the ANSWER and OBJECT locations. Given the sentence: `John Lennon died on December 8th, 1980 during a public dramatic interpretation of J.D. Salinger's "Catcher in the Rye"`, the QITEKAT system would tag 1 DATE entity (December 8th, 1980) and 2 PERSON entities (John Lennon and J.D. Salinger) the QITEKAT system would dynamically produce 2 regular expressions:

```
1. (John Lennon)\sdied\s((on|in|around
   )\s(December 8th, 1980)
2. (J.D.Salinger)\sdied\s((on|in|around
   )\s(December 8th, 1980)
```

These would then be applied to the sentence to extract any matches which would be transformed into *Knows* relations. In this case, option 1 would match, resulting in the following relation (given that the "knowledgeable" agent who produced the document text referred to as *A*).

```
    Knows(A, "Domain: PEOPLE",
       "When did John Lennon die?",
       "December 8th, 1980", 1.0).
```

Further examples of extracted *Knows* relations:

```
K1 = Knows(A, "Domain: PEOPLE", "Who is
George W. Bush?", "United States
President", 1.0).
K2 = Knows(A, "Domain: PEOPLE", "When
was George W. Bush born?", "July 6th
1946", 1.0).
```

These *Knows* relations are then used to populate suitable *KnowsAbout* relations such as the following:

```
    KnowsAbout(A, "Domain: PEOPLE",
       "George W. Bush", {K1, K2},
       1.0).
    KnowsAbout(A, "Domain: PEOPLE",
       "John Lennon", Ka, 1.0).
```

A small number of broad domain types are used (PEOPLE, GEOGRAPHY, HISTORY, SPORT, MISC), and all relations are stored within the Knowledgeable Agents using serialized vectors, in order to achieve persistent data storage between executions.

B.2.3.2 Distribution

In developing the QITEKAT system, consideration was given to its use as a prototype for a Knowledge Grid (Cannataro and Talia, 2003), and for knowledgeable agents to communicate effectively with one another. This concept pointed toward the need for some kind of distributed system where agents could show mobility, and the ability to reside on a network, wherever there was data to process.

In addition, the large amount of data that was being handled for the Q&A task (1 million+ documents) lent itself to exploiting distributed paradigms to share the workload of examining this data and extracting suitable relations.

The QITEKAT system uses a simple UDP based system to handle communication between agents. This allows each agent to determine what other resources are available on the grid, and also inform others about the knowledge it possesses. As more agents are added to the grid, each becomes aware of what knowledge resources are available, and where a certain domain of questions may be best answered.

The system handles 5 message types:

Ping	Ask an agent if they are active.
Broadcast	Inform other agents in the grid that this agent is active.
Send_Question	Post a question to a specific agent on the grid.
Send_Answer	Send an answer back.
Send_KnowsAbout	Tell another agent what this agent has information about.

This approach allows knowledge to propagate through the system, as each question is sent from agent to agent to discover answers. When an answer is found, the response is returned, and the agents in the chain are each able to 'learn' that fact. A user only needs to enquire of a single agent in the grid, and that agent will be able to find the other agents on the grid that may be capable of answering the users query, and forward the question as required. A typical interaction between Knowledgeable Agents on this grid system is outlined below:

Agent 1 starts up, loads Knows and KnowsAbout relations and Agent IPs and sends a broadcast message on the local network.

Receives responses from other agents and updates its KnowsAbout relations.

Receives question from user.

Checks its own Knows relations for a suitable answer - none found.

Checks its KnowsAbout relations for another agent that may have an answer - one found (**Agent 2**).

Tags the question and forwards it to **Agent 2.**

Agent 2 finds an answer to the question and sends it back to **Agent 1**

Agent 1 updates it's knows relations so it now knows the answer and won't need to ask **Agent 2** next time.

Agent 1 forwards the answer to the user, and updates its local disk storage.

Figure B.5 – Typical Agent Interaction

B.2.4 Confidence Ranking

In the specific area of question answering it is often the case that systems are able to generate a number of candidate answers for a particular query. In this year's TREC Q&A track for example, an entire section of questions is devoted to returning multiple results for a single query (the so called List questions).

This poses the problem of determining the *best* result for a particular query, which is what is required by the standard questions in the Q&A track, and is likely to be the requirements of any practical application of a Question-Answering system.

The way in which this is often achieved is through a confidence ranking for an answer, reflecting the degree of certainty the system places on the answer returned being correct. The confidence ranking is often returned as a decimal value in the range 0.0 (zero confidence that the answer is correct) to 1.0 (completely confident that the answer is correct).

Past Q&A systems have used a number of means for determining a confidence measure from answers. Weighting based on matching NE types from the answer to that expected by a specific question type (i.e. A *where* type question expects a LOCATION type answer, and so a corresponding answer gets a higher weighting) is popular. Other popular measures include keyword densities in the answer document, and vector matching of question and answer pairs.

We adopted a new approach based on corroboration with external data sources (popular search engines)

B.2.4.1 Search Engine Corroboration

Search engines provide a large document base – Google for example currently claims to index over 3.3 billion Web pages, and as a result are likely to contain many examples of the correct answer to any query likely to be posed to a Q&A system. Although this offers scope to use Web search results as a source corpora for practical Q&A applications, the TREC Tracks require that all answers are found in the AQUAINT document collection. This does not preclude, however, the use of web search results to aid in the Q&A process, and we have adopted a novel approach for confidence ranking of answers, based on the results of an appropriate Web search query.

The fact that a suitable query to a search engine, based on the original question, is likely to result in many examples of the correct answer means that we can use the proportion of each possible answer within these search results to determine a relevance rank for that answer.

The QITEKAT system achieves this through a simple search API, developed in Java, which queries a number of popular search engines. Noun and verb phrase chunks from the question text are used to form a suitable search query, and the abstracts of the first 1000 results are retrieved from the search engine. These results are then scanned to determine the frequency of each of the possible results as produced by the Q&A system. The proportion of these frequencies are then used to calculate a relevance ranking.

This is better explained using a simple example:

- Given the question:
  When did John Lennon die?
- We extract the noun and verb phrases
  John Lennon
  Die
- These are then passed as a search query to Google
  "John Lennon" + "die"
- The first 1000 abstracts are retrieved
- The Knowledgeable Agents return three possible answers
  8th December
  15th August
  19th July
- Thus we find frequency matches for each of these answers in the Google abstracts, and calculate a relevance rating:

ANSWER	FREQ	CALC	RELEVANCE
8th December	462	462/533	**0.87**
15th August	28	28/533	**0.05**
19th July	43	43/533	**0.08**

Table B.1 – Relevance Ranking Calculation

So we have a corroborated relevance for each of the answers, and the Q&A system is able to return the answer 8th December as the most favourable.

B.3. Results

Preliminary testing of the QITEKAT system showed positive results on previous TREC question sets, and these are confirmed by the TREC 2003 evaluations.

B.3.1 Trained Question Types

In developing the regular expression rules to extract *Knows* relations from source corpora we used the question data supplied as part of the TREC 2001 Question-Answering track. We constructed 400 regular expression rules, although time constraints meant we were unable to construct rules for all question types.

B.3.2 TREC 2002

Initial testing of the QITEKAT system was carried out on TREC 2002 Q&A Track questions in order to provide an indication of how the system would perform under typical application. Manual examination shows that of the 500 questions provided, rules have been constructed that should be able to find answers to 122 of them, assuming those answers exist within the AQUAINT source documents.

The system registered 107 correct answers, of which 4 were NIL answer questions, as no answer existed in the AQUAINT corpus. 15 incorrect answers were registered, of which 2 should have been NIL answers.

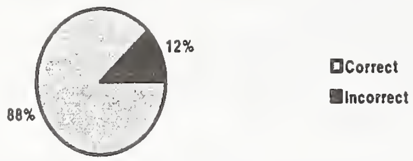


Figure B.6 - Results on TREC 2002 Questions

B.3.3 TREC 2003

Manual evaluation of the TREC 2003 question set showed that the system should have been able to answer 124 of the 500 questions made available with its current regular expression definitions. Evaluation of the TREC 2003 run showed 107 completely correct answers and 6 answers judged as being inexact. 11 incorrect answers were registered, which included answers that were judged as being unsupported answers.

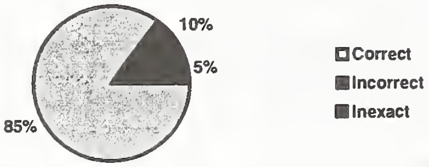


Figure B.7 - Results on TREC 2003 Questions

B.4. Analysis

The results produced by the QITEKAT system, both from in-house tests on previous Q&A data, and on the current TREC Q&A track questions are promising, particularly given the timescale of the development process. With levels of correct answers exceeding 80% in both tests, this implies a positive first step on the Q&A ladder, and a solid foundation to build on the work in Knowledgeable Agents and the concepts of *'Knowing About Knowledge'*.

B.4.1 Question Types

The results gained by the QITEKAT system need to be considered in the context of the question types that were addressed in order to gain a more accurate indication of the performance of the system.

It could be argued that the *When* and *Who* question types are the simpler of the main types used in the TREC evaluations, offering a definite answer type, and often more simple sentence constructs where an answer may be found. We felt this to be the case in this respect, and deliberately chose these types in order to aid the speed of system development in order to meet the deadline for run submission. We hope, however, that the underlying concepts of the system that we have adopted should be able to achieve similar results on all of the major question types, given suitable Regular Expressions on which to match.

B.4.2 Speed

Analysis of the AQUAINT documents which formed the source corpora for the TREC 2003 Q&A evaluation demonstrated the benefits of the distributed design adopted as the basis for the QITEKAT system, but also indicated a need for further speed improvements.

The final evaluation was carried out using a distributed network of 8 Pentium III computers, each using a 128Mb of local memory, and approximately 500Mb of local storage. The parsing and analysis of the 1 million documents took approximately 72 hours on this configuration. Although this level of performance is manageable, it would need to be improved if the system were to be applied to practical applications, or larger corpora, such as Web search results.

B.5. Future Directions

As a foundation for future Q&A and language processing research, the QITEKAT system has performed well, although a number of areas have been targeted as areas for improvement. In particular it is important that we are able to handle a greater number of question types in order to perform a more accurate evaluation of the systems performance, and allow for a direct comparison to other research systems participating in the TREC tracks. Further additional features that we feel may improve system performance, both in terms of speed of execution, and the ability to determine answers are outlined below.

B.5.1 Improved NE Classification

Although the NE classifier developed as part of the QITEKAT system performs well, for the purposes of Q&A it is important to broaden the scope of the system, and introduce further NE types in order to allow for more accurate answer matching. Sekine et al present a system offering a far greater number of NE classifications (Sekine et al, 2002), which we feel would be a beneficial addition to the QITEKAT architecture.

B.5.2 Synonym substitution

The present system architecture offers no methods for word substitution, which is a limiting factor, both in terms of matching questions with appropriate knowledge relations, and also extracting relations from document texts. The addition of a synonym system, such as WordNet (Miller, 1990) would enable a greater number of sentence constructs to be identified and extrapolation of questions to form multiple queries, offering a far greater chance of successful responses.

As an example, take the question text

When did Charles Bronson die?

In the present QITEKAT system, this will match only those relations with an equivalent question construct, which may result in no answer being found. With synonym substitution, however, the query would be reformulated as:

When did Charles Bronson pass away?

which may provide a positive match.

B.5.3 Past Participle Determination

In a similar vein to synonym substitution, it would be useful to develop a feature within the system to automatically generate past participles of verbs, particularly for Search Engine Corroboration.

When querying a search engine, the system passes the main subjects of a question, so for example, given the question:

When did Charles Bronson die?

The system forms a query using *Charles Bronson* and *Die.* It is likely however that in any documents retrieved by a search engine, the information that we are interested in would be described using the past participle (died), i.e.

Charles Bronson died on .....

Substituting the past participle may result in a more useful query string, and ultimately a greater number (or more accurate) results.

B.5.4 Automate RE Production

Manual production of Regular Expressions to extract information from document texts was one of the more time consuming aspects of the initial QITEKAT development, and as a result meant we were only able to focus the tool at a limited number if question types in order to meet the TREC deadline. A key idea for future development of the system is to implement an automated system, capable of producing generic expressions which could then be used to extract further information. Initial thoughts are that this issue may lend itself to a transformation based system, similar to that found in Brill-type POS tagging systems (Brill, 1992), where it would be possible for the system to learn a set of rules, based on existing, manually produced REs.

B.6. References

E. Brill. 1992. "A simple rule-based part-of-speech tagger" In *Proceedings of ANLP-92*, pp 152–155, 1992.

M. Cannataro and D. Talia. 2003. "The Knowledge Grid". In *Communications of the ACM* Vol 46, Number 1, pp 89-93.

J. Chu-Carroll, J. Prager, C. Welty, K. Czuba and D. Ferrucci. 2002. "A Multi-Strategy and Multi-Source Approach to Question Answering". In *TREC 2002 Proceedings.*

J. Hobbs, D. Appelt, J. Bear, D. Israel, M. Kameyama, M. Stickel, and M. Tyson. 1996. "Fastus: A Cascaded Finite-State Transducer for Extracting Information from Natural-Language Text" In *Finite State Devices for Natural Language Processing.* MIT Press, Cambridge MA.

G. Miller. 1990. "WordNet: An On-Line Lexical Database" In *International Journal of Lexicography.*

S. Sekine, K. Sudo, and C. Nobata. 2002 "Extended Named Entity Hierarchy" In *Proceedings of the LREC-2002 Conference*, pp 1818–1824, 2002.

W. Teahan and D. Harper. 2003. "Using Compression-Based Language Models for Text Categorization" *Language Modelling for Information Retrieval,* Chapter 7, pp 141-165. Kluwer Academic Publishers.

W. Teahan. 2003. "Knowing About Knowledge: Towards a Framework for Knowledgeable Agents and Knowledge Grids". Artificial Intelligence and Intelligent Agents Tech Report AIIA03.2, School of Informatics, University of Wales Bangor.

Generating GeneRIFs for a Multi-Agent-based Biomedical Information Retrieval System

Abstract

This section describes work that was done for the 2003 Text Retrieval Conference (TREC) Genomics Track second task. It also describes preliminary work on implementing a multi-agent Biomedical information retrieval system. The proposed multi-agent system will apply knowledgeable agents using a logic-based question and answering framework. Part of this work requires the specifying of context for the questions and answers, and one means of doing this is to generate GeneRIFs for each biomedical document, where a GeneRIF is a MEDLINE standard for describing the contents of the biomedical document in terms of gene function. Various methods are explored to generate the GeneRifs automatically.

C.1 Introduction

The Text Retrieval Conference (TREC) Genomics Track was started to provide a forum for information retrieval in the genomics area. The Secondary Task for this year's track is to analyse automatic methods to reproduce the GeneRIF notation for biomedical papers. Below shows the official definition of what a GeneRIF should be:

> *A concise phrase describing a function or functions (less than 255 characters in length, preferably more than a restatement of the title of the paper.)*
> (www.ncbi.nlm.nih.gov/LocusLink/GeneRIFhelp.html)

The GeneRIF annotation is designed to allow the contents of a bioscience paper to be summarised in such a way as to show the function of the gene, which is the subject of the bioscience paper. An analysis by Mork and Aronson (2003) of NLM found that 95% of GeneRIF snippets contained some text from the title or abstract of the article. About 42% of the matches were taken directly from the title or abstract, 25% contained significant runs of words from pieces of the title or abstract.

The data provided for the Secondary Task consists of 139 GeneRIFs representing all of the articles appearing in five journals – Journal of Biological Chemistry, Journal of Cell Biology, Nucleic Acids Research, Proceedings of the National Academy of Sciences, and Science – during the latter half of 2002.

C.2 Methodology

The main objective of our research is to design and implement a multi-agent based system for knowledge-based retrieval to biomedical literature. The purpose is to provide easy-to-use and seamless interfaces to biomedical literature both for the expert and for the layperson. The main components of the multi-agent systems we envisage will consist of peer-to-peer based communicating agents which are knowledgeable about biomedical resources. The biomedical information retrieval systems will provide scientists in the biomedical community with better support for searching the latest literature, therefore enabling them to be better informed about the latest developments in the biomedical field. The systems will also serve the wider community that will enable researchers, whether they are experts or non-specialists, a seamless and natural interface that will require minimal training.

C.3 Knowledgeable Agent Framework

We are in the process of designing and developing a novel logic-based framework for implementing knowledgeable agents that will become the core components for our multi-agent biomedical information retrieval system. The logic is based on three relations that describe whether an agent is "knowledgeable" or not (Teahan, 2003; also see the first part of this paper).

C.3.1 Context modelling

We feel that context has a very important role to play in specifying knowledge. For example, for an agent to answer the question "What is entropy?"' in a knowledgeable way, the agent must first appreciate the context in which the question is asked. A different answer is required to this question depending on whether the context concerns the domain of physics or the domain of information theory. If neither domain is apparent given the context, then it might be appropriate for an agent (if it wishes to be knowledgeable) to inform the person asking the question that more than one answer is possible to the question. We feel that different representations of context are required in different applications depending on the nature of the knowledge that needs to be specified and/or manipulated.

C.3.2 GeneRIF as a context

For the application that we investigate in this paper, we explore the possibility of using GeneRIF-based annotation for specifying the context of biomedical documents in the Genomics domain. Importantly, in this "context", we consider the MEDLINE GeneRIFs as provided in the training data for the TREC 2003 Genomics Secondary Task experiments as only a representation of the "true"' GeneRIF. By "true", we mean what a correct annotation of the article might be if it was performed by a group of experts, rather than the GeneRIFs encountered in the MEDLINE database. Note that some of the MEDLINE GeneRIFs in the training data evidently fall short of what a group of experts might assign to the "correct" or "true" GeneRIF, if they had been allocated this task.

In light of this, we can also partially ignore the differences between the MEDLINE provided GeneRIF and the candidate GeneRIFs we are automatically generating, since we are only interested in finding a GeneRIF description that encompasses some notion of a true GeneRIF. Hence, providing GeneRIF strings that are potentially longer than the MEDLINE GeneRIF, but have a much greater chance of encompassing the "true" GeneRIF description makes sense, as our goal is to ensure as comprehensive description as possible is generated for information retrieval purposes. Our contention is this will make it more likely that a relevant document is retrieved when the GeneRIF is used in an IR system – in our case, within the distributed IR framework that we are developing based on Knowledgeable Agents.

The consequence of this is that we feel that a modified co-efficient is more suitable for our purposes to evaluate the "goodness" of the generated GeneRIF. This modified co-efficient is a measure of the percentage of words in *X* that occur in *Y*, so we are effectively ignoring the extraneous words in *Y*. The equation below is for the modified measure:

$$D'_{(A,B)} = Z/X \qquad [1]$$

as opposed to the original DICE co-efficient:

$$D'_{(A,B)} = 2 \times Z/(X+Y) \qquad [2]$$

where *A* is the MEDLINE GeneRIF, *B* is the generated GeneRIF, *Z* is the number of words that occur in both *A* and *B*, *X* is the number of words in *A*, and *Y* is the number of words in *B*.

Changing the DICE co-efficient in this way shows that we are only interested in the following questions: "Does the generated GeneRIF contain the MEDLINE GeneRIF?" or "How much of it does it contain?" These questions are noticeably different to what the standard, unmodified DICE co-efficient measures which is: "How similar are the MEDLINE and generated GeneRIFs?"

C.3.3 Experimental Results

Table C.1 shows the results for the secondary task experiments, using both the standard DICE co-efficients (Classic DICE (see equation [2]), Modified Unigram DICE, Modified Bigram DICE, and Modified Bigram Phrases DICE), and the modified measure (see equation [1]). The table is divided into sections by experimental run, and these sections are divided further by the co-efficient used to generate the average result for the particular run. The runs processed the following data: *uwb2* used a concatenation of the document titles and the last line of the document abstracts as input data, *uwb3* used the document titles as input data, *uwb4* used the last line of the document abstracts as input data, and *clairvoyant* used pre-calculated DICE co-efficients to find the best possible generated GeneRIF from a choice of document title, first line of document abstract, penultimate line of the document abstract, and last line of document abstract. The *clairvoyant* run is therefore a measure of the best possible result that the chosen generated GeneRIF selection technique could produce, but used data that would not be available to the system when it operates autonomously. Due to this, only runs *uwb2*, *uwb3*, and *uwb4* were submitted to TREC for evaluation. The use of the modified measure in the *clairvoyant* run is justified as the run produces the best standard DICE co-efficient results and the second best modified measure result.

The result the modified measure achieved (61.48%) indicates the better coverage of the input data of run *uwb2*. This is unsurprising, as the input data for this run was a concatenation of two strings; so the modified measure had more data to work with. However, incorrect data for run *uwb2* was sent to TREC, but this did not have an effect on the results for the classic DICE, the modified unigram DICE, or the modified measure in this run. Our own analysis of the corrected data gives the results shown in table C.2.

The best run using the standard DICE co-efficients was the Modified Unigram DICE co-efficient in run *uwb3*, which produced 48.25%. This result is interesting as it shows that document titles do contain pertinent data for generating GeneRIFs. However, this advantage is lost when document titles are combined with other parts of the document, such as the last line of the document abstracts in run *uwb2*.

Table C.1: Experimental Results

CO-EFFICIENT TYPE	Average %
uwb2	
Classic DICE	44.41
Modified Unigram DICE	44.07
Modified Bigram DICE	2.33
Modified Bigram Phrases DICE	1.80
Modified Measure	61.48
uwb3	
Classic DICE	46.48
Modified Unigram DICE	48.25
Modified Bigram DICE	29.53
Modified Bigram Phrases DICE	32.82
Modified Measure	42.74
uwb4	
Classic DICE	36.28
Modified Unigram DICE	35.21
Modified Bigram DICE	22.73
Modified Bigram Phrases DICE	24.52
Modified Measure	38.80
clairvoyant	
Classic DICE	58.16
Modified Unigram DICE	59.61
Modified Bigram DICE	45.04
Modified Bigram Phrases DICE	47.94
Modified Measure	54.43

Table C.2: Corrected Results for *uwb2*

CO-EFFICIENT TYPE	Average %
uwb2	
Classic DICE	44.53
Modified Unigram DICE	40.08
Modified Bigram DICE	29.20
Modified Bigram Phrases DICE	31.80
Modified Measure	-

C.4 References

J. Mork and L. Aronson. July 2003. medir.ohsu.edu/~genomics/protocol.html.

W. Teahan, 2003. "Knowing about knowledge: Towards a framework for knowledgeable agents and Knowledge Grids", Tech Report AIIA 03.2, School of Informatics, University of Wales, Bangor.

BioText Team Report for the TREC 2003 Genomics Track

G Bhalotia†, PI Nakov†, AS Schwartz†, MA Hearst§
†Computer Science Division
§School of Information Management and Systems
University of California Berkeley
Berkeley, CA 94720
{bhalotia, nakov, sariel}@cs.berkeley.edu, hearst@sims.berkeley.edu

Abstract

The BioText project team participated in both tasks of the TREC 2003 genomics track. Key to our approach in the primary task was the use of an organism-name recognition module, a module for recognizing gene name variants, and MeSH descriptors. Text classification improved the results slightly. In the secondary task, the key insight was casting it as a classification problem of choosing between the title and the last sentence of the abstract, although MeSH descriptors helped somewhat in this task as well. These approaches yielded results within the top three groups in both tasks.

1 Introduction

The paper reports on the work conducted by the BioText project team at UC Berkeley for the TREC 2003 Genomics track. In 2003 this track consists of two tasks and the document collection consists of 525,938 MEDLINE records dating between 4/1/2002 and 4/1/2003. Task 1 was intended to be similar to standard information retrieval queries and was stated as follows:

> For gene *X*, find all MEDLINE references that focus on the basic biology of the gene or its protein products from the designated organism. Basic biology includes isolation, structure, genetics and function of genes/proteins in normal and disease states.

The relevance judgements for this task were drawn from GeneRIF references from the National Library of Medicine's LocusLink database. Participants were allowed to make use of the gene name variation information associated with the GeneRIF.

The secondary task was intended to require more detailed analysis, in order to allow groups to make use of sophisticated language processing technology that is generally considered to be important for genomics and other bioscience text. However, due to limited resources, a specialized annotated collection was not yet available for this task. Instead, the goal of the secondary task was to reproduce the GeneRIF textual description for a given gene/document pair. Because these descriptions were often extracted verbatim from the document's title or abstract, and systems were judged on how closely the extracted text overlapped with the original, the task was better approached as a classification problem than as a language analysis and generation problem.

There were some commonalities in our approaches for the two tasks: in both cases we made use of classification algorithms and a special module for recognizing gene name variants and identifying MeSH descriptors in the text. Below approaches to both tasks are described in detail.

2 The Primary Task

2.1 Overview

The main challenges in the primary task are to improve recall by finding all appropriate variations of the given gene name, to improve precision by removing documents that describe genes that do not pertain to the target organism, and to demote the ranking of documents that mention the target gene but have not been assigned to a GeneRIF.

To improve recall, we created a special-purpose algorithm for generating and recognizing gene name variants. Our three-fold approach consisted of normalizing gene names by replacing special characters with spaces, developing a set of expansion rules that generate possible variants of the gene names that are not included in LocusLink, and looking in the citations for MeSH terms that pertain to gene names.

To improve precision we developed a semi-automated method to convert LocusLink organism names to MeSH organism descriptors, and used these to filter out papers that were not relevant to the target organism.

We submitted two runs, illustrated in Figure 1. For the first run, the relevance ranking ccmponent consisted of a weighted sum over 5 different sub-queries. For the second run, this score was combined with that of a statistical model that was trained to distinguish documents that are referred to by GeneRIFs from those that are not. We used as our backend retrieval system the IBM DB2 Net Search Extender, which allows convenient combination of relational and full-text queries.

In the following subsections we describe the modules for gene name recognition, organism filtering, MeSH term mapping, GeneRIF classification, and the method of combining the scores of the sub-queries.

2.2 MeSH Descriptors

In a number of places in the paper below we make use of MeSH (Medical Subject Heading)[1] lexical hierarchy.

In MeSH, each concept is assigned a unique identifier (e.g., *Eye* is D005123) and one or more alphanumeric tree numbers (corresponding to particular positions in the hierarchy). For example, A (*Anatomy*), A01 (*Body Regions*), A01.456 (*Head*), A01.456.505 (Face), A01.456.505.420 (*Eye*). Eye is ambiguous according to MeSH and has a second tree number: A09.371 (A09 represents *Sense Organs*).

In addition, each MeSH concept is assigned a semantic type (there are over 200): e.g., *Enzyme*, *Gene or Genome*, *Mammal*, *Tissue*, *Virus*, etc.

In some cases in the work below we use MeSH tree numbers and truncate them at period breaks to generalize across sub-hierarchies of the trees. In other cases we use the unique descriptor or the semantic type.

2.3 Identifying Variations in Gene Names

In order to capture the variations in gene names we had to expand the original synonym list from LocusLink since using only the gene synonyms available from LocusLink produced relatively less accurate results.

We created a semi-automated technique to identify such variations. We analyzed a large set of gene names to try to determine rules for converting a given representation into a canonical form. First we used n-gram matching to find candidate sets of similar gene names. Then we inspected the results of this matching to make a set of rules for such conversion (see Table 2). Some of these rules were more accurate and so were assigned more weight than the others. The details appear in the next two subsections.

N-Gram Overlap

The first step in identifying patterns of variation is to locate the variant form of the gene name in the article text. We automated this step by using an n-gram overlap measure [3]. "n-grams" are simply n-long strings of continuous characters in a given document/string. The distribution of n-grams between pairs of strings is compared, and a score is computed that represents the similarity between them. The main idea behind using n-grams is that similar words will have a high proportion of n-grams in common. Typical values for n are 2 or 3 corresponding to the use of digrams or trigrams, respectively.

To compute the similarity of two strings using this method, we first compute the n-gram sets for the strings being compared and then calculate the overlap using the Dice Coefficient [10]. The Dice coefficient $\mathcal{D}$ for two sets A and B of sizes $|A|$ and $|B|$ is given by

$$\mathcal{D} = \frac{2|A \cap B|}{|A| + |B|}$$

This overlap measure penalizes the presence of extra characters beyond the ones common to the two strings. Thus two strings with the same amount of overlap get a higher score when the non-overlapping regions are smaller in size.

We take the abstract and title of the articles associated with the genes and compute the n-gram overlap of all the possible subsequences of words against all known alias forms of the gene. We use character level digrams and trigrams with Dice Coefficient as the overlap measure. The word sequences in the abstracts/title that have high similarity to one of the known alias forms of the query gene are reported.

Inspection and Rule Generation

The procedure outlined above yields high similarity pairs of strings with one of them corresponding to the known alias form of a gene name and the other to the actual variant form of representation found in article text. We process this list to remove the

[1] http://www.nlm.nih.gov/mesh

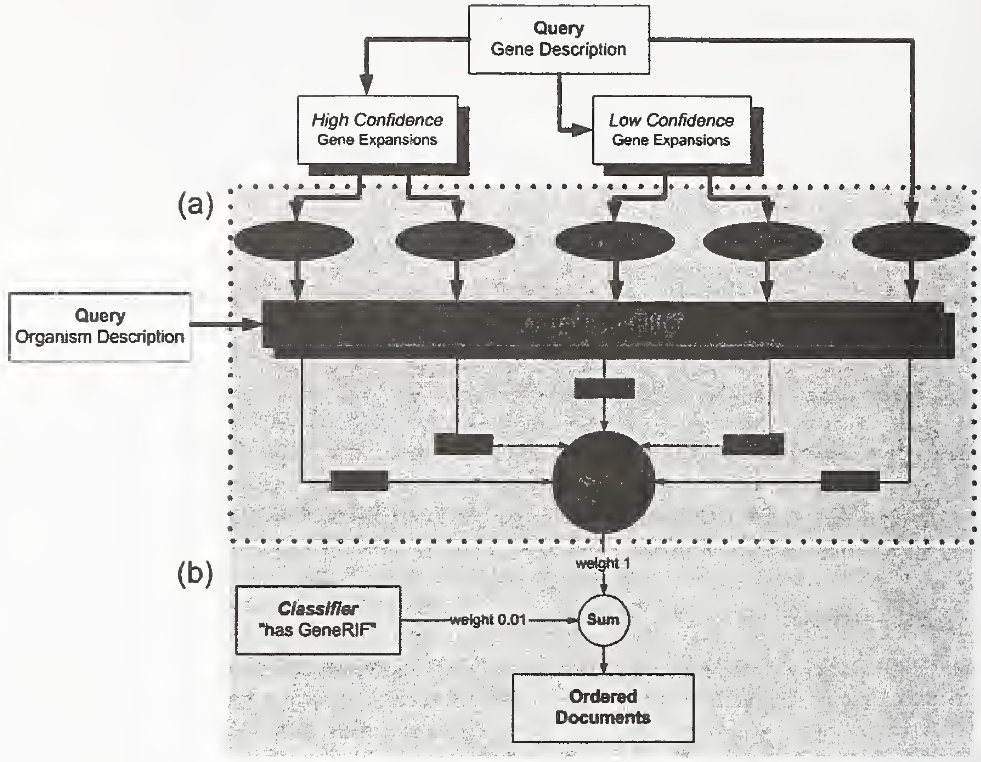


Figure 1: Software architecture for the primary task. Part (a) was used for both runs; part (b) for the second run only.

Known Alias Name	Best match variant in text
HLA-DQB1	hla-dqb
DNA synthesis inhibitor	inhibitors of dna synthesis
phospholipase C, gamma 1	phospholipase c gamma 1
adrenergic receptor, alpha 1d	alpha 1d-adrenergic receptor
Janus kinase 2 (a protein tyrosine kinase)	protein tyrosine kynase
golgi protein, 73-kD	golgi protein
luteinizing hormone/choriogonadotropin	luteinizing hormone–choriogonadotropin (lh/hcg)

Table 1: Some selected overlap pairs for gene names and their variants

ones that are exactly identical (note that identical strings will receive the highest similarity coefficient of 1). Next we remove those that lie below a threshold, that is obtained using quick inspection of the list (we used a threshold of 0.5). This yields a set of original forms and their variants. Selected overlap matches are shown in Table 1.

We inspect the pairs obtained to identify the patterns of variation in gene names. These patterns are used to generate rules to transform thc names to obtain a broader set of alias forms for the gene names. The rules that we generated are shown in Table 2. Such rules are syntactic in nature; sometimes there are variations of semantic nature that cannot be captured this way.

2.4 Organism Filtering

We filtered out documents that do not correspond to the organism that the query gene belongs to (note that each query in the TREC task consists of a gene name and a corresponding organism). Similar genes with the same name can occur in multiple organisms, e.g., the gene named c-myc which stands for "cellular myelocytomatosis oncogene" can be found in different organisms including humans and chickens. In humans it is located on chromosome 8 and is involved in the pathogenesis of Burkitt's lymphoma. In chickens, c-myc activation by avian leukosis virus appears to result in the development of lymphoid leukosis. Most of the time we are interested in documents that talk about the function of the gene corresponding to a given organism.

MEDLINE records do not contain an "organism annotation" for the documents. However this information can be inferred from the MeSH terms assigned to the article. For example, a document that talks about the fruitfly "Drosophila melanogaster" contains the MeSH term *Drosophila melanogaster.* However the organism names used in LocusLink usually do not match the corresponding MeSH term. For example, the term *Human* is used in MeSH instead of "Homo sapiens" used by LocusLink.

We used the combined information in LocusLink and MEDLINE to identify the descriptors used to characterize the organisms for MEDLINE documents. We collected the MEDLINE references (as described before, LocusLink has a set of references to MEDLINE documents relevant to the gene) for documents corresponding to each organism in LocusLink.[2] Each query produced a set of documents corresponding to a LocusLink organism. We then ran a query to compute what the top MeSH terms were for each set of documents.

By looking at the most frequent MeSH descriptors for each of the document classes, we can infer the term that is used to denote the organism in MEDLINE. We also checked if the LocusLink organism is used in MEDLINE name in full or partially for the same one; e.g., the terms *Drosophila* and *Caenorhabditis* also appear in the MeSH headings contained in MEDLINE. None of the other organism names appears in their original LocusLink form. The terms that we ultimately use to map the documents in the collection to organisms are shown in Table 3. Some of the documents map to multiple organisms and a few map to none.

2.5 Mapping Query Terms to MeSH

In order to further improve the system's performance, we also retrieved documents using their MeSH annotations. Both MeSH Main Headings and MeSH Supplementary Concepts (Chemical List) map to MeSH concepts. Each MeSH concept has one or more textual synonyms that are called MeSH terms. Given a query term the system retrieves all the documents that are annotated with a MeSH concept that has a MeSH term that exactly matches one of the gene names or one of its original synonyms (not including the expanded forms).

Adding MeSH mappings to the query helped mainly in ranking the retrieved documents. Documents that are retrieved by using the MeSH mapping in addition to the text search are more likely to be relevant, and therefore are given higher scores.

2.6 GeneRIF Classification Module

The goal of the classification module is to estimate the probability that a given document has been assigned to a GeneRIF. Our approach is based on the idea that articles which discuss gene function contain a distinct set of features which can be learned using automated techniques. The resulting models can be used to classify new documents.

Feature Selection

We experimented with a number of different feature sets; a comparison of their corresponding classification accuracy is presented in Figure 2. We compared (a) using MeSH descriptors as complete phrases, (b) using MeSH tree numbers, (c) using words in abstracts with stemming, and (d) using MeSH descriptors combined with tree numbers from levels one and two. The best results are obtained for MeSH descriptors used as complete phrases.

[2] LocusLink contains genes from eight different organisms.

A, B	$\Rightarrow$	$A\ B$ $B\ A$	*removal of comma* *rearrangement of tokens*
$A([n])$ $A\_[n]$ $A\text{-}[n]$	$\Rightarrow$	$A\ [n]$	*where* $[n]$ *is the set of numerals*
$A[n]$	$\Rightarrow$	$A\ [n]$	*addition of spaces (normalization of numerals)*
$A\ [n]$	$\Rightarrow$	$A[n]$	*removal of spaces (denormalization of numerals)*
$A(B)$	$\Rightarrow$	$A\ B$ A	*removal of parentheses* *removal of terms in parentheses*

Table 2: Rules for expanding gene names

Organism name from LocusLink	MeSH descriptor to look for
Bos taurus	Cattle
Caenorhabditis elegans	Caenorhabditis elegans, Caenorhabditis
Danio rerio	Zebrafish
Drosophila melanogaster	Drosophila melanogaster, Drosophila
Homo sapiens	Human
Human immunodeficiency virus 1	HIV-I
Mus musculus	Mice
Rattus norvegicus	Rats

Table 3: MeSH terms used to map documents to organisms

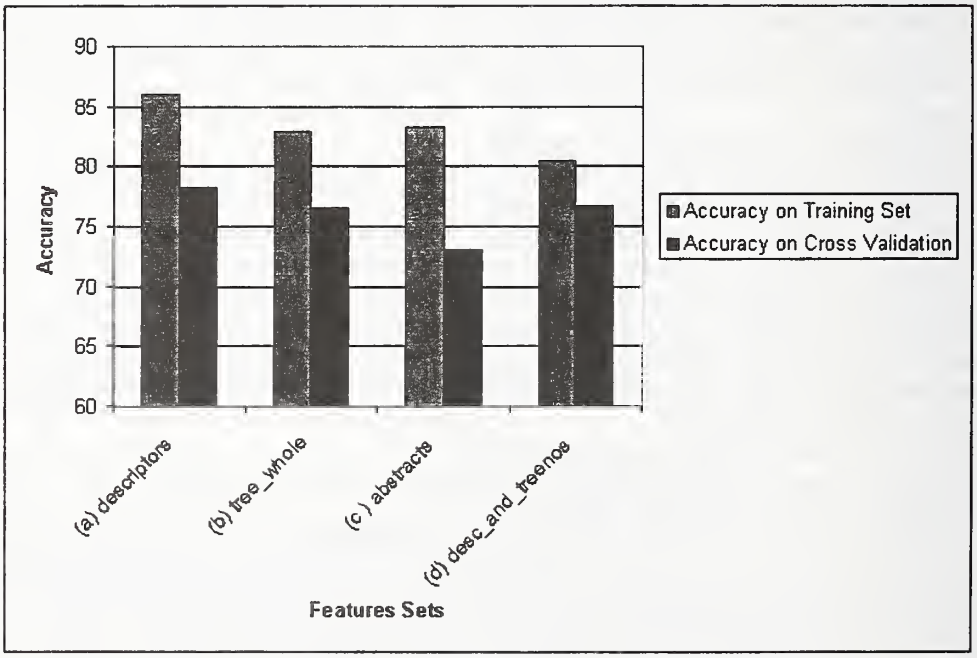


Figure 2: Classification accuracy for different feature sets: (a)descriptors as phrases (b) Whole tree numbers (c) Abstracts cleaned and stemmed (d) descriptors together with tree numbers

Model Training

Once the feature vectors are obtained, we train a classifier to build a model that can predict whether a document talks about gene function. We use a Naive Bayes classifier [5, 9]. Its fundamental idea is the assumption that the values of the feature variables $F = (F_1, F_2, ..., F_n)$ are conditionally independent given the class variable S. The joint probability is given by the expression:

$$p(S,F) = p(S)\prod_{i=1}^{N} p(F_i|S) \qquad (1)$$

The model parameters are given by the probabilities $p(S)$ and $p(F_i|S)$, which are usually estimated from the text by means of maximum likelihood estimates (MLE). The classification of a new concept is determined by the most likely category:

$$S_{ML} = \arg\max_{S_k} p(S_k|F) \qquad (2)$$

Naive Bayes classifiers are among the most successful algorithms for document classification. The Naive Bayes classifier is known to be optimal when attributes are independent given the class, but Domingos et al. [4] show that it will often outperform more powerful classifiers for common training set sizes and numbers of attributes even if the independence assumption is not met.

The implementation of the Naive Bayes classifier we currently use is part of the open source machine learning package WEKA (Waikato Environment for Knowledge Analysis [1, 11]) from the University of Waikato, New Zealand. They provide excellent Java implementation of Naive Bayes and other machine learning algorithms.

One model was trained for a set of 50 gene names that are not part of the TREC training or test set. We used the retrieval module to extract the relevant documents for each target gene, and the first 1000 documents of each query as the training set.

2.7 Document Ranking

As mentioned above, we used IBM DB2 Universal Database to store MEDLINE documents including the abstracts, titles and other annotations. We have built text indexes on these fields using DB2 Net Search Extender, which is then used to search for documents that contain a given set of terms.

We retrieve all the documents that match one of the known alias forms of the gene or the variations created using the expansion rules (variant forms are generated for all the aliases). The query against the database is composed of five sub-queries combined with the SQL UNION operator, as follows.

Let G be the various forms of the gene name as computed by the conversion rules shown in Table 2 and let LG be other lower-confidence rules for normalizing the gene names that have a higher rate of false positives. For the first run, the score is computed as follows:
Score(R) = the aggregated SUM over the result of the UNION operator GROUP BY document id of:

(a) J * (G compared to terms in titles)
(b) J * (LG compared to terms in abstracts)
(c) K * (LG compared to terms in titles)
(d) K * (LG compared to terms in abstracts)
(e) L * (MeSH concepts compared to MeSH terms assigned to documents)

where $J = 1, K = 0.015$, and $L = 1.4$ (determined experimentally on training data).

As shown above, the scores of the documents in each sub-query are weighted and then aggregated using the UNION operator and the SUM aggregate function. We experimented with using the MAX aggregate function instead, but the results obtained using the SUM function were substantially better. This is due to the fact that documents that are retrieved in multiple sub-queries get a higher total score, and are in fact more likely to be more relevant to the query. We also experimented with giving higher weights to titles over abstracts, but this did not appear to help. Increasing the weights of specific types of aliases (official terms for example) did not improve the system's performance either.

For the second run, in order to combine the retrieval scores and the classification scores we normalized the weighted scores for each query into values between 0 and 1 by dividing the score by the highest document score of the query. The combined retrieval-classification score is a weighted sum of the two scores. We used 1 and 0.01 as the weights for the retrieval and the classification scores respectively.

The retrieval score is a value between 0 and 1 for each document and is obtained in part by combining the frequency of occurrence of the term in the document and the relative size of the retrieved document. The exact details of the scoring function are not available as they are part of the DB2 proprietary system.[3]

[3] However, we have been told via personal communication from James Cooper of IBM, reporting information from Roy Byrd of IBM, that the algorithm is based on the Guru ranking algorithm [8], which is a Bayesian computation of a document's probability of being relevant to the query, with lexical affinities mixed in.

2.8 Evaluation

We submitted two runs for the primary task. The first run uses only the retrieval module, and the second run combines it with the classification module.

When training our system we noticed that some of the topics in the training set did not have any GeneRIFs associated with them. We therefore removed them from the training topic list. Also, some correct GeneRIFs were not listed in the list of qrels. After fixing these errors our system achieved 0.5028 mean average precision (MAP) on the training set with the retrieval module alone, and 0.5101 with the modules combined. The classification module helps but not dramatically.

On the test set our system achieved 0.3753 MAP with the retrieval module alone, and 0.3912 MAP using both modules combined. Again, the classification module helps but not markedly. In 12 out of 50 queries the retrieval module alone achieves MAP higher than the combined modules.

The big gap in performance between the test and training sets suggests that the system parameters might be over-fitting the training set. However, an initial sensitivity analysis of the system performance on the test set, shows that only a minor improvement can be achieved by tuning the parameters to fit the test set. Another explanation for the performance gap might be that the test set is inherently harder than the training set. This hypothesis is in agreement with the analysis presented in [6], which shows a high degree of variation in MAP across topics in general, and between the training and test sets in particular.

Both our runs fell within statistical significance of the top performing group [6]. In 43 out of 50 queries the MAPs of both runs were higher then the median MAP. Analysis of the 7 remaining queries yields some interesting insights about possible improvements of our system. In 3 out of 7 queries the low MAP was a result of a low recall. This is mainly due to some limitations of the current gene name expansion rules. For example, in query 37 the system retrieved only 36 relevant documents out of the possible 61. This is due to the fact that one of the query terms was *peroxisome proliferative activated receptor gamma* while in many relevant documents it appears as *peroxisome proliferative activated receptor gamma isoform 1*, and another query term, *PPAR gamma*, appears in the text as *PPARgamma*. Additional low-confidence expansion rules could improved the MAP in these cases without affecting the performance of other queries. In many cases the system retrieved much less than the allowed 1000 documents. In these cases, recall could also be improved by retrieving additional documents with low ranking scores like the ones that were filtered out by the organism-filter, or documents that could be retrieved using single query terms instead of full phrases. Of the other 4 sub-par queries, 1 was due to a small bug in the implementation of one of the high-confidence expansion rules that resulted in the addition of an over-generalized term with a high weight into the query, and the other 3 were due to sub-optimal rankings.

3 The Secondary Task

3.1 Overview

For the secondary task, our initial intention was to try a linguistically motivated approach but we soon realized that the data was too noisy due to a lack of clear definition of what a GeneRIF is. Instead, we addressed it as a text classification problem.

Our investigations showed that most of the time the GeneRIF text was pulled verbatim, or with slight modifications, from the abstract text or title. Most of the time the extract came from the title and in the majority of the remaining cases – from the last sentence of the abstract. Thus we assumed the baseline was always choosing the title since it was quite difficult to find an algorithm that performed better than this.

After much experimentation, we ended up training a Naive Bayes classifier that, given an abstract text, predicts whether the last sentence or the title is a better candidate for GeneRIF text. Our feature set was limited to verbs, MeSH terms (cut at level 2, e.g. G14.330), genes (a single feature for the frequency of all genes), all weighted by TF.IDF, and the appearance of the target gene (a Boolean feature). For training we focused on the abstracts coming from the 5 target journals that were announced on the genomics track Web site (about 6,500 abstracts in total), split them into 10 sets and performed a stratified 10-fold cross validation.

3.2 GeneRIF Mapping into the Abstract Text

Looking at the GeneRIFs we found that most of the time the GeneRIF text was pulled verbatim, or with slight modifications, from the abstract text. To quantify this, we investigated 33,662 MEDLINE abstracts that had a GeneRIF assigned and we tried to find a substring in the text that is most similar to the GeneRIF description. Given a particular abstract, we considered all possible sequences, respecting the

Run	Modified Unigram Dice
In title	35.17%
In abstract	45.81%
In both	4.98%
In last sentence	20.63%
Exact title match	19.67%
Total matched	76.02%

Table 4: Finding the best mapping of the GeneRIF text against the corresponding abstract.

word and sentences boundaries, and for each one we calculated the *Modified Unigram Dice* (*MUD*) score. We accepted a mapping as successful if the score was above some threshold.

When the MUD threshold was set to 80%, a successful mapping for 25,590 of the documents (76.02%) could be obtained directly from the title and/or abstract. For 11,847 (35.17%) of them an acceptable match was a substring of the title (and for 6,620 (19.67%), the whole title was taken verbatim: this is 65.10% of the cases when the mapping was found in the title). In 15,421 documents (45.81%), the best match was inside a sentence from the abstract body, and in 1,678 (4.98%) it was found in both the title and the body. In 6,943 of the cases (20.63%) the best match was found in the last sentence of the abstract (this is 50.52% of the cases when it was found in the abstract body). (Note that there is always further opportunity to truncate some unused part of the sentence and improve the score.)

Given the fact that for most of the abstracts an acceptable matching was found in the title and that in 65.10% of them the best match was the whole title taken verbatim, an obvious baseline was "pick the title". This resulted in a MUD score of 53.39%.

The last sentence of an abstract usually summarizes its contents, so it was not surprising that it often contained the best match. The title and the last sentence together account for 73.40% of the matches that pass the threshold. Thus, an algorithm that chooses between them would have the potential to perform better than always choosing the title. We calculated that if we limited the choice to title or last sentence and always selected the one that leads to a higher score (we select the *whole* last sentence or the *whole* title), this would result in a MUD score of 66.33%. This is the upper bound for any algorithm that relies on whole sentences and chooses between the title and the last abstract sentence only. In practice this algorithm may not perform better than the "pick the title" baseline since it may choose incorrectly. We calculated that if it always made the wrong choice it would end up with a MUD score of 26.62%.

We performed similar calculations using as similarity measures *Classic Dice* (*CD*), *Modified Bigram Dice* (*MBD*), *Modified Bigram Dice Phrases* (*MBDP*) as well as a combination of MBU and MBD. The results were all similar, although sometimes the best choices under the different scores differed. We also performed a more general mapping that allowed the target GeneRIF text description to split into two parts, each of which can be mapped to two different parts of the abstract text. Although this way we found better matches for some of the GeneRIFs, the impact was limited: 77.10% matched as compared to 76.02% for the case when a single string was allowed.

3.3 The Features

We experimented with a number of different features, including: *words/stems*, *verbs* (the most frequent ones only: e.g. *bind*, *block*, *inhibit*, *accept*, *involve* etc.; they are stemmed so nominalized verbs are considered as well: e.g. *inhibition*), *genes*, *genes_freq* (frequency: how many gene names are mentioned in the sentence), *MeSH_unique_ID* (e.g. *D005796*), *MeSH_tree_number* cut at a certain level (level 1: *G14*, or level 2: *G14.330*), *MeSH_semantic_type*, *journal*, *publication_date* (month and year taken together, e.g. *10_2003*). We used also three Boolean features: *target_gene* (is the target gene mentioned?), *is_title* (is the current sentence the title?), *is_last_sentence* (is this the last sentence?). The features were weighted according to the TF.IDF measure (except for the Boolean ones). We also experimented without weighting (i.e. using the raw frequency information) as well as treating all features as Boolean.

The *journal* and *publication_date* features were introduced in order to account for possible journal- or time-dependent regularities, but these did not prove useful. The *MeSH_semantic_type* features were too general. The *words* and *stems* lead to a dramatic increase in the vector space dimensionality, which made training some particular classifiers intractable so we did not use them. The same applies to *genes_freq* and *MeSH_unique_ID*.

The best combination of features was (we will refer to it as *the standard feature set* below): *verbs*, *genes_freq*, *MeSH_tree_number* (cut at level 2), *target_gene*, *is_title* and *is_last_sentence*. The three Boolean features were especially important, while the impact of *genes_freq* was minor. The non-Boolean features were weighted using TF.IDF.

3.4 Choosing the Title vs. the Last Sentence

For the classification experiments we used the WEKA Machine Learning Software in Java again. We used mainly Naive Bayes with kernels [7] but tried several others classifier as well. Decision trees were helpful to identify the useful features: e.g. the MeSH terms turned up often.

As mentioned above, we addressed the problem as a text classification task at the sentence level, and ended up stating it as a choice between the title and the last sentence. Using the *standard feature set*, we trained a Naive Bayes classifier that was able to distinguish between when the best sentence is a title vs. a non-title with an accuracy of 81.22%, as measured on a stratified 10-fold cross-validation on a corpus of 4,000 GeneRIF texts.

We wanted to extend this idea in the direction of a two-step classification: the first classifier chooses between title and non-title. In case non-title is chosen, then a second classifier chooses the best abstract sentence (here the most important feature is *is_last_sentence*). The second classifier is to be trained on non-title sentences only. Unfortunately, comparing the title to the abstract body was problematic due to substantial length differences. We decided to simplify the things further and compare the title to the last sentence only.

We trained a Naive Bayes classifier with kernels that, given a gene and a document, chooses between the title (*class A*) and the last sentence (*class B*). We used the *standard feature set*, but without the *is_title* and *is_last_sentence* features. In order to label the training examples as belonging to class A/B we compared the MUD overlap of the target GeneRIF text with both the title and the last sentence and assigned the label A or B depending on which get a higher score. We then concatenated the title and the last sentence and extracted the features from the resulting string. So, each abstract produced a single example labeled either A or B.

We tried marking the features (e.g. MeSH terms) to indicate whether they came from the title or from the last sentence in order to allow the classifier to distinguish between them, but this lead to decreased performance and so was dropped. Finally, we limited the training to the abstracts coming from the 5 target journals that were announced on the genomics track Web site: about 6,500 abstracts in total. We split them into 10 sets and performed a stratified 10-fold cross-validation.

3.5 Evaluation

We performed several experiments for the different formulations of the problem in order to find the best feature set and the best classification algorithm using stratified 10-fold cross-validation and collections of different sizes: 343, 1000, 2000, 5000, 10000, 20000, 33662 etc. Then we fixed the feature set (the *standard* feature set) and the classifier (Naive Bayes; class A/B) and concentrated on a set of 6,500 abstracts that have GeneRIFs and come from the 5 journals. We ran series of experiments in order to find the best thresholds for feature selection.

The baseline results of always choosing the title for the various scoring metrics are shown on line 1 of Table 5. For the training set cross validation runs, the best results were obtained for minimum verb frequency of 5 and minimum MeSH tree number frequency of 12 (see line 2 of Table 5).

For the TREC run we trained on all the abstracts from the 5 journals, except the 139 ones used for testing. We used the feature thresholds found above (5 for verbs and 12 for MeSH tree numbers) and we obtained the results shown on line 3 of Table 5. Although these are lower than those of line 2, they are still about 3–4% above the baseline shown on line 1.

We also calculated the best possible score we could have obtained if our algorithm had always made the correct choice between the title and the last sentence (see line 5 of Table 5). If we had tuned the parameters to the test set, we would have used minimum verb frequency of 12 and minimum MeSH frequency of 11 and gotten the results shown in line 4 of Table 5. (Of course, we cannot use the test set to compute the thresholds for the TREC run.)

Finally, our scores 57.83%, 59.63%, 46.75%, 49.11% (for CD, MUD, MBD and MBDP) were the second best ones after those of Erasmus University (emc4): 53.04%, 54.65%, 38.62%, 41.17, who used a similar classification technique but managed to successfully train a classifier to choose among *all* sentences, not just between the title and the abstract. In fact, emc4 was the only other group that was able to beat the baseline. However, according to [6], neither of these results represents a statistically significant improvement over just using the titles.

4 Discussion

4.1 Primary Task

It appears that the definition of GeneRIF is quite fuzzy. This limits the potential contribution of our classification module. However, we believe that in

	Run	CD	MUD	MBD	MBDP
1	Baseline	50.47%	52.60%	34.82%	37.91%
2	Cross Validation	58.06%	59.11%	44.74%	47.29%
3	TREC run	53.04%	54.65%	38.62%	41.17%
4	Tuned thresholds	54.88%	56.66%	40.66%	43.31%
5	Upper bound	61.72%	64.19%	50.88%	54.00%

Table 5: TREC run result compared to baseline, best possible result and a better features selection.

cases where the subset of relevant documents could be defined more precisely combining such a classifier with a more traditional IR system could result in a significant boost in performance.

In order to further improve the performance of our system, semantic information has to be incorporated. In the future we plan to add syntactic and semantic annotations to the text in order to support much more powerful algorithms for information retrieval and extraction from bioscience literature.

4.2 Secondary Task

Better results could potentially be obtained with a careful feature selection algorithm [12]: all we do at present is to remove the least frequent features and our algorithm is very sensitive to setting the correct threshold (compare lines 3 and 4 in Table 5). This would allow the introduction of some carefully selected stems as features without a dramatic increase of the vector space dimensionality, and can account for predictive phrases of the kind: *our results show that ....* In addition, although good, our gene and MeSH tagging utilities are not perfect. MeSH ambiguity is another source of problems and the verb nominalization introduces some noise as well.

A promising improvement would be a more careful truncation of the unnecessary part of the selected sentence. It would be also interesting to try some specialized algorithm that tries to learn a ranking directly (e.g. [2]) as opposed to the classification approach described above.

Finally, it would be good to have a similarity measure that takes into account the semantics, e.g., not penalize those cases in which a synonym name for the same gene is used.

References

[1] Waikato environment for knowledge engineering. http://www.cs.waikato.ac.nz/ml/weka/.

[2] Crammer, K., and Singer, Y. A new family of online algorithms for category ranking. In *25th annual international ACM SIGIR conference on Research and development in information retrieval* (2002), pp. 151–158.

[3] Damashek, M. Gauging similarity with n-grams: Language-independent categorization of text. In *Science* (1995), vol. 267, pp. 843 – 848.

[4] Domingos, P., and Pazzani, M. Beyond independence: Conditions for the optimality of the simple bayesian. In *International Conference on Machine Learning, ICML* (1996).

[5] Duda, R., and Hart, P. Pattern classification and scene analysis, 1973.

[6] Hersh, W., Bhupatiraju, R., Kraemer, D., Corley, S., Aronson, A., Mork, J., Hearst, M., Nakov, P., and Mitchell, J. Enhancing access to the bibliome: The trec genomics track, 2004. In preparation.

[7] John, G., and Langley, P. Estimating continuous distributions in bayesian classifiers. In *Eleventh Conference on Uncertainty in Artificial Intelligence* (1995), Morgan Kaufmann, pp. 338–345.

[8] Maarek, Y., and Smadja, F. Full text indexing based on lexical relations. In *Proceedings of the 12th International ACM SIGIR Conference on Research and Development in Information Retrieval* (1989), pp. 198–206.

[9] Mitchell, T. *Machine Learning.* McGraw Hill, 1997.

[10] van Rijsbergen, C. *Information Retrieval*, 2nd ed. Butterworth-Heinemann, 1979.

[11] Witten, I., and Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann, 1999.

[12] Yang, Y., and Pedersen, J. A comparative study on feature selection in text categorization. In *Proceedings of the Fourtheenth Intenational Conference on Machine Learning* (1997).

Edinburgh-Stanford TREC 2003 Genomics Track: Notebook Paper

Miles Osborne*, Jeffrey Chang†, Mark Cumiskey*, Nipun Mehra†,
Veronica Rotemberg†, Gail Sinclair*, Matthew Smillie*,
Russ B. Altman†, Bonnie Webber*

* School of Informatics, University of Edinburgh
{mcumiske,miles,csincla1,bonnie}@inf.ed.ac.uk
m.b.smillie@sms.ed.ac.uk

† Department of Genetics, Stanford University
{russ.altman,jeffc,nmehra}@stanford.edu
vmr@smi.stanford.edu

Abstract

We describe our participation in both tasks in the 2003 TREC Genomics track. For the primary task we concentrated mainly upon query expansion and species-specific document searching. An analysis of the variance of possible retrieval results suggested that the official TREC-supplied test set is only a crude approximation of the true system performance. The secondary task we treated as an extraction problem, using a maximum entropy scorer trained on GeneRIF sentences as positives and other sentences as negatives. While our results were not always equivalent to the actual GeneRIFs, on biological grounds many of them appeared better descriptors than the GeneRIFs themselves.

1 Introduction

The School of Informatics at the University of Edinburgh, Stanford's Center for the Study of Language and Information, and the Laboratory for Informatics in the Department of Genetics at Stanford have an on-going collaboration, aimed at improving access to and use of the biomedical literature. In connection with this collaboration, a group of staff and students at the two institutions decided to mount a joint team to participate in the first TREC-Genomics track. Here we outline how we went about dealing with the two tasks.

2 Primary Task: Ad-hoc Document Retrieval

2.1 Partitioning the document set and indexing

Because many species have genes with the same name, we decided to try to eliminate "wrong species" as a source of false positives, by only searching species-specific sub-collections of the original document set (525,938 documents). We produced these species-specific sub-collections (one for each "official" TREC species: *human* (315,714 documents, 60% of the collection), *rat* (31,859 documents, 6% of the collection), *mouse* (28,280 documents, 5.4% of the collection), and *drosophilia* (8,377 documents, 1.6% of the collection), as well as one for *any* "official" TREC species (373,906 documents, 71.1% of the collection)) through a combination of intuition and machine learning. Specifically, we used Rainbow's classification system [1], choosing as classification features, those that were thought to be indicators of species. These included both obvious features (e.g. the MeSH term *Human* as an indicator of human), but also some idiosyncratic ones (e.g. the MeSH term *Predisposition to Disease* also as an indicator of human). Both obvious and idiosyncratic features were selected through careful analysis of the training qrels. Features indicative of other species (e.g.*yeast, frog*) were also used to help classify documents as **not** being about an "official" species. False positives were also avoided by excluding matches against MedLine fields that might contain distracting information (e.g AD - Institutional affiliation and address). In the end, we only considered the title, abstract, MeSH terms and registry number fields of individual documents in creating these species-specific and *all four species* sub-collections.

The 325 relevant documents (qrels) specified in the official TREC training data were used as the training data for species classification, as was an equal number of documents that were about "non-official" species (i.e. *bos taurus*, *danio rerio* and *c. elegans* - obtained through Locuslink). A separate binary classifier was built for each "official" species, as well as one to classify whether a document is about **at least one** of them. To further eliminate possible false positives from the search space, features were added that would allow us to exclude documents that were considered *non-genetic* and therefore unlikely to contain a GeneRIF. The resulting classification accuracy is shown below for a locally-developed held-out test dataset:

Species	Accuracy (%)
Human	88.05
Mouse	89.51
Rat	88.80
Drosophila	98.45
All 4	96.46

We also tried using a variety of features and Medline fields to see if we could improve on these results, but none were found that would increase accuracy in the majority of cases.

Another possibility would have been to simply classify species on the basis of their being indexed by the appropriate MeSH term for the species (e.g. query = *Homo Sapiens*: MeSH term = *Human*). While this may have produced a slight increase in classification accuracy (and therefore a slight increase in the document retrieval recall), it would also have substantially increased the search space, which can lead to a substantial decrease in the document retrieval precision. By restricting our search to species-specific sub-collections, we may pay the price in "false negatives" – relevant documents that are missing from the sub-collection because they lack sufficient features to have been included.

The *all 4 species* sub-collection was intended, in part, to deal with this problem in a fall-back strategy, where our initial retrieval using the individual species sets resulted in poor performance. Time, however, denied us the opportunity to research this thoroughly. One technique that was applied was where the first n (500, 700, 800) documents retrieved using the species specific set were retained and the first 1000-n documents retrieved (distinct from the n documents) using the all 4 species document

[1] http://www-2.cs.cmu.edu/~mccallum/bow/rainbow/

set were tagged on to make up the 1000 documents. This increased our total recall scores. However, this increase was not deemed justifiable for the resultant decrease in precision.

2.2 Query expansion

The recent surge in IR and IE within biological text has highlighted the inconsistency of the terminology within the text. Genes can be referred to in a variety of different ways: the full name, an acronym derived from the full name, by the name of a homologous gene, the resultant product of the gene. Even Locus Link itself is not complete and does not specify all synonymous names of a gene. Conversely, more than one distinct gene can be referred to with the same handle. Thus, for more effective retrieval, we looked at ways to expand our query. Synonyms from genetic databases were sought to complement the set from LocusLink. Analysis of the training queries and their corresponding qrel documents showed other discrepencies within gene symbols. For example, while the same letters may be used, they may differ in case (upper vs lower). Punctuation characters may be added, as well as different representations of the same numeric specifiers (e.g. 2=II=beta). These issues also had to be taken into consideration when formulating our query.

At run-time, the original, TREC-supplied queries were first processed into *unexpanded queries*, with the same format as the queries in Hersh's third preliminary run. Figure 1 shows an example of a TREC-supplied query.

1	1026	Homo sapiens	OFFICIAL_GENE_NAME cyclin-dependent kinase inhibitor 1A (p21, Cip1)
1	1026	Homo sapiens	OFFICIAL_SYMBOL CDKN1A
1	1026	Homo sapiens	ALIAS_SYMBOL P21
1	1026	Homo sapiens	ALIAS_SYMBOL CIP1
1	1026	Homo sapiens	ALIAS_SYMBOL SDI1
1	1026	Homo sapiens	ALIAS_SYMBOL WAF1
1	1026	Homo sapiens	ALIAS_SYMBOL CAP20
1	1026	Homo sapiens	ALIAS_SYMBOL CDKN1
1	1026	Homo sapiens	ALIAS_SYMBOL MDA-6
1	1026	Homo sapiens	PREFERRED_PRODUCT cyclin-dependent kinase inhibitor 1A
1	1026	Homo sapiens	PRODUCT cyclin-dependent kinase inhibitor 1A
1	1026	Homo sapiens	PRODUCT cyclin-dependent kinase inhibitor 1A
1	1026	Homo sapiens	ALIAS_PROT DNA synthesis inhibitor
1	1026	Homo sapiens	ALIAS_PROT CDK-interaction protein 1
1	1026	Homo sapiens	ALIAS_PROT wild-type p53-activated fragment 1
1	1026	Homo sapiens	ALIAS_PROT melanoma differentiation associated protein 6

Figure 1: Original TREC-supplied query

This TREC-supplied query was transformed into the unexpanded query format shown in Figure 2.

> "CDKN1A"$^{2.9}$ "P21" "CIP1" "SDI1" "WAF1" "CAP20" "CDKN1" "MDA-6" "cyclin-dependent kinase inhibitor 1A" "cyclin-dependent kinase inhibitor 1A" "cyclin-dependent kinase inhibitor 1A" "DNA synthesis inhibitor" "CDK-interaction protein 1" "wild-type p53-activated fragment 1" "melanoma differentiation associated protein 6"

Figure 2: An unexpanded query

This unexpanded query uses the Lucene retrieval engine syntax [2]. Terms in quotations must match as a unit, whilst the $^{2.9}$ modifier is an ad-hoc method for boosting the importance of matching against the associated term. The entire set of terms is taken as a disjunction. For this example, since the TREC supplied query indicated that the relevant gene belonged to *homo sapiens*, we used all documents in our *human* sub-collection of the MEDLINE documents when retrieving documents.

We used different query expansion strategies in our two runs: one emphasised precision over recall, and the other tried to recover more documents, with a possible decrease in precision. In the precision-optimised run:

- Parentheticals in the official gene name or a protein product were extracted and appended to the end of the unexpanded query.

- Conjunctions of two words preceding and two words following a hyphen were also appended to query, provided that none of the words were common English. For example, given the query term *CDK-interaction protein 1*, where at least two words follow the hyphen, the term *("CDK" && "interaction protein")* is appended to the query. (&& is Lucene syntax for logical "and".) Applying this method for punctuation marks other than hyphenation was not found to be successful.

- Acronyms were generated for terms that had more than three words. Again, such acronyms were simply appended to the set of terms.

- Numeric specifiers were removed and the resultant term appended.

- Non-digit specifiers (such as Roman numerals) were converted to digits and vice versa. Again, these extra terms were appended to the end of the query.

- Lowercase and uppercase versions of all terms were also appended.

Our recall-optimised run started with the same query expansion approach as when emphasising precision. In addition though, we retrieved from SwissProt the official gene, associated gene and proteins synonyms for the relevant species. These were appended to the query.

Query expansion of the query shown in Figure 2 results in the much larger query shown in Figure 3, in which the original query terms are followed by upper-case variants, extracted parentheticals, variants with hyphens removed, lower-case variants, etc.

We also tried expanding the query using GO terms, throwing-away common English (non-biological) words, stemming etc, but none of these led to any improvement in recall and/or precision.

After expansion, a query was sent to the Lucene retrieval engine, and all retrieved documents (from a given document sub-collection) were used in our submission. As a cross-check, we also tried the Lemur retrieval engine (http://www.cs.cmu.edu/~lemur) and obtained similar results. This showed that our system performance was not simply a consequence of using Lucene.

2.3 Results

On the 50-document development set, we obtained the following results using the query expansion strategy that concentrated on obtaining high-precision results (at the expense of possibly not retrieving some documents).

[2] http://jakarta.apache.org/lucene

”cyclin-dependent kinase inhibitor 1A (p21, Cip1)”$^{2.9}$ ”CDKN1A”$^{2.9}$ ”P21” ”CIP1” ”SDI1” ”WAF1” ”CAP20” ”CDKN1” ”MDA-6” ”cyclin-dependent kinase inhibitor 1A” ”cyclin-dependent kinase inhibitor 1A” ”cyclin-dependent kinase inhibitor 1A” ”DNA synthesis inhibitor” ”CDK-interaction protein 1” ”wild-type p53-activated fragment 1” ”melanoma differentiation associated protein 6” ”CYCLIN-DEPENDENT KINASE INHIBITOR 1A (P21, CIP1)”$^{2.9}$ ”CDKN1A”$^{2.9}$ ”P21” ”CIP1” ”SDI1” ”WAF1” ”CAP20” ”CDKN1” ”MDA-6” ”CYCLIN-DEPENDENT KINASE INHIBITOR 1A” ”DNA SYNTHESIS INHIBITOR” ”CDK-INTERACTION PROTEIN 1” ”WILD-TYPE P53-ACTIVATED FRAGMENT 1” ”MELANOMA DIFFERENTIATION ASSOCIATED PROTEIN 6” ”P21, CIP1” ”CYCLINDEPENDENT KINASE INHIBITORA” ”cyclin-dependent kinase inhibitor 1a (p21, cip1)”$^{2.9}$ ”cdkn1a”$^{2.9}$ ”p21” ”cip1” ”sdi1” ”waf1” ”cap20” ”cdkn1” ”mda-6” ”cyclin-dependent kinase inhibitor 1a” ”dna synthesis inhibitor” ”cdk-interaction protein 1” ”wild-type p53-activated fragment 1” ”melanoma differentiation associated protein 6” ”p21, cip1” ”cyclindependent kinase inhibitora” ”CDK 1C” ”CDK 1” ”CIP1” ”WTPAF1” ”MDAP6” ”cyclindependent kinase inhibitorA (p, Cip)” ”cyclindependent kinase inhibitorA” ”cyclindependent kinase inhibitorA” ”cyclindependent kinase inhibitorA” ”CDKinteraction protein” ”wildtype pactivated fragment” ”melanoma differentiation associated protein” ”cyclindependent kinase 1A (p21, Cip1)”$^{2.9}$ ”MDA6” ”cyclindependent kinase 1A” ”CDKinteraction protein 1” ”wildtype p53activated fragment 1” ”cyclin-dependent kinase 1A (p21 Cip1)”$^{2.9}$ ”p21 Cip1” (”1A p21” && ”Cip1”) (”p21 p21” && ”Cip1”)

Figure 3: An expanded query

Retrieved (out of 331)	185 (55.9%)	280 (84.6%)	265 (80.1%)
Average precision	0.3523	0.3939	0.4870
Exact	0.2959	0.3330	0.4711

The first column shows the results without expanding the query; the second column shows the results after expanding the query. The final column shows the results after using the expanded query and the relevant species-specific document sub-collections.

On the official 50-document test set, we obtained the following results, with column labels the same as above:

Retrieved (out of 566)	360 (63.6%)	523 (92.4%)	495 (87.5%)
Average precision	0.1625	0.1870	0.2888
Exact	0.1458	0.1584	0.2636

As can be seen, using our species-specific document sub-collections leads to a significant gain in precision (54.4%), offset by a small drop in recall (5.4%).

We had mostly similar results using our expansion strategy aimed at increasing recall. The development set produced the following results:

Retrieved (out of 331)	185 (55.9%)	331 (100%)	314 (94.9%)
Average precision	0.3523	0.3838	0.4661
Exact	0.2959	0.3342	0.4340

On the official test set, our results were as follows:

Retrieved (out of 566)	360(63.6%)	501(88.5%)	533(94.2%)
Average precision	0.1625	0.1609	0.2690
Exact	0.1458	0.1385	0.2275

The latter figures show that the addition of terms from SwissProt only improved overall performance when applied to the much small species-specific sub-collections. This leads us to conclude that "profligate" query expansion can pay off in the context of "intelligently" targetting the retrieval set.

2.4 (Failed) Attempts at Re-ranking

Information retrieval and question-answering systems often use *re-ranking* techniques, with a view to promoting the rank of relevant documents. We first tried re-ranking the retrieved documents using a maximum entropy model and purely lexico-syntactic features. To give an indication of how well this might work, Hersh's third set of results (http://medir.ohsu.edu/~genomics/preliminary.html), with a mean average precision of 0.30 could be boosted to a mean precision score of around 0.60. Unfortunately, we could not discover a set of lexico-syntactic features which could capture the notion of relevance when applied to an unseen set of documents and associated queries. We think there may be two reasons for this: First, the relevance of a document to a particular query appears not to be obviously encoded in the document. Secondly, there is considerable variation in how relevant documents correctly match against a query, and no way of obvious general, systematic way to re-rank them.

We then hypothesized that re-ranking documents based on some "function score" would lead to better results. Here we tried three methods, all without success.

First, we attempted to distinguish between articles that talked about function as opposed no function at all. We devised a "Go-iness" score, based on the the number of prominent GO terms that appeared in an article's abstract. We performed an analysis of GO-terms by first tokenizing each GO term. We then counted GO-tokens in each abstract. Indeed, abstracts with more GO-tokens do tend to discuss biological "function" more frequently. We checked the distribution of GO terms across abstracts that talk about function and those that do not. We created a training set of function articles by searching Pubmed with the Mesh term "protein" and for the negative set we searched for clinical and disease abstracts. We then ran a chi-square test of significance over GO-tokens. We weighted more heavily the scores of tokens that were statistically more likely to be found in function articles. While this measure produced a reasonable distinction between function vs non-function articles, it did not create a clear bi-modal distribution, and overall classification performance based on go-iness was not good.

As an alternative, we tried to develop a Maximum Entropy classifier, to distinguish between function and non-function articles. As positive examples, we took articles that were annotated with a molecular function GO term. As negative examples, we downloaded genetics articles without the MeSH term "protein". While the performance of the resulting classifier was good, precision fell drastically when it was used to re-rank documents, either alone or combined with Lucene scores. As far as we could discern, the lack of improvement was based on the classifier's inability to distinguish which gene's function is being talked about. Since Lucene returns documents that talk about more than one gene, highly functional articles discussing the wrong gene were often highly ranked: We found the top-scoring documents do talk about function but not the function of the gene in question.

It also appeared that certain phrases could help distinguish between the function of queried genes and the function of other genes. For example, "on <gene-name>" seemed to be frequently used in documents that were talking about the effects of other genes on the queried gene (but not the function of the queried gene itself). Since we did not have time to devote to parsing, we attempted to use information about word order around a gene name. Since the Maximum Entropy classifier we were using treats each document as a bag of words, we artificially induced ordering by replacing each word that occurred before a gene in a sentence by "XXX<word>" and each that occurred after, by "<word>XXX". This created two subtypes of <word> for the classifier to use as features. The method still did not improve performance relative to the original ordering produced by Lucene.

Our final attempt at reordering involved a "one gene theory", based on the observation that an article that talked about many genes seemed less likely to describe a gene's function, since it was not a focus of the article. For this we ran the gene name identifier by first blacking out the topic gene and checking if an article contains any other genes. We took all sentences that talked only about one gene and tried to create a distinction using maximum entropy. We also experimented with using the title, first sentence that mentions the gene and last sentence to mention the gene. None of our manuevres aided precision. They did, however, inform our approach to the secondary task (Section 3).

2.5 Result Stability

The official test set contained only 50 queries. One could argue that retrieval results using such a small set are unlikely to be a good estimate of the true system performaance.

To test this hypothesis, random sets of 50 TREC-like queries and qrels were formulated from LocusLink and tested on our system for both optimum recall and optimum precision. We tested 150 of these random sets for each expansion method, with results as follows:

	No expansion (%)	Precision optimised (%)	Recall optimised (%)
Minimum Precision	23.87	29.43	30.65
Maximum Precision	47.63	49.41	45.04
Mean Precision	34.37	37.33	37.43
Standard Deviation	4.29	3.83	3.52
Precision Range	23.76	19.98	14.39
Minimum Recall	41.31	58.48	67.89
Maximum Recall	79.63	90.14	98.28
Mean Recall	61.95	76.94	80.61
Standard Deviation	7.18	5.43	5.94
Recall Range	38.32	31.66	30.39

There are two significant things to note: First is the wide range in precision and recall that can result from a change in 50-element test set. The second is that our precision scores for the actual test set fall significantly below the mean for our random sets, even taking into account the standard deviation. This indicates that, assuming our random sets were representative of typical queries, the TREC test set was a particularly poor query sample (for our system). What we cannot tell from this experiment is whether relative system performance (i.e., comparing systems against one another) is stable or varies with the choice of test set. This would be something worth assessing.

3 Secondary Task: Reproducing GeneRIF Annotation

Using insights from the first task about gene function, we devised two techniques for picking key phrases out of abstracts that are known to be GeneRIFS (Task 2). Both worked well on cursory subjective evaluation by biologists, but did not score well on the DICE measure.

The first method was based on the density of GO terms in a given sentence. Our hypothesis was that a sentence with a very dense concentration of GO terms would be the most likely GeneRIF text. We used the same tokenized GO-terms as described for Task 1. The method can produce good results. Consider for example, a sentence we picked up from Pubmed ID 12080061 for the Locus Link ID 4012. The actual GeneRIF for this document is:

> *Identification of a tankyrase-binding motif in this protein*

The sentence we picked from the same abstract is:

> *TRF1 binding allows tankyrase to regulate telomere dynamics in human cells, whereas IRAP binding presumably allows tankyrase to regulate the targeting of IRAP.*

In many respects, the sentence picked by our method is a better GeneRIF candidate than the actualy GeneRIF itself, thus raising the usual questions about the quality of the gold standard, and the conditions under which GeneRIFs are selected.

The second method we implemented was based on a Maximum Entropy classification of sentences as either GeneRIF or non-GeneRIF. Using a feature selection method based on chi-square distribution of words, we selected features that helped distinguish between sentences with signal (i.e., GeneRIF) and those without. The positive set consisted of all GeneRIFs other than the test set itself. The negative set consisted of all sentences from the GeneRIF containing abstracts other than those sharing more than three non-stop words in common with the GeneRIF. We created a score algorithm for which the output was the sentence with the highest probability of being a GeneRIF.

To determine the probability that a GeneRIF would be found in a particular position, we annotated a set of 200 MedLine entries from LocusLink associated with GeneRIFs. We located the words from the GeneRIF within the title and abstract. In this data set, the GeneRIF came from the title in 74 of 187 cases. So we used 0.4 as probability for the title being taken as the GeneRIF. The GeneRIF was drawn from the final sentence of an abstract in in 49 of 187 cases, and so this probability was set to 0.26. Finally, the GeneRIF came from the first sentence in 3 of the 187 cases. Assuming all other sentences to be equally likely, we uniformly distributed the probabilities on other sentences by dividing by (total sentences, including title - 3)(decomposition of 3 - 1 for title, 1 for first sentence and 1 for last sentence).

In the end, we submitted one run from this method. The final submission was based on a rather ad hoc combination of internal features (using the Maximum Entropy classifier to find GeneRIF-like sentences) and the prior probability that a GeneRIF would be found in the title, first, last or other sentence. We also used the GO-based metric, in which sentences were weighted in terms of their concentration of lexical tokens found in GO terms.

While our methods identified sentences that appeared very reasonable sources of GeneRIFs, our DICE score for Task 2 was not very high. Because our methods stayed at the sentence-level, they suffered when the correct GeneRIF was a specific phrase within the sentence.

4 Discussion

In future work on query expansion, rather than expand all queries equally with the same method, we want to consider a strategic approach to tailor query expansion. This can be achieved by gathering simple statistics for each term looking for occurrences of punctuation used, protein name containing and gene names and vice versa, parentheticals, specifier types etc. So each query can have a unique expansion based on it term constituents.

While GeneRIFs provide a good measure of comparison, we feel that they are neither exhaustive nor the best candidates in many cases. This makes machines unable to objectively match the GeneRIFs. As explained in the example above, some of the senences/phrases that the computer picked up were in fact better at explaining function than the actual GeneRIFs themselves. The other examples are not discussed here but they provide ample evidence to the claim that machines can be used to extract biologically relevant information from text. This can potentially save thousands of man-hours of work. Since the machine works objectively, the criteria for selection of such sentences will objective and standard. A cursory checking by Biologists for most of these candidates would be enough. These methods can also be used to extract GeneRIFs from abstracts before 4/2002 when GeneRIFs were not available.

Acknowledgements

This work has been supported by in part by Scottish Enterprise, through the Stanford-Edinburgh LINK project (Sinclair, Mehra). Jeffrey Chang was funded under NIH grant GM61374, and Veronica Rotemberg, under the Stanford Summer Research program. We would like to thank James Curran for his suggestions and ideas, and Beatrice Alex for annotating the MedLine entries used in developing our approach to Task 2.

QED: The Edinburgh TREC-2003 Question Answering System

Jochen L. Leidner Johan Bos Tiphaine Dalmas James R. Curran
Stephen Clark Colin J. Bannard Bonnie Webber Mark Steedman
School of Informatics, University of Edinburgh
2 Buccleuch Place, Edinburgh, EH8 9LW, UK.
trec-qa@inf.ed.ac.uk

Abstract

This report describes a new open-domain answer retrieval system developed at the University of Edinburgh and gives results for the TREC-12 question answering track. Phrasal answers are identified by increasingly narrowing down the search space from a large text collection to a single phrase. The system uses document retrieval, query-based passage segmentation and ranking, semantic analysis from a wide-coverage parser, and a unification-like matching procedure to extract potential answers. A simple Web-based answer validation stage is also applied. The system is based on the Open Agent Architecture and has a parallel design so that multiple questions can be answered simultaneously on a Beowulf cluster.

1 Introduction

This report describes QED, a new question answering (QA) system developed at the University of Edinburgh for TREC-12. A key feature of QED is the use of natural language processing (NLP) technology at all stages in the QA process; recent papers have shown the benefit of using NLP for QA (Moldovan et al., 2002). In particular, we parse both the question and blocks of text potentially containing an answer, producing dependency graphs which are transformed into a fine grained semantic interpretation. A matching phase then determines if a potential answer is present, using the relations in WordNet to constrain the answer.

In order to process very large text collections, the system first uses shallow methods to identify text segments which may contain an answer, and these segments are passed to the parser. The segments are identified using a "tiler", which uses simple heuristics based on the words in the question and the text being processed.

We also use additional state-of-the-art text processing tools, including maximum entropy taggers for POS tagging and named entity (NE) recognition. POS tags and NE-tags are used during the construction of the semantic representation. Section 2 describes each component of the system in detail.

The main characteristics of the system architecture are the use of the Open Agent Architecture (OAA) and a parallel design which allows multiple questions to be answered simultaneously on a Beowulf cluster. The architecture is shown in Figure 1.

2 Component Description

2.1 Pre-processing and Indexing

The ACQUAINT document collection which forms the basis for TREC-2003 was pre-processed with a set of Perl scripts, one per newspaper collection, to identify and normalize meta-information. This meta-information included the document id and paragraph number, the title, publication date and story location. The markup for these last three fields was inconsistent, or even absent, in the various collections, and so collection-specific extraction scripts were required.

The collection was tokenized offline using a combination of the Penn Treebank sed script and Tom Morton's statistical tokenizer, available from the OpenNLP project. Ratnaparkhi's MXTERMINATOR program was used to perform sentence boundary detection (Reynar and Ratnaparkhi, 1997). The result was indexed with the Managing Gigabytes (MG 1.3g) search engine (Witten et al., 1999). For our TREC-2003 experiments, we used case-sensitive indexing without stop-word removal and without stemming.

2.2 Query Generation and Retrieval

Using ranked document retrieval, we obtained the best 100 documents from MG, using a query generated from the question. The question words were

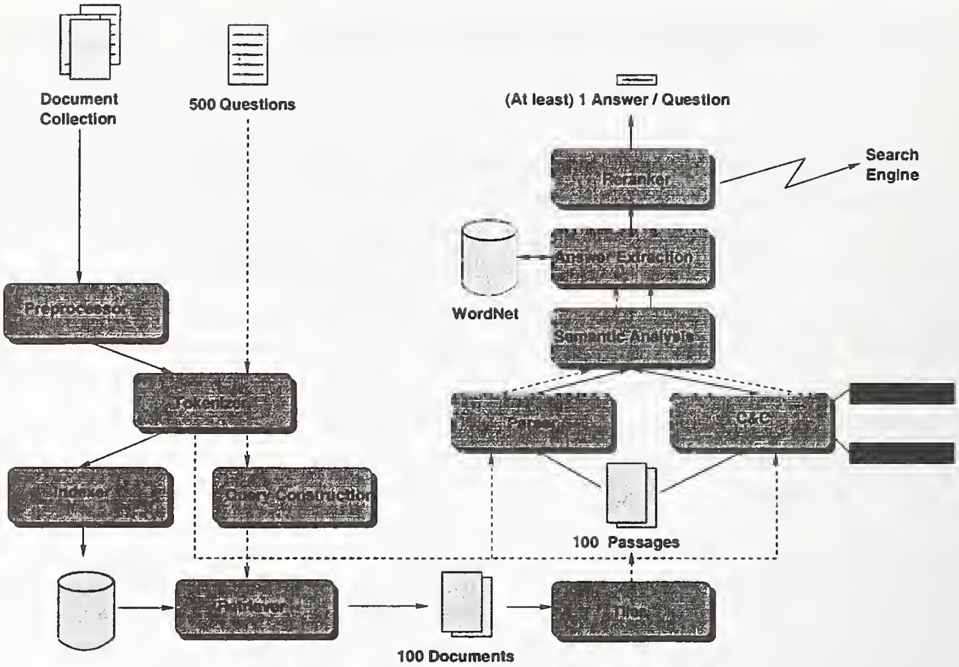


Figure 1: The QED system architecture.

first augmented with their base forms obtained from Minnen et al. (2001)'s morphological analyser, which also performs case normalisation. Stopwords were then removed to form the query keywords. Any remaining uppercase keywords were required to be in the returned documents using MG's plus operator. All remaining lower case keywords were weighted by a factor of 12 (determined by experimentation). Without such a weighting, MG's ranking is too heavily influenced by the uppercase keywords (which are predominantly named entities).

2.3 Passage Segmentation and Ranking

Since our approach involves full parsing to obtain grammatical relations in later stages, we need to reduce the amount of text to be processed to a fraction of the amount returned by the search engine. To this end, we have implemented QTILE, a simple query-based text segmentation and passage ranking tool. This "tiler" extracts from the set of documents a set of segments ("tiles") based on the occurrence of relevant words in a query, which comprises the words of the question. A sliding window is shifted sentence by sentence over the text stream, retaining all window tiles that contain at least one of the words in the query and contain all upper-case query words.

Each tile gets assigned a score based on the following: the number of non-stopword query word tokens (as opposed to types) found in the tile; a comparison of the capitalization of query occurrence and tile occurrence of a term; and the occurrence of 2-grams and 3-grams in both question and tile. The score for every tile is multiplied with a window function (currently a simple triangle function) which weights sentences in the centre of a window higher than in the periphery.

Our tiler is implemented in C++, has linear asymptotic time complexity and requires constant space. For TREC-2003 we use a window size of 3 sentences and pass forward the top-scoring 100 tiles (with duplicates eliminated using a hash signature test).

2.4 Tagging And Syntactic Analysis

The C&C maximum entropy POS tagger (Curran and Clark, 2003a) is used to tag the question words and the text segments returned by the tiler. The C&C NE-tagger (Curran and Clark, 2003b) is also applied to the question and text segments, identifying named entities from the standard MUC-7 data set (locations, organisations, persons, dates, times and monetary amounts). The POS tags and NE-tags are used to construct a semantic representation from the output of the parser (see Section 2.5).

We used the RADISP system (Briscoe and Carroll, 2002) to parse the question and the text segments returned by the tiler. The RADISP parser returns syntactic dependencies represented by grammatical relations such as `ncsubj` (non-clausal subject), `dobj` (direct object), `ncmod` (non-clausal modifier), and so on. The set of dependencies for a sentence are annotated with POS and NE information and converted into a graph in Prolog format. The next section contains an example dependency graph.

To increase the quality of the parser output, we reformulated imperatives in "list questions" (e.g. *Name countries in Europe*) into proper question form (*What are countries in Europe?*). The RADISP parser was much better at returning the correct dependencies for such questions, largely because the RADISP POS tagger typically assigned the incorrect tag to *Name* in the imperative form. We applied a similar approach to other question types not handled well by the parser.

2.5 Semantic Analysis

The aim of this component is to build a semantic representation from the output of the parser. It is used for both the question under consideration and the text passages that might contain an answer to the question. The input to the semantic analysis is a set of dependency relations (describing a graph) between syntactic categories, as Figure 2 illustrates. Categories contain the following information: the surface word-form, the lemmatized word-form, the word position in the sentence, the sentence position in the text, named-entity information, and a POS tag defining the category.

```
top(1, node(’originate’, 9) ).

cat(1, ’croquet’, node(’croquet’, 8), ’NN1’, ’O’ ).
cat(1, ’the’, node(’the’, 5), ’AT’, ’O’ ).
cat(1, ’did’, node(’do’, 4), ’VDD’, ’O’ ).
cat(1, ’originate’, node(’originate’, 9), ’VV0’, ’O’ ).
cat(1, ’game’, node(’game’, 6), ’NN1’, ’O’ ).
cat(1, ’what’, node(’what’, 2), ’DDQ’, ’O’ ).
cat(1, ’country’, node(’country’, 3), ’NN1’, ’O’ ).
cat(1, ’of’, node(’of’, 7), ’IO’, ’O’ ).
cat(1, ’In’, node(’In’, 1), ’II’, ’O’ ).

edge(1, node(’originate’, 9), ncsubj, node(’game’, 6) ).
edge(1, node(’what’, 2), detmod, node(’country’, 3) ).
edge(1, node(’of’, 7), ncmod1, node(’game’, 6) ).
edge(1, node(’of’, 7), ncmod2, node(’croquet’, 8) ).
edge(1, node(’the’, 5), detmod, node(’game’, 6) ).
edge(1, node(’originate’, 9), aux, node(’do’, 4) ).
edge(1, node(’In’, 1), ncmod1, node(’originate’, 9) ).
edge(1, node(’In’, 1), ncmod2, node(’country’, 3) ).

id([’Q_ID’:’1394’,’Q_TYPE’:’factoid’], [1]).
```

Figure 2: Dependency output for the question *In what country did the game of croquet originate?*

Our semantic formalism is based on Discourse Representation Theory (Kamp and Reyle, 1993), but we use an enriched form of Discourse Representation Structure (DRS), combining semantic information with syntactic and sortal information. DRSs are constructed from the dependency relations in a recursive way, starting with an empty DRS at the top node of the dependency graph, and adding semantic information to the DRS as we follow the dependency relations in the graph, using the POS information to decide on the nature of the semantic contribution of a category.

Following DRT, DRSs are defined as ordered pairs of a set of discourse referents and a set of DRS-conditions. The following types of basic DRS-conditions are considered: `pred(x,S)`, `named(x,S)`, `card(x,S)`, `event(e,S)`, and `argN(e,x)`, `rel(x,y,S)`, `mod(x,S)`, where `e`, `x`, `y` are discourse referents, `S` a constant, and `N` a number between 1 and 3. Questions introduce a special DRS-condition of the form `answer(x,T)` for a question type `T`. We call this the *answer literal*; answer literals play an important role in answer extraction (see Section 2.6).

Implemented in Prolog, we reached a recall of around 80%. (By *recall* we mean the percentage of categories that contributed to semantic information in the DRS). Note that each passage or question is translated into one single DRS; hence DRSs can span several sentences. Some basic techniques for pronoun resolution are implemented as well. However, to avoid complicating the answer extraction task too much, we only considered non-recursive DRSs in our TREC-2003 implementation, i.e. DRSs without complex conditions introducing nested DRSs for dealing with negation, disjunction, or universal quantification.

Finally, a set of DRS normalisation rules are applied in a post-processing step, thereby dealing with active-passive alternations, question typing, inferred semantic information, and the disambiguating of noun-noun compounds. The resulting DRS is enriched with information about the original surface word-forms and POS tags, by co-indexing the words, POS tags, the discourse referents, and DRS-conditions (see Figure 3).

```
id([’Q_ID’:’1394’,’Q_TYPE’:factoid],1).

sem(1,

    [p(1001,’In’), p(1002,what), p(1003,country), p(1004,did),
     p(1005,the), p(1006,game), p(1007,of), p(1008,croquet),
     p(1009,originate)],

    [i(1001,’II’), i(1002,’DDQ’), i(1003,’NN1’), i(1004,’VDD’),
     i(1005,’AT’), i(1006,’NN1’), i(1007,’IO’), i(1008,’NN1’),
     i(1009,’VV0’)],

    [drs([0:x1008,1002:x1003,1004:e1004,1005:x1006,1009:e1009],
          [1001:rel(e1009,x1003,’In’),
           1003:answer(x1003,country),
           1006:pred(x1006,game),
           1007:rel(x1006,x1008,of),
           1008:pred(x1008,croquet),
           1009:arg1(e1009,x1006),
           1009:event(e1009,originate) ])]
   ).
```

Figure 3: Example DRS for the question *In what country did the game of croquet originate?*

2.6 Answer Extraction

The answer extraction component takes as input a DRS for the question, and a set of DRSs for selected passages. The task of this component is to extract answer candidates from the passages. This is realised by performing a match between the question-DRS and a passage-DRS, by using a relaxed unification method and a scoring mechanism indicating how well the DRSs match each other.

Taking advantage of Prolog unification, we use Prolog variables for all discourse referents in the question-DRSs, and Prolog atoms in passage-DRSs. We then attempt to unify all terms of the question DRSs with terms in a passage-DRS, using an A* search algorithm. Each potential answer is associated with a score, which we call the DRS score. High scores are obtained for perfect matches (i.e., standard unification) between terms of the question and passage, low scores for less perfect matches (i.e., obtained by "relaxed" unification). Less perfect matches are granted for different semantic types, predicates with different argument order, or terms with symbols that are semantically familiar according to WordNet (Fellbaum, 1998).

After a successful match the answer literal is identified with a particular discourse referent in the passage-DRS. Recall that the DRS-conditions and discourse referents are co-indexed with the surface word-forms of the source passage text. This information is used to generate an answer string, simply by collecting the words that belong to DRS-conditions with discourse referents denoting the answer. Finally, all answer candidates are output in an ordered list. Duplicate answers are eliminated, but answer frequency information is added to each answer in this final list.

Figure 4 gives an example output file. The columns designate the question-id, the source, the ranking score, the DRS score, the frequency of the answer, and a list of sequences of surface word-form, lemma, POS tag and word index.

2.7 Heuristic Candidate Reranking

The system uses a final answer reranking and filtering component, defined slightly differently for each question type. For factoid questions, we rerank the top 5 answers using a function of the candidate answer frequency and the score assigned by the DRS matcher. For definition questions, the same process is used but with a filter which removes any candidate answers with a DRS score below a certain threshold. For list questions, the top 10 answers are considered and the same scoring function is used.

We also use two variations on this reranking algorithm. The first simply uses the DRS score directly, without the candidate answer frequency. The second uses frequency counts from Google to filter out improbable question-answer combinations. A query is sent to Google based on a combination of keywords from the question and the candidate answer. If the document count returned by Google is below some threshold, the answer candidate is removed.

3 Evaluation

Three runs were submitted: run A (EdinInf2003A) used Google as a filter; run B is the system using a function of the candidate answer frequency and the DRS score; and run C is the system using the DRS score directly. On factoid questions, we obtained an accuracy of 0.073 (372 wrong, 5 correct but unsupported, 6 inexact, 30 correct) for runs A and B. For run C we obtained a score of 0.058. Figure 5 gives some example extracted answers for factoid questions.

```
What country is Aswan High Dam located in?
R 1900 XIE19960828.0011 Egypt
What business was the source of John D. Rockefeller's fortune?
R 1909 NYT19991109.0441 Standard Oil
How many Earth days does it take for Mars to orbit the sun?
R 2000 NYT19991220.0063 687
What river is under New York's George Washington bridge?
R 2330 NYT20000203.0416 the Hudson River
What instrument did Louis Armstrong play?
R 2356 NYT19990830.0439 trumpet
What is the name of the Chief Justice of the Supreme Court?
R 2198 XIE19971101.0185 Sajjad Ali Shah
What membrane controls the amount of light entering the eye?
R 1941 APW19980609.1138 The iris
What museum in Philadelphia was used in "Rocky"?
R 2044 NYT20000411.0123 the Museum of Art
When was the first Star Wars movie made?
R 2069 NYT19990315.0214 1977
What composer wrote "Die Gotterdammerung"?
R 2301 NYT19980629.0183 Wagner

How late will airlines let you fly in pregnancy?
W 1952 NYT19990101.0001 NIL
How fast can a nuclear submarine travel?
W 1937 APW20000814.0076 24 nuclear armed
                        cruise missiles
How many floors are in the Empire State Building?
W 1938 NYT19990121.0328 only the top
                        22 floors
What did George Washington call his house?
W 1944 APW19990728.0148 the picture of
                        George Washington
```

Figure 5: Some correct (R) and wrong (W) answers from the EdinInf2003A run. The second column of the system response contains the question number; the third column contains the document the answer was retrieved from.

See Figure 6 for a breakdown of factoid questions by *wh*-word for runs A and B. We obtained correct, but unsupported, answers for factoid questions 1971, 2023, 2048, 2115, 2245 in runs A and B and a similar list excluding the latter two questions for run C. Our average F score over 37 list questions was 0.013; for the 50 definition questions we obtained an F score of 0.063. As a result, our final main task score is 0.056.

4 Discussion and Future Work

In TREC 2003, the overall accuracy of the 54 runs submitted to the QA track ranged between 0.034 and 0.700

```
1394 NYT19990821.0176 0.0687983 0.50 8 Degnan Degnan NNP 157001
1394 NYT19990821.0176 0.0687983 0.43 3 the the DT 158010 nation nation NN 158011
1394 APW19990616.0182 0.0923594 0.37 1 Tarzan Tarzan NNP 21011
1394 APW20000827.0133 0.0651768 0.37 2 English English NN 219015
1394 APW20000827.0133 0.0651768 0.37 1 Additionally Additionally NNP 220001
1394 APW20000827.0133 0.0651768 0.37 4 the the DT 220010 U.S. U.S. NNP 220011
```

Figure 4: Example output file of answer extraction.

WHAT	25/230
WHAT + LOCTYPE	16
WHAT + BE	2
WHAT [OTHER]	7
WHEN	4 / 39
HOW + ADV	1 / 100

Figure 6: Breakdown of correct factoid answers by *wh*-word.

(median 0.177). For list questions, the best, median, and worst average F-scores were 0.396, 0.069, and 0.000, respectively. For definition questions, the F-scores ranged from 0 to 0.555 (with a median of 0.192).

In relation to this interval, our low score reflects the fact that our first year of track QA participation required a large resource commitment to develop a solid basic infrastructure. Such long-term investment will provide the basis for subsequent performance analysis, which in turn will lead to replaced components with superior performance.

Acknowledgements

We are grateful Malvina Nissim for advice on Google and RASP and to the authors of all external progams we utilized for making them available. Leidner is supported by the German Academic Exchange Service (DAAD) under scholarship D/02/01831 and by Linguit GmbH (research contract UK-2002/2). Dalmas is supported by the School of Informatics, University of Edinburgh. Curran is supported by a Commonwealth scholarship and a Sydney University Travelling scholarship. Clark is supported by EPSRC grant GR/M96889. Bannard is supported by ESRC Grant PTA-030-2002-01740.

References

[Briscoe and Carroll2002] Ted Briscoe and John Carroll. 2002. Robust accurate statistical annotation of general text. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, pages 1499–1504, Las Palmas, Gran Canaria.

[Curran and Clark2003a] James R. Curran and Stephen Clark. 2003a. Investigating GIS and smoothing for maximum entropy taggers. In *Proceedings of the 11th Annual Meeting of the European Chapter of the Association for Computational Linguistics (EACL'03)*, pages 91–98, Budapest, Hungary.

[Curran and Clark2003b] James R. Curran and Stephen Clark. 2003b. Language independent NER using a maximum entropy tagger. In *Proceedings of the Seventh Conference on Natural Language Learning (CoNLL-03)*, pages 164–167, Edmonton, Canada.

[Fellbaum1998] Christiane Fellbaum, editor. 1998. *WordNet. An Electronic Lexical Database*. The MIT Press.

[Kamp and Reyle1993] Hans Kamp and Uwe Reyle. 1993. *From Discourse to Logic. An Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and DRT*. Kluwer, Dordrecht.

[Minnen et al.2001] Guido Minnen, John Carroll, and Darren Pearce. 2001. Applied morphological processing of English. *Journal of Natural Language Engineering*, 7:207–223.

[Moldovan et al.2002] Dan Moldovan, Sanda Harabagiu, Roxana Girju, Paul Morarescu, Finley Lacatusu, Adrian Novischi, Adriana Badulescu, and Orest Bolohan. 2002. LCC tools for Question Answering. In *Proceedings of the Eleventh Text Retrieval Conference (TREC-2002)*, Gaithersburg, Maryland.

[Reynar and Ratnaparkhi1997] Jeffrey C. Reynar and Adwait Ratnaparkhi. 1997. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, Washington, D.C.

[Witten et al.1999] Ian A. Witten, Alistair Moffat, and Timothy C. Bell. 1999. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann, Los Altos, CA, 2nd edition.

University of Glasgow at the Robust Track - A Query-based Model Selection Approach for the Poorly-performing Queries

Ben He
Department of Computing Science
University of Glasgow
ben@dcs.gla.ac.uk

Iadh Ounis
Department of Computing Science
University of Glasgow
ounis@dcs.gla.ac.uk

ABSTRACT

In this newly introduced Robust Track, we mainly tested a novel query-based approach for the selection of the most appropriate term-weighting model. In our approach, we cluster the queries according to their statistics and associate the best-performing term-weighting model to each cluster. For a given new query, we assign a cluster to the query according to its statistical features, then apply the model associated to the cluster. As shown by the experimental results, our query-based model selection approach does improve the poorly-performing queries compared to a baseline where a unique retrieval model is applied indifferently to all queries. Moreover, it seems that query expansion has detrimental effect on the poorly-performing queries, although it significantly achieves a higher mean average precision over all the 100 queries.

1. INTRODUCTION

Many term-weighting models have been proposed for information retrieval. For a given collection and a given query, it is an interesting and challenging problem to automatically select the term-weighting model, which would provide the best retrieval performance. It is referred to as the *model selection* problem.

Previous works on the model selection problem, including [6, 8, 7], are based on the analysis of the term-weighting models relevance scores. Therefore, using these approaches, the system cannot select the optimal model prior to the retrieval process.

On the contrary, our approach to the model selection is a pre-retrieval strategy. For a given query, it automatically selects a term-weighting model without the need to wait for the system's relevance scores.

Our work for the model selection problem is based on Amati & van Rijsbergen's Divergence From Randomness (DFR) probabilistic framework [2]. Their framework deploys more than 50 models for term weighting. However, for a given retrieval task/query, the framework does not have a strategy to single out a model that would provide a reliable performance. Table 1 and Table 2 list the mean average precision (MAP) obtained by different models on the TREC-7 and TREC-8 ad-hoc tasks respectively. Here we just list the results given by the most stable and effective models in Amati & van Rijsbergen's framework. As shown by the results, even on the same collection, the optimal model could be different for each task. Also, for each task, the best model could achieve up to 5.65% higher MAP than the poorest one.

Table 1: The mean average precision (MAP) for the TREC-7 ad-hoc task on the disk4, 5 (No CR) of the TREC collections using 11 different DFR models.

Model	MAP	Model	MAP
I(F)B2	.1985	PL2	.1894
I(n_exp)L2	.1937	PB2	.1906
I(n_exp)C2	.2001	BB2	.1985
I(n_exp)B2	.1986	I(n)B2	.1987
I(n)L2	.1958	BL2	.1932
I(F)L2	.1933		

Table 2: The mean average precision (MAP) for the TREC-8 ad-hoc task on the disk4, 5 (No CR) of the TREC collections using 11 different DFR models.

Model	MAP	Model	MAP
I(F)B2	.2623	PL2	.2571
I(n_exp)L2	.2611	PB2	.2526
I(n_exp)C2	.2637	BB2	.2632
I(n_exp)B2	.2630	I(n)B2	.2649
I(n)L2	.2626	BL2	.2608
I(F)L2	.2607		

The aim of this study is to test a query-based approach for the selection of the most appropriate term-weighting model. For a given query and a given collection, we propose to automatically select the best performing term-weighting model. For the Robust Track, our assumption is that the proposed pre-retrieval model selection mechanism would improve the poorly-performing queries, by applying a term-weighting model that maximises the average precision of each query.

The remainder of the paper is organized as follows. In Section 2 we introduce our query-based model selection approach. In Sections 3 and 4, we describe our experimental setup for the Robust Track, and provide the evaluation results and the related analysis. In the experiments, the performance of the proposed model selection approach is particularly assessed. Finally, we conclude our work in Section 5.

2. QUERY-BASED MODEL SELECTION

Our query-based model selection mechanism assumes that the performance of a term-weighting model depends on the statistics of the query. Therefore, the statistical features of a query could constitute a good indication for the model selection mechanism.

Our mechanism involves a training process where the queries are clustered according to their statistical characteristics, and the best term-weighting model for each cluster of queries is obtained by taking previous relevance judgements into consideration. For a given new query, we assign a cluster to the query according to its statistical features, and then apply the term-weighting model associated to the cluster.

A possible approach to clustering the queries is to take the users' feedback into account, and cluster together the queries for which users have visited similar documents [10]. However, since the retrieved document are only known after the retrieval process, this approach is not appropriate for our pre-retrieval model selection mechanism.

2.1 Query Clustering

We propose a query clustering method that is independent of the retrieval procedure. For each query, we construct a feature vector, then cluster the queries according to the similarity of each vector pair. The underlying problem of this approach is the right choice of the features required to represent a query. In this paper, we propose the following three factors for the feature vector of a query:

- Query length

  According to Zhai & Lafferty's work [12], query length has a strong effect on the smoothing methods of language models. In our previous work, we also found that query length heavily affects the length normalisation methods of the probabilistic models [5]. Therefore, it can be an important feature in the clustering process.

  In this work, we use $\rho \cdot ql$ to represent this feature, where:

  - ρ is a parameter. We experimentally set it to 0.2.
  - ql is the query length, i.e. the number of unique terms in the query.

- The difference of the informative amount in the query terms

  To describe the informative amount that a term carries, we usually associate an inverse document frequency (*idf*) to the term. The *idf* factor is a decreasing function of the number of documents containing the term.

  Since the informative amount of a query term is correlated with the utility of a term in the retrieval process, the most informative term in a query, which has the highest *idf*, is supposed to be the most useful term in discriminating the relevant documents from the others in a collection. On the contrary, the least informative term in a query would add little weight to the relevant documents. Indeed, the least informative term in a query tends to be a "stop-word", which could have a detrimental effect on the retrieval. Hence, the difference of the informative amount among the query terms can be an important feature of a query.

  In this work, this difference is defined as the quotient of the minimum *idf* divided by the maximum *idf* among the query terms:

$$\gamma = \frac{idf_{min}}{idf_{max}} = \frac{\log(n_{t,max}/N)}{\log(n_{t,min}/N)}$$

  where:

  - γ is the factor representing the distribution of the informative amount in the query terms.
  - idf_{max} and idf_{max} are the minimum and maximum *idf* among the query terms respectively.
  - n_t is the number of documents containing a particular query term t.
  - $n_{t,max}$ and $n_{t,min}$ are the maximum and minimum n_t among the query terms respectively.
  - N is the number of documents in the whole collection.

- The clarity/ambiguity of a query

  In [4], Cronen-Townsend et. al. proposed the clarity score of a query to measure the coherence of the language usage in documents, whose models are likely to generate the query. In their definition, the clarity of a query is the sum of the Kullback-Leibler divergence between the probability of generating each term in the vocabulary from the query and from the whole collection.

  Another clue for the clarity of a query is the size of the document set containing (at least one of) the query terms. In [9], Plachouras et. al. suggested that it is an important property of a query.

  In this work, we follow the idea in [9], and use

$$\omega = -\frac{\log(n\_q/N)}{\log N}$$

  to represent the clarity a query, where:

  - N is again the number of documents in the whole collection.
  - n_q is the number of documents containing (at least one of) the query terms.

  Thus, when n_q is small, we will obtain a large ω value, which implies that the query is very specific, and therefore is of high clarity.

As a consequence, the feature vector $\overline{qf}$ for a query is given as:

$$\overline{qf} = (\rho \cdot ql, \gamma, \omega)$$

Finally, the feature vectors have to be clustered. A specific similarity measure, and a clustering algorithm need to be specified. In this work, we use the cosine of the angle between two feature vectors in the above three-dimensional space and the agglomerating hierarchical clustering (AHC) algorithm [11] for the clustering process. In the AHC algorithm, initially, each vector is an independent cluster. The similarity between two clusters is measured by the cosine similarity of their centroids. If we have n vectors to be processed, we start with n clusters. Then, we merge the closest pair of clusters (according to the cosine similarity measure) as a single cluster. The merging process is repeated until it results in k clusters. Here the number k of clusters is the halting criterion of the algorithm.

2.2 The Model Selection Mechanism

Based on the query clustering method introduced in the previous section, our model selection mechanism can be summarised as follows:

- We cluster a set of training queries according to their intrinsic features, as proposed in the previous section.
- For each cluster, we select the best-performing model in terms of the precision/recall measures.
- Then for a new query, we assign its closest cluster and trigger the best-performing model associated to the assigned cluster.

The proposed mechanism aims to optimise the average precision for each query, which leads to a maximised overall performance for all the queries.

In Sections 3 and 4, we describe our experimental setting for the Robust Track, and provide the evaluation results and the related analysis. The experiments aim to evaluate the proposed model selection mechanism, especially its performance on the poorly-performing queries.

3. EXPERIMENTAL SETUP

The document collection used in the Robust Track is the disk4 and disk5 (no CR database) of the TREC collections. The 100 topics given by TREC for the Robust Track have two parts. The first part consists of the 50 poorly-performing queries of the TREC-6, 7, 8 ad-hoc tasks, namely 50 old queries. The remaining part consists of 50 new queries introduced by the Robust Track.

Each query consists of 3 fields: title, description and narrative. As required, we use only the description field.

Two different query sets are used as the training set of the model selection mechanism, i.e. the 50 old queries and the 100 queries of the TREC-7, 8 ad-hoc tasks. The latter includes 35 queries of the former.

For the experiments, our model selection mechanism involves 11 term-weighting models developed within Amati & van Rijsbergen's DFR probabilistic framework. These models are: I(n_exp)C2, I(n)L2, I(n)B2, I(n)B2, I(n_exp)B2, BL2 , I(F)B2, I(n_exp)L2, BB2, PL2 and PB2. An effective and stable length normalisation method, i.e. the *normalisation 2*, is applied in these models. The details of these models and the normalisation 2 can be found in [1].

In our experiments for the Robust Track, the model selection mechanism was evaluated through 6 runs (5 official runs and 1 additional run). Also, its performance with or without the use of query expansion was tested. The runs are designed as follows (Table 3 lists the IDs of the runs, Sel78 was the additional run):

- InexpC2

  This is the baseline. It applies a single model, i.e. the I(n_exp)C2 model [1], for all the queries. The I(n_exp)C2 model is developed within Amati & van Rijsbergen's DFR framework and is considered as an effective and robust model on the collection used in this Robust Track. Its formula is:

  $$w(t,d) = \frac{F+1}{n_t \cdot (tfn_e + 1)}(tfn_e \cdot \log_2 \frac{N+1}{n_e + 0.5})$$

  where:

  - $w(t,d)$ is the weight of the term t in the document d.
  - F is the term frequency of the term t in the whole collection.
  - n_t is the document frequency of the term t.
  - N is the number of documents in the collection.
  - n_e is given by $N \cdot \left(1 - (1 - \frac{n_t}{N})^F\right)$.
  - tfn_e is the normalised term frequency. It is given by the modified version of the normalisation 2 [1]:

    $$tfn = tf \cdot \log_e(1 + c \cdot \frac{avg\_l}{l}) \qquad (1)$$

    where c is a parameter; l and avg_l are the document length of the document d and the average document length in the collection respectively; tf is the raw term frequency.

- Sel50 and Sel78

  In order to compare our query-based model selection mechanism to the baseline, we proposed two runs. The two runs use different training queries sets. Sel50 uses the 50 poorly-performing queries (50 old queries), and Sel78 uses the 100 queries of the TREC-7, 8 ad-hoc tasks. Thus, the effect of the training set on the model selection mechanism performance could be tested. We experimentally set the halting criterion of our query clustering method to $k = 4$ for Sel50, and $k = 3$ for Sel78.

- InexpC2QE

  We also tested the model selection mechanism with the use of a query expansion methodology. This run constitutes our baseline for the runs applying the query expansion methodology. The run InexpC2QE applies I(n_exp)C2 and a query expansion methodology for all the queries. The query expansion methodology follows

Table 3: The IDs of the 6 involved runs. +QE and -QE indicate that query expansion is applied or not respectively. Sel78 is an additional run.

	-QE	+QE
Run ID	InexpC2	InexpC2QE
	Sel50	Sel50QE
	Sel78	Sel78QE

the idea of measuring divergence from randomness [1]. The approach can be seen as a generalisation of the approach used by Carpineto and Romano in which they applied the Kullback-Leibler divergence to the un-expanded version of BM25 [3]. For each query, we extract the 40 most informative terms from the top 10 retrieved documents as the expanded terms.

- Sel50QE and Sel78QE

  Both the runs Sel50QE and Sel78QE use the same setting as the runs Sel50 and Sel78 for model selection. However, they also apply the query expansion mechanism for each query. Moreover, we experimentally set the halting criterion of our query clustering method to $k = 4$ for Sel50QE, and $k = 2$ for Sel78QE.

The parameter c of the normalisation 2 (see Equation (1)) was estimated by our new tuning approach, which measures the normalisation effect on the term frequency distribution [5]. Using this tuning approach, we automatically set the parameter to c = 1.96.

4. EXPERIMENTAL RESULTS

There are mainly three quantitatives measures that could be used to evaluate the experiments in the Robust Track:

$\#\overline{Rel}$: The number of queries with no retrieved relevant documents in the top 10 ranks, computed over the complete set of queries.

MAP(X): The area under the curve when MAP (mean average precision) of the worst X queries is plotted against X.

MAP: The mean average precision over the complete set of queries.

Tables 4 and 5 list the results of our runs. Tables 6 and 7 list the involved models in the model selection process and the performance of each single model over all the 100 queries. From the tables, we can see that MAP(X) is quite low in all the cases. Therefore, we mainly compare the MAP and $\#\overline{Rel}$ measures of the runs.

As shown by the results, for the poorly-performing queries, using the 50 old queries as the training query set, Sel50 and Sel50QE achieve better MAP and $\#\overline{Rel}$ than InexpC2 and InexpC2QE respectively (see Tables 4 and 5).

Using more training queries, for the poorly-performing queries, the performance of Sel78 is nearly the same as InexpC2 (see Table 4). Moreover, with query expansion, Sel78QE outperforms InexpC2QE (see Table 5).

Compared to all the 6 runs, we can see that Sel78QE achieves the highest MAP over all the 100 queries (see Tables 4 and 5).

Table 4: Results of the runs without query expansion. Sel78 is an unofficial run.

Queries	Run ID	$\#\overline{Rel}$	MAP(X)	MAP
50 old	InexpC2	10	.0056	.1019
	Sel50	*7*	.0055	*.1054*
	Sel78	10	.0049	.1015
50 new	InexpC2	4	.0246	.3478
	Sel50	4	.0162	.3327
	Sel78	3	.0219	.3446
all 100	InexpC2	14	.0094	.2249
	Sel50	11	.0071	.2190
	Sel78	13	.0081	.2231

Table 5: Results of the runs with query expansion.

Queries	Run ID	$\#\overline{Rel}$	MAP(X)	MAP
50 old	InexpC2QE	17	.0011	.1169
	Sel50QE	*13*	.0045	*.1295*
	Sel78QE	16	.0032	.1238
50 new	InexpC2QE	7	.0191	.3600
	Sel50QE	8	.0053	.3480
	Sel78QE	9	.0071	.3626
all 100	InexpC2QE	24	.0039	.2384
	Sel50QE	21	.0037	.2387
	Sel78QE	25	.0030	*.2432*

The run Sel78QE results into the constitution of two query clusters, to which the models PL2 and I(n_exp)B2 are associated respectively (see Table 6). It is encouraging to see that although I(n_exp)B2 has a better performance in terms of MAP than PL2 does (see Table 7), using PL2 for most of the queries (i.e. 86 out of 100), and I(n_exp)B2 for the rest, our model selection mechanism achieves even higher MAP. This observation suggests that the performance of a term-weighting model is dependent on the statistics of the query. Indeed, with the use of query expansion, this run (i.e. Sel78QE) outperforms the use of each single model indifferently for all the 100 queries (see Table 7).

Moreover, it seems that the query expansion methodology significantly improves the MAP, but has detrimental effect on the poorly-performing queries in terms of $\#\overline{Rel}$.

The performance of the query expansion methodology could be explained by its underlying assumption. It assumes that the top 10 ranked documents are highly relevant, then extracts the 40 most informative terms from them as the expanded query terms. Therefore, for the poorly-performing queries, the top 10 returned documents are likely to give false relevance information. As a consequence, the poorly-performing queries lead to the failure of the query expansion methodology.

5. CONCLUSIONS

In the Robust Track, we mainly evaluated our query-based model selection mechanism based on query clustering. The performance of the model selection mechanism was tested with two different training query sets and, with or without query expansion.

According to the evaluation results, if a proper training query set is used, our query-based term-weighting model selection does improve the performance of the poorly-performing queries compared to the baseline, where a unique term-

Table 6: Statistics of the model selection mechanism. M_Cluster is the term-weighting model associated to a cluster of queries. #Queries is the number of queries (in the whole 100 queries) belonging to the cluster.)

Run ID	M_Cluster	#Queries
Sel50 ($k = 4$)	PB2	18
	I(n_exp)C2	6
	I(F)B2	7
	I(F)L2	69
Sel78 ($k = 3$)	I(n_exp)B2	86
	I(n_exp)C2	14
	PL2	0
Sel50QE ($k = 4$)	I(F)B2	7
	PB2	6
	PL2	18
	I(F)L2	69
Sel78QE ($k = 2$)	PL2	86
	I(n_exp)B2	14

Table 7: Single model Vs. Model selection over all the 100 queries.

Without QE			With QE		
Model	#*Rel*	MAP	Model	#*Rel*	MAP
BB2	13	.2203	BB2	24	.2367
BL2	19	.2115	BL2	26	.2338
PB2	15	.2102	PB2	23	.2240
PL2	14	.2098	PL2	*18*	.2329
I(F)B2	14	.2230	I(F)B2	22	.2396
I(F)L2	17	.2156	I(F)L2	24	.2368
I(n)B2	*12*	.2181	I(n)B2	26	.2375
I(n)L2	15	.2160	I(n)L2	21	.2381
I(n_exp)B2	13	.2234	I(n_exp)B2	22	.2404
I(n_exp)C2	14	*.2250*	**I(n_exp)C2**	24	.2396
I(n_exp)L2	17	.2148	I(n_exp)L2	26	.2386
Sel50	14	.2190	Sel50QE	21	.2387
Sel78	13	.2231	Sel78QE	25	*.2432*

weighting model has been applied uniformly to all queries.

Moreover, it seems that query expansion has detrimental effect on the poorly-performing queries, although it achieves a significantly higher average precision measure over all the 100 queries. This observation can be explained by the underlying mechanism of the query expansion methodology.

6. ACKNOWLEDGMENTS

This work is funded by a UK Engineering and Physical Sciences Research Council (EPSRC) project grant, number GR/R90543/01. The project funds the development of the Terrier Information Retrieval framework (URL: http://ir.dcs .gla.ac.uk/terrier).

7. REFERENCES

[1] G. Amati. *Probabilistic Models for Information Retrieval based on Divergence from Randomness.* PhD thesis, Department of Computing Science, University of Glasgow, 2003.

[2] G. Amati and C. J. van Rijsbergen. Probabilistic models of information retrieval based on measuring the divergence from randomness. In *ACM Transactions on Information Systems (TOIS)*, volume 20(4), pages 357 – 389, October 2002.

[3] C. Carpineto, R. de Mori, G. Romano, and B. Bigi. An information-theoretic approach to automatic query expansion. *ACM Transactions on Information Systems*, 19(1):1 – 27, 2001.

[4] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 299 – 306, Tampere, Finland, 2002.

[5] B. He and I. Ounis. A study of parameter tuning for term frequency normalization. In *Proceedings of the Twelveth ACM CIKM International Conference on Information and Knowledge Management*, pages 10 – 16, New Orleans, LA, November 2003.

[6] R. Jin, C. Falusos, and A. G. Hauptmann. Meta-scoring: Automatically evaluating term weighting schemes in IR without precision-recall. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 83–89, New Orleans, LA, 2001.

[7] S. Luo and J. Callan. Using sampled data and regression to merge search engine results. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 19–26, Tampere, Finland, 2002.

[8] R. Manmatha, T. Rath, and F. Feng. Modeling score distributions for combining the outputs of search engines. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 267–275, New Orleans, LA, 2001.

[9] V. Plachouras, I. Ounis, G. Amati, and C. J. van Rijsbergen. University of glasgow at the web track: Dynamic application of hyperlink analysis using the query scope. In *Proceedings of the Twelth Text REtrieval Conference (TREC 2003)*, Gaithersburg, MD, 2003.

[10] J. Wen, J. Nie, and H. Zhang. Query clustering using user logs. *ACM Transactions on Information Systems (TOIS)*, 20(1)(1046-8188):59 – 81, 2002.

[11] J. R. Wen and H. J. Zhang. *Information Retrieval and Clustering*, chapter Query Clustering in the Web Context, pages 1 – 30. Kluwer Academic Publishers, 2002.

[12] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 334 – 342, New Orleans, LA, 2001.

University of Glasgow at the Robust Track - A Query-based Model Selection Approach for the Poorly-performing Queries

Ben He
Department of Computing Science
University of Glasgow
ben@dcs.gla.ac.uk

Iadh Ounis
Department of Computing Science
University of Glasgow
ounis@dcs.gla.ac.uk

ABSTRACT

In this newly introduced Robust Track, we mainly tested a novel query-based approach for the selection of the most appropriate term-weighting model. In our approach, we cluster the queries according to their statistics and associate the best-performing term-weighting model to each cluster. For a given new query, we assign a cluster to the query according to its statistical features, then apply the model associated to the cluster. As shown by the experimental results, our query-based model selection approach does improve the poorly-performing queries compared to a baseline where a unique retrieval model is applied indifferently to all queries. Moreover, it seems that query expansion has detrimental effect on the poorly-performing queries, although it significantly achieves a higher mean average precision over all the 100 queries.

1. INTRODUCTION

Many term-weighting models have been proposed for information retrieval. For a given collection and a given query, it is an interesting and challenging problem to automatically select the term-weighting model, which would provide the best retrieval performance. It is referred to as the *model selection* problem.

Previous works on the model selection problem, including [6, 8, 7], are based on the analysis of the term-weighting models relevance scores. Therefore, using these approaches, the system cannot select the optimal model prior to the retrieval process.

On the contrary, our approach to the model selection is a pre-retrieval strategy. For a given query, it automatically selects a term-weighting model without the need to wait for the system's relevance scores.

Our work for the model selection problem is based on Amati & van Rijsbergen's Divergence From Randomness (DFR) probabilistic framework [2]. Their framework deploys more than 50 models for term weighting. However, for a given retrieval task/query, the framework does not have a strategy to single out a model that would provide a reliable performance. Table 1 and Table 2 list the mean average precision (MAP) obtained by different models on the TREC-7 and TREC-8 ad-hoc tasks respectively. Here we just list the results given by the most stable and effective models in Amati & van Rijsbergen's framework. As shown by the results, even on the same collection, the optimal model could be different for each task. Also, for each task, the best model could achieve up to 5.65% higher MAP than the poorest one.

Table 1: The mean average precision (MAP) for the TREC-7 ad-hoc task on the disk4, 5 (No CR) of the TREC collections using 11 different DFR models.

Model	MAP	Model	MAP
I(F)B2	.1985	PL2	.1894
I(n_exp)L2	.1937	PB2	.1906
I(n_exp)C2	.2001	BB2	.1985
I(n_exp)B2	.1986	I(n)B2	.1987
I(n)L2	.1958	BL2	.1932
I(F)L2	.1933		

Table 2: The mean average precision (MAP) for the TREC-8 ad-hoc task on the disk4, 5 (No CR) of the TREC collections using 11 different DFR models.

Model	MAP	Model	MAP
I(F)B2	.2623	PL2	.2571
I(n_exp)L2	.2611	PB2	.2526
I(n_exp)C2	.2637	BB2	.2632
I(n_exp)B2	.2630	I(n)B2	.2649
I(n)L2	.2626	BL2	.2608
I(F)L2	.2607		

The aim of this study is to test a query-based approach for the selection of the most appropriate term-weighting model. For a given query and a given collection, we propose to automatically select the best performing term-weighting model. For the Robust Track, our assumption is that the proposed pre-retrieval model selection mechanism would improve the poorly-performing queries, by applying a term-weighting model that maximises the average precision of each query.

The remainder of the paper is organized as follows. In Section 2 we introduce our query-based model selection approach. In Sections 3 and 4, we describe our experimental setup for the Robust Track, and provide the evaluation results and the related analysis. In the experiments, the performance of the proposed model selection approach is particularly assessed. Finally, we conclude our work in Section 5.

2. QUERY-BASED MODEL SELECTION

Our query-based model selection mechanism assumes that the performance of a term-weighting model depends on the statistics of the query. Therefore, the statistical features of a query could constitute a good indication for the model selection mechanism.

Our mechanism involves a training process where the queries are clustered according to their statistical characteristics, and the best term-weighting model for each cluster of queries is obtained by taking previous relevance judgements into consideration. For a given new query, we assign a cluster to the query according to its statistical features, and then apply the term-weighting model associated to the cluster.

A possible approach to clustering the queries is to take the users' feedback into account, and cluster together the queries for which users have visited similar documents [10]. However, since the retrieved document are only known after the retrieval process, this approach is not appropriate for our pre-retrieval model selection mechanism.

2.1 Query Clustering

We propose a query clustering method that is independent of the retrieval procedure. For each query, we construct a feature vector, then cluster the queries according to the similarity of each vector pair. The underlying problem of this approach is the right choice of the features required to represent a query. In this paper, we propose the following three factors for the feature vector of a query:

- Query length

  According to Zhai & Lafferty's work [12], query length has a strong effect on the smoothing methods of language models. In our previous work, we also found that query length heavily affects the length normalisation methods of the probabilistic models [5]. Therefore, it can be an important feature in the clustering process.

  In this work, we use $\rho \cdot ql$ to represent this feature, where:

  - ρ is a parameter. We experimentally set it to 0.2.
  - ql is the query length, i.e. the number of unique terms in the query.

- The difference of the informative amount in the query terms

  To describe the informative amount that a term carries, we usually associate an inverse document frequency (*idf*) to the term. The *idf* factor is a decreasing function of the number of documents containing the term.

  Since the informative amount of a query term is correlated with the utility of a term in the retrieval process, the most informative term in a query, which has the highest *idf*, is supposed to be the most useful term in discriminating the relevant documents from the others in a collection. On the contrary, the least informative term in a query would add little weight to the relevant documents. Indeed, the least informative term in a query tends to be a "stop-word", which could have a detrimental effect on the retrieval. Hence, the difference of the informative amount among the query terms can be an important feature of a query.

  In this work, this difference is defined as the quotient of the minimum *idf* divided by the maximum *idf* among the query terms:

$$\gamma = \frac{idf_{min}}{idf_{max}} = \frac{\log(n_{t,max}/N)}{\log(n_{t,min}/N)}$$

  where:

  - γ is the factor representing the distribution of the informative amount in the query terms.
  - idf_{max} and idf_{max} are the minimum and maximum *idf* among the query terms respectively.
  - n_t is the number of documents containing a particular query term t.
  - $n_{t,max}$ and $n_{t,min}$ are the maximum and minimum n_t among the query terms respectively.
  - N is the number of documents in the whole collection.

- The clarity/ambiguity of a query

  In [4], Cronen-Townsend et. al. proposed the clarity score of a query to measure the coherence of the language usage in documents, whose models are likely to generate the query. In their definition, the clarity of a query is the sum of the Kullback-Leibler divergence between the probability of generating each term in the vocabulary from the query and from the whole collection.

  Another clue for the clarity of a query is the size of the document set containing (at least one of) the query terms. In [9], Plachouras et. al. suggested that it is an important property of a query.

  In this work, we follow the idea in [9], and use

$$\omega = -\frac{\log(n\_q/N)}{\log N}$$

  to represent the clarity a query, where:

  - N is again the number of documents in the whole collection.
  - n_q is the number of documents containing (at least one of) the query terms.

  Thus, when n_q is small, we will obtain a large ω value, which implies that the query is very specific, and therefore is of high clarity.

As a consequence, the feature vector $\overline{qf}$ for a query is given as:

$$\overline{qf} = (\rho \cdot ql, \gamma, \omega)$$

Finally, the feature vectors have to be clustered. A specific similarity measure, and a clustering algorithm need to be specified. In this work, we use the cosine of the angle between two feature vectors in the above three-dimensional space and the agglomerating hierarchical clustering (AHC) algorithm [11] for the clustering process. In the AHC algorithm, initially, each vector is an independent cluster. The similarity between two clusters is measured by the cosine similarity of their centroids. If we have n vectors to be processed, we start with n clusters. Then, we merge the closest pair of clusters (according to the cosine similarity measure) as a single cluster. The merging process is repeated until it results in k clusters. Here the number k of clusters is the halting criterion of the algorithm.

2.2 The Model Selection Mechanism

Based on the query clustering method introduced in the previous section, our model selection mechanism can be summarised as follows:

- We cluster a set of training queries according to their intrinsic features, as proposed in the previous section.
- For each cluster, we select the best-performing model in terms of the precision/recall measures.
- Then for a new query, we assign its closest cluster and trigger the best-performing model associated to the assigned cluster.

The proposed mechanism aims to optimise the average precision for each query, which leads to a maximised overall performance for all the queries.

In Sections 3 and 4, we describe our experimental setting for the Robust Track, and provide the evaluation results and the related analysis. The experiments aim to evaluate the proposed model selection mechanism, especially its performance on the poorly-performing queries.

3. EXPERIMENTAL SETUP

The document collection used in the Robust Track is the disk4 and disk5 (no CR database) of the TREC collections. The 100 topics given by TREC for the Robust Track have two parts. The first part consists of the 50 poorly-performing queries of the TREC-6, 7, 8 ad-hoc tasks, namely 50 old queries. The remaining part consists of 50 new queries introduced by the Robust Track.

Each query consists of 3 fields: title, description and narrative. As required, we use only the description field.

Two different query sets are used as the training set of the model selection mechanism, i.e. the 50 old queries and the 100 queries of the TREC-7, 8 ad-hoc tasks. The latter includes 35 queries of the former.

For the experiments, our model selection mechanism involves 11 term-weighting models developed within Amati & van Rijsbergen's DFR probabilistic framework. These models are: I(n_exp)C2, I(n)L2, I(n)B2, I(n)B2, I(n_exp)B2, BL2 , I(F)B2, I(n_exp)L2, BB2, PL2 and PB2. An effective and stable length normalisation method, i.e. the *normalisation 2*, is applied in these models. The details of these models and the normalisation 2 can be found in [1].

In our experiments for the Robust Track, the model selection mechanism was evaluated through 6 runs (5 official runs and 1 additional run). Also, its performance with or without the use of query expansion was tested. The runs are designed as follows (Table 3 lists the IDs of the runs, Sel78 was the additional run):

- InexpC2

  This is the baseline. It applies a single model, i.e. the I(n_exp)C2 model [1], for all the queries. The I(n_exp)C2 model is developed within Amati & van Rijsbergen's DFR framework and is considered as an effective and robust model on the collection used in this Robust Track. Its formula is:

  $$w(t,d) = \frac{F+1}{n_t \cdot (tfn_e + 1)}(tfn_e \cdot \log_2 \frac{N+1}{n_e + 0.5})$$

  where:

  - $w(t,d)$ is the weight of the term t in the document d.
  - F is the term frequency of the term t in the whole collection.
  - n_t is the document frequency of the term t.
  - N is the number of documents in the collection.
  - n_e is given by $N \cdot \left(1 - (1 - \frac{n_t}{N})^F\right)$.
  - tfn_e is the normalised term frequency. It is given by the modified version of the normalisation 2 [1]:

  $$tfn = tf \cdot \log_e(1 + c \cdot \frac{avg\_l}{l}) \qquad (1)$$

  where c is a parameter; l and avg_l are the document length of the document d and the average document length in the collection respectively; tf is the raw term frequency.

- Sel50 and Sel78

  In order to compare our query-based model selection mechanism to the baseline, we proposed two runs. The two runs use different training queries sets. Sel50 uses the 50 poorly-performing queries (50 old queries), and Sel78 uses the 100 queries of the TREC-7, 8 ad-hoc tasks. Thus, the effect of the training set on the model selection mechanism performance could be tested. We experimentally set the halting criterion of our query clustering method to $k = 4$ for Sel50, and $k = 3$ for Sel78.

- InexpC2QE

  We also tested the model selection mechanism with the use of a query expansion methodology. This run constitutes our baseline for the runs applying the query expansion methodology. The run InexpC2QE applies I(n_exp)C2 and a query expansion methodology for all the queries. The query expansion methodology follows

Table 3: The IDs of the 6 involved runs. +QE and -QE indicate that query expansion is applied or not respectively. Sel78 is an additional run.

	-QE	+QE
Run ID	InexpC2	InexpC2QE
	Sel50	Sel50QE
	Sel78	Sel78QE

the idea of measuring divergence from randomness [1]. The approach can be seen as a generalisation of the approach used by Carpineto and Romano in which they applied the Kullback-Leibler divergence to the un-expanded version of BM25 [3]. For each query, we extract the 40 most informative terms from the top 10 retrieved documents as the expanded terms.

- Sel50QE and Sel78QE

  Both the runs Sel50QE and Sel78QE use the same setting as the runs Sel50 and Sel78 for model selection. However, they also apply the query expansion mechanism for each query. Moreover, we experimentally set the halting criterion of our query clustering method to $k = 4$ for Sel50QE, and $k = 2$ for Sel78QE.

The parameter c of the normalisation 2 (see Equation (1)) was estimated by our new tuning approach, which measures the normalisation effect on the term frequency distribution [5]. Using this tuning approach, we automatically set the parameter to $c = 1.96$.

4. EXPERIMENTAL RESULTS

There are mainly three quantitatives measures that could be used to evaluate the experiments in the Robust Track:

$\#\overline{Rel}$: The number of queries with no retrieved relevant documents in the top 10 ranks, computed over the complete set of queries.

MAP(X): The area under the curve when MAP (mean average precision) of the worst X queries is plotted against X.

MAP: The mean average precision over the complete set of queries.

Tables 4 and 5 list the results of our runs. Tables 6 and 7 list the involved models in the model selection process and the performance of each single model over all the 100 queries. From the tables, we can see that MAP(X) is quite low in all the cases. Therefore, we mainly compare the MAP and $\#\overline{Rel}$ measures of the runs.

As shown by the results, for the poorly-performing queries, using the 50 old queries as the training query set, Sel50 and Sel50QE achieve better MAP and $\#\overline{Rel}$ than InexpC2 and InexpC2QE respectively (see Tables 4 and 5).

Using more training queries, for the poorly-performing queries, the performance of Sel78 is nearly the same as InexpC2 (see Table 4). Moreover, with query expansion, Sel78QE outperforms InexpC2QE (see Table 5).

Compared to all the 6 runs, we can see that Sel78QE achieves the highest MAP over all the 100 queries (see Tables 4 and 5).

Table 4: Results of the runs without query expansion. Sel78 is an unofficial run.

Queries	Run ID	$\#\overline{Rel}$	MAP(X)	MAP
50 old	InexpC2	10	.0056	.1019
	Sel50	7	.0055	*.1054*
	Sel78	10	.0049	.1015
50 new	InexpC2	4	.0246	.3478
	Sel50	4	.0162	.3327
	Sel78	3	.0219	.3446
all 100	InexpC2	14	.0094	.2249
	Sel50	11	.0071	.2190
	Sel78	13	.0081	.2231

Table 5: Results of the runs with query expansion.

Queries	Run ID	$\#\overline{Rel}$	MAP(X)	MAP
50 old	InexpC2QE	17	.0011	.1169
	Sel50QE	*13*	.0045	*.1295*
	Sel78QE	16	.0032	.1238
50 new	InexpC2QE	7	.0191	.3600
	Sel50QE	8	.0053	.3480
	Sel78QE	9	.0071	.3626
all 100	InexpC2QE	24	.0039	.2384
	Sel50QE	21	.0037	.2387
	Sel78QE	25	.0030	*.2432*

The run Sel78QE results into the constitution of two query clusters, to which the models PL2 and I(n_exp)B2 are associated respectively (see Table 6). It is encouraging to see that although I(n_exp)B2 has a better performance in terms of MAP than PL2 does (see Table 7), using PL2 for most of the queries (i.e. 86 out of 100), and I(n_exp)B2 for the rest, our model selection mechanism achieves even higher MAP. This observation suggests that the performance of a term-weighting model is dependent on the statistics of the query. Indeed, with the use of query expansion, this run (i.e. Sel78QE) outperforms the use of each single model indifferently for all the 100 queries (see Table 7).

Moreover, it seems that the query expansion methodology significantly improves the MAP, but has detrimental effect on the poorly-performing queries in terms of $\#\overline{Rel}$.

The performance of the query expansion methodology could be explained by its underlying assumption. It assumes that the top 10 ranked documents are highly relevant, then extracts the 40 most informative terms from them as the expanded query terms. Therefore, for the poorly-performing queries, the top 10 returned documents are likely to give false relevance information. As a consequence, the poorly-performing queries lead to the failure of the query expansion methodology.

5. CONCLUSIONS

In the Robust Track, we mainly evaluated our query-based model selection mechanism based on query clustering. The performance of the model selection mechanism was tested with two different training query sets and, with or without query expansion.

According to the evaluation results, if a proper training query set is used, our query-based term-weighting model selection does improve the performance of the poorly-performing queries compared to the baseline, where a unique term-

Table 6: Statistics of the model selection mechanism. M_Cluster is the term-weighting model associated to a cluster of queries. #Queries is the number of queries (in the whole 100 queries) belonging to the cluster.)

Run ID	M_Cluster	#Queries
Sel50 ($k = 4$)	PB2	18
	I(n_exp)C2	6
	I(F)B2	7
	I(F)L2	69
Sel78 ($k = 3$)	I(n_exp)B2	86
	I(n_exp)C2	14
	PL2	0
Sel50QE ($k = 4$)	I(F)B2	7
	PB2	6
	PL2	18
	I(F)L2	69
Sel78QE ($k = 2$)	PL2	86
	I(n_exp)B2	14

Table 7: Single model Vs. Model selection over all the 100 queries.

Without QE			With QE		
Model	*#Rel*	MAP	Model	*#Rel*	MAP
BB2	13	.2203	BB2	24	.2367
BL2	19	.2115	BL2	26	.2338
PB2	15	.2102	PB2	23	.2240
PL2	14	.2098	PL2	*18*	.2329
I(F)B2	14	.2230	I(F)B2	22	.2396
I(F)L2	17	.2156	I(F)L2	24	.2368
I(n)B2	*12*	.2181	I(n)B2	26	.2375
I(n)L2	15	.2160	I(n)L2	21	.2381
I(n_exp)B2	13	.2234	I(n_exp)B2	22	.2404
I(n_exp)C2	14	*.2250*	**I(n_exp)C2**	24	.2396
I(n_exp)L2	17	.2148	I(n_exp)L2	26	.2386
Sel50	14	.2190	Sel50QE	21	.2387
Sel78	13	.2231	Sel78QE	25	*.2432*

weighting model has been applied uniformly to all queries.

Moreover, it seems that query expansion has detrimental effect on the poorly-performing queries, although it achieves a significantly higher average precision measure over all the 100 queries. This observation can be explained by the underlying mechanism of the query expansion methodology.

6. ACKNOWLEDGMENTS

This work is funded by a UK Engineering and Physical Sciences Research Council (EPSRC) project grant, number GR/R90543/01. The project funds the development of the Terrier Information Retrieval framework (URL: http://ir.dcs .gla.ac.uk/terrier).

7. REFERENCES

[1] G. Amati. *Probabilistic Models for Information Retrieval based on Divergence from Randomness.* PhD thesis, Department of Computing Science, University of Glasgow, 2003.

[2] G. Amati and C. J. van Rijsbergen. Probabilistic models of information retrieval based on measuring the divergence from randomness. In *ACM Transactions on Information Systems (TOIS)*, volume 20(4), pages 357 – 389, October 2002.

[3] C. Carpineto, R. de Mori, G. Romano, and B. Bigi. An information-theoretic approach to automatic query expansion. *ACM Transactions on Information Systems*, 19(1):1 – 27, 2001.

[4] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 299 – 306, Tampere, Finland, 2002.

[5] B. He and I. Ounis. A study of parameter tuning for term frequency normalization. In *Proceedings of the Twelveth ACM CIKM International Conference on Information and Knowledge Management*, pages 10 – 16, New Orleans, LA, November 2003.

[6] R. Jin, C. Falusos, and A. G. Hauptmann. Meta-scoring: Automatically evaluating term weighting schemes in IR without precision-recall. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 83–89, New Orleans, LA, 2001.

[7] S. Luo and J. Callan. Using sampled data and regression to merge search engine results. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 19–26, Tampere, Finland, 2002.

[8] R. Manmatha, T. Rath, and F. Feng. Modeling score distributions for combining the outputs of search engines. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 267–275, New Orleans, LA, 2001.

[9] V. Plachouras, I. Ounis, G. Amati, and C. J. van Rijsbergen. University of glasgow at the web track: Dynamic application of hyperlink analysis using the query scope. In *Proceedings of the Twelth Text REtrieval Conference (TREC 2003)*, Gaithersburg, MD, 2003.

[10] J. Wen, J. Nie, and H. Zhang. Query clustering using user logs. *ACM Transactions on Information Systems (TOIS)*, 20(1)(1046-8188):59 – 81, 2002.

[11] J. R. Wen and H. J. Zhang. *Information Retrieval and Clustering*, chapter Query Clustering in the Web Context, pages 1 – 30. Kluwer Academic Publishers, 2002.

[12] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 334 – 342, New Orleans, LA, 2001.

University of Glasgow at the Web Track: Dynamic Application of Hyperlink Analysis using the Query Scope

Vassilis Plachouras[1], Fidel Cacheda[2], Iadh Ounis[1], and
Cornelis Joost van Rijsbergen[1]

[1]University of Glasgow, G12 8QQ Glasgow, UK
[2]University of A Coruña, Campus de Elviña s/n, 15071 A Coruña, Spain

Abstract

This year, our participation to the Web track aims at combining dynamically evidence from both content and hyperlink analysis. To this end, we introduce a decision mechanism based on the so-called *query scope* concept. For the topic distillation task, we find that the use of anchor text increases precision significantly over content-only retrieval, a result that contrasts with our TREC11 findings. Using the query scope, we show that a selective application of hyperlink analysis, or URL-based scores, is effective for the more generic queries, improving the overall precision. In fact, our most effective runs use the decision mechanism and outperform significantly the content and anchor text retrieval. For the known item task, we employ the query scope in order to distinguish the named page queries from the home page queries, obtaining results close to the content and anchor text baseline.

1 Introduction

Our participation in both tasks of the Web track is a continuation of our TREC11 evaluation of a modular probabilistic framework for Web Information Retrieval, which integrates content and link analysis [6]. This year, we introduce a new notion called *query scope*, which is employed in order to decide dynamically about the most appropriate combination of content and hyperlink analyses. Our assumption is that a query-based application of link analysis performs better than a uniform approach, where the same method is applied indifferently for all queries.

The estimation of the query scope is based on statistical evidence obtained from the set of retrieved documents. The query scope is used as an input into a simple decision mechanism, with which we select the most appropriate retrieval approach for each query. The approaches we employ are content-only retrieval, retrieval based on the content of documents and the anchor text of their incoming links, and a combination of the last approach with a score obtained from the URL length of a document, or from the Static Utility Absorbing Model, a hyperlink analysis model [7].

This year, we find that the combination of content and anchor text retrieval is highly effective, increasing the precision over content-only retrieval. This contrasts with our findings from last year's TREC11 topic distillation task, where the content-only retrieval outperformed any other approach for the topic distillation task. In addition, we find that using a URL length-based score for the most generic queries increases precision for the same task.

The known item task for this year is a mixture of named and home page queries. The query scope is used to detect the probable type of each query, and apply an appropriate retrieval approach. For example, we assume that home page queries could benefit from the combination of content and anchor text with the URL length score. We find that content and anchor text is still a very effective retrieval approach for this task.

The rest of the paper is organised as follows. In Section 2, the query scope is defined. Sections 3 and 4 contain a description of our official runs and a set of additional runs for the topic distillation task. In Section 5, we present our experiments for the known item task, while Section 6 contains our conclusions from this year's participation in the Web track.

2 The Query Scope

Assuming that not all queries benefit from the same retrieval approach, we need to find which of the available approaches is most appropriate for a specific query. Hence, we introduce a decision mechanism that will associate to each query the most appropriate approach. For example, for specific queries we employ a content-only retrieval, while for more generic queries, we use evidence from the hyperlinks, or the URLs. The decision mechanism is based on a composite measure, the query scope, which addresses three important statistical aspects of the set of retrieved documents.

The first aspect addressed is related to the number of retrieved documents. We assume that for the more generic queries, there will be many documents that contain all the query terms. In these cases, the queries address a topic that is widely covered in the collection. Therefore, evidence from hyperlink analysis may be more useful in detecting high quality documents, or homepages of relevant sites.

The $query\_extent$ is the number of retrieved documents that contain all the query terms, normalised between 0 and 1 by dividing with a given fraction α of the total number of documents in the test collection:

$$query\_extent = \min\left(\frac{\{\text{number of retrieved documents containing all query terms}\}}{\alpha}, 1\right) \tag{1}$$

The normalisation is introduced as most of the queries tend to retrieve only a small fraction of documents from the collection, and therefore dividing by α leads to a better distributed measure. In our experiments, we normalised the query extent by dividing with 1% of the number of documents in the collection.

The second aspect is about finding whether there are sites devoted to the query's topic. If there are such sites, we expect that they will contain a high number of documents with all the query terms, and that their homepages will be more useful than other documents. Similarly to the approaches used in [5, 8], we assume that a site is defined by the domain of the document's URL, and we group the documents according to their domain.

We denote by $size_j$ the number of documents from the jth site. In addition, let μ_{size} and σ_{size} be the average and the standard deviation respectively of $size_j$ for $j \in [1, n]$, where n is the number of the retrieved sites. We define the $result\_extent$ as the number of sites for which the size is higher than $\mu_{size} + 2 \times \sigma_{size}$:

$$result\_extent = \{\text{number of sites for which } size_j > \mu_{size} + 2 \times \sigma_{size}\} \tag{2}$$

The third aspect is related to the distribution of *root*, *subroot* and *path* documents among the top ranks. Kraaij et al. [4] have classified the documents into four types according to the type of their URL:

- *root* documents (e.g. http://www.dcs.gla.ac.uk/)
- *subroot* documents (e.g. http://www.dcs.gla.ac.uk/ir/)
- *path* documents (e.g. http://www.dcs.gla.ac.uk/ir/projects/)
- *file* documents (e.g. http://www.dcs.gla.ac.uk/ir/people.html)

We expect that for the home page finding queries, there will be a higher number of documents of the first three types distributed in the top ranks, than for named page finding or topic distillation queries. For this reason, we employ the sum of the reciprocal ranks of documents from the first three categories as an indication of the user's intent to retrieve a home page, so that both the ranks and the number of these documents are taken into account.

In order to obtain the sum of reciprocal ranks, we rank the documents according to their content, and we denote the ith document by d_i. Then, if d_i is either a root, a subroot, or a path, we set RR(i) to the reciprocal rank of d_i, otherwise we set it equal to zero:

$$\mathrm{RR}(i) = \begin{cases} \frac{1}{i} & \text{if } d_i \text{ is a } \textit{root, subroot}, \text{ or } \textit{path} \text{ document} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

The sum of the reciprocal ranks is given by:

$$rank\_sum = \sum_{i=1}^{k} \mathrm{RR}(i) \tag{4}$$

In all our experiments we used $k = 100$.

Having defined the three components of query scope in Equations (1), (2) and (4) respectively, we can proceed with defining a decision mechanism that will employ these measures. Depending on the Web track task, we use the most appropriate of the components defined above, in order to decide how to combine the content and the hyperlink analysis. For the topic distillation task, we employ the $query\_extent$ and the $result\_extent$, while for the known item finding task, we use the $query\_extent$ and the $rank\_sum$.

More specifically, for the topic distillation task, the decision mechanism is shown in Table 1. The queries that have high $query\_extent$ and $result\_extent$ (case I), are treated as more generic. On the contrary, the queries, which result in low values for the query scope components (case IV), are considered to be specific. For the rest of the queries (cases II and III), we cannot say with confidence whether they are generic or specific queries. Both thresholds t_{qe} and t_{re} are determined experimentally.

	$result\_extent \geq t_{re}$	$result\_extent < t_{re}$
$query\_extent \geq t_{qe}$	case I (generic queries)	case III (unknown)
$query\_extent < t_{qe}$	case II (unknown)	case IV (specific queries)

Table 1: The decision mechanism for the topic distillation task

For the known item finding task, our aim is to distinguish between the named page finding queries and the home page finding queries. We expect that the home page finding queries will retrieve more documents containing all the query terms and that there will be more root, subroot or path documents distributed in the top ranks. Therefore, in the decision mechanism for the known item finding task, we employ the $query\_extent$ and the $rank\_sum$, as shown in Table 2. The named page queries are expected to have low values for both $rank\_sum$ and $query\_extent$ (case IV), while home page queries should have either high $rank\_sum$, or high $query\_extent$ (cases I, II and III). Again, both thresholds t_{qe} and t_{rs} are determined experimentally.

	$rank\_sum \geq t_{rs}$	$rank\_sum < t_{rs}$
$query\_extent \geq t_{qe}$	case I (home page query)	case III (home page query)
$query\_extent < t_{qe}$	case II (home page query)	case IV (named page query)

Table 2: The decision mechanism for the known item finding task

3 Topic Distillation Task

Our aim is to investigate the appropriateness of the query scope and the corresponding decision mechanism described in Section 2. We test four different retrieval approaches: content-only, content and anchor text, content and anchor text with URL length, content and anchor text with the Static Utility Absorbing Model [7].

In all runs, we removed stop words and applied stemming during indexing. For the content analysis, we used the weighting model $PL2$ from Amati and van Rijsbergen's Divergence From Randomness (DFR) framework [1], with $c = 1.28$. As opposed to last year, the estimation of the c value is done using a new parameter-tuning methodology for term frequency normalisation, which is collection-independent [3].

The first run, uogtd1c, is a content-only baseline, where only the content of the documents is used. For the second run, uogtd2ca, we test the usefulness of combining the content of documents with the anchor text of their incoming links. Comparing the results of the first two runs, we can see that there is a significant improvement from using anchor text, for both precision at 10 and R precision (see Table 3).

With the third run, we introduce a simplified version of the decision mechanism, as defined for the topic distillation task in Section 2. For the queries where $result\_extent < t_{re}$, we use the content of documents only. For the rest of the queries, where $result\_extent \geq t_{re}$, we employ the content and anchor text of documents.

Official run	Prec. at 10	R precision	Features
uogtd1c	0.0680	0.0730	content-only
uogtd2ca	0.1020	0.1325	content and anchor text
uogtd3cas	0.0820	0.1053	using dynamically content and anchor text
uogtd4cahs	**0.1140**	**0.1391**	using dynamically content, anchor text and URL length
uogtd5cass	0.1080	0.1361	using dynamically content, anchor text and SUAM

Table 3: Topic distillation official results

Therefore, in this configuration, we only use $result\_extent$, for which the threshold is experimentally set to $t_{re} = 7$ (Table 4). In Table 5, we show the number of queries for which each retrieval approach was applied, along with the number of queries for which the applied approach was at least as effective as the alternative retrieval approach.

$result\_extent \geq 7$	$result\_extent < 7$
content and anchor text	content-only

Table 4: The simplified decision mechanism for run uogtd3cas

Approach	Applied for # queries	Effective for # queries (Pr. at 10)	Effective for # queries (R pr.)
content-only	19	10	12
content and anchor text	31	28	29

Table 5: The number of queries for which each approach was applied in run uogtd3cas, and the number of queries for which the applied approach was at least as effective as the alternative approach.

Comparing the effectiveness of this approach (i.e. uogtd3cas) with the first two runs, we can see that both precision at 10 and R precision are between the corresponding measures of the content-only and the content and anchor text runs. However, it is not clear whether this is due to a failure of the decision mechanism, or due to the poor effectiveness of the content-only retrieval.

In the fourth run, uogtd4cahs, we employ a more fine-grained classification by using both $query\_extent$ and $result\_extent$. For the most generic queries (case I), we use both content and anchor text, and we re-rank the top 1000 documents by taking into account their URL length, in order to boost the homepages. Let sc_i be the content analysis score of the ith document. In addition, let $urlpath\_len_i$ be the length in characters of the ith document's URL path[1]. The final score for this document will be:

$$score_i = sc_i \times \frac{1}{\log_2(urlpath\_len_i + 1)} \tag{5}$$

The decision mechanism used for this run is shown in Table 6. The thresholds t_{re} and t_{qe} were set experimentally to the values 7 and 0.5 respectively. Table 7 contains the number of queries for which each retrieval approach was applied, along with the number of queries for which the applied approach was at least as effective as the others.

	$result\_extent \geq 7$	$result\_extent < 7$
$query\_extent \geq 0.5$	content, anchor text and URL length	content-only
$query\_extent < 0.5$	content and anchor text	content-only

Table 6: The decision mechanism for run uogtd4cahs

Approach	Applied for # queries	Effective for # queries (Pr. at 10)	Effective for # queries (R pr.)
content-only	19	10	12
content and anchor text	16	15	15
content, anchor text and URL length	15	14	14

Table 7: The number of queries for which each approach was applied in run uogtd4cahs, and the number of queries for which the applied approach was at least as effective as the others.

As shown in Table 3, the above approach improves both precision at 10 and R precision significantly over run uogtd3cas. Moreover, looking in Table 7, we find that both combinations of content with anchor text, and content with anchor text and URL length are very effective for the queries on which they were applied respectively. On the other hand, content-only retrieval is not very appropriate for the selected queries.

For our last official run of the topic distillation task, uogtd5cass, we combine the content and anchor text of documents with the Static Utility Absorbing Model (SUAM) [7], a hyperlink analysis approach. Let sc_i be the

[1] For example, for the URL http://trec.nist.gov/data/intro_eng.html, the path is data/intro_eng.html

content analysis score and sc_{AMi} be the Absorbing Model score for the ith document. The final score is given by:

$$score_i = sc_i^A \times (-\log_2(sc_{AMi}))^B \tag{6}$$

where $A = B = 1$. The Static Utility Absorbing Model scores documents according to both their incoming and outgoing links, aiming to boost the key entry points for a given topic. The decision mechanism employed is shown in Table 8, where the thresholds t_{re} and t_{qe} are set to 7 and 0.5 respectively. In all cases, we employ the content and anchor text of documents, while SUAM is only used for the most generic queries.

When comparing this run to uogtd2ca, where content and anchor text is used for all the cases, we note that precision at 10 increases slightly (Table 3), due to improvement in 2 out of the 15 most generic queries, for which SUAM is applied (Table 8). For the rest of the queries, effectiveness remains stable. If we consider R precision, we find that SUAM results in improvements for 2 queries, but is also detrimental for 3 queries on which it is applied.

	$result\_extent \geq 7$	$result\_extent < 7$
$query\_extent \geq 0.5$	content, anchor text and SUAM	content and anchor text
$query\_extent < 0.5$	content and anchor text	content and anchor text

Table 8: The decision mechanism for run uogtd5cass

4 Additional Topic Distillation Experiments

After obtaining the official results and the relevance assessments, we run additional experiments for the topic distillation task. The aim is to learn more about the effectiveness of the query scope and the decision mechanism.

In the additional runs, we test separately the effectiveness of $query\_extent$ and $result\_extent$. To facilitate the analysis, employing the three retrieval approaches used in uogtd4cahs (Table 7), we use the two most effective ones, in terms of average precision at 10 over all queries. As shown in Table 9, these are content and anchor text retrieval, and content with anchor text and the URL length-based score (Equation (5)).

Employing the decision mechanism as shown in Table 1, we use Equation (5) for the cases I and II and only the content and anchor text for the cases III and IV. This corresponds to employing only the $result\_extent$, for which the threshold t_{re} takes the values $4, 7, 10, 13$ (runs re$i$ in Table 9). On the other hand, if we use Equation (5) for the cases I and III, and we employ only the content and anchor text of documents for the cases II and IV, then this approach corresponds to applying only the $query\_extent$, for which the threshold t_{qe} takes the values $0.25, 0.45, 0.50, 0.55, 0.65$ (runs qe$i$ in Table 9).

Run	Threshold	Prec. at 10	R precision
content-only (uogtd1c)		0.0680	0.0730
content and anchor text (uogtd2ca)		0.1020	0.1325
content, anchor text and URL length		0.1400	0.1369
re1	$t_{re} = 4$	0.1360	0.1340
re2	$t_{re} = 7$	0.1440	0.1428
re3	$t_{re} = 10$	**0.1480**	0.1528
re4	$t_{re} = 13$	0.1240	0.1395
qe1	$t_{qe} = 0.25$	0.1280	0.1547
qe2	$t_{qe} = 0.45$	0.1340	**0.1657**
qe3	$t_{qe} = 0.50$	0.1340	**0.1657**
qe4	$t_{qe} = 0.55$	0.1300	0.1635
qe5	$t_{qe} = 0.65$	0.1260	0.1542

Table 9: Additional results for the topic distillation

From the additional results, we can see that by employing $result\_extent$, we achieve higher precision at 10 for two out of the four threshold values tested. More specifically, for $t_{re} = 10$, we obtain 0.1480 precision at 10. When we use $query\_extent$, precision at 10 is between the average precision at 10 of the two most effective retrieval approaches employed. Now, if we consider R precision, both $result\_extent$ and $query\_extent$ lead to improvements over the best performing retrieval approach. In addition, $query\_extent$ is more effective than $result\_extent$, for the tested threshold values. When t_{qe} is equal to 0.45 or 0.50, we obtain 0.1657 R precision.

5 Known Item Finding Task

For the known item finding task, we employ the query scope in order to discriminate between home page and named page queries. We assume that the most effective method for named page queries is to employ content and anchor text retrieval. For home page queries, we use content and anchor text retrieval, and re-rank all the retrieved documents by applying the formula from Equation (5).

Official run	Aver. Recip. Rank.	Found in top 10	Not Found	Features
uogki1c	0.363	167 (55.7%)	82 (27.3%)	content-only
uogki2ca	**0.615**	238 (79.3%)	34 (11.3%)	content and anchor text
uogki3cah	0.273	117 (39.0%)	92 (30.7%)	content and anchor text and URL length
uogki4cahs	0.595	227 (75.7%)	30 (10.0%)	using dynamically content, anchor text and URL length

Table 10: Known item finding official results

In the first run, uogki1c, we use only the content-only retrieval. The used weighting model is $PL2$, as in the case of topic distillation, but with $c = 1$ this time. As expected from last year's named page finding task [2], the content-based approach is not the most efficient approach (see Table 10). In the second run, uogki2ca, where the document's content and the anchor text of the incoming links are used, the effectiveness increases significantly.

For the third run, uogki3cah, we also employ the URL length, in the same way as in Equation (5), in order to boost the home pages, which tend to have shorter URLs. Although this approach is not appropriate for all queries, it is expected to be effective at least for the home page finding queries.

Before proceeding with the fourth run, we will look at the effectiveness of content and anchor text retrieval, and content with anchor text and URL length, across the sets of named page and home page finding queries. For the named page queries, content and anchor text retrieval results in 0.613 average reciprocal rank, and for the home page queries the average reciprocal rank is 0.617. On the other hand, content with anchor text and URL length is not so stable across the two sets of queries. For the named page queries, the average reciprocal rank is 0.060, while for the home page finding task it is 0.487. These results show that content and anchor text retrieval is a very robust approach for the known item finding task, and indicate that there is no significant gain in trying to combine different methods.

However, if we manually select the best approach for each query, then we get 0.680 average reciprocal rank across the two sets of queries. For the named page finding queries, we would get 0.613 and for the home page finding queries we would get 0.747. These figures show that there is room for improvement if a successful combination is found.

Our last official run intends to simulate automatically the above manual selection. Therefore, we employ the query scope as defined in Table 2 for the known item task. We use two components of the query scope, the $query\_extent$ and the $rank\_sum$, with threshold values set experimentally to $t_{rs} = 1$ and $t_{qe} = 0.8$ as shown in Table 11. For all queries, the content and the anchor text of the incoming links is used, and for the most generic ones, where $t_{rs} \geq 1$ and $t_{qe} \geq 0.8$, the URL's length is used as defined in Equation (5). As shown in Table 10, precision is close to that of run uogki2ca, where content and anchor text is used.

We find that the average reciprocal rank of named page queries is 0.554, while for the home page queries it is 0.636. More specifically, the URL length is employed for 56 queries, of which 19 were named page queries and 37 were home page queries. For those 19 named page queries where the URL length is employed, there is no additional improvement. However, for 11 out of the 37 home page queries where the URL length is used, there is an improvement over using only the anchor text. These results show that our decision mechanism succeeds in increasing the effectiveness for the home page queries, but on the other hand, it does not benefit named page queries. More work is needed in order to refine the decision mechanism and improve its effectiveness, compared to that of the manual selection of the most appropriate approaches per query.

	$rank\_sum \geq 1$	$rank\_sum < 1$
$query\_extent \geq 0.8$	content, anchor text and URL length	content, anchor text and URL length
$query\_extent < 0.8$	content, anchor text and URL length	content and anchor text

Table 11: The decision mechanism for run uogki4cahs

6 Conclusions

We introduce a dynamic approach for combining content and hyperlink analysis, based on the query scope, a composite measure for quantifying three aspects of how appropriate a query is for hyperlink analysis.

For the topic distillation, we find that employing the anchor text of incoming links outperforms significantly a content-only retrieval. This result contrasts with our findings from TREC11, where the content-only retrieval was the most effective approach for topic distillation. In addition, URL information is very effective in identifying the relevant homepages. As for the dynamic combination of different retrieval approaches, it appears that the decision mechanism is a very effective method, leading to significant improvements in our official results. Moreover, when the most effective individual approaches are employed, we find that the *result_extent* is more effective with respect to precision at 10, while *query_extent* achieves better results when R precision is used for the evaluation.

For the known item finding task, the content and anchor text retrieval is a robust approach, independently of the query type. However, the URL information is useful for the home page queries, and if it is applied appropriately, it results into significant improvement. Our decision mechanism performed nearly as well as the content and anchor text baseline. It resulted in increased effectiveness among the home page queries, while it didn't benefit the named page queries.

In conclusion, we have shown that by employing simple statistical mechanisms, it is possible to improve the retrieval effectiveness by combining dynamically evidence from content and hyperlink analysis.

Acknowledgements

The work of the first, third and fourth authors is funded by a UK Engineering and Physical Sciences Research Council (EPSRC) project grant, number GR/R90543/01. The project funds the development of the Terrier Information Retrieval framework (url: http://ir.dcs.gla.ac.uk/terrier).

The work of the second author has been partially supported by the 'Comisión Interministerial de Ciencia y Tecnologia' (CICYT) of the Spanish government, under project TIC2001-0547 and by the 'Fundación Caixa Galicia' (Beca curso 2002/2003 para Estudios de Postgrado en Universidades y en Centros de Investigación de Excelencia Académica).

References

[1] G. Amati and C. J. van Rijsbergen. Probabilistic models of information retrieval based on measuring divergence from randomness. *ACM Transactions on Information Systems*, 40(4):1–33, 2002.

[2] N. Craswell and D. Hawking. Overview of the TREC-2002 Web Track. In *NIST Special Publication: 500-251 The 11th Text REtrieval Conference (TREC 2002)*, pages 86–93. NIST, 2002.

[3] B. He and I. Ounis. A study of parameter tuning for term frequency normalization. In *Proceedings of the 12th international Conference on Information and Knowledge Management (CIKM)*, pages 10–16. ACM Press, 2003.

[4] W. Kraaij, T. Westerveld, and D. Hiemstra. The importance of prior probabilities for entry page search. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and Development in Information Retrieval*, pages 27–34. ACM Press, 2002.

[5] K. L. Kwok, P. Deng, N. Dinstl, and M. Chan. TREC2002 Web, Novelty and Filtering Track Experiments using PIRCS. In *NIST Special Publication: SP 500-251 The 11th Text Retrieval Conference (TREC)*, pages 520–528, NIST, 2002.

[6] V. Plachouras, I. Ounis, G. Amati, and C. J. van Rijsbergen. University of Glasgow at the Web track of TREC 2002. In *NIST Special Publication: SP 500-251 The 11th Text Retrieval Conference (TREC)*, pages 645–651, NIST, 2002.

[7] V. Plachouras, I. Ounis, and G. Amati. A Utility-oriented Hyperlink Analysis Model for the Web. In *Proceedings of the 1st Latin Web Conference*, pages 123–131. IEEE Press, 2003.

[8] M. Zhang, R. Song, C. Lin, S. Ma, Z. Jiang, Y. Jin, Y. Liu, and L. Zhao. THU TREC 2002: Web Track Experiments. In *NIST Special Publication: SP 500-251 The 11th Text Retrieval Conference (TREC)*, pages 586–594, NIST, 2002.

UIC at TREC-2003: Robust Track (Draft)

Shuang Liu, Clement Yu

Database and Information System Lab
Computer Science Department, University of Illinois at Chicago
{sliu, yu}@cs.uic.edu

Abstract

In TREC 2003, the Database and Information System Lab (DBIS) at University of Illinois at Chicago (UIC) participate in the robust track, which is a traditional ad hoc retrieval task. The emphasis is based on average effectiveness as well as individual topic effectiveness.

Noun phrases in the query are identified and classified into 4 types: proper names, dictionary phrases, simple phrases and complex phrases. A document has a phrase if all content words in a phrase are within a window of a certain size. The window sizes for different types of phrases are different. We consider phrases to be more important than individual terms. As a consequence, documents in response to a query are ranked with matching phrases given a higher priority. WordNet is used to disambiguate word senses and bring in useful synonyms and hyponyms once the correct senses of the words in a query have been identified. The usual pseudo-feedback process is modified so that the documents are also ranked according to phrase and word similarities with phrase matching having a higher priority. Five runs which use either title or title and description have been submitted.

1. Introduction

We believe that the robust retrieval result can be achieved by: (1) effective use of phrases, (2) a new similarity function capturing the use of phrases, and (3) utilizing suitable synonyms and hyponyms which are properly chosen in a word sense disambiguation process.

Noun phrases, if exist in a query, are recognized and classified into four types: proper names of people or organizations; dictionary phrases which can be found in dictionaries such as WordNet; simple phrases which do not have any embedded phrases; complex phrases which are more complicated phrases.

A document has a phrase if all the content words in the phrase are within a window of a certain size. The window size depends on the type of the phrase. For a proper name, essentially all content words have to be adjacent. That is, the window size for a proper name, after excluding non-content words and words in the name, is close to 0. Content words in a dictionary phrase need not to be adjacent so its window size can be larger. The window size for a simple phrase is larger than that for a dictionary phrase but smaller than that for a complex phrase.

We consider phrases to be more important than individual content words in retrieving documents. As a consequence, the similarity measure between a query and a document has two components (phrase-sim, term-sim), where phrase-sim is the similarity obtained by matching the phrases of the query against those in the document and term-sim is the usual similarity between the query and the document based on term matches. The latter similarity can be computed by the standard Okapi similarity function [RW00]. Documents are ranked in descending order of (phrase-sim, term-sim). That is, documents with higher phrase-sim will be ranked higher. When documents have the same phrase-sim, they will then be ranked according to term-sim.

In traditional pseudo-feedback, new terms which are highly correlated with the original query terms in the top ranked documents are added to the query [BR99, GF98]. In our framework, we impose an additional constraint, namely a new term is added only if it is highly positively globally correlated with a query term/phrase. This information is used to compute phrase-sim as well as term-sim.

Machine readable dictionaries such as WordNet [Mill90] have been utilized in document retrieval. In our approach, WordNet is used to disambiguate word senses. Once a correct or a dominant sense has been identified, additional synonyms and hyponyms are added into the query.

In the remaining part of this paper, how phrases in a query are identified and how they are classified into the four types are discussed in section 2. Section 3 presents a new similarity function capturing the use of phrases. Section 4 describes how WordNet can be utilized. In section 5 we analysis our submitted 5 runs. Section 6 concludes this paper.

2. Phrase Identification

Noun phrases in a query are classified into proper names, dictionary phrases, simple phrases and complex phrases. Brill's tagger [Brill] is used to assign a part of speech (POS) to each word. The POS information will be used to recognize simple and complex phrases.

We first classify phrases into the following types: named entities which are the names of persons and organizations. Dictionary phrases, they are actual phrases which appear in dictionaries such as WordNet. An example is "prime factor". Simple phrases: they are not found in a dictionary such as WordNet, but they are two word phrases as recognized by a simple grammar. An example is "school uniform". Complex phrases: they are not found in a dictionary and are phrases which are more complicated than simple phrases. A complex phrase may have one or more dictionary or simple phrase embedded in it. For example, a complex phrase is "Canadian building code".

While the words in a named entity are required to be adjacent and be in the same order, the words in the other phrases need not be adjacent. In fact, dictionary phrase, simple phrase and complex phrase may appear in different forms in different documents. For example, the phrase "information retrieval" may appear as "retrieval of information" in a document. As a consequence, we impose different proximity conditions for their recognitions in documents. Specifically, for a dictionary phrase to be recognized in a document, we require its component content words to be within certain number of words, say w1; for a simple phrase, the constraint is that the component words are to be within w2 words, where w2 > w1; for a complex phrase, the component words are within w3 words, with w3 > w2. These constants w1, w2 and w3 will be determined by a learning algorithm to be given below. In addition, we require the component content words in a simple phrase and a complex phrase to be highly correlated in the documents to be considered as having the phrase; otherwise, a phrase is not formed and we match individual content words. Intuitively, a dictionary phrase should have its component words in close proximity with each other, say within 3 words apart. However, this might not be true in practice. For example, for the query "drugs for mental illness" which contains the dictionary phrase "mental illness", an example relevant document (in a TREC collection) has the two words quite far apart as shown in the following paragraph:

"Earlier this week, Rose and the family acknowledged that Joseph Lynch was hospitalized for mental problems more than 2 years ago and had been on lithium for his illness until recently. "

Based on the above observation, we decide that suitable distances between components of different types of phrases should be learned instead of determined rather arbitrarily based on intuition. Specifically, for a set of queries, we identify the types of phrases, the distances of the components of the phrases in the relevant documents and the irrelevant documents having high similarities with the queries (in the TREC collections) and have the information fed to a decision tree (C4.5). The decision tree will then supply for each type of phrases a suitable distance d such that for most relevant documents having components of that type of phrases, the component words are within the distance and for most irrelevant documents having high similarities with the queries, their component words have distances exceed d. This information is then applied to a different set of queries which are disjoint from the set of training queries.

3. Similarity Functions

Our hypothesis is that phrases convey more semantic information than individual words. As a consequence, we design a new similarity function which places more emphasis on phrase matching than word matching. Specifically, the new similarity function produces, for each query and each document, a pair (phrase-sim, term-sim), where phrase-sim is the similarity due to matching of phrases and term-sim is the similarity due to matching of individual words. Term-sim can be computed using the Okapi formula. Suppose the similarities of two documents D1 and D2 with respect to a query Q are (p1, t1) and (p2, t2) respectively. Then the similarity of D1 is higher than that of D2 if p1 > p2 or if p1 = p2, then t1 > t2. In other words, phrase similarity dominates term similarity and only when the two phrase similarities are equal, then term similarities are used to break the tie.

The phrase similarity of a document with a query can be computed as follows. For a named entity, a dictionary phrase or a simple phrase, if the document has the phrase, then its phrase-sim increases by the idf weight (inverse document frequency weight) of the phrase. For a complex phrase, a document may have the entire complex phrase, a phrase embedded in the complex phrase or none of it. In the first case, the phrase-sim of the document is increased by the sum of the idfs weights of all phrases embedded in the complex phrase, including the complex phrase itself. In the second case, it is increased by the idf weight of the phrase embedded in the complex phrase.

4. Synonyms and Hyponyms Utilization

In Wordnet, synonyms having the same meaning are grouped together to form a synset. For example, the noun baby forms a synset with the word infant in one sense, while in a different sense, baby and sister form a different synset. Associated with each synset of a word, there is a definition of the synset. In addition, there is a frequency value which indicates the extent the word is utilized in this sense. Suppose the noun baby and the word infant form a synset with frequency 611. Suppose the noun baby has other synsets with a total of frequency values 54. Then the noun baby is more likely to be used in the sense of infant than any other sense. In general, if a word with a part of speech has multiple senses and one of its synset, say S, has its frequency value higher than the sum of the frequency values of all other synsets of the same word, then the synset S is called the dominant synset of the word in that part of speech.

If a synonym s of a query word or phrase t is to be added to the query, one of the two following rules will be followed: (i) s and t are identical synonyms i.e. there is an unique synset containing both s and t, and there is no other synset containing s or t. (ii) there is a synset containing both s and t and that synset is dominant for both s and t.

Clearly, if there is a unique synset containing both s and t, the synset is dominant for both terms. We now attempt to disambiguate word senses while at the same time add new words into the query. Our strategy to add new terms and to disambiguate word senses is based on the following principles.

4.1 Utilize the adjacent words in the query for sense disambiguation

A given query is parsed. The POS of each word as well as the phrases in the query are recognized. Suppose there are two adjacent terms t1 and t2 in the query and they form a phrase. Each of t1 and t2 has (a) one or more synsets, (b) for each synset, there is a definition, (c) zero or more synsets containing hyponyms (a hyponym of a term t satisfies the IS-A relationship with respect to t; for example, boy is a hyponym of male). These three items (a), (b), and (c) can be used to disambiguate the senses of the two terms. We now consider determining the sense of t1. The sense of t2 can be determined in a similar manner.

Step (1) Check if term t2 or a synonym of t2 is found in the definition of a synset of t1, say S. In this case, the sense of t1 is determined to be the synset S whose definition contains term t2 or its synonym.

Example 1: Suppose a query contains the phrase "incandescent light". The definition of a synset of incandescent contains the word light. Thus, this synset of incandescent is used.

Whenever the sense of a term t is determined, we examine the possibility of adding the synonyms of t in its synset S to the query. For any term t' in S, if S is a dominant synset of t', then t' is added to the query. The weight of t' is given by:

$$W(t') = f(t', S)/F(t') * f(t, S)/ F(t) \qquad (1)$$

where f(t',S) is the frequency value of t' in S; f(t, S) is the frequency value of t in S; F(t) is the sum of frequency values of t in all synsets which contain t and have the same part of speech as t and F(t') is the corresponding value for t'.

The first and the second components of the equation represent the likelihood that t' and t have the same meaning as that conveyed by the synset S respectively. Thus, we may interpret the weight of t' to be the likelihood that t' has the same meaning as t.

Example 2: This is a continuation of Example 1. The synset containing incandescent also contains "candent". It can be verified that the synset is dominant for candent and therefore candent is added to the query.

Special case 1: A synonym of a term t can be a single term or a phrase containing multiple words. Sometimes, the phrase p contains the term t. In that situation, adding the phrase p to the query will not be useful, as a document having the phrase must necessarily have the term t. We therefore examine if the terms in p - t can be added to the query. The criterion to add the words in p - t to the query is that each such word must appear in the definition of the determined synset containing t.

Example 3: A determined synset containing the word induction (see Example 5) is generalization, generalisation, induction, inductive reasoning (reasoning from detailed facts to general principles). This synset is dominant for both the word induction and the phrase inductive reasoning. As a consequence, the phrase is being considered for addition to the query. However, induction and inductive have the same stem. Thus, we consider adding the word reasoning to the query. It is included in the query, because it occurs in the definition of the synset.

In addition to possibly adding terms from the synset S to the query, we also examine the definition of S and attempt to add some terms which are similar to some query term. Specifically, if a term t' in the definition of t has a prefix which is sufficiently similar to a prefix of a query term t", then t' can be added to the query. The reason is that stemming is not a perfect process and very often two words t' and t" are variants of the same word but stemming [Porter] does not reduce them to the same stem.

When the sense of a query term t is determined, we also want to determine if direct hyponyms of t should be added to a query. If the determined synset of t has a unique child (hyponym synset) U, then for each term t' in the synset U, we check if the synset U is dominant in the synsets containing t'; if so, t' is added to the query, with a weight similar to that given by the formula (1).

Step (2): If the sense of some query term t has yet to be determined, then decide whether there is a dominant synset for t. If there is, the sense of t is assumed to be that dominant synset. For a term t' in the synset, if the synset is also dominant for t', then t' is added to the query with its weight given by formula (1).

4.2 Modification of the query and the similarity function

In the last section, if the senses of query terms are determined, then new terms may be added to the query. Such new terms may make the resulting query a Boolean query as explained in the following paragraph.

Consider a query consisting of two query terms t1 with idf weight w1 and term t2 with idf weight w2. Suppose t1 brings in new terms t1' with weight fi' as given by formula (1) and term t2 brings in term t2' with weight f2' as given by the same formula. The weight fi' can be considered as a relative significance between an occurrence of ti versus an occurrence of ti'. That is, an occurrence of ti' is equivalent to fi occurrence of ti. The idf weight of ti' is assumed to be min. {wi, actual idf weight of ti'}. Thus, having x occurrences of ti' results in an idf weight no higher than that of the actual ti and x * fi occurrences of ti.

If there is no phrase in this query, then a document with a1 occurrences of t1' and a2 occurrences of t2' will get a similarity based on the Okapi formula, in which the idf weight of term i is min. {wi, actual idf of ti'} and the term frequency of ti is ai * fi.

If terms t1 and t2 actually form a phrase, then a document having t1' and t2, or t1 and t2', or t1 and t2, or t1' and t2' will get a phrase-sim (phrase similarity) value of the phrase. In addition, it will get the term similarity due to the terms t1, t1', t2 or t2' using the standard Okapi formula. In effect, in the computation of phrase-simt, the query is equivalent to (t1 AND t2') or (t1' AND t2') or (t1' AND t2).

As an example, if a document has both t1 and t2', then its phrase-sim will be the same as if it has t1 and t2. However, its term similarity is computed based on the occurrences of both t1 and t2'.

4.3 Utilize pseudo-feedback for reinforcement

It is known that pseudo-feedback helps in improving retrieval effectiveness. However, it usually brings in a reasonably large number of extraneous terms. It is also possible that the synonyms and hyponyms which are brought in by the above sense disambiguation process may also consist of both useful and useless terms. Based on this intuitive idea, we suggest the following.

(1) Each of the terms brought in by one of the two processes (pseudo-feedback and sense disambiguation) will be initially be given a weight which is dependent on its correlation with the query terms (in the pseudo-feedback process) or a weight which is dependent on its frequency value in a synset (in the sense disambiguation process). These weights will be adjusted such that they are significantly below that of the original query term.

(2) When a term is brought in by both processes, their weights are added together. Based on the above description, a term which is brought in by both processes is likely to be a useful term and is therefore given a high weight. A term brought in by one but not both of the two processes will get a relatively low weight. As a consequence, even if it is extraneous, it will not adversely affect retrieval effectiveness significantly.

5. Robust Track

In the robust track, we submit 5 runs. Run 1, 3, and 5 use both the title and the description; run 2 and run 4 use the title only. WordNet is used to disambiguate word senses and supply synonyms and hyponyms in each run. Pesudo-feedback is applied to all runs. Table 1 gives the average precisions of the 5 runs over the 50 old topics, the 50 new topics, and the entire set of 100 topics.

	Run1	Run2	Run3	Run4	Run5
50 Old Queries	0.1674	0.1548	0.1622	0.1527	0.1608
50 New Queries	0.3133	0.3037	0.3125	0.3065	0.3172
100 queries	0.2404	0.2293	0.2373	0.2296	0.2390

Table 1. Average Precision for TREC 2003 Robust Track

The average precision gives the overall performance. The individual effectiveness is measured by the (a) number of topics with no relevant document retrieved in the top 10 positions and (b) the area under MAP(X)-vs-X measure where X is the number of topics (queries) having the worst mean precision and MAP(X) is the mean precision of the Xth worst topic [Robust]. These two measures reflect the robustness of any given retrieval strategy. Table 2 gives the number of topics with no relevant document in the top 10 positions for the new, the old and overall queries sets.

	Run1	Run2	Run3	Run4	Run5
50 Old Queries	8	10	10	11	10
50 New Queries	5	6	6	6	5
100 Queries	13	16	16	17	15

Table 2. Number of topics with no relevant document in the top 10 positions

Table 3 lists the area under MAP(X)-vs-X statistic information. For the entire set of 100 topics, X ranges from 1 to 25 only. For the two sets of 50 topics (50 old and 50 new), X ranges from 1 to 12 only. Worst topics are defined with respect to the individual run.

	Run1	Run2	Run3	Run4	Run5
Old 50 Queries	0.0076	0.0052	0.0065	0.0050	0.0061
New 50 Queries	0.0409	0.0354	0.0402	0.0387	0.0452
Overall	0.0141	0.0114	0.0126	0.0107	0.0124

Table 3. Area under MAP(X)-vs-X evaluation

6. Conclusion

Our TREC-2003 experiment shows that the intuitions regarding the robust retrieval are reasonable. That is the robust retrieval result can be achieved by: (1) effective use of phrases, (2) a new similarity function capturing the use of phrases, and (3) utilizing suitable synonyms and hyponyms which are properly chosen in a word sense disambiguation process. We are experimenting with more complicated techniques of word senses disambiguation in the document retrieval, which hopefully will yield much better effectiveness in the future.

Reference

[BR99] R. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval, ACM Press, Addison-Wesley, 1999.

[BS95] C. Buckley and G. Salton. Optimization of relevance feedback weights. ACM SIGIR95, p351-357

[Brill] Eric Brill. Penn Treebank Tagger, Copyright by M.I.T and the University of Pennsylvania

[GF98] David Grossman and Ophir Frieder, Ad Hoc Information Retrieval: Algorithms and Heuristics, Kluwer Academic Publishers, 1998.

[GF98] David Grossman and Ophir Frieder, Ad Hoc Information Retrieval: Algorithms and Heuristics, Kluwer Academic Publishers, 1998.

[Porter] Martin Porter, Porter Stemmer http://www.tartarus.org/~martin/PorterStemmer/index.html

[RW00] Robertson S E, Walker S. Okapi/Keenbow at TREC-8. *TREC-8*, 2000

[Robust] Robust Track Guidelines. http://trec.nist.gov/act_part/tracks/robust/robust.03.guidelines.html

Active Feedback – UIUC TREC-2003 HARD Experiments

Xuehua Shen, ChengXiang Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign

Abstract

In this paper, we report our experiments on the HARD (High Accuracy Retrieval from Documents) Track in TREC 2003. We focus on active feedback, i.e., how to intelligently propose questions for relevance feedback in order to maximize accuracy improvement in the second run. We proposed and empirically evaluated three different methods, i.e., top-k, gapped top-k, and k-cluster centroid, to extract a fixed number of text units (e.g. passage or document) for feedback. The results show that presenting the top k documents for user feedback is often not as beneficial for learning as presenting more diversified documents.

1 Introduction

For interactive information retrieval such as Web search, a user may need to interact with the search engine many times because of the mismatch of the returned results and the information need. In this case, the user often has to initiate a refined query to do the retrieval several times. But the search engine just uses the current query as the only clue about the user's information need and neglects other apparently useful information such as the user's previous queries in the same search session [10]. In this sense, the search engine responds to a user's query passively.

We believe that a search engine can actively participate in this interactive information retrieval process so that the user's effort can be reduced and retrieval performance can be improved. One interesting way for a search engine to actively participate in this process is to decide what retrieval results the search engine should present to the user during an interactive information retrieval process. Since there are several interactions in this process, when the search engine decides which documents to present to the user, it need consider not only the relevance of the documents to the user's query, but also whether presenting these documents will help the system gain feedback information from the user to improve the next search activity. In this case, the search engine should actively learn which are the best candidate documents to show to the user at any specific moment.

The HARD track of TREC 2003 makes it possible to explore this direction. In the HARD track, the number of times of information retrieval interaction is set to 2. So a search engine would have an opportunity to make use of the first interaction to improve the performance of the second (also final in this case) interaction. In the end of the first interaction, the search engine can propose questions to the user to clarify the user's information need. The search engine can then obtain answers to these questions (e.g., whether a passage is relevant) and some metadata about the information need (e.g., the purpose of the user's search activity), which can presumably be exploited to improve the performance in the second round of retrieval. An interesting and challenging research question is thus how we can best utilize the first interaction to maximize the performance improvement in the second interaction.

We focus our exploration on *active feedback*, i.e., how to intelligently propose passages/documents for user feedback. More specifically, we propose and study three methods, i.e., top-k, gapped top-k, and k-cluster centroid, to extract a fixed number of documents or passages from the initial retrieval results and present them to the user for feedback. Then we use the obtained relevant documents or passages from the feedback process to update our query model and do the second-time retrieval.

The organization of this paper is as follows. In Section 2, we briefly introduce the HARD track. Then we introduce the active feedback in Section 3. In Section 4, we describe how active feedback is used in HARD track. In Section 5, we describe our experiments and result analysis. Section 6 gives our conclusions.

2 HARD Track

The HARD track in TREC2003 is an exploration of how to achieve high accuracy retrieval from documents by leveraging additional information about the searcher and/or the search context, through techniques such as passage retrieval and using very targeted interaction with the searcher [1].

There are two runs to submit in HARD track. In the first (baseline) run, just like the traditional TREC track, given a document database and topics (each topic consists of title, description and narrative), participants use their search engines to do retrieval and submit the retrieval results. At the same time, participants submit a clarification form for each topic, which is used to solicit answers to some questions from the assessors who originally initiated the information need described by the topic. A search engine can freely propose all kinds of questions, e.g., whether some document is relevant to this topic or not. The constraint on the clarification form is that clarification form should be held in a small web page and the assessor will spend no more than 3 minutes filling out the form. We consider this step as the first interaction.

After half a month, participants get the filled out clarification forms with answers from the assessor. At the same time, some metadata about each topic such as relevant terms and searching purpose of the user are also distributed. Search engines can make full use of such information to improve retrieval performance, and submit the second-run retrieval results for the assessor to evaluate. We consider this step as the second interaction.

The HARD track puts search into the context, which allows search engines to actively infer user's information need and improve retrieval performance. We focus on how to intelligently choose passages/documents for user feedback through the clarification forms. There is a limitation on the number of questions to be asked in a clarification form, which is also true in a real interactive retrieval scenario. We want to maximize the amount of feedback information that can be obtained subject to these constraints, in hope of maximizing the retrieval performance in the second run.

3 Active Feedback

Instead of considering information retrieval as only one independent query submission activity, we consider it as an iterative process, in which the user would initiate a query, get retrieval results, and refine the query and submit it again [3]. This provides opportunities for a search engine to actively participate in the retrieval process. For example, a search engine can obtain useful information from the interaction (e.g., inferring relevance of top ranked documents through clickthrough data [4] and/or extracting informative terms from query history [10]) and improve retrieval performance in later interactions of the same search session. Currently, most, if not all, search engines passively respond to user queries and ignore the search context. For example, most search engines only use the information in the current query to generate a ranked list of documents for the user. If the user is not satisfied with the result, (s)he generally has to refine the query and submit it again. Clearly, if the search engine can play a more active role and propose intelligent questions to probe the user's further information need, the user's effort will be reduced, and the final retrieval performance will be improved as well.

Here we consider search as an iterative process. During the retrieval interaction, the documents returned by a search engine have two roles [13]: one is to provide information to the user and the other is to obtain user feedback explicitly or implicitly when the user browses these documents [5]. A search engine can be expected to learn from such explicit or implicit feedback information to improve retrieval performance later in the same search session. In order to maximize the effectiveness of such learning, especially when explicit feedback is possible, the search engine should intelligently choose appropriate questions to ask the user. For example, a question could be whether a document/passage is relevant, or whether a term describes the user's information need. We refer to this problem as *active feedback*. Essentially, active feedback is a problem of applying active learning [9, 11] to ad hoc information retrieval. A similar problem is introduced for learning a text classifier in [7], where a sequential sampling method which chooses most uncertain examples is proposed.

4 Active Relevance Feedback Experiment Design

In HARD track, for each topic, participants can make a clarification form to probe the user with questions. The first decision we face is what kind of questions we want to ask in order to obtain information for active feedback. Perhaps the most natural question to ask is whether some text unit is relevant to the topic or not. The next decision to make is what kind of unit we should present to the user. Individual terms seem to be a good choice, but presenting individual terms has two disadvantages. One is that they are often ambiguous and it is often hard for an assessor to judge precisely whether a particular term is relevant to the topic or not. The other disadvantage is judging individual terms is a boring work for the assessor since the assessor benefits very little from judging individual term relevance. However, presenting documents may not be a good choice either, because a document is generally long and sometimes the assessor may not be able to finish reading a single long document in 3 minutes. Therefore, passages appear to be a good compromise. Accordingly, we make each clarification form contain several passages (6 in this HARD track due to the limited size of the form), so that we can obtain relevance feedback on these passages. An additional benefit of presenting passages is that the assessor can benefit from reading these passages while judging their relevance.

Considering the computation efficiency, we presegment each document into several passages of similar lengths (average is 68.8 words and maximum is 208 words). We build an inverted index for all the passages, do passage retrieval and get a ranked list of passages for each topic (We do not submit this result since it is only used to create clarification forms. Instead we submit document retrieval results in the baseline run).

We proposed and explored three strategies to choose passages for the clarification form. The first one is to choose *top-k* passages from the passage retrieval results, which is the most natural method and is what an existing retrieval system would do. The second one is to choose *gapped top-k* passages from the results. For example, if we set the gap to 3 and k to 6 as we actually did in this HARD track, we will end up choosing the 1st, 4th, 7th, ..., 16th passage from the retrieval results. The third one is to choose *k-cluster centroid* passages from the results. We cluster top N passages of passage retrieval results (we set N to be 100 in this HARD track) into k clusters and choose centroid of k clusters. We use the k-medoid clustering algorithm to do clustering of top N document. And we choose J-Divergence [8] of two passages as the distance function. J-divergence is a divergence metric similar to KL-divergence. But unlike the non-symmetry of KL-divergence, J-divergence is symmetric. The underlying hypothesis for choosing gapped top-k and k-cluster centroid is that the top-k passages may be very similar so that they have redundant information. If search engines instead choose diversified top passages for the clarification form, search engines can also benefit from the active feedback similarly or even more.

When we get the feedback from the assessor, we select relevant passages/documents from the feedback and update the query model. We use mixture model [12] to update the original query used in the baseline run. Then we do the second run retrieval and get the ranked document list.

5 Experiments and Results

We use the Lemur toolkit as our search engine[2] and the KL-Divergence language retrieval model as our retrieval method[6, 12].

In the evaluation stage of HARD track, two judgment files are distributed. One is the hard evaluation judgment file and the other is the soft evaluation judgment file. In the hard evaluation judgment file, a document is relevant if not only the document is relevant to the topic but also the document matches the metadata of the topic. For the soft evaluation judgment file, a document is relevant if the document is relevant to the topic. We pick the soft evaluation judgment file as our judgment file. Since we do not use any metadata information, soft evaluation judgment file is more fair to our experiment evaluation, and is the judgments that we use in all our evaluation.

We submitted five runs. The first one is the baseline run. Then we do passage retrieval to get a ranked passage list. We use gapped top-k and k-cluster centroid strategies to create two clarification forms. After we get answers from the clarification form, we extract relevant passages(we only use relevant passages in this HARD track). We use two methods to update query model. One is to use passage index to update the query model. The other is to assume documents which have relevant passages are relevant and use document index to update the query model. The feedback method is the mixture model as presented in [12]. Then we do document retrieval and obtain results for four runs.

We summarize the mean average precision and pr@20docs in Table 1. From Table 1, we can see retrieval performance using active feedback is significantly better than that of baseline, which indicates that our feedback method is effective. It is also clear that the performance of using relevant *passages* to update the query model is better than that of using relevant *document* to update the query model. The performance of our four active feedback methods is all higher than the median performance of all HARD submissions. Among our four active feedback submissions, the UIHARD4 submission is best for average precision, which uses gapped top-k and updates the query model using passage index. But for pr@20docs, our UIHARD3 submission is the best, which uses k-cluster centroid and also updates the query model using passage index.

Submission		avg prec	pr@20docs
Baseline		0.3077	0.4854
Cluster	doc (UIHARD1)	0.3286	0.5015
	passage (UIHARD3)	0.3465	**0.5219**
	improvement(3 vs. 1)	5.4%	4.0%
Gap	doc (UIHARD2)	0.3321	0.5031
	passage (UIHARD4)	**0.3510**	0.5167
	improvement(4 vs. 2)	5.7%	2.7%

Table 1: Mean Average Precision and pr@20docs of 5 HARD track submissions.The best performance is shown in bold.

To test the hypothesis that diversified documents may be better for feedback than the natural top-k documents, we can use the top-k method as a baseline and compare it with the other two methods. Unfortunately, in our official HARD submissions, we forgot to include the results using the top-k strategy, which makes it impossible to do such analysis with the official runs.

Thus we ran some post-TREC experiments and use the judgments provided by NIST to simulate user feedback. Specifically, we do regular document retrieval and use three active feedback strategies to select k documents for relevance feedback. Then we use relevant documents to update the query; our feedback method can only learn from relevant documents. We then use the updated query to do a second retrieval. The results are shown in Table 2 and Figure 1. We can see that the k-cluster centroid method performs better than the gapped top-k method, which in turn is better than the top-k method in both average precision and precision at 20 documents, though the difference is generally small. Table 2 also shows the total number of relevant documents obtained from the feedback process for all the 48 topics for each method. It is interesting to see that the best performing method – k-cluster centroid – actually has obtained least number of relevant document examples. This suggests that the quality of the examples obtained by k-cluster centroid is probably higher than that of the examples obtained by the other two methods.

Figure 1 shows the precision-recall curve for these three methods. We can see again that at low recall level(0, 0.1, 0.2 and 0.3), performance of gapped top k strategy and k cluster centroid strategy are better than that of top k strategy. In high recall level, performance of top k strategy are slight better.

Active Feedback	avg prec	pr@20docs	#rel
top-k	0.3247	0.4979	146
gapped top-k	0.3278	0.5042	150
k-cluster centroid	**0.3299**	**0.5135**	105

Table 2: Mean Average Precision and pr@20docs of Post-HARD track.The best performance is shown in bold.

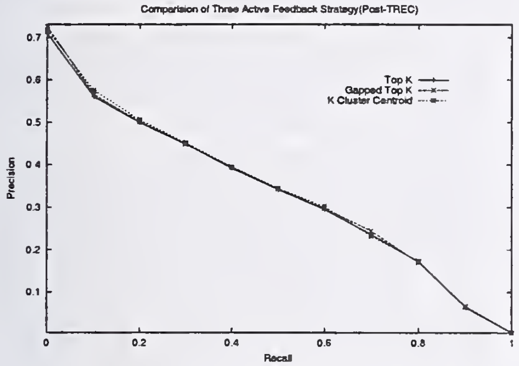


Figure 1: Average Precision at different recall levels.

Active Feedback	avg prec	pr@20docs	#rel
top-k	0.3016	0.4698	146
gapped top-K	0.3114	0.4770	150
k-cluster centroid	**0.3255**	**0.5031**	105

Table 3: Mean Average Precision and pr@20docs of Post-HARD track excluding documents used for active feedback.The best performance is shown in bold.

The results shown in Table 2 are generated based on all the relevance judgments, which means the judgments obtained from the feedback process are also included, which may be biased. Intuitively, the user really does not care where the feedback documents are ranked because the user has already seen these documents. Thus another more meaningful way to evaluate these methods is to exclude any document presented to the user in the feedback stage. This gives us the results shown in Table 3. Here we see again the same order of methods in terms of their relative performance. In fact, the difference between the methods appears to be amplified. Note that this evaluation strategy might be unfair for a method that has obtained more "easy" relevant documents in feedback, since the task becomes harder as more "easy to retrieve" relevant documents are excluded.

These results strongly suggest that just presenting the top-k documents is not optimal for active feedback. Methods that intend to return k diverse documents, such as k-cluster centroid, can be more effective.

The main difference between the experiments that we have just described and our official HARD track submissions is that we use documents instead of passages for judging relevance. Since there is no way for us to obtain equivalent feedback information judged by the official assessors based on passages for the top-k method, we decide to generate results of an approximate top-k baseline.

For the UIHARD1 and UIHARD2 official submissions, we do passage retrieval to get top k passages. Then we use documents which contain at least one of the top k passages for relevance feedback. This top-k results obtained in this way are comparable (not strictly) with UIHARD1 and UIHARD2. The three active feedback methods are compared in Table 4 and Figure 2. This time, we see that the top k method has slightly better performance in both average precision and pr@20docs,and the gapped top-k method obtained the largest number of relevant passages.

Active Feedback	avg prec	pr@20docs	#rel
top-k	**0.3373**	**0.5125**	134
gapped top-k	0.3319	0.5021	155
k-cluster centroid	0.3286	0.5063	121

Table 4: Mean Average Precision and pr@20docs of HARD track using document index to update query model. The best performance is shown in bold.

Active Feedback	avg prec	pr@20docs
top-k	0.3400	0.5177
gapped top-k	**0.3510**	0.5167
k-cluster centroid	0.3465	**0.5219**

Table 5: Mean Average Precision and pr@20docs of HARD track using passage index to update query model. The best performance is shown in bold.

For the UIHARD3 and UIHARD4 official submissions, we use passage retrieval to get top k passages, then we check relevance judgment file and consider a passage as relevant if the document containing this passage is relevant. Then we use passage index to update the query

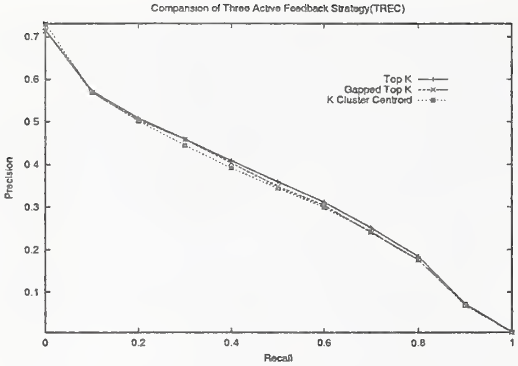


Figure 2: Average Precision at different recall levels.

model, which is used to retrieve the final results. We list our results in Table 5. The top-k results obtained in this way are comparable (again, not strictly) with UIHARD3 and UIHARD4. The three active feedback methods are compared in Table 5. This time, we see that the top K method is, again, inferior to the other two methods.

Since, strictly speaking, the top-k baseline results presented in Table 4 and Table 5 are not really comparable, it is actually hard to make any reliable inference from these two tables.

6 Conclusions

In HARD track of TREC 2003, we focused on the issue of active feedback. We proposed and evaluated three techniques for active relevance feedback, which are the top-k, gapped top-k, and the k-cluster centroid method. We found that the top-k method is not optimal for active feedback, and is worse than both the gapped top-k method and the k-cluster centroid method in a controlled design of experiments. The k-cluster centroid method, which emphasizes returning diversified documents, performs better than both the top-k and gapped top-k methods with fewer examples of relevant documents, suggesting that diversity in the presented documents may be a desirable property.

Clearly, our work represents only a very preliminary exploration of this important topic. We need to do more experiments on other data sets to draw more reliable conclusions. It would be very interesting to develop and test principled models for active feedback.

References

[1] http://ciir.cs.umass.edu/research/hard/.

[2] Lemur Toolkit 2003. http://www.cs.cmu.edu/ lemur.

[3] Tommi Jaakkola and Hava Siegelmann. Active information retrieval. In *Proceedings of Neural Information Systems(NIPS)*, 2001.

[4] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining(KDD)*, 2002.

[5] Diane Kelly and Jaime Teevan. Implicit feedback for inferring user preference: A bibliography.

[6] John Lafferty and Chengxiang Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of 24th Annual International ACM SIGIR Conference*, 2001.

[7] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of 17th International ACM SIGIR Conference*, 1994.

[8] Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991.

[9] Nicholas Roy and Andrew McCallum. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of 18th International Conf. on Machine Learning*, 2001.

[10] Xuehua Shen and ChengXiang Zhai. Exploiting query history for document ranking in interactive information retrieval (poster). In *Proceedings of 26th Annual International ACM SIGIR Conference*, 2003.

[11] Simon Tong and Daphne Koller. Active learning for parameter estimation in bayesian networks. In *Proceedings of Neural Information Systems(NIPS)*, 2000.

[12] Chengxiang Zhai and John Lafferty. Model-based feedback in kl divergence retrieval model. In *Proceedings of the 10th International Conference on Information and Knowledge Management(CIKM)*, 2001.

[13] Yi Zhang, Wei Xu, and James P. Callan. Exploration and exploitation in adaptive filtering based on bayesian active learning. In *Proceedings of the 20th International Conference on Machine Learning(ICML)*, 2003.

Improving the robustness of language models – UIUC TREC-2003 Robust and Genomics Experiments

ChengXiang Zhai, Tao Tao, Hui Fang, Zhidi Shang
Department of Computer Science
University of Illinois at Urbana-Champaign

Abstract

In this paper, we report our experiments in the TREC 2003 Genomics Track and the Robust Track. A common theme that we explored is the robustness of a basic language modeling retrieval approach. We examine several aspects of robustness, including robustness in handling different types of queries, different types of documents, and optimizing performance for difficult topics. Our basic retrieval method is the KL-divergence retrieval model with the two-stage smoothing method plus a mixture model feedback method. In the Genomics IR track, we propose a new method for modeling semi-structured queries using language models, which is shown to be more robust and effective than the regular query model in handling gene queries. In the Robust track, we experimented with two heuristic approaches to improve the robustness in using language models for pseudo feedback.

1 Introduction

Recent work on using language models for information retrieval has shown that probabilistic language models have several advantages over the more traditional retrieval models, including being able to optimize retrieval parameters automatically [7] and improve retrieval performance through better language models or estimation methods [5, 2]. Language models have also shown promising empirical results in different TREC tasks (e.g., [3]). In this year's TREC experiments, we focus on the issue of the robustness of language modeling approaches, and examine several aspects of it, including robustness in handling different types of queries, different types of documents, and optimizing performance for difficult topics.

Our basic retrieval method is the KL-divergence retrieval model with the two-stage smoothing method plus a mixture model feedback method [5, 7]. The KL-divergence retrieval model can be regarded as a generalization of the query likelihood method. It has an additional advantage of supporting model-based feedback [5]. The two-stage smoothing was originally proposed in the context of query likelihood, but we can adapt it in a straightforward way to handle a query model generated by a feedback method. We test this basic approach in the Genomics IR track to see how robust such an approach is in handling the Medline abstract documents and the gene description queries. In the Genomics IR track, we propose a new method for modeling semi-structured queries using language models, which is shown to be more robust and effective than the regular query model in handling gene queries. In the Robust Track, we explore how we can automatically set parameters and evaluate how well our approach perform on difficult topics. In the Robust track, we experimented with two heuristic approaches to improve the robustness in using language models for pseudo feedback.

2 Retrieval with language models

In this section, we describe our retrieval approaches. The basic retrieval formula we use is the Kullback-Leibler (KL) divergence between the query language model and the document language model [1, 5]. This method is a generalization of the query likelihood retrieval method proposed in [4] and can support feedback more naturally than the query likelihood method.

Suppose that a query $\mathbf{q}$ is "generated" by a unigram language model $p(\mathbf{q} \mid \theta_Q)$ with θ_Q denoting the parameters, and similarly, a document $\mathbf{d}$ is generated by a unigram model $p(\mathbf{d} \mid \theta_D)$ with θ_D denoting the parameters. Let $\widehat{\theta}_Q$ and $\widehat{\theta}_D$ be the estimated query and document language models respectively. The KL-divergence retrieval formula scores d with respect to q with the following *negative* KL-divergence function [5]:

$$\begin{aligned} -D(\widehat{\theta}_Q \,\|\, \widehat{\theta}_D) &= \sum_w p(w \mid \widehat{\theta}_Q) \log p(w \mid \widehat{\theta}_D) \\ &\quad +(-\sum_w p(w|\widehat{\theta}_Q) logp(w|\widehat{\theta}_Q)) \end{aligned}$$

Since the second term on the right-hand side of the formula does not depend on d, it can be ignored for the purpose of ranking documents. Thus the scoring is essentially based on the first term, i.e., the cross entropy of the document model and the query model. In general, the computation of this cross entropy involves a sum over all the words that have a non-zero probability according to $p(w|\widehat{\theta}_Q)$. However, when $\widehat{\theta}_D$ is based on the following general smoothing scheme, the computation would only involve a sum over those that both have a non-zero probability according to $p(w|\widehat{\theta}_Q)$ and occur in document d. Such a sum can be computed much more efficiently with an inverted index. See [3] for a detailed explanation of this.

Clearly, the retrieval performance of the KL-divergence would depend on how we estimate the document model θ_D and the query model θ_Q. Smoothing of θ_D is very important, and an effective smoothing method is the two-stage smoothing method proposed in [7], where it has been shown to achieve optimal or near optimal performance with completely automatic parameter setting. However, this smoothing method as presented in [7] is based on the query likelihood retrieval method, which is unable to incorporate feedback naturally. Although some techniques for tuning retrieval parameters automatically are proposed in [7], they are based on without pseudo feedback. This is also why the performance reported in [7] is not as good as the best official TREC results on the same data, which are usually obtained based on pseudo feedback. The KL-divergence retrieval formula can be shown to be a generalization of the query likelihood method, and can naturally support feedback as query model updating. Two specific methods for performing feedback using language models are proposed in [5]. In order to automatically tune the retrieval parameters for pseudo feedback, we extend the two-stage smoothing method so that it can work with a query model using KL-divergence. Since maximizing the likelihood is equivalent to minimizing the KL-divergence, this extension is not unnatural. We now explain this extension in detail.

The basic idea of two-stage smoothing is to decouple the two roles of smoothing (i.e., the estimation role and query modeling role [6]) so that we can capture the intrinsic connections between the parameter values and the data. Specifically, the document model θ_D is smoothed first with Dirichlet prior (to implement the estimation role) and then with a simple linear interpolation with some query background model (to implement the query modeling role). To apply this idea to the KL-divergence retrieval formula, we also use Dirichlet prior as the first-stage smoothing method. That is, our estimated document language model θ_D is given by Dirichlet prior [6]. According to this method,

$$p(w|\widehat{\theta}_D) = \frac{c(w,d) + \mu p(w|\mathcal{C})}{|d| + \mu}$$

where $c(w, d)$ is the count of word w in d, μ is a query independent smoothing parameter, and $p(w|\mathcal{C})$ is reference language model estimated using the whole document collection. μ can be set using leave-one-out cross validation on the document collection [7]. To implement the second-stage smoothing, we first interpolate the estimated document model $\widehat{\theta}_D$ with the query background model $p(w|U)$ to obtain a two-stage smoothed document model $\widetilde{\theta}_D$, and then compute the KL-divergence $D(\widehat{\theta}_Q||\widetilde{\theta}_D$. That is,

$$p(w|\widetilde{\theta}_D) = (1-\lambda)p(w|\widehat{\theta}_D) + \lambda p(w|U)$$

where λ can be interpreted as the amount of noise that we believe exists in the query, and $p(w|U)$ is the user's query background model. Since no information is available about the noise in the query, we simply approximate $p(w|U)$ by $p(w|C)$. λ can also be estimated in a very much similar way as in [7], except that we minimize the KL-divergence between the document mixture model and the query model instead of maximizing the query likelihood given the document mixture model.

The collection language model $p(w|\mathcal{C})$ is typically estimated by $\frac{c(w,\mathcal{C})}{\sum_{w'} c(w',\mathcal{C})}$, or a smoothed version $\frac{c(w,\mathcal{C})+1}{V+\sum_{w'} c(w',\mathcal{C})}$, where V is an estimated vocabulary size (e.g., the total number of distinct words in the collection). One advantage of the smoothed version is that it would never give a zero probability to any term, but in terms of retrieval performance, there will not be any significant difference in these two versions, since $\sum_{w'} c(w', \mathcal{C})$ is often significantly larger than V.

Having discussed how we use two-stage smoothing to estimate a document model, let us now look at the query model. The simplest way to estimate θ_Q is to use the maximum likelihood estimator based on the query text, which gives us essentially the empirical query distribution:

$$p_{ml}(w|\widehat{\theta}_Q) = \frac{c(w, \mathbf{q})}{|q|}$$

Using this estimated value, the KL-divergence scoring formula is essentially the same as the query likelihood retrieval formula, thus we essentially have the two-stage smoothing retrieval method as presented in [7]. This is what we use for the initial round of retrieval. A more interesting way of computing $p(w|\widehat{\theta}_Q)$ is to exploit feedback documents. Specifically, we can interpolate the simple $p_{ml}(w|\widehat{\theta}_Q)$ with a *feedback model* $p(w|\theta_F)$ estimated based on feedback documents. That is,

$$p(w|\widehat{\theta}_Q) = (1-\alpha)p_{ml}(w|\widehat{\theta}_Q) + \alpha p(w|\theta_F) \quad (1)$$

where, α is a parameter that needs to be set empirically.

To compute θ_F, we assume a two component mixture model for the feedback documents, where one component model is $p(w|\theta_F)$ and the other is $p(w|\mathcal{C})$. The likelihood of the feedback documents is thus

$$\log p(\mathcal{F} \,|\, \theta_F) =$$

$$\sum_{i=1}^{k}\sum_{w} c(w; d_i)\log((1-\nu)p(w \mid \theta_F) + \nu p(w \mid \mathcal{C}))$$

where, $F = \{d_1, ..., d_k\}$ is the set of feedback documents, and ν is yet another parameter that indicates the amount of "background noise" in the feedback documents, and that needs to be set empirically. Given ν, the feedback documents $\mathcal{F}$, and the collection language model $p(w|\mathcal{C})$, we can use the EM algorithm to compute the maximum likelihood estimate of θ_F [5]. For the sake of efficiency, we truncate the model θ_F so that we only keep k word with the highest probabilities according to $p(w|\theta_F)$. We thus have four parameters to set in performing pseudo feedback: (1) α controls the influence of feedback documents on the new query model; (2) ν indicates the amount of noise that we believe exists in the feedback documents; (3) k is the number of terms to select for query model updating; and (4) the number of documents to use for feedback. It is reasonable to assume that the choice of k has a relatively insignificant influence since the words excluded generally have smaller probabilities. But we know that the feedback performance is sensitive to both α and ν [5], and no doubt, also to the number of top documents to be used for pseudo feedback. This makes the feedback approach less robust.

Thus a major research question we address is how we can make this feedback process more robust. We propose two strategies:

- **Weighted pseudo feedback:** The basic idea of this strategy is to discount the contribution of documents according to their rank in the results. Specifically, we assume the probability of relevance of a document ranked at rank r to be 1/r. Thus the top document has "full" contribution in feedback, whereas the second and third documents are discounted by a factor of $1/2$ and $1/3$ respectively. We hoe this strategy would make the performance less sensitive to the choice of the number of documents used for pseudo feedback because if we use more documents, those documents would just be discounted more.

- **Applying two-stage smoothing after feedback:** The idea of this strategy is to re-estimate the second-stage smoothing parameter λ *after* we obtain an updated query model using feedback so that we can "compensate" for any non-optimality in the feedback parameter setting by adjusting λ according to the "noise" in the newly updated query model.

Both strategies have shown some positive effect in our preliminary experiments with the past TREC data.

3 Genomics IR Track

We participated in the primary task of the genomics IR track, and focused on studying how we may apply language modeling approaches to this new retrieval task.

A critical difference between the genomics IR task and a regular ad hoc task is that a query in the genomics IR task has a structure. More specifically, a gene query has several parts, including an official name, which is usually a long noun phrase, several symbols, each being a unique identifier for the gene, and protein product names. Such a query has a clear disjunctive structure in that matching either a gene symbol or the *complete* gene name would indicate relevance. For example, topic 1 has the following name and symbols:

```
OFFICIAL_GENE_NAME
activating transcription factor 2
OFFICIAL_SYMBOL ATF2
ALIAS_SYMBOL HB16
ALIAS_SYMBOL CREB2
ALIAS_SYMBOL TREB7
ALIAS_SYMBOL CRE-BP1
```

These symbols are *unique* identifiers for the gene, so matching one of the symbols is a sufficiently strong evidence for a document to be relevant, and is as good as matching the whole phrase "activating transcription factor 2". Thus although both "transcription" and "CREB2" are a single word term, "CREB2" should be weighted much higher than "transcription".

Such a semi-structured query clearly violates the basic assumption made in a language modeling approach that the query can be treated as a sample drawn from a language model. As a result, if we use the basic language modeling approach as is, the gene symbols may be underweighted. Indeed, in our preliminary experiments with the training topics, we found that the basic language modeling approaches did not perform as well as well-tuned vector space models.

To address this problem, we propose a new way to estimate our query language model θ_Q so that it can better model a disjunctive query. Our idea is to first compute the empirical word distribution for each "component query", and then take an average of them. Formally, suppose $q = \{q_1, ..., q_k\}$ is a disjunctive semi-structured query, where q_i is a "component query". Our estimated query model is given by

$$p(w|\theta_Q) = \frac{1}{k}\sum_{i=1}^{k}\tilde{p}(w|q_i)$$

where $\tilde{p}(w|q_i)$ is the empirical word distribution of component query q_i, and is computed as $\frac{c(w,q_i)}{|q_i|}$ where $c(w, q_i)$ is the count of word w in the component query q_i and $|q_i|$ is the length of q_i. Note that each gene symbol is one component query so if q_i is a gene symbol s, then we have $\tilde{p}(s|q_i) = 1$, whereas if w is a word in the name, then $\tilde{p}(w|q_i) = 1/|q_i|$, which is usually smaller than 1 because a name phrase almost always has more than one

Query	**No feedback**				**Pseudo feedback**			
	MAP	Pr@0.1	Pr@10doc	Rec@1000	MAP	Pr@0.1	Pr@10doc	Rec@1000
Official Name	0.089	0.168	0.086	367/566	0.101	0.176	0.094	389/566
Symbols	0.147	0.289	0.131	390/566	0.169	0.313	0.142	466/559
All words	0.138	0.277	0.116	391/566	0.149	0.278	0.11	424/566
All distinct words	0.160	0.310	0.140	493/566	0.171	0.321	0.134	514/566
Name + Symbols (ad hoc weighting)	0.178	0.338	0.152	519/566	**0.193**	**0.366**	**0.158**	**524/566**
Name + Symbols (model avg.)	0.185	0.385	0.154	496/566	**0.200**	**0.393**	**0.150**	**511/566**

Table 1: Different query models with/without pseudo feedback. Boldface indicates the results of two official submissions.

word. Thus, in effect, our average query model would favor matching a symbol. Intuitively, this average query model normalizes word counts so that the total contribution from each component query is equal, reflecting the disjunctive semantics of the query. As a baseline method, we also experimented with heuristically duplicating each gene symbol in a query to "boost" its weight.

Our two official submissions represent these two different approaches to model semi-structured queries. In both official submissions, we used symbols and the official name in the gene query description, and used the retrieval approach as described in Section 2 except that we did not apply two-stage smoothing after pseudo feedback. We used top 10 documents for pseudo feedback, set both α and ν to 0.5, and used top 20 terms for query model updating.

Since many biology names involve digits and hyphens, we used a slightly different tokenization method than what we normally use to handle the special syntax of gene names. Specifically, we retain all the digits as well as any hyphen that connects at least one digit; in other words, we only remove a hyphen when it connects two letters. This method gives a slight improvement in performance based on the training topics. To test the robustness of our method, we deliberately did not remove any stop word, as we believe that this is best handled through appropriate language modeling. We did not apply stemming either.

Table 3 shows the results of our two official submissions along with some other experiments where we use different types of queries and test them with/without feedback. The results of the two official runs are highlighted in boldface.

We can make the following observations:

1. Using gene name alone is least effective, significantly worse than using any other versions of the query. This is probably because the words in a name are too general and thus not discriminative. Indexing the whole name as a phrase may help improve the performance.

2. Using symbols is more effective than using all the words, which means that we treat the whole gene description as one long text query. This suggests that the symbols alone are the most important information in the query, and there are noise terms in other part of the query (mostly the gene name and protein names).

3. Using all words but ignoring word frequency (i.e., "All distinct words" in the table) improves performance significantly. Since symbols always have a frequency of 1, this suggests that when pooling all words together, we are overweighting the regular words in those names. The fact that "all distinct words" also performs significantly better than using the symbols alone indicates that the gene name is also very useful.

4. The two official runs all use names combined with symbols and both put more weight on a symbol term. We see that both perform better than other runs, indicating that treating such a disjunctive query as a regular text query is non-optimal and we need to increase the weight for short component queries (i.e,. symbols).

5. Comparing the two different methods for modeling disjunctive queries (i.e., ad hoc weighting by duplication and language model averaging), we see that the average query model approach performs slightly better in mean average precision (MAP) and much better in precision at recall level of 0.1 (i.e., the front end precision). However, it has a lower recall at 1,000 documents and a lower precision at 10 documents in feedback runs (i.e., our official runs).

4 Robust Track

Our goal for the robust track is to evaluate and improve the robustness of the language model based retrieval approach described in Section 2. This approach is a combination of the two-stage smoothing method which has been shown to

Topics	Query Type	MAP	Pr@0.1	Pr@10doc	Rec@1000	percent(pr@10=0)	MAP area (worst 12 topics)
Old	title	0.108	0.275	0.268	2034/4416	18%	0.0057
	description	0.113	0.293	0.266	1827/4416	26%	0.0031
	title+desc	0.140	0.380	0.326	2178/4416	12%	0.0080
New	title	0.303	0.608	0.430	1392/1658	6.0%	0.024
	description	0.372	0.684	0.494	1404/1658	12%	0.0207
	title+desc	0.392	0.741	0.528	1476/1658	10%	0.0466

Table 2: Different types of queries on old and new topics.

Topics	Method	MAP	Pr@0.1	Pr@10doc	Rec@1000	percent (pr@10=0)	MAP area (worst 12 topics)
Old	no re-est (20 terms)	0.113	0.293	0.266	1827/4416	26%	0.0031
	2s re-est (20 terms)	0.1178	0.311	0.280	1892/4416	24%	0.0036
	2s re-est (50 terms)	0.1250	0.326	0.282	1933/4416	26%	0.0034
New	no re-est (20 terms)	0.372	0.684	0.494	1404/1658	12%	0.0207
	2s re-est (20 terms)	0.364	0.680	0.486	1402/1658	16%	0.0193
	2s re-est (50 terms)	0.375	0.675	0.498	1409/1658	16%	0.0189

Table 3: Effect of two-stage smoothing after feedback on old and new topics.

be quite robust w.r.t., parameter setting [7], and the mixture model feedback approach [5], which is effective but sensitive to several parameters.

As discussed in Section 2, we propose two strategies to reduce the sensitivity of our approach to the setting of parameters: (1) Estimate the probability of relevance for each document to be used for pseudo feedback so as to discount the contribution of documents based on their ranks. (2) Apply the two-stage smoothing method to re-estimate the smoothing parameters after query model updating. In some sense, this method is to "bypass" the problem of choosing optimal values for the some of the feedback parameters. In our preliminary experiments with the old topics, these methods perform better than several baseline approaches

We did minimum preprocessing (only stemming, no stop word removal) to test the robustness of our method, and submitted five runs with one run using title of the query (UIUC03Rt1, three runs using the description part of the query (UIUC03Rd1, UIUC03Rd2, UIUC03Rd3), and one run using both the title and the description (UIUC03Rtd1). In all our submissions we applied the first strategy (i.e., discounting documents based on ranks), and in UIUC03Rd2 and UIUC03Rd3, we also applied the second strategy. UIUC03Rd2 and UIUC03Rd3 differ in the number of top terms selected to update the query model (20 and 50 respectively). In all cases, we used top 10 documents for pseudo feedback, and set α and ν both to 0.5 as in all our experiments.

In Table 4, we compare the performance of different versions of the queries on both old topics and new topics. It is clear that, in all cases, using the regular precision measures over all the topics, "title+description" performs much better than "description only", which performs better than "title only", though in the case of old topics, the recall of "description only" is worse than that of "title only". However, it is very interesting to see that for both old and new topics, "title only" has the least number of topics with no relevant document in top 10 documents, and its MAP area for the worst 12 topics is also better than the description. "title+description" has the largest MAP area for the worst 12 topics, but, compared with the "title only" run, it actually has more topics which have no relevant document in top 10 documents. These results clearly show that using titles appear to help improve the performance on difficult topics. This may be because the words in the query title are more accurate and when we use more words from other parts they just bring up many non-relevant documents perhaps because the relevant documents do not have a good match in vocabulary with the words in these other parts. Another possibility is that our approach may not be discriminative enough to give title words more weight and due to the fact that we do not remove stop words, the description part just adds too many noisy words. It would thus be interesting to apply our disjunctive query model to model the title and description parts to see if it can improve the performance, which we will explore in the future.

In Table 4, we compare the three "description only" runs. One (baseline) run applies two-stage smoothing only for the initial run and after the feedback the same smoothing parameters are used. The other two runs further apply two-stage smoothing to *re-estimate* the smoothing parameters after we obtain an updated query model. These two runs differ in the number of top terms to be selected for query model updating. Here we can make the following observations:

1. Looking at all runs in which we use 20 terms for feedback, we see that for old topics, the re-estimation strategy helps improve performance by all the regular measures averaged on all the topics, but it does not improve the performance on difficult topics. Indeed, it actually hurts the performance for difficult topics. For new topics, the re-estimation strategy not only hurts the performance for difficult topics, it is worse by all measures.

2. Comparing all the runs involving re-estimation, we see that for old topics, using 50 terms for feedback is better than using 20 terms when averaging over all the topics, but is worse on difficult topics. For new topics, it appears to perform similarly to using 20 terms.

Thus our re-estimation strategy does not seem to help improve the robustness. We have not yet had more experiments to examine whether the first strategy – discounting feedback documents based on their ranks – is useful.

5 Conclusions

In this paper, we report our experiments in the TREC 2003 Genomics Track and the Robust Track. A common theme that we explored is the robustness of a basic language modeling retrieval approach. We examine several aspects of robustness, including robustness in handling different types of queries, different types of documents, and optimizing performance for difficult topics. Our basic retrieval method is the KL-divergence retrieval model with the two-stage smoothing method plus a mixture model feedback method. In the Genomics IR track, we propose a new method for modeling semi-structured queries using language models, which is shown to be more robust and effective than the regular query model in handling gene queries. In the Robust track, we experimented with two heuristic approaches to improve the robustness in using language models for pseudo feedback. One is the discount feedback documents based on their ranks and the other is to apply two-stage smoothing after feedback to re-estimate the smoothing parameters. Preliminary result analysis appears to suggest that the re-estimation of smoothing parameters does not really help. But we need to do more experiments and analysis in order to make reliable conclusions.

References

[1] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of SIGIR'01*, pages 111–119, Sept 2001.

[2] V. Lavrenko and B. Croft. Relevance-based language models. In *Proceedings of SIGIR'01*, pages 120–127, Sept 2001.

[3] P. Ogilvie and J. Callan. Experiments using the lemur toolkit. In *Proceedings of the 2001 TREC conference*, 2002.

[4] J. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the ACM SIGIR'98*, pages 275–281, 1998.

[5] C. Zhai and J. Lafferty. Model-based feedback in the KL-divergence retrieval model. In *Tenth International Conference on Information and Knowledge Management (CIKM 2001)*, pages 403–410, 2001.

[6] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of ACM SIGIR'01*, pages 334–342, Sept 2001.

[7] C. Zhai and J. Lafferty. Two-stage language models for information retrieval. In *Proceedings of ACM SIGIR'02*, pages 49–56, Aug 2002.

Relevance Propagation for Topic Distillation UIUC TREC-2003 Web Track Experiments

Azadeh Shakery, ChengXiang Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign

Abstract

In this paper, we report our experiments on the Web Track TREC-2003. We submitted five runs for the topic distillation task. Our goal was to evaluate the standard language modeling algorithms for topic distillation, as well as to explore the impact of combining link and content information.

We proposed a new general relevance propagation model for combining link and content information, and explored a number of specific methods derived from the model. The experiment results show that combining link and content information generally performs better than using only content information, though the amount of improvement is sensitive to the document collection and tuning of parameters.

1 Introduction

We participated in the topic distillation task of TREC-2003 Web Track with two goals:

1. Evaluating our basic language modeling algorithms for topic distillation.
2. Exploring how to effectively combine the link information with the content information.

The reports about this task from TREC-2002 seem to indicate that a standard content-based retrieval algorithm, such as Okapi, performs very well for topic distillation. So we decided to test how a basic language modeling approach would perform. Specifically, we used the standard query likelihood method with Dirichlet prior smoothing [8] as well as the two-stage language modeling algorithm, which is supposed to tune the parameters automatically according to the query [9].

Intuitively, the link information may provide some clues as to whether a page is a key resource or not. It is thus interesting to explore how we may combine the link information with the content information to improve the accuracy in finding key resources. We propose a new general relevance propagation model for combining link information with the content information. The relevance propagation model naturally captures the intuition that a web page's value depends on the page's content value (self relevance) as well as the values of all the pages that are linked to this page (through either inlinks or outlinks). It allows every page to propagate its self relevance value to the neighboring pages through links to generate a "hyper-relevance" value for each page.

We consider several interesting special cases of this general relevance propagation model, and derive several specific methods for combining link information with content information. We use the query likelihood method with Dirichlet prior smoothing as well as the two-stage smoothing for computing the self relevance value of a page (solely based on the content). We then apply these different propagation methods to propagate every page's self relevance value along links to obtain a hyper-relevance value for each page. The hyper-relevance values are used to produce the final ranking for selecting key resources.

The experiment results show that combining link and content information generally performs better than using only content information, though the amount of improvement is sensitive to the document collection and tuning of parameters.

The rest of this paper is organized as follows. In Section 2, we briefly describe the language modeling algorithms that we experimented with. In Section 3, we present the general Relevance Propagation Model and three specific propagation methods. In Section 4, we briefly discuss how to implement the general relevance propagation model. In Section 5, we analyze the results of our experiments on TREC-2002 and TREC-2003 data. Finally, in Section 6, we give the conclusions and future research directions.

2 Language Modeling Algorithms

Our basic content-based retrieval algorithm is the Kullback-Leibler (KL) divergence between the query language model and the document language model [1, 7]. This method is a generalization of the query likelihood retrieval method proposed in [5] and can support feedback more naturally than the query likelihood method. In this retrieval method, in order to compute a score for a document w.r.t. a query, we first compute a query language model and a document language model, and then compute the KL-divergence of these two models. The main issue in computing the document language model is smoothing, and we explore two smoothing methods – Dirichlet prior and two-stage smoothing. We also explore two different ways of computing a query language model, one using the query text alone and one using both the query text and some pseudo feedback documents. The details of these methods can be found in [10].

From these different combinations, we decide to use the following approaches as our baseline "content only" runs:

1. **Dirichlet Prior Smoothing, no feedback**: This is the simplest language modeling approach.

2. **Two-stage smoothing, mixture model feedback**: This is a relatively sophisticated language modeling approach.

3 Relevance Propagation Model

The results of TREC-2002 Web Track did not seem to favor approaches combining link and content information. The best official results were from Tsinghua university [11]; they explored the use of out-degree, as well as anchor text to find key resources. What they found was that anchor-text was useful, but out-degree was not. The second and third best results on TREC-2002 Web Track were from City University [2] and Chinese-Academy [6] respectively. None of these groups used the link information in finding the key resources.

However, intuitively, the content similarity of a page to a query, on its own, may not be sufficient for selecting a key resource, and the link information can be useful in finding key resources. A good resource is a page whose content is related to the query topic, and which has links to and/or from other related resources. So two factors are important in selecting good resources: the content of the page and the relevance of the pages which have links to/from the page.

Motivated by these observations, we propose a new general relevance propagation model for combining link and content information. Specifically, We define the "hyper-relevance" score of each page as a function of three variables, the content similarity of the page to the query ("self relevance"), a weighted sum of the hyper-relevance scores of the pages that point to the page, and a weighted sum of the hyper-relevance scores of the pages that are pointed to by the page. Formally, the relevance propagation model can be written as :

$$h(p) = f(S(p), \sum_{p_i \rightarrow p} h(p_i) w_I(p_i, p), \sum_{p \rightarrow p_j} h(p_j) w_O(p, p_j))$$

where h(p) is the hyper-relevance score of page p, S(p) is the content similarity between the page p and the query (i.e., "self relevance"), and w_I and w_O are weighting functions for in-link and out-link pages, respectively.

In principle, the choice of function f could be arbitrary. An interesting choice is a linear combination of the variables shown below:

$$\begin{aligned} h(p) &= \alpha S(p) + \beta \sum_{p_i \rightarrow p} h(p_i) w_I(p_i, p) \\ &\quad + \gamma \sum_{p \rightarrow p_j} h(p_j) w_O(p, p_j)) \end{aligned}$$

$$\alpha + \beta + \gamma = 1$$

The hyper-relevance scores can be computed iteratively until they converge to a limit, which is the final score of the page for ranking; more detail is presented in Section 4.

We now consider three special cases of this general relevance propagation model. Each special case corresponds to some reasonable user behavior.

3.1 Weighted In-Link

This model of user behavior is quite similar to the model of PageRank [4], except that it is not query-independent. The random surfer is given a start page at random. He keeps traversing links until he gets bored and starts from another random page. The probability that the random surfer visits a page is its hyper-relevance score.

This model has some characteristics which distinguish it from PageRank. First, it works on a subset of pages which are in the working set, rather than working on the whole set of data. (The details of constructing the working set is given in section 4.2) One of the properties of pages in the working set is that their content similarity to the query is above a threshold, so they can not be completely unrelated to the query. Second, the probability that the random surfer traverses an edge is proportional to the similarity of the target page and the query. That is, it is more probable that the user traverses and edge which leads to a more similar page, than jumping to a less similar page. Besides, when the random surfer gets bored, it jumps to new pages based on their similarity to the query.

This behavior can be formally modeled as follows. In each iteration, the new score of each page is computed as:

$$h(p) = \alpha S(p) + (1 - \alpha) \sum_{p_i \rightarrow p} h(p_i) w(p_i \rightarrow p)$$

$$w(p_i \rightarrow p) \propto S(p)$$

3.2 Weighted Out-Link

In this model, we assume that given a page to a user, he reads the content of the page with probability α and he traverses the outgoing edges with probability $(1 - \alpha)$. We also assume that it is more likely that the user traverses an edge that leads to a page whose content-similarity is higher.

Formally, we compute the hyper-relevance of each page iteratively using:

$$h(p) = \alpha S(p) + (1 - \alpha) \sum_{p \rightarrow p_j} h(p_j) w(p \rightarrow p_j)$$

$$w(p \rightarrow p_j) \propto S(p_j)$$

In each iteration, the hyper-relevance of each page is computed as a combination of its self-relevance and the hyper-relevance of the pages that it points to. The pages that are linked from a page do not have the same impact on its weight. Pages whose contents are more similar to the query are assumed to have more impact on the score of the page than those which are less similar. This effect is indicated in $w(p \rightarrow p_j)$. The hyper-relevance scores of pages are computed iteratively until they converge to a value, which are used to rank the pages.

3.3 Uniform Out-Link

In this special case, we assume that at each page, the user reads the content of the page, and with probability $(1 - \alpha)$ he reads all the pages that are linked from the page. So the score of each page will be equal to a combination of its self relevance and the scores of all its linked pages.

Formally the hyper-relevance of each page is computed iteratively. In each iteration, the hyper-relevance of each page is:

$$h(p) = S(p) + (1 - \alpha) \sum_{p \rightarrow p_j} h(p_j)$$

4 Implementation Issues

4.1 Preprocessing

We indexed all the web pages before dealing with queries. We used the Lemur toolkit for document indexing [3]. For document tokenization, we used the Fox stopword list and Porter stemmer.

4.2 Constructing the Working Set

We do not run our experiments on the whole set of data. Instead, for each query, we first construct a working set of pages and then find the top ranked pages among the pages of this subset.

To construct the working set, we first find the top 100000 pages which have the highest content similarity to the query from the whole collection of pages. We assume that the pages that are not among these pages can not be key resources for the given query. From these 100000 pages, a small number (about 200) of the most similar pages are selected to be the core set of pages. We then expand the core set to the working set by adding the pages that are among the 100000 pages and which point to the pages in the core set or are pointed to by the pages in the core set. We then run our experiments on these working sets. Note that each working set is specific to a query.

4.3 Computing the Hyper-relevance Values

In order to be able to compute the scores efficiently, we should come up with a way to find the hyper-relevance scores easily and in a feasible amount of time. In our implementation, we use matrix multiplication iteratively to compute the scores. We can use existing matrix multiplication methods to speed up the computation process.

Suppose that query Q and parameters are given. Our goal is to compute the limit hyper-relevance scores. To this end, we construct a square matrix $U_{n \times n}$ where n is the number of pages in the working set.

To construct the matrix, we first set all the entries to *zero* ($u_{ij} = 0, 1 \leq i, j \leq n$). We then add the influence of out-links: for each $1 \leq i, j \leq n$, we add $\beta \times w(p_i \rightarrow p_j)$ to the entry u_{ij}. Then we add the influence of in-links by adding $\gamma \times w(p_i \rightarrow p_j)$ to the entries u_{ji}. The only thing that remains is to add the effect of self content similarity. To add this effect, we add $\alpha \times S(p_i)$ to all the entries $u_{ij}, 1 \leq j \leq n$.

We then construct a vector H_n of hyper-relevance scores. At the beginning, we set all the entries in H to be $\frac{1}{n}$. The vector H should maintain the property that the sum of all the entries equal to *one* in every step.

In each step, we multiply U by H and we normalize the H vector. Each entry h_i in H corresponds to the hyper-relevance value of page p_i. It is easy to show that after multiplication, the hyper-relevance values are updated according to the general relevance propagation model.

If we do the multiplication and normalization iteratively, the hyper-relevance scores will converge to a limit, which is the final score we use for sorting results.

5 Experiment Results

5.1 Preliminary Experiments

Run	P@10	Content	Links	
			In	Out
Dir. Base	0.255	√		
Dir + W. IN	0.267	√	√	
Dir + W. OUT	0.265	√		√
Dir + U. OUT	0.265	√		√

Table 1: Experiments on TREC-2002 Data

The results of our algorithms on TREC-2002 data were quite promising. We explored the three presented methods, as well as a couple of other ways of propagating the relevance scores, and all the methods outperformed the baseline content-only language modeling method.

Table 5.1 summarizes our experiments on TREC-2002 data. Figure 1 shows the results of our algorithms compared to the baseline method.

As can be seen from the chart, all the methods have improvements over baseline.

Uniform out-link always perform better than baseline, while Weighted out-link has improvements for $\alpha > 0.2$ and Weighted in-link has improvements for $\alpha > 0.7$.

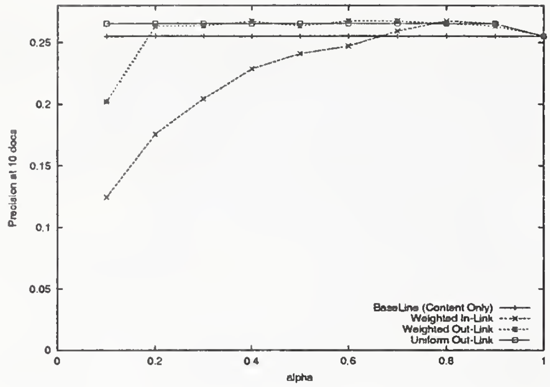


Figure 1: Experiments on TREC-2002 Data

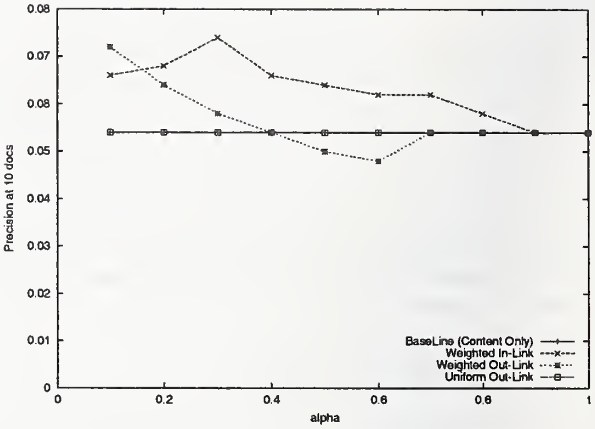


Figure 2: Experiments on TREC-2003 Data

5.2 Official TREC-2003 Results

Table 5.2 summarizes our experiments on TREC-2003 data. The second column in the table is precision at 10 for the submitted runs, while the third column shows precision at 10 when the parameters are tuned to get the optimal results.

Run	P@10		Cont.	Links	
	$\alpha = 0.8$	$\alpha = 0.1$		In	Out
Dir. Base	0.054*	0.054	√		
Dir+W. In	0.058*	0.066	√	√	
Dir+W. Out	0.054*	0.072	√		√
Dir+U. Out	0.054*	0.054	√		√
2s Base	0.064*	0.064	√		
2s+W. In	0.066	0.078	√	√	
2s+W. Out	0.062	0.082	√		√
2s+U. Out	0.062	0.062	√		√

Table 2: Experiments on TREC-2003 Data
$*$: Submitted Runs

Unlike our results on TREC-2002 data, our experiments on TREC-2003 data were not that promising. Figure 2 shows the results of our algorithms on TREC-2003 data. We see that for a majority settings of the parameter, the weighted out-link approach actually improves the performance clearly, but the α value that we obtained by training on the 2002 data corresponds to a bad setting for TREC-2003. When we optimize the parameter α on the TREC 2003 test set, both the weighted in-link and weighted out-link methods outperform the baseline.

As can be seen from the chart, the baseline method did not give good results on this year's data and although the proposed algorithms improved the ranking a bit, the overall precision was not good.

Comparing our two baseline runs (i.e., "2s Base"and "Dir Base"), we see that the two-stage smoothing coupled with mixture model for feedback performs significantly better than the the Dirichlet prior smoothing, confirming the effectiveness of two-stage smoothing and dictionary based feedback.

5.3 Discussion

We tried to find the reason for our poor results for this year's task. One difference between TREC-2002 data and TREC-2003 data is the number of relevant documents for each query. The average number of relevant documents per query is 31.48 for TREC-2002, while it is only 10.32 for TREC-2003 data. To find out whether this is a reason for our poor performance. We do experiments on two subsets of queries: a selected subset of queries from TREC-2002 whose average number of relevant documents is smaller than the average over all the queries and a selected subset of queries from TREC-2003 whose average number of relevant documents is larger than the average over all queries.

Our subset of queries from TREC-2002 data contains 25 queries with 10.48 relevant documents on average. The subset of queries from TREC-2003 data contains 20 queries with 19 relevant documents on average.

We tried our algorithms on these two subsets of queries. Figure 3 and Figure 4 show the results for the TREC-2002 subset and TREC-2003 subset, respectively.

As can be seen from Figure 3, the performance is not as good as the performance we obtained using the whole set of queries, but is not as poor as the results we obtained for TREC-2003 either.

On the other hand, Figure 4 shows that the performance is better than the performance we obtained using the whole set of queries, but is not as good as the TREC-2002 results.

What we can conclude is that small number of relevant documents per query can be a factor in our poor performance in TREC-2003, but there should be other reasons as well, that we are trying to find out.

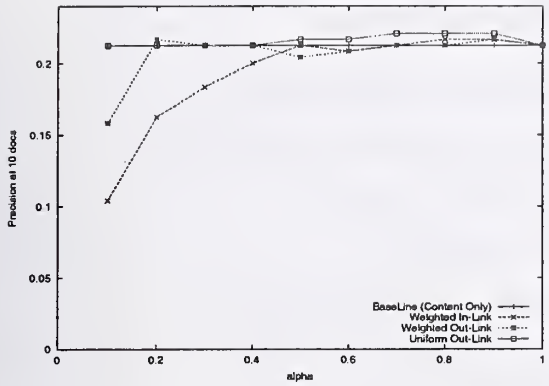


Figure 3: Experiments on the Subset of TREC-2002 Data

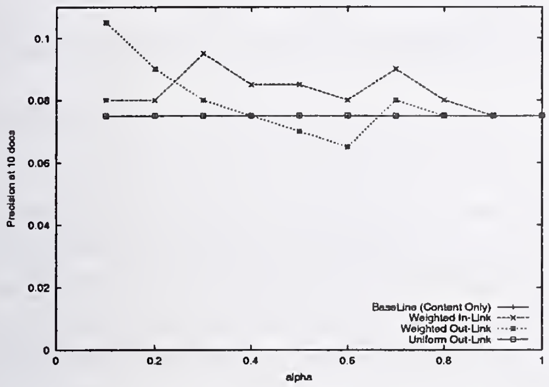


Figure 4: Experiments on the Subset of TREC-2003 Data

6 Conclusions and Future Directions

We explored two language modeling approaches for the topic distillation task: (1) basic query likelihood method with Dirichlet prior smoothing, and (2) two-stage smoothing with mixture model feedback. The results show that the the two-stage smoothing with feedback significantly outperforms the query likelihood method, confirming the effectiveness of two-stage smoothing [9] and mixture model feedback method [7]

We also proposed a new general relevance propagation model for combining link and content information, and explored a number of specific methods derived from the model. The experiment results show that combining link and content information generally performs better than using only content information, though the amount of improvement is sensitive to the document collection and tuning of parameters.

For the future work, we plan to do more experiments to find out what factors affect the performance of our algorithms. We will also explore how to tune the parameters for obtaining the optimal results.

References

[1] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of SIGIR'2001*, pages 111–119, Sept 2001.

[2] A. MacFarlane. Pliers at trec 2002. In *Proceedings of TREC 2002*, 2002.

[3] P. Ogilvie and J. Callan. Experiments using the lemur toolkit. In *Proceedings of the 2001 TREC conference*, 2002.

[4] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[5] J. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the ACM SIGIR*, pages 275–281, 1998.

[6] H. Xu, Z. Yang, B. Wang, B. Liu, J. Cheng, Y. Liu, Z. Yang, and X. Cheng. Trec 11 experiments at cas-ict: Filtering and web. In *Proceedings of TREC 2002*, 2002.

[7] C. Zhai and J. Lafferty. Model-based feedback in the KL-divergence retrieval model. In *Tenth International Conference on Information and Knowledge Management (CIKM 2001)*, pages 403–410, 2001.

[8] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of SIGIR'2001*, pages 334–342, Sept 2001.

[9] C. Zhai and J. Lafferty. Two-stage language models for information retrieval. In *Proceedings of SIGIR'2002*, pages 49–56, Aug 2002.

[10] C. Zhai, T. Tao, H. Fang, and Z. Shang. Improving the robustness of language models: Uiuc trec-2003 robust and genomics experiments. In *Notebook of TREC 2003*, 2003.

[11] M. Zhang, R. Song, C. Lin, S. Ma, Z. Jiang, Y. Jin, Y. Liu, and L. Zhao. Thu trec 2002: Web track experiments. In *Proceedings of TREC 2002*, 2002.

Experiments in Novelty, Genes and Questions at The University of Iowa

David Eichmann,[1,2] Padmini Srinivasan,[1,3] Marc Light,[1,4]
Hudong Wang,[2] Xin Ying Qiu,[3] Robert J Arens[2] and Aditya Sehgal[2]
[1]School of Library and Information Science
[2]Computer Science Department
[3]Department of Management Sciences
[4]Department of Linguistics
The University of Iowa
{david-eichmann, padmini-srinivasan, marc-light}@uiowa.edu

The University of Iowa participated in the novelty, genomics and question answering tracks of TREC-2003.

1 – Novelty

Our system for novelty this year is a refinement of that used for last year. One of the challenges in preparing for the 2002 novelty track was the nature of the training data. Our experiments with using the 2002 evaluation data as training data for this year have shown that the novelty task can in fact be tuned to trade off precision and recall - at least across the range of what a given system can detect as novel. Our tuning involved establishing a similarity threshold for sentence relevance and an new entity threshold for novelty.

We decided to focus our development experiments for this year on a composite precondition of simple similarity matches between the topic definition and the candidate document and the topic and the candidate sentence. If both measures exceed the declared threshold, a sentence is declared relevant. Additionally, if the number of novel elements present in the sentence is above a declared number, the sentence is declared novel. ‘Element’ here can be a noun phrase or a named entity. For the available training topics, this proved to be remarkably responsive to tuning between precision-focused runs and recall-focused runs for novelty as well as the more predictable relevance decision.

Our official runs involved the following approaches for the defined tasks:

Task 1 (detect relevance and novelty). Proceed as described above, making a judgement on relevance based upon similarity, and given that as a guard, make a judgement on novelty based upon the existence of new entities.

Task 2 (given relevance, detect novelty). Load the given relevance judgements, and proceed as per task 1 for novelty.

Task 3 (given relevance and novelty for first 5, detect relevance and novelty for last 20). Load relevance judgements and entities present in the first five documents, and then proceed as per task 1 for both relevance and novelty.

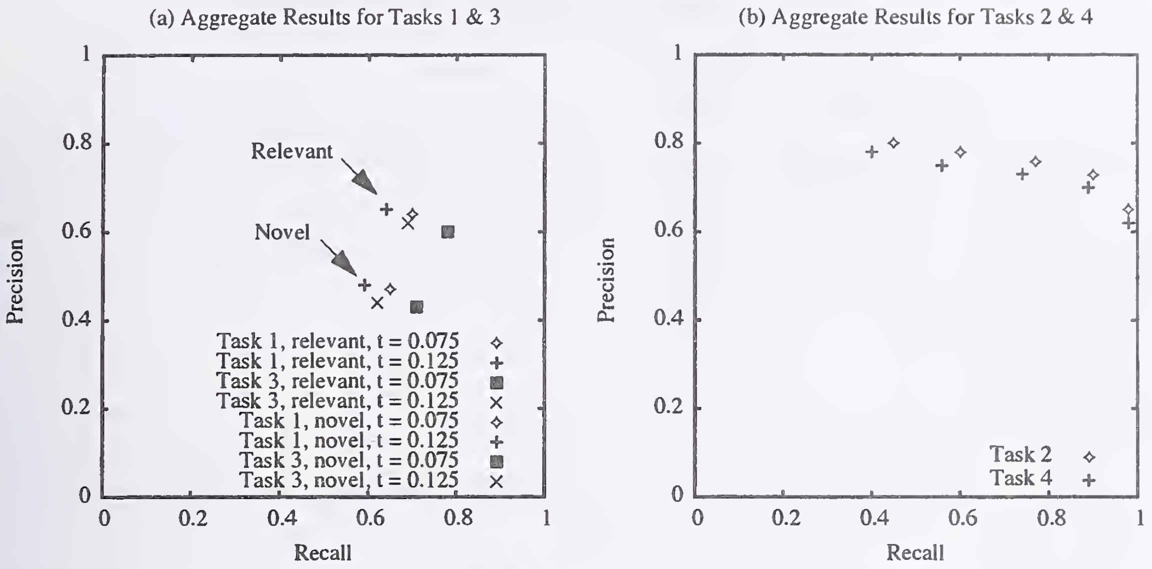


Figure 1: Novelty Task Results

Task 4 (given relevance for all and novelty for first 5, detect novelty for last 20). Load as per task 2 for relevance and task 3 for novelty, run as per task 2.

Aggregate Results for All Tasks

We submitted two sets of runs for tasks 1 and 3, with similarity thresholds of 0.075 and 0.125 and a new entity/noun phrase threshold of 1 and five runs each for tasks 2 nd 4, using new entity/noun phrase thresholds of 0-4. Table 1 shows the full results for all runs. As show in Figure 1a, there is a distinct performance differential between relevance and novelty detection, but relative performance among the four threshold/task conditions is comparably positioned for relevance and novelty. As might be expected, increasing the similarity threshold slightly improves precision at a slight cost to recall. More interestingly, precision of the task 1 configurations is similarly higher than their task 3 counterparts. Having the additional information regarding the first five documents for each topic slightly improves recall, but at the cost of precision. In other words, we can achieve better precision in both relevance and novelty by *not* looking at the initial pool of documents available in task 3.

As shown in Figure 1b, there is a very regular recall/precision trade-off achieved when varying the number of entities and/or noun phrases required to declare a sentence novel, given that it is relevant. It is also interesting to note that the oddity noted for tasks 1 and 3 is still present for tasks 2 and 4. Indeed, in this case, task 4 with relevance and novelty information available for the first five documents uniformly performs less well for both precision and recall for all thresholds. We find this intriguing and plan on further analyzing the cause of these results.

Conditioning by Topic Type

Figures 2 and 3 show the performance per topic for relevance and novelty, broken out by event and opinion topics. There appear to be no major trends to distinguish event topics from opinion

Table 1: Summary Results for Novelty Track

Run	Task	Sim. Thresh.	Entity/NP Thres.	Relevant			New		
				Prec.	Recall	F	Prec.	Recall	F
UIowa03Nov01	1	0.075	1	0.64	0.70	0.594	0.47	0.65	0.480
UIowa03Nov02	1	0.125	1	0.65	0.64	0.568	0.48	0.59	0.461
UIowa03Nov03	2	–	0	–	–	–	0.65	0.98	0.767
UIowa03Nov04	2	–	1	–	–	–	0.73	0.90	0.794
UIowa03Nov05	2	–	2	–	–	–	0.76	0.77	0.746
UIowa03Nov06	2	–	3	–	–	–	0.78	0.60	0.659
UIowa03Nov07	2	–	4	–	–	–	0.80	0.45	0.555
UIowa03Nov08	3	0.075	1	0.60	0.78	0.606	0.43	0.71	0.466
UIowa03Nov09	3	0.125	1	0.62	0.69	0.585	0.44	0.62	0.448
UIowa03Nov10	4	–	0	–	–	–	0.62	0.98	0.741
UIowa03Nov11	4	–	1	–	–	–	0.70	0.89	0.767
UIowa03Nov12	4	–	2	–	–	–	0.73	0.74	0.712
UIowa03Nov14	4	–	3	–	–	–	0.75	0.56	0.617
UIowa03Nov15	4	–	4	–	–	–	0.78	0.40	0.505

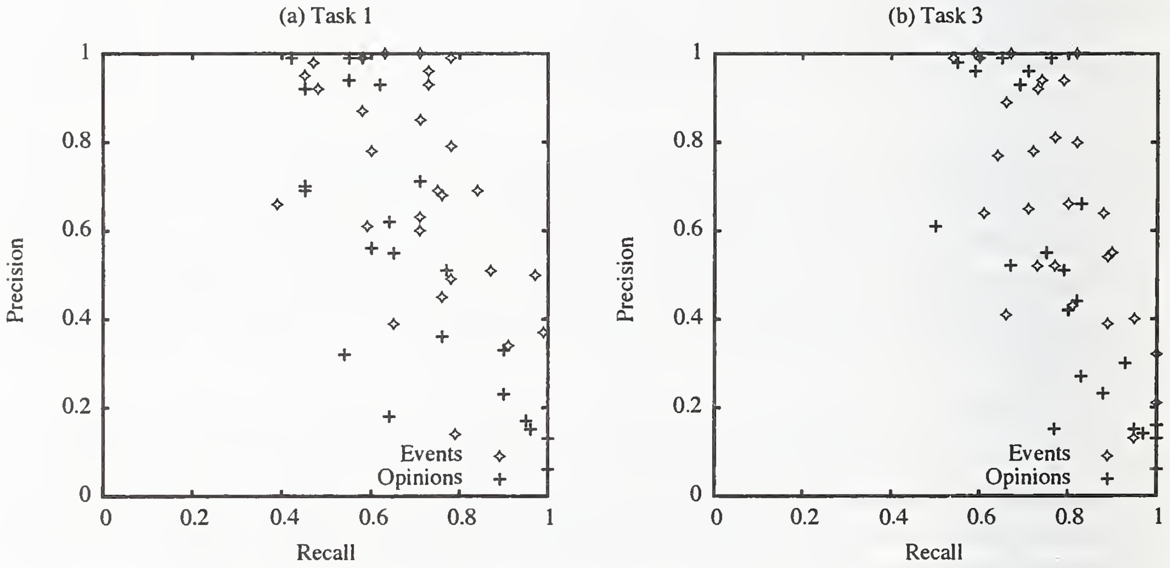


Figure 2: Novelty Task, Relevant by Topic Type

topics, although events do seem to edge opinions out in general. It does appear that the additional information available in task 3 results in a 'tightening' of the topic clouds for both relevant and novelty over the topic clouds for task 1.

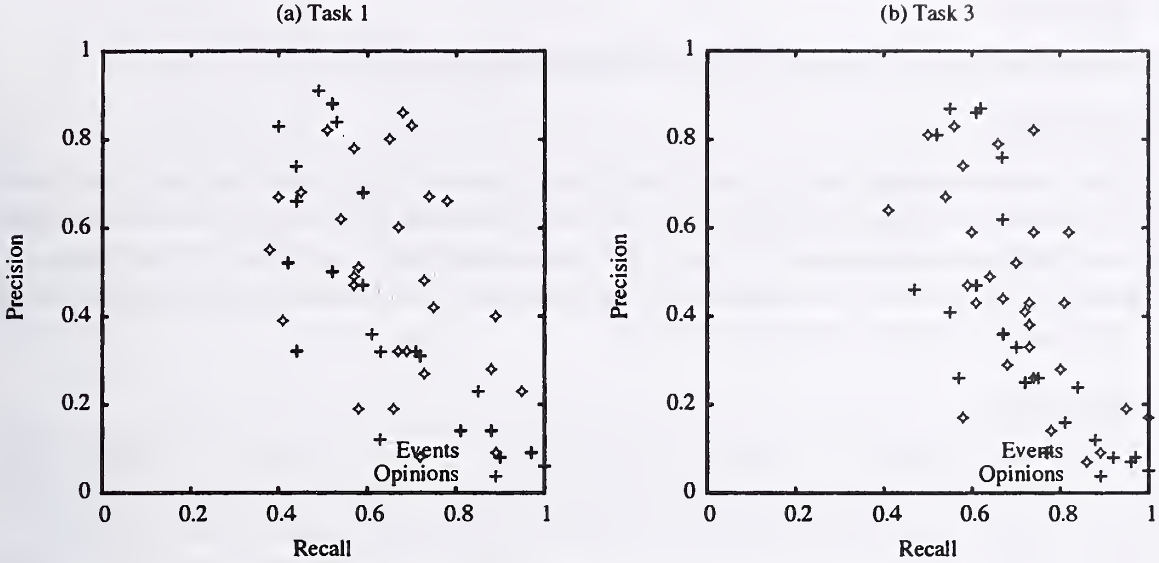


Figure 3: Novelty Task, New by Topic Type

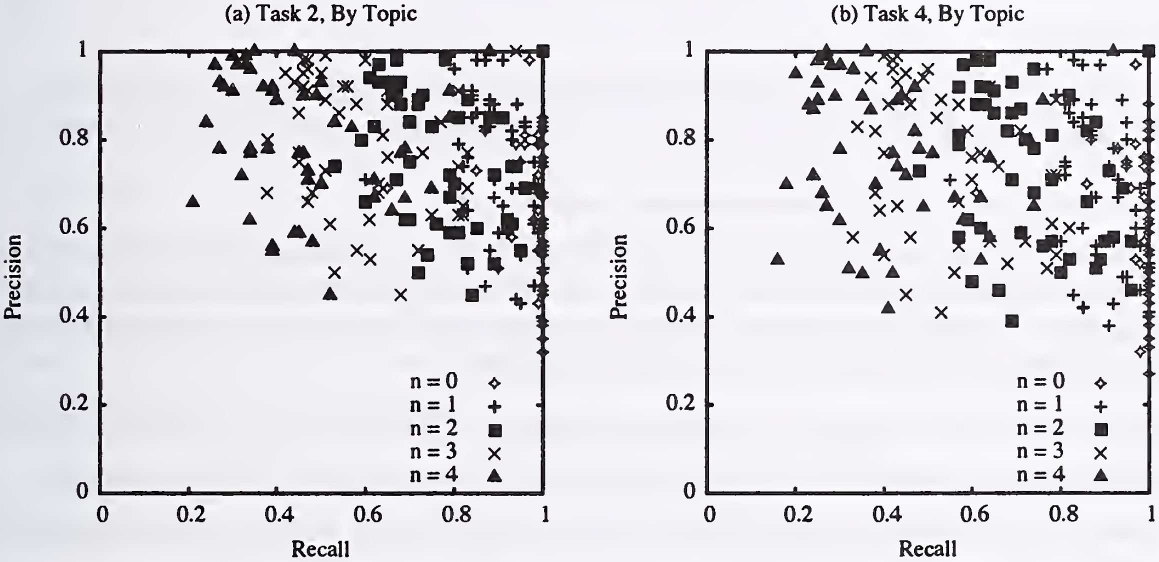


Figure 4: Novelty Task, New by Entity/NP Threshold

Effect of New Entities and Noun Phrases

Figure 4 shows a topic-level breakout of performance for each run for tasks 2 and 4. For the degenerate condition, n = 0, we see perfect recall generally, but our dual guard of both sentence and document level similarity does lower recall for some topics. Increasing the threshold to n = 1 improves precision overall, as seen in Figure 1b in aggregate. Further increases in the threshold generate little benefit with respect to precision and seriously erodes recall. Based upon this we have concluded that a single new entity or noun phrase can serve as an indicator of novelty. Our future work will focus on the analysis of poorly performing topics for n=1.

2 – Genomics

We participated in both the primary and secondary tasks in this track.

Primary task:

This is a baseline run where we used SMART as the retrieval system with atc weighting on queries and documents. Queries were generated from the different fields provided to us including gene name, symbols, product names. A few low level experiments were conducted with different weighting schemes and stemming options. We also tried using document classifiers (SVM) to limit the document set, but the results were not good.

Secondary task:

Each abstract sentence was classified to gauge its likelihood as a source of a GeneRIF. A sentence classifier was built using GeneRIF entries in LocusLink excluding those that were in the secondary.txt file and their abstracts. For feature selection an in house tokenizer was used and idf weights computed against a reference subset of 211,457 MEDLINE abstracts selected independent of this track.

Training Set:

GeneRIF entries (excluding the 'test' set as described above) were used to identify abstracts. 90% of the abstracts were used as training and 10% as testing for model/parameter selection.

Selection of positive and negative sentence samples.

Several methods were tried. But first, sentences in title, last sentence and first sentence are found to be most relevant, thus other sentences are discarded. Our methods involve a measure called pDice. This measures the percentage of words in a sentence that are in the GeneRIF entry corresponding to the abstract in which the sentence occurs.

Method 1: GeneRIF sentences are positive samples, low pDice sentences are negative samples.

Method 2: High pDice sentences are positive samples, low pDice ones are negative samples.

Method 3: GeneRIF and high pDice sentences are positive samples, low pDice ones are negative samples.

We used SVM classifier technology, specifically LIBSVM java classifier, with most parameters at default value. We also used EPSILON_SVR SVM, RBF kernel function. Positive/Negative class ratio is not used because it doesn't help. The best model found uses GeneRIF statements as positive samples and sentences with pDice<0.25 as negative samples. SVM gives each sentence score, the larger the score the more likely it is to be a GeneRIF. A weighting scheme was also used to emphasize titles, first sentences, and last sentences. The best weighting scheme on the test set was 5:0:1 respectively which is almost the same as saying select titles only. The best model and parameters was selected for use on secondary.txt and corresponding abstracts to generate result, for the official TREC run.

3 – Question Answering

This year marks the first evaluation of our complete implementation of an extraction-based QA system. By shifting natural language parsing forward in the process, we can amortize this very expensive step against a number of downstream extraction processes that mine the text for named entities, relationships, etc. Redefinition of extraction specifications hence does not require reparsing of the source text. We have implemented tgrep-like extraction grammar designed for predicate-based extensibility using it in mapping sentence parse trees to relational structure. This overall approach handles not only factoid answers, but definitional answers and those requiring inference across multiple extracted relationships.

Each document in the corpus is decomposed into doc-id / sentence pairs, with the sentence being the unit of analysis from that point. Each sentence is then POS-tagged and fed to the CMU link grammar parser. The parse tree for the sentence is then attributed with the POS tags for each word. Processing both queries and documents using this scheme allows us to establish both the nature of the query (using a fairly typical taxonomy) and the nature of the needed answer. This is particularly useful with respect to identification of candidate phrases in sentences and scoring of these phrases against the goal of the query. Sentences are then matched against the set of extraction patterns, populating a set of relations used to answer queries derived from the questions.

The availability of the parse tree for the phrase allows for elision of subordinate clauses that can cause answers to span too long a string and for extraction of likely answers through heuristic matching of, for example, a subordinate clause immediately trailing a mention of a candidate named entity.

We view our results for this year as very preliminary for two key reasons. The first is operational – a few days before the deadline a database failure cost us the full parse of the corpus and we were only able to reparse the top fifty documents for each question in the time remaining. The second is a developmental one – we have only begun the specification of our extraction pattern framework, and coverage is limited to

- persons' titles, ages and a minimal set of interpersonal relationships;
- location of organizations (e.g., "Seattle-based Microsoft"); and
- relative location of place names (e.g., "the resort, five miles east of Seattle").

Factoid Questions

The preliminary nature of our extraction patterns is probably most evident for factoid questions. Our pattern sets are insufficiently rich to provide sufficient coverage of potential questions, and hence the number of correct answers we generate is modest. As shown in Table 2, there is interesting potential in the low levels of unsupported and inexact answers relative to correct answers. We also have a comparatively high level of NIL answer recall, particularly given our level of correct answers. This is easily explained when the number of NIL answers returned is considered - ~20% of all questions. This is directly attributable to failure to extract sufficient information with the available patterns – we are returning so many NILs that we are catching those questions that actually have no answer in the corpus.

Table 2: QA Track, Factoids

Run	U	X	R	Accuracy	# NIL returned	NIL P	NIL R
UIowaQA0301	3	4	14	0.034	100	0.100	0.333
UIowaQA0302	2	2	17	0.041	173	0.087	0.500
UIowaQA0303	3	2	17	0.041	98	0.102	0.333

List Questions

Our implementation for this year had no support for list identification or extraction. Any coverage of answers in this category was purely accidental...

Table 3: QA Track, Lists

Run	Ave. F
UIowaQA0301	0.002
UIowaQA0302	0.002
UIowaQA0303	0.004

Definition Questions

We do believe that the approach that we are taking with extraction holds good promise for definition questions. As shown in Table 4, performance for this category of question is very different than that for factoids and lists.

Table 4: QA Track, Definitions

Run	Ave. F
UIowaQA0301	0.214
UIowaQA0302	0.231
UIowaQA0303	0.048

Breaking out performance of individual questions, as shown in Figure 5, we see that there is a broad spread of performance, but there are a large number of questions with no answers provided.

Figure 6 shows our performance in relation to the number of vital and total facts connected to a question. For questions where our system is performing well, there are a relatively small number (~2-5) of vital facts and a modest number (~10) of total facts.

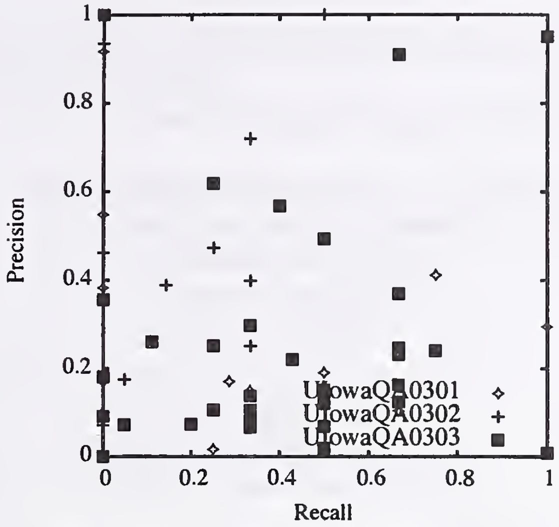


Figure 5: QA Task, Definition Questions

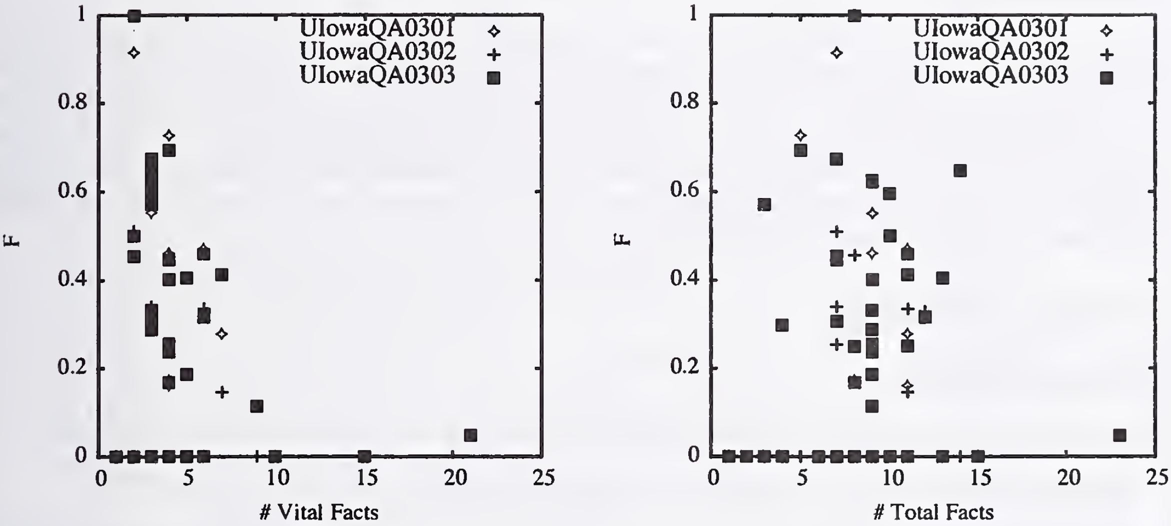


Figure 6: QA Task, Definition Richness

Question Answering using the DLT System at TREC 2003

Richard F. E. Sutcliffe, Igal Gabbay, Michael Mulcahy, Kieran White

Documents and Linguistic Technology Group
Department of Computer Science
and Information Systems
University of Limerick
Limerick, Ireland

+353 61 202706 Tel
+353 61 202734 Fax
Richard.Sutcliffe@ul.ie Email
www.csis.ul.ie/staff/richard.sutcliffe URL

1. Introduction

This article outlines our participation in the Question Answering Track of the Text REtrieval Conference organised by the National Institute of Standards and Technology. This was our second year in the track and we hoped to improve our performance relative to 2002. In the next section we outline the general strategy we adopted, the changes relative to last year and the approaches taken to the three question types, namely factoid, list and definition. Following this the individual system components are described in more detail. Thirdly, the runs we submitted are presented together with the results obtained. Finally, conclusions are drawn based on our findings.

2. Outline of System

2.1 Overall Strategy

In common with most other QA systems we use the following general strategy to answer questions:

- **Question analysis**: Process the input query attempting to find its type (e.g. who or colour).
- **Document retrieval**: Formulate a search query based on the results of the previous stage. Use this together with a search engine indexed on the document collection to produce a list of candidate documents which are likely to contain answers to the question.
- **Named entity recognition**: Based on the query type identified in the first stage, search for appropriate named entities (NEs) in the candidate documents which co-occur with terms derived from the query.
- **Answer selection**: Decide which NE (or NEs) should be chosen as the answer.

We now outline how this strategy was adapted to handle the three types of question.

2.2 Factoids

Factoid questions formed the majority (413) of the total of 500 in the collection. They are intended to ask about straightforward pieces of information which can be extracted from free text fairly readily. Our approach to these was based entirely on traditional NEs, i.e. numbers, places, person names and so on.

Question Type	Example Question	Candidate Answer
abbrev	’What does ACLU stand for?’	American Civil Liberties Union
airport_name	’What is the name of the airport in Dallas Ft. Worth?’	Dallas Fort Worth International Airport
colour	’What color is the top stripe on the U.S. flag?’	Red
company	What company manufactures Sinemet?’	Hangzhou MSD Pharmaceutical
currency	’What is the currency of Denmark?’	Kroner
distance	’How far is it from Earth to Mars?’	249 million miles
gen_name	’What is the name of the chart that tells you the symbol for all chemical elements?’	Periodic Table
how_did_die	’How did Cleopatra die?’	asp bite
how_many3	’How many time zones are there in the world?’	24
how_much	’What percent of the nation’’s cheese does Wisconsin produce?	28 percent
length_of_time	' How long is a quarter in an NBA game?'	12 minutes
name_part	' What is Britney Spears' ' middle name?'	Jean
nickname_state	' What is the Bluegrass state?'	Kentucky
population	' What is the population of Iceland?'	275000
sci_name	' What is the scientific name for red ants?'	Solenopsis invicta
speed	' How fast does light travel through space?'	186,000 miles per second
temp	' How hot is the sun?'	two million degrees centigrade
title	' What book did Rachel Carson write in 1962?'	Silent Springs
translat	' How do you say "cat" in the French language?'	chat
unknown	' What passage has the Ten Commandments?'	Exodus
what_city	' What city is Disneyland in?'	Anaheim
what_continent	' What continent is the world' ' s largest dessert on?'	Africa
what_county	' What county is San Antonio Texas in?'	Bexar
what_country	' What country is Aswan High Dam located in?'	Egypt
what_state	' What state was Amelia Earhart born in?'	Kansas
when	' When was "Cold Mountain" written?'	1997
when_date	' What date did the U.S. civil war start?'	April 12th 1861
where	' Where is Mount Olympus?'	Greece
who	' Who created the literary character Phineas Fogg?'	Jules Verne

Table 1: Question Types used in the DLT system. The second column shows a sample question for each type. The third column shows sample answers. These are all of appropriate types for the question but are not necessarily correct. Fifteen question types handled by the system did were not used. They are anthem, atomic_number, atomic_weight, country_religion, element, planet, profession, state_motto, state_nickname (the opposite of nickname_state), what_airport_code, when_interval, when_month, when_week_day, when_year and where_airport.

The type of the question as identified in the first stage was directly mapped onto one or more named entities which were then searched for in the text. For example if the question was identified as being of type what_city then the NEs used were nea_x_us_city and nea_x_non_us_city. The answer to the question was defined to be the NE which scored best by one of two measures. The highest_scoring method looked at how many query-derived terms co-occurred with the NE. The highest_google method used the World Wide Web (WWW) to predict the NE most likely to be the correct answer by adopting an algorithm similar to that of Magnini et al. (2002).

2.3 Lists

The approach to answering the X list questions was identical to that for factoids except in the answer selection stage. Here, multiple answers were selected based on their exceeding a simple threshold.

2.4 Definitions

In order to answer the Y definition questions, the NE recognition stage was adapted to find instances of simple phrasal patterns based around terms from the query. These had been developed in another project concerned with scientific definitions. All such phrases were extracted during answer selection. The first two stages of processing were the same as for factoids and lists.

In the next section we describe the various components of the system in more detail.

3. DLT System Components

3.1 Summary of Enhancements

Our 2002 TREC system was constructed in a short time frame and was thus very basic. A number of significant extensions were made to the system for this year which we summarise here. Firstly additional query types were added, taking the total from 19 plus ‘unknown’ up to 43 plus ‘unknown’. Examples of the 29 types actually used this year are shown in Table 1 with the remaining fifteen types listed in the caption. Secondly the process of query analysis was completely changed in order to allow a strategy for formulating and subsequently simplifying search expressions to be implemented. Thirdly documents in the Aquaint collection were indexed and searched using a commercial engine (DTSearch, 2000) rather than relying on the NIST TOPDOCS files. Fourthly recognisers for various NEs were added, including one for recognising general names. Finally, an answer re-ranking component using WWW hit counts was developed. These enhancements are described below.

3.1 Query Type Identification

As before this was accomplished using simple keyword combinations and without carrying out syntactic parsing of the query. 24 additional types were added relative to last year. However, many of these did not in fact come up in the test questions (see Table 1). Once again an ‘unknown’ type was adopted to allow a default strategy for queries not falling into other categories.

3.2 Query Analysis

Following type identification the query is subjected to a detailed analysis to assist in the process of search expression formulation as follows:

- Initial words and phrases are removed;
- Capitalised word sequences and expressions within quotation marks are recognised;
- Alternatives for all-capitals words and initial sequences are computed (e.g. ACLU could also be A.C.L.U. and T.E. could be TE);

Query Type	C	NC	R	X	U	W	Total
abbrev	2	0	0	0	0	2	2
airport_name	1	0	0	0	0	1	1
colour	3	0	1	0	0	2	3
company	3	0	0	0	0	3	3
currency	1	0	0	0	0	1	1
distance	17	3	2	0	0	18	20
gen_name	5	2	0	0	0	7	7
how_did_die	23	0	4	0	0	19	23
how_many3	39	6	4	0	0	41	45
how_much	6	1	1	0	0	6	7
length_of_time	3	0	0	0	0	3	3
name_part	2	0	0	0	0	2	2
nickname_state	1	1	0	0	0	2	2
population	2	1	0	0	0	3	3
sci_name	1	0	0	0	0	1	1
speed	4	1	0	0	0	5	5
temp	5	0	0	0	0	5	5
title	3	1	0	0	0	4	4
translat	3	0	1	0	0	2	3
unknown	51	97	6	1	0	141	148
what_city	16	0	2	0	1	13	16
what_continent	4	0	3	0	0	1	4
what_county	1	0	0	0	0	1	1
what_country	22	2	1	0	0	23	24
what_state	0	2	0	0	0	2	2
when	39	0	3	0	0	36	39
when_date	6	0	0	2	0	4	6
where	1	0	0	0	0	1	1
who	23	9	5	0	0	27	32
	287	126	33	3	1	376	413

Table 2: Results by Query Type. The columns C and NC show the numbers of queries of a particular type which were classified correctly and not correctly. Those classified correctly are then broken down into Right, ineXact, Unsupported and Wrong.

- Alternative formulations for numbers are computed (e.g. Apollo-11 could be Apollo-Eleven);
- Stopwords are removed;
- The tense of any remaining verbs is changed to simple past.

3.3 Search Expression Formulation

Based on the results of query analysis a search expression is composed. All searches are boolean. Remaining query terms (i.e. quotations, capitalised word sequences or individual words, each possibly in disjunction with alternatives) are assigned an importance score using a scheme reminiscent of Magnini et al. (2002). Quotations score 10, capitalised word sequences 9, numbers 8, pure nouns and verbs 7, superlative adjectives 6, pure adjectives 2, pure adverbs 1 and every other term 5. A 'pure' noun is one which can not have any other part-of-speech and similarly for pure verbs etc. Terms are then ordered by increasing score and then joined with AND operators to make a single boolean query. This is then used to search for documents.

3.4 Document Retrieval

Before the runs, the Aquaint collection was indexed by treating each paragraph (marked by a <p> tag) as a separate document. During retrieval a boolean query as formulated in the previous stage is submitted to the search engine and the first *n* matching documents (i.e. paragraphs) are returned. *n* was set to 30 throughout. If no documents are returned the least significant term (i.e. the first) is removed and the search is repeated. This process continues until at least one document is returned or no terms remain.

3.5 Named Entity Recognition

NE recognition is similar to last year and uses our own module which is based on grammars. Some extra types were added including nea_x_title and nea_x_general_name. The former recognises quotes expressions while the latter can recognise capitalised expressions with spurious matches being eliminated using simple heuristics. Following Clarke et al. (2003) queries of unknown type are answered by searching for general names.

3.6 Answer Selection

In order to decide which NE candidate (or candidates in the case of list questions) should be returned, two strategies for answer selection were used. The first is highest_scoring, where we return the NE occurring in a context which matches terms in the query best. The second is highest_google which uses a similar algorithm to Magnini et al. (2002). Specifically for candidate answer we

- Submit the candidate answer to Google with search terms and record the number of hits;
- Submit the candidate answer to Google alone and record the number of hits (it will be many more);
- Divide the first value by the second.

During answer selection for factoid questions the candidate with the highest score is chosen. For list questions a threshold of 0.03 was used.

4. Runs and Results

Two runs were submitted. The first used highest_scoring for answer selection while the second used highest_google. The results of Run 2 are shown in Table 2. The first two columns show the numbers of queries which were classified correctly (C) and incorrectly (NC) broken down by query type. 287 of the 413 factoid questions were classified correctly i.e. 69%. However if unknown and gen_name queries (which are effectively the same) are disregarded the rate of success in recognising known query types is 231 out of 258 which is 90%. As noted before, fifteen query types did not come up at all.

Query Num	Query Text	Answer	Supporting Doc	Text Extract
1925	What did Ozzy Osbourne bite the head off of?	bat	NYT19981111.0479	Maybe this is sacrilege (or at least the devil-metal equivalent), but I've always preferred bat-biting rocker Ozzy Osbourne's solo material to his work with Black Sabbath.
1925	What did Ozzy Osbourne bite the head off of?	bat	APW20000823.0047	One of his most notorious stunts was biting off a bat's head during a 1982 concert in Des Moines, Iowa.
1939	How did Einstein die?'	ruptured abdominal aortic aneurysms'	20000314_NYT	Some 15,000 die from ruptured abdominal aortic aneurysms each year. <new para> Albert Einstein died from one in 1955 when he was 76.
2004	What do the opposite sides of a die add up to?	seven	NYT19981202.0181	For instance, the opposite sides of a die always add to seven.
2037	How did Iowa get its name?	ouaouia	NYT19991026.0143	Iowa (from the Siouan "ouaouia," meaning "one who puts to sleep"),
2059	How did Chicago get its name?	"she-kag-ong,"	NYT19981008.0164	have gotten its name from the Ojibwa words "she-kag-ong," meaning "wild onion place."
2259	How did Hawaii become a state?	NO ANSWER		
2262	How did Cincinnati get its name?	given its name by Arthur St. Clair	NYT19991028.0282	Founded in 1788, it was given its name by Arthur St. Clair, governor of the Northwest Territory.
2262	How did Cincinnati get its name?	Cincinnatus	NYT19991028.0282	The society was named for Cincinnatus, a Roman patriot who returned to his farm after saving his city in battle, and this city, in turn, was named for the society.
2287	What divides Haiti from the Dominican Republic?	NO ANSWER		
2311	How did the Lindy Hop get its name?	Charles Lindbergh	NYT19990820.0230	"Shorty" Snowden invents the Lindy Hop, a dance craze named for Charles Lindbergh's trans-Atlantic flight.
2313	What does an English stone equal?	14 pounds	NYT20000203.0201	And nearly all English people weigh themselves in "stones" (14 pounds) rather than pounds or ... shudder, kilograms.
2334	What did Peter Minuit buy for the equivalent of $24.00?	Manhattan	NYT19991120.0177	New York recalls how Peter Minuit bought Manhattan Island from local Indians for a Dutch song. <new para> Legend has it that Minuit paid $24, possibly in beads and trinkets.
2334	What did Peter Minuit buy for the equivalent of $24.00?	Manhattan	NYT19980723.0209	Everyone knows that, back in 1626, Peter Minuit bought Manhattan from the Indians for $24 worth of trinkets.

Table 3: Examples of Hard Questions at TREC 2003.

The columns marked R, X, U and W show the numbers of answers judged Right, ineXact, Unsupported and Wrong by the NIST assessors. The overall rate of success was thus 33 out of 413 (8%) or 36 out of 413 (9%) including inexact answers.

5. Conclusions

After carrying out a significant number of enhancements to our system relative to last year, the results are still no better. This is a disappointing result. There are several reasons for this. Firstly, the questions appear to be much harder this year. A significant number of the factoid questions could not even in principle be answered by our type of system based mainly on NEs. A selection of hard queries can be seen in Table 3. Secondly, certain queries are conventional in form but the answer is not stated in a way which is easy to find. An example of this is numbers with unexpected spacing as in the answer to Q-1980 of '250 , 000 miles'. Another example is anaphoric reference as in Q-1939. Thirdly, we are missing a lot of categories for 'normal' forms of TREC query such as what things made of, animals, rock bands and pop groups, baseball teams, musical instruments, how often, how late etc.

References

Clarke, C. L. A., Cormack, G. V., Kemkes, G., Laszlo, M., Lynam, T. R., Terra, E. L., & Tilker P.L. (2003). Statistical Selection of Exact Answers (MultiText Experiments for TREC 2002). In E. M. Voorhees and L. P. Buckland (Eds) *Proceedings of the Eleventh Text REtrieval Conference (TREC 2002), Gaithersburg, Maryland, November 19-22, 2002*. NIST Special Publication 500-251. Gaithersburg, MD: Department of Commerce, National Institute of Standards and Technology.

DTSearch (2000). www.dtsearch.com .

Magnini, B., Negri, M., Prevete, R., & Tanev H. (2002). Is it the Right Answer? Exploiting Web Redundancy for Answer Validation. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia*, 425-432.

Acknowledgement

Many thanks to Ken Litkowski for providing answer patterns and supporting documents for the question collection.

Robust and Web Retrieval with Document-Centric Integral Impacts

Vo Ngoc Anh *Alistair Moffat*

Department of Computer Science and Software Engineering
The University of Melbourne
Victoria 3010, Australia

{vo,alistair}@cs.mu.oz.au

Abstract: *This paper reports the experiments done at The University of Melbourne for the Robust and Web tracks of* TREC-2003. *We explore the idea of determining the impact of a term locally within the document and in a qualitative manner instead of a quantitative one. The impact of each distinct term in a document or query text is defined to be a small integer. The scalar product of the impact vector for a document and the impact vector for a query is taken to be the similarity score between them, an arrangement that allows very fast query evaluation.*

1 Document-Centric Integral Impacts

Consider a document collection with N documents and n distinct terms. Use is often made of the TF-IDF rule to assess the similarity degree between a query q and any document d of the collection. An n-dimensional vector space is constructed, with each dimension representing a term that appears in the collection. In the space each document d is represented as

$$d = (w_{d,t_1}, w_{d,t_2}, \ldots, w_{d,t_n}),$$

and the query q as

$$q = (w_{q,t_1}, w_{q,t_2}, \ldots, w_{q,t_n}).$$

In this framework, the t_i are the distinct terms of the collection, and $w_{x,t}$ is the projection of document or query x in dimension t. That is, $w_{x,t}$ is the "importance" of t in x, and can be calculated by any formulation obeying the TF-IDF requirement. A similarity score $S(d,q)$ between d and q calculated by the cosine measure is of the form:

$$S(d,q) = \frac{\sum(w_{d,t} \cdot w_{q,t})}{\sqrt{\sum w_{d,t}^2} \cdot \sqrt{\sum w_{q,t}^2}}\,, \tag{1}$$

where the three summations are over all n terms.

Anh et al. [Anh et al., 2001, Anh and Moffat, 2002a] consider $w_{d,t}/\sqrt{\sum w_{d,t}^2}$ as the *impact* of the term t in d and propose a transformation that maps the impacts into small integers. The transformation is done in the context of the whole document collection. They obtained better effectiveness than using non-transformed impacts even when only $k = 32$ different values are used for transformed impacts.

In our TREC-2003 experiments we define the same mapping locally within each document rather than globally in the whole collection. The rationale for this change is to give all the documents an equal spread of high impact terms and low impact ones.

Because the transformation is done locally within each document, we do not need to rely on the document length factor in Equation 1. We can choose a step further by not to be bound to any particular formulation of $w_{d,t}$, which is often very diverse. In its original form, TF-IDF is not a precise definition – it is a philosophy. It is our human desire for precision in computation that has led to overly precise formulations, with their various tuning factors. Adopting a philosophy that "rank is more important than value" allows us to reduce the effect that numerical constants have on the effectiveness of the ranking [Anh and Moffat, 2002b].

Ideally, we would like to create the sorted list of terms corresponding to each document without doing any detailed computation. The surrogate weights of the n_d terms in document d are then computed for use in the query processing regime, without any further recourse to collection or document statistics.

To achieve this independence, we focus on one document at a time, and divide the process of determining the transformed impacts of terms in a document d into two phases. The first *sorting* phase orders the list of n_d terms d in decreasing order of term contribution. In the second *mapping* phase, each value in the term list is converted to an integer in range 1 to k. As the end of the process, these integer values serve as document term impacts.

The same process can be done to define query term impacts for the terms in any query q.

Finally, the similarity score between d and q is computed as the sum of the products of document term impact and query term impact of the terms shared by d and q.

The question to be faced is how to specify the sorting and the mapping phases. We sorted the terms in decreasing term frequency $f_{d,t}$, with ties broken using increasing f_t as a secondary key. This two-part sort key implicitly imports the TF-IDF rule into our scoring system. Note that the reverse sorting order, where f_t is used as the primary key, and $f_{d,t}$ as the secondary can not be expected to work as well, as the large number of different f_t values means that the TF factor would rarely be brought into play.

In the mapping phase, we follow the main idea of Fibonacci that in a structure, the number of "important" elements should be less than the number of less important ones. Hence we choose the mapping so that the number of elements in a document corresponding to each surrogate weight, in decreasing order of the weights, forms a geometric subsequence. That is, $x_i \approx x_{i+1} \cdot B$, where $B = k^{1/k}$ and $x_0 \approx n_d \cdot (B-1)/(k-1)$ is the number of elements with the highest impact value.

As an example, when a document contains $n_d = 250$ distinct terms and $k = 10$, the most frequent (within that document) 7 terms are assigned the highest impact of ten, and the least frequent 57 terms in that document are assigned the lowest impact value of one.

2 Task Setting and Performance

Robust Track

In the robust track, the collection TREC45-CR (that is, disk 4 and disk 5 of the TREC corpus, minus the *Congressional Record*) is employed with 50 old topics and 50 new topics. The opportunity of using the 50 old topics for training was not exploited.

Our goal in this track is to measure the effectiveness of the retrieval methods respective a range of query length. The following type of queries, that are automatically generated from the original topics, are used as input: title-only queries, set T; description-only, D; title-and-description, TD; and the whole topic, query set TDN.

The standard settings described in the previous section were applied directly for the Robust track. The number of different impacts was set to $k = 10$, a value found to be appropriate based upon experiments carried out on the TREC-2002 data [Anh and Moffat, 2002b].

Our preliminary experiments with other collection show that using document meta data to compute quasi term frequencies, and using them as surrogate to the raw term frequencies in ordering, leads to improvement on effectiveness. The quasi frequencies can be calculated, for example, as weighted sum

Query type	Precision at 10	No of topics not found in top 10	Area underneath MAP(X) for worst 25 topics
For 50 old topics			
T	0.2800	**8**	0.0029
D	0.2680	**9**	0.0041
TD	**0.3340**	**7**	**0.0093**
TDN	0.2980	8	0.0079
For 50 new topics			
T	**0.4200**	**4**	0.0365
D	0.4480	**5**	**0.0205**
TD	**0.5060**	**1**	**0.0456**
TDN	0.4880	4	0.0383
For all 100 topics			
T	**0.3500**	**12**	0.0087
D	0.3580	**14**	**0.0090**
TD	**0.4200**	**8**	**0.0175**
TDN	0.3930	12	0.0155

Table 1: Effectiveness performance on different type of queries. Figures in bold represent values that are not inferior to the median performance of the 2003 TREC runs for the same category of query.

of frequencies of the terms in different text components. Unfortunately, the different text structures in the sub collections of TREC45-CR meant that we did not assign a weight to each component, and only raw term frequencies were used in our runs.

Table 1 presents the performance of our system with respect to different types of queries. Overall, the TD queries obtain the best performance, and TDN is the worst. The poor performance of the TDN queries is presumed to be a consequence of the noise information in the narrative fields of the original topics. The same observation explains the relative weakness of the D in comparison with the query set T. Another possible reason for the relative good performance of the short queries is that we do not apply any pruning technique to prevent the low impact terms in long queries adding their contributions to the final score.

Web Track

In the Web track, the corpus .GOV is used, with 300 topics for the named page finding task, and another 50 for the topic distillation task.

Meta data is used consistently in .GOV, and we made use of its availability for ordering terms in documents. Incoming and outgoing anchor text is also employed. The way to use these extra information is to count the frequency of terms separately in different component of the text, and use a combination of the partial frequencies to determine a primary key for sorting, prior to impact assignment. The second sorting key is either not used or bound to the IDF factor as in the standard settings.

As the first step, we divide each text into the following components:

- *URL text.* The words in the URL can be valuable for both content search and web search, as it is indicated by many of the last year participants. One possible problem is that in many cases acronyms (such as nist) are used. Content management systems that emit alphanumeric strings of gibberish are also an issue. As an initial remedy to this problem, we preprocess the URL text before indexing. The details of this preprocessing are given in the next subsection.

Run name	Description
W3	Weights of $(1, 2, 2, 4, 4, 8)$ are assigned to components (content, meta text, outgoing anchor, incoming anchor, title, URL); then the weighted sum of these components is used as the sorting key. The IDF factor is not employed.
W4	As for W2, but the component weight vector is $(1, 2, 2, 2, 4, 16)$; and IDF is used as a secondary sort key.
S5	The primary sort key is defined as the complex key that represents the term frequency in the sequence (URL, incoming anchor, outgoing anchor, title, meta, content); with IDF used as a secondary sort key.
W1	As for W3, but only entry pages are indexed. Non-entry pages are ignored.

Table 2: Different sorting methods used to define impacts. The left column is a method identification, with the digit being used as the last digit in the names of our two sets of official runs.

- *Titles.* The title field represent a concise summary of the document, and is usually provided by the author of the document. Other visible markup – for example, headings and bold terms – can also be used with some confidence.
- *Incoming anchor text.* Incoming anchor text can be regarded as being an assessment of the content of this document that is provided by a reader of it, rather than the author of it. The ubiquitous "click here" is not helpful in this regard, and nor are clickable images.
- *Outgoing anchor text.* Outgoing anchor text (provided in links in this document that lead to other documents) do not normally reflect the content of this document. It can, however, be valuable in inferring site structure and page connectivities for the topic distillation task.
- *Metadata.* Other meta text, such as keywords, subject fields, and explicit metadata, may also be of assistance. But there is also a risk that it will have been inherited from a previous document and not edited when the content was revised. Even if it has been edited, it will not have been proofread with the same care as the visible text is.
- *Content text.* The plain text of the page is also a reflection of its content, but not of its importance. Content is the baseline resource of all document retrieval systems.

For each document, after counting the term frequency in each text component, different methods can be used to order the terms in a document based on the number of occurrences of each term in each of these categories. The methods used for the TREC runs are listed in Table 2. In all of the methods, the number of different impact values is set to $k = 10$ (as was also the case for the Robust track).

Manipulating URL text

To facilitate use of the URL text, we have employed a simple process that attempts to de-acronymize it. The two parts considered for expansion are the host name, and the last component of the path name of the document (not counting the default page specification such as `index.html`).

The expansion is based on the title of the page and incoming anchor text, to look for possible variants or abbreviations that might give rise to the words in the URL for that page. If the metadata field "Subject" is available, it is also used as part of this process.

Method	Topic Distillation		Home Page Finding		
	P10	AVP	MRR	F10	NF
W3	0.0660	0.0537	**0.508**	**76.7%**	10.3%
W4	0.0400	0.0559	**0.530**	**75.0%**	10.7%
S5	0.0580	0.0728	**0.527**	**76.0%**	11.0%
W1	**0.0920**	**0.1096**	0.288	40.0%	58.0%

Table 3: Effectiveness of the configurations described in Table 2 for the topic distillation and home page finding tasks of the TREC-2003 Web track. The metric P10 is the precision at 10 documents retrieved; AVP is the average precision over all relevant documents; MRR is the mean reciprocal rank of the first relevant document; F10 is the percentage of topics for which the named page found in top 10; and NF is the percentage of topics for which no named page was found in the top 50 documents retrieved. Figures in bold represent values that are higher than the median performance of the 2003 TREC runs for the same task. Comparative figures are not available for the last column.

For example, document `G00-00-0000000` in `.GOV` was crawled from the URL `http://www.aspe.hhs.gov`. Many of the incoming links that lead to it use variants of "Office Assistant Secretary Planning Evaluation" as their anchor text. The algorithm assigns the words "aspe assistant secretary evaluation planning evaluation hhs" by sorting the set of title, subject, and incoming anchor text words into decreasing frequency, and then choosing for inclusion one matching word for each letter in the URL. Similarly, with `http://www.cs.mu.oz.au`, "computer science cs mu oz" should be identified from anchor text and title text, and used subsequently as URL-related terms.

For document `G00-00-0114013`, the original URL was `http://wwwcalfed.water.ca.gov`, and the URL-based terms are just the words "wwwcalfed water ca". This example is not based upon an acronym, and is correspondingly more complex. It is not satisfactorily handled by the ad-hoc process, and the small number of anchors containing "CALFED Bay Delta Program" were overlooked. The URL `www.unimelb.edu.au` is also hard to handle using the simple mechanism.

As a final example, incoming anchor text for document `G00-00-0497574` from the URL `http://wrf.fsl.noaa.gov` is all "WRF", but the title words "Weather Research and Forecasting (WRF) Model" contribute to the expansion, and allow "wrf weather forecasting research fsl noaa" as URL-derived index terms.

Topic Distillation and Home Page Finding Tasks

This year's topic distillation task seems biased in favor of entry pages. While this is generally true, there are also exceptions – for example, a person with a particular hobby might build an excellent resource page which can be the target of a topic distillation query, but is probably within that person's web site. Consequently, we treat all pages equally in three of our four runs for this task. In the other run only entry pages are indexed, as listed in Table 2. The same configurations were used for the home page finding task. The performance for both of the Web tasks is shown in Table 3.

As can be expected, for the topic distillation, W1 is the best, since it indexes only entry pages. However, this policy dramatically degrades the effectiveness of the named page finding task. A clear conclusion from here is that, if the topic distillation keeps its present definition we need to find another way to address it in order to get a compromise with the home page finding task.

For both of the tasks, all of the three remaining methods give about the same level of effectiveness. That means that the weighting scheme we applied is not effective enough. There should be another, better, way to combine the term frequencies in different components.

3 Practical Considerations

Our experiments were conducted in a 933 MHz Intel Pentium III with 1 GB RAM, a 9 GB SCSI disk for system needs, and four 36 GB SCSI disks in a RAID-5 configuration for data. The computer is also a server for a Beowulf cluster of 16 nodes, each an 800 MHz Intel Pentium III with 256 MB RAM and local hard disk of 40 GB. There is a link with the capacity of 100 Mbits/second to and from each node as well as the server with a network switch. The system runs Debian GNU Linux, and all the programs are written on ANSI C, compiled with gcc, with the optimization flag -O3. For the Robust task, the experiments are done in the server, while all the Web tasks are done by the cluster in a parallel manner.

Except for the methods used determining the impacts, all the index structure and query processing are as described by Anh et al. [2001], and are based on the *MG* software (available at `http://www.cs.mu.oz.au/mg/`, and described by Witten et al. [1999]).

Index construction using the integer impacts is straight forward. After an additional pass to link the incoming anchor text to the target documents (this pass also computes the document frequencies for each term, if needed) the process of assigning impacts can be done locally within a document.

Parallelism is implemented as follow. The data collection is roughly divided between the nodes of the cluster, and each part transferred to the local disk of the corresponding node. The server plays the role of interface – it receives requests for indexing or querying, and sends them to all the nodes. In the case of indexing, the nodes work independently. In the case of querying, after receiving a query, each node produces a list of answers of its own based on its sub-collection, then the results are merged between pairs of node until one of them finally has the complete results. That node then send the results to the server.

Acknowledgement This work was supported by the Australian Research Council.

References

V. N. Anh, O. de Kretser, and A. Moffat. Vector-space ranking with effective early termination. In W. B. Croft, D. J. Harper, D. H. Kraft, and J. Zobel, editors, *Proc. 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 35–42, New Orleans, LA, September 2001. ACM Press, New York.

V. N. Anh and A. Moffat. Impact transformation: Effective and efficient web retrieval. In M. Beaulieu, R. Baeza-Yates, S. H. Myaeng, and K. Järvelin, editors, *Proc. 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–10, Tampere, Finland, August 2002a. ACM Press, New York.

V. N. Anh and A. Moffat. Vector space ranking: Can we keep it simple? In J. Kay, editor, *Proc. Australian Document Computing Symposium*, pages 7–12, Sydney, December 2002b. University of Sydney, Australia.

I. H. Witten, A. Moffat, and T. C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann, San Francisco, second edition, 1999.

UMBC at TREC 12

Srikanth Kallurkar, Yongmei Shi, R. Scott Cost, Charles Nicholas, Akshay Java, Christopher James, Sowjanya Rajavaram, Vishal Shanbhag, Sachin Bhatkar, and Drew Ogle

University of Maryland, Baltimore County
Baltimore, MD USA
{skallu1,yshi1,cost,nicholas,aks1,cjames2,rs2,vshan1,sachin1,og1}@csee.umbc.edu

Abstract. We present the results of UMBC's participation in the Web and Novelty tracks. We explored various heuristics-based link analysis approaches to the Topic Distillation task. For the novelty task we tried several methods for exploiting semantic information of sentences based on the SVD technique. We used SVD to expand the query and to filter redundant sentences. We also used a clustering algorithm that is also based on SVD.

1 Web Track - Topic Distillation Task

Web Retrieval has long been characterized as a web-traversal problem [6, 8]. We centered our efforts in the topic distillation task at this year's TREC around this concept. As in last year's approach, we used our CARROT II (C2) [4] agent-based Distributed Information Retrieval system to implement our approach and to observe if the task itself could be augmented by a distributed retrieval perspective. In a C2 system, an agent's task is to maintain a collection of documents and handle all retrieval operations, including the operations of collection selection and results fusion, as applied to the domain of distributed retrieval.

1.1 Approach

The main idea of our approach for the topic distillation task was to generate an initial set of results for a topic and traverse through the links to reach pages of interest. The initial results were generated from a text search. A system of C2 agents was deployed, with agents maintaining a disjoint subset of the 18 gigabyte web collection. The subsets were created from a depth first traversal of the collection. There were two reasons for creating such subsets:

1. Most search engines obtain web-pages through crawls and,
2. Pages linked together may contain similar content

We simulated a crawl by a depth first traversal. A document from the collection was picked at random and its entire links explored sequentially in a depth first manner, down to a pre-determined depth. A set number of such crawled pages were then assigned to a set and removed from the collection, and the procedure was repeated with another random page as a starting point until all the pages were assigned to sets. All the children of a page, as obtained from the traversal, were assigned to the same set even if the number of pages in the set crossed the limit. Pages with no links were assigned to a particular set. The links of the pages were determined from the links information files accompanying the web collection. These sets of pages were assigned as collections to C2 agents. The agents created metadata about their collection for the purpose of collection selection. The metadata in a C2 system is a vector of terms of the collection and document frequencies [3]. Each agent maintained such metadata about their own collection and shared it with all the other agents. The reason for each agent individually maintaining metadata about all the agents in the system is that collection selection and/or results fusion can be performed by any agent in the system. The C2 agents retrieval operations are performed by using the tf-idf based cosine similarity function.

The Topics were represented by a query consisting of the topic keywords. The initial results set for a topic was generated by first selecting the K best agents, as determined by a comparison of the query with the metadata about the agent-collections and then selected agents returning a list of M best results. We did not perform results fusion because in the next step of the approach, link analysis, we pooled the results from the agents into a single set.

1.2 Link Analysis

1. The top M pages from each of K agents were placed in set A
2. For each page in A, in-links of A were placed in set B (all the immediate parents of A)
3. For each page in B thefollowing two attributes were determined
   - The number of out-links to pages in A and,
   - An associated sub-site

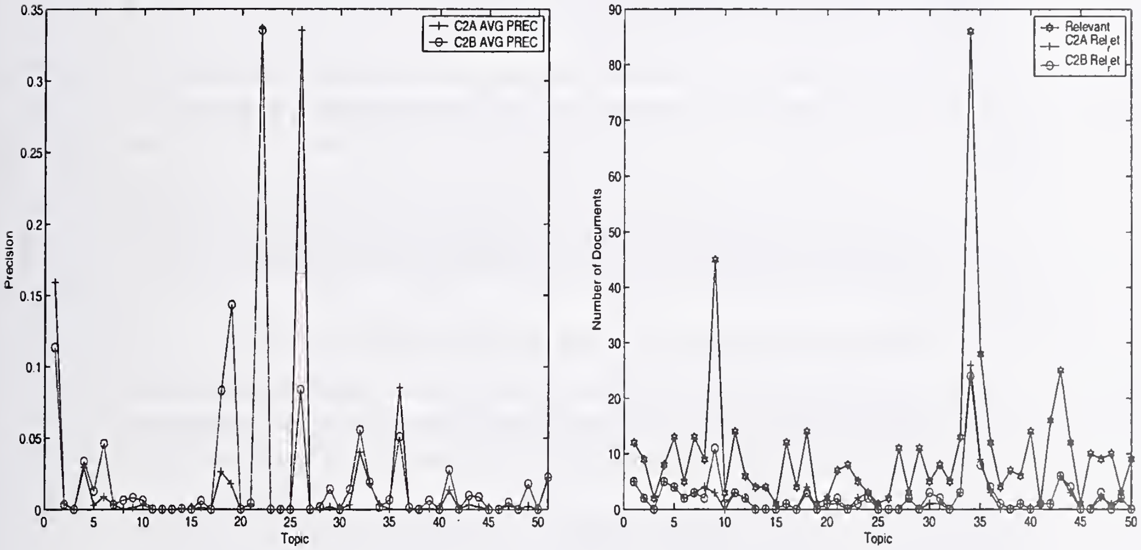
The intuitive reasoning behind our link analysis was a belief that linked pages, in general, may contain similar information. So, the parents of the initial results were assigned scores that were reflective of the number of their children in the results set. Since the task was to find the relevant resources, the representative entry-pages of the pages with the best scores were returned as the final answers. Thus, the more children a page had in the initial results set, the higher was its value in terms of being a good resource.

The in and out-links for a page were obtained from the link information files. The root website for a page was the one with only the head in its URL. The pages in the collection were classified as sub-sites if their URL has a directory listing at the end or contained an "index.html" or an "index.htm". Similar URLs were grouped together as belonging to a site. An associated sub-site was one with the longest common URL with that of the page.

1.3 Experiments and results

We observed that very few sites had more than 10,000 pages. So, we chose the number 10,000 as the maximum number of pages in a set.

We produced two runs with values for K as 20 and M as 100, i.e. selecting the top 100 documents (pages) from the first 20 agents for each topic. In the first run, we simply ordered the pages in B by the out-link count (to pages in A only), using the highest value as a scaling factor. In the second run, we assigned to the each sub-site a score equal to the number of pages that had identified it as their sub-site. The results were again scaled by the largest value. The results are shown in Figure 1.



(a)

(b)

Fig. 1. (a) Average Precision per Topic (b) Number of Relevant Documents Retrieved per Topic

2 Novelty Track

This was our first time taking part in the Novelty Track. We explored SVD based techniques for relevant sentence detection, query modification and filtering redundant sentences.

2.1 Select Relevant Sentences

Selecting relevant sentences was the first step in the Novelty Track. Our method consisted of the following three steps:

1. Modify query
2. Cluster sentences
3. Select the top clusters using the generated queries and return all the sentences in these clusters

Query Modification The initial query was the Topic description. We modified the query by

- Finding highly co-occurring terms with the query terms
- Adding meaningful terms from narrative section

We determined the term co-occurrences using a SVD technique as described by Furnas et al. [5]. In summary, by applying SVD to a term-sentence matrix T, we obtain;

$$T = USV^{\top} \tag{1}$$

The matrix $TT^{\top}$ contains the term-term similarities. Using SVD, we can select the K largest singular values and obtain the term-term similarity matrix $(US_k)(US_k)^{\top}$.

The following methods were used to generate our runs:

1. In the first run the matrix $TT^{\top}$ was calculated directly. The terms that had normalized similarity scores greater than 0.3 to the initial query terms were selected as co-occurring terms and added to the query. All query terms were assigned equal weights.
2. In the second run the narrative sections of the topics were used, (without using $TT^{\top}$). The narrative sections of the topics contain useful information about the topics. To extract useful terms we used several heuristics. First, we eliminated the sentences in which the words "irrelevant" or "not relevant" were used. Then, words such as "opinion" and "description" were deleted, since they do not provide information about the topics commensurate to

their high frequency of occurrences. The remaining sentences were then parsed with a part of speech tagger written by Eric Brill [2]. Words identified as nouns, verbs, decimal numerals and adjectives were added to the query. All query terms were assigned equal weights.
3. The third run was similar to first run except that we used matrix 2.1. The terms were selected based on the normalized similarity scores and the gap between the scores. If a term was selected more than once, its weight was calculated by adding up these scores, with an upper bound of 1.

Cluster Creation and selection For each of the 50 topics, all sentences from each of the 25 documents were pooled together. The sentence pool was clustered using the PDDP [1] clustering algorithm. The intuitive reason for clustering the sentences was that a sentence, as opposed to a document, describes a theme about a topic. Thus, a cluster of sentences sould contain sentences about very similar topical themes. Then, if a sentence was relevant to a topic, then all the sentences in its cluster sould be relevant to that topic. To select the clusters, we compared the query (see section 2.1) representing the topic with the cluster centroid and selected the top clusters.

Results and Analysis The results for relevant sentences are shown in Figure 2. One of the difficulties we faced in our approach was the selection of appropriate number of clusters such that all relevant sentences were chosen. We set our threshold to top 15% of the clusters based on a heuristic from last year's results, where the percentage of relevant sentences was low. The other problem was related to the accuracy of cluster selection and the quality of the clustering algorithm, both of which affected our results. The effects of these challenges can be seen in the results. We will analyze their effects in the near future.

2.2 Select Novelty Sentences

The novelty sentences were a subset of the relevant sentences, such that each new sentence provided novel information. To find the Novelty sentences we used the following approaches:

1. Our first method to find novel sentences was based on a text summarization technique [7]. In this technique, sentences of a document were first clustered to identify the diverse topics of the document. The document was then summarized by selecting the most important sentence from each cluster. For the task of finding novel sentences, the rationale of clustering sentences is intuitive because selecting any one sentence from each of the relevant clusters

would result in a set of novel sentences, where the novelty is determined by the cluster property that the clusters are orthogonal. Since the task was to find novel sentences across all the documents, we used the same approach as in the first task, i.e. we clustered all relevant sentences for a topic, and selected the earliest occurring sentence (determined by the sentence's ordinal information) from each of the relevant clusters. We used the clustering algorithm PDDP to cluster the relevant sentences. The result of this method was submitted as run 1 of task 2.

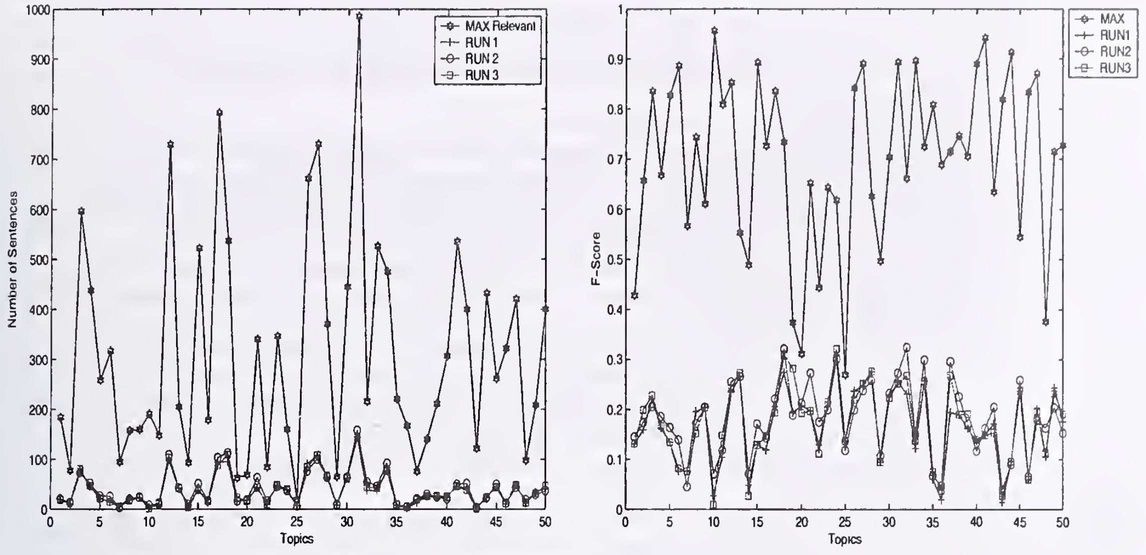
2. Since the novel sentences are a subset of the relevant sentences, we calculated the similarity between relevant sentences so as to eliminate redundantly similar sentences. The assumption here is that if two sentences are adequately dissimilar, then they should provide novel information.

   For a term-sentence matrix T, the matrix $T^{\top}T$ consists of sentence-sentence similarity scores. Then, as in matrix 2.1, we can select the K largest singular values to compute the matrix $(VS_k)(VS_k)^{\top}$.

   (a) In run 2 of task 2 , the sentence-sentence similarity matrix $T^{\top}T$ was computed directly. Novel sentences were selected based on their similarities to any of the previously selected novel sentences. A sentence was marked novel if its similarity was less than a set threshold value with each of the novel sentences selected so far.

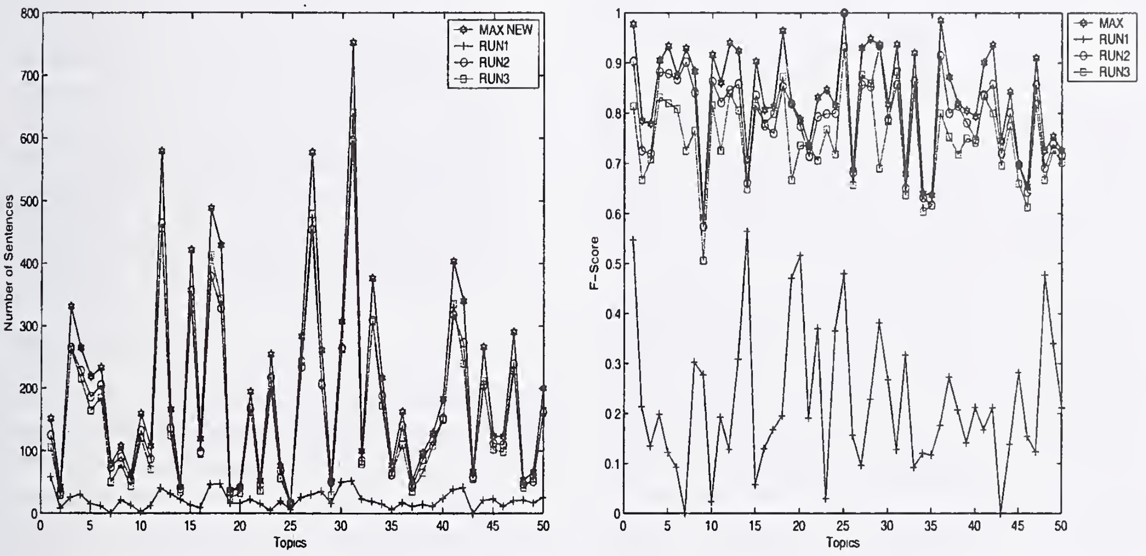
   (b) In run 3 we used matrix 2 to determine similarities between the sentence. The novel sentences were selected as described in run 2.

Results and Analysis The results for novel sentences are shown in Figure 3. From the results of run 1, we can conclude that either the clustering algorithm could not distinguish small variations among the sentences or that the cluster selection procedure was imperfect. One of the problems with cluster selection was that the query-centroid similarity scores were uniformly distributed with no sharp or detectable gaps. This reinforced our first conclusion and also forced us to rely on a heuristic approach to cluster selection. Although runs 2 and 3 appear promising, they were still dependent on another heuristic. During our experiments, we tried to choose thresholds for full-dimension and reduced-dimension methods to determine the novelty of sentences. In run 2, the threshold was 0.7, and 0.5 in run 3. Surprisingly, the results from these two runs are extremely comparable which is encouraging, because we can obtain similar retrieval effectiveness by reduced dimension SVD computation. Thus, we wish to improve upon methods to obtain meaningful thresholds both for cluster selection and SVD-based sentence selection.



(a) (b)

Fig. 2. (a) Task 1: Number of relevant sentences retrieved per run (b) Task 1: FScore values per run



(a) (b)

Fig. 3. (a) Task 2: Number of novel sentences retrieved per run (b) Task 2: FScore values per run

References

1. D. L. Boley. Principal direction divisive partitioning. *Data Mining and Knowledge Discovery*, 2(4):325–344, 1998.
2. E. Brill. http://www.cs.jhu.edu/ brill/code.html.
3. J. P. Callan, Z. Lu, and W. B. Croft. Searching distributed collections with inference networks. In E. A. Fox, P. Ingwersen, and R. Fidel, editors, *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–28, Seattle, Washington, 1995. ACM Press.
4. R. Cost, S. Kallurkar, Y. Shi, H. Majithia, and C. Nicholas. Integrating distributed information sources with CARROT II. In *International Workshop on Cooperative Information Agents*, Madrid, Spain, August 2002.
5. G. W. Furnas, S. Deerwester, S. T. Dumais, T. K. Landauer, R. A. Harshman, L. A. Streeter, and K. E. Lochbaum. Information retrieval using a singular value decomposition model of latent semantic structure. In *Proceedings of the Eleventh International Conference on Research and Development in Information Retrieval*, pages 465–480, 1988.
6. J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
7. T. Nomoto and Y. Matsumoto. A new approach to unsupervised text summarization. In *Proceedings of the Twenty Fouth annual international ACM SIGIR conference on Research and Development in Information Retrieval*, pages 26–34, New Orleans, 2001.
8. L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

HARD Experiment at Maryland: from Need Negotiation to Automated HARD Process

Daqing He
Institute for Advanced Computer Studies
University of Maryland
College Park, Maryland, 20742
daqing@umiacs.umd.edu

Dina Demner-Fushman
Department of Computer Science
Institute for Advanced Computer Studies
University of Maryland
College Park, Maryland 20742
demner@cs.umd.edu

Abstract

Our aim of participating in this year's High Accuracy Retrieval from Documents (HARD) track is to explore the possibility of developing an automated HARD retrieval model by leveraging existing models and theories about information need negotiation in information science. The clarification questions we developed are related to four different aspects of search topic, and four different techniques were developed to fully use the information collected from the user through these questions. Our initial analysis of the results indicates that this is a promising approach.

1 Introduction

Searching for information is increasingly common in people's life. Modern techniques based on "free text" indexing and ranked retrieval have proven to be scalable and robust. Batch mode information retrieval (IR), which essentially studies retrieval algorithms, receives a great deal of attention. Significant improvements have been achieved both in academic and commercial paradigms. Many people associate the improvements, especially those achieved in academic paradigm, with controlled experiment frameworks, such as Text Retrieval Conference (TREC).

However, the initiative of searching for information ultimately lies with human. It is people who pose the questions, interpret what they read, and determine when their needs have been met. Especially in modern retrieval process, end users, who are not necessarily search experts, nor domain experts, leverage easy access to full text to support increasingly focused exploratory searches via iterative refinement (Marchionini, 1995). Therefore, the ultimate goal of retrieval systems is not to generate the best possible ranked list for a given search query, but to provide the best information access mechanisms to users so that they can easily find needed information, and have a pleasant search experience.

Based on this view of retrieval process, many researchers concentrated on interactions in information retrieval process, and designed experiments within a controlled experiment framework. The interactive track in TREC was an effort dedicated to this task. Many interesting research results have been achieved via this approach, but many of its limitations have also been shown. It is widely accepted that interactive IR experiments are difficult to design, expensive to conduct, limited in their small scales, and hard to compare cross-site. Our past experience with interactive IR, especially experiments we conducted for interactive track of Cross Language Evaluation Forum (iCLEF) (He et al., 2002; Dorr et al., 2003b), provides us with some first-hand knowledge of these limitations.

We see HARD, a new track of TREC 2003, as an opportunity to ask interesting questions about the real human retrieval process, especially the interactions between human and the retrieval process, and at the same time, to design a relatively easy, cheap, and large scale controlled experiment to find answers to our questions, and to compare the results to these of other sites.

To us, HARD experiment models the retrieval process differently to both batch and interactive IR. For better representation of the actual retrieval process, HARD allows interactions between the users and the retrieval system, which is like interactive IR. However, to avoid the difficulty of managing full interactions, HARD only allows one iteration of interactions. The interaction is conducted by letting the system generate a set of clarification questions to be answered by the user. Then the system uses the answers to improve its search effectiveness. In addition, HARD uses measures on returned rank list as indicators of the performance rather than measures on relevance judgments often used in interactive IR.

HARD experiment, to some extent, can be viewed as a simplified model of information need negotiation services, which is a well studied area in information and library science (Taylor, 1968). The process of generating clarification questions in HARD experiment is a simplified version of information need negotiation, or reference interview. Once we achieve this transformation of models, we will have opened a rich resource for us to borrow. Therefore, our approach in this year's HARD experiment is to leverage existing theories, models, ideas, and resources in information need negotiation to design and implement an automated process of generating clarification questions and utilizing their answers to improve the ranked list of documents for a given query statement.

In the rest of the paper, we are going to briefly introduce the idea of information need negotiation in information science, and then move to present our approach of leveraging existing models to our automated HARD process. In the remaining sections, we discuss some preliminary analysis of our experimental results, and finally we conclude with some indications to future directions.

2 Information Need Negotiation

Information need negotiation is a reference interview in the library setting. It is a communication between an information specialist and a user, in which the users present their information requirements, and the specialist clarifies these needs to develop an appropriate, mutually agreeable search strategy. In his classic paper about information need negotiation (Taylor, 1968), Taylor identifies this process as question negotiation, since he believes that the query issued by a user is not a command, but a question that the user wants to be answered by the information specialist. Because a user in IR tries to search for something that he/she does not know, the negotiation process contains complex actions where two persons interact to achieve the goal of identifying the need and find an appropriate search strategy. This is why the negotiation of reference questions is one of the most complex acts of human communication.

Based on studies of librarians and information specialists, Taylor identified the following five general types of information often occurring during an information need negotiation process:

1. determination of the subject that the user is searching on;
2. objective and motivation for the current search;
3. personal characteristics of the user;
4. relationship between the search statement and the file organization in the collection; and
5. anticipated or acceptable answers.

With the advancement of information technology, not all information need negotiations are conducted face to face between a user and an information specialist. One of the examples of remote need negotiation services is conducting the need negotiation via Email. Abels conducted a three-phased project at University of Maryland to explore the email negotiation process (Abels, 1996). She identified five approaches often used in email negotiation process: (1) piecemeal, (2) feedback, (3) bombardment, (4) assumption, and (5) systematic. Her analyses showed that the systematic approach yielded successful need negotiation, and it was clearly the most efficient in terms of number of messages needed in the process. In the systematic approach, the specialists responded to a request with a list of questions covering all related aspects. The questions were organized in a logical way and included both open- and closed-end questions. At the later stage of the project, Abels designed a request form to include all the questions that would be asked in the systematic approach of need negotiation. The essential content of the form are questions about the personal data of the user, subject to be searched, and preference/constraints on the search results.

3 Constructing automated HARD process

Our HARD process naturally divided into two stages. The first one was to automatically generate a set of clarification questions to probe for more information about the topic, the person who had the need, and his/her preferences regarding search results. The second stage was to automatically utilize the answers to questions in the clarification forms. The design of the questions in the clarification forms and the methods utilizing the answers to the questions are closely related. The questions were selected based on our understanding of how the answers will be applied to improve the retrieval effectiveness. The automated process for utilizing these answers was designed to include as much as possible of the information from the answers.

3.1 Generating Clarification Questions

We combined Taylor's question negotiation model and Abels' email reference forms, and designed our own set of clarification question types. The clarification questions came from four aspects of context information related to a given query. They are presented in each of the following sub-sections, respectively.

3.1.1 Users' preference to sub-topic areas

Documents that are retrieved for a given query can be classified into multiple sub-topic areas. One reason is that the search topic naturally has multiple facets. Another reason could be that the query terms have multiple senses, which results in the search system retrieving documents related to several sub-topics. For example, topic 87 is about *Egyptian cotton*, and its top ranked documents retrieved by our baseline system covered sub-topics ranging from an advertisement for Macy's white sale to the history of Egypt, to cotton leaf worm or child labor in Egypt's cotton industry.

Although there could be cases when the user is interested in more than one sub-topic in the same search, prevalently the user is interested in only one of them. Differentiating the intended sub-topic area from those irrelevant ones would help the system to avoid placing irrelevant documents at the top of the ranked list. Therefore, the first aspect of the clarification questions we tried was to probe the user's preference to the sub-topic areas.

Two tasks were needed to enable us to generate questions for the user to select among sub-topics. The first one was to identify the sub-topic areas existing in the top ranked baseline retrieval results. The second task was to present the user with a concise description for each sub-topic area so that the user could select.

For the first task, we used automatic clustering method. Clustering has been shown as an effective method for organizing and presenting closely related documents (Hearst and Pedersen, 1996). During clustering, we only used top ten retrieved documents for each query because our training results demonstrated that this part of retrieval results contains little noise. In addition, space limitation imposed on clarification forms by HARD track guideline meant that only a small number of clusters (i.e., less than 5 clusters) were possible. Restricting the document pool size also helped to obtain tight small-sized clusters of closely related documents.

We clustered the documents using the Lighthouse implementation of the Ward's hierarchical clustering algorithm (Leuski and Allan, 2000). The distance measure used in the clustering was the cosine value of the angle between the vector representations of two documents. The weight for each term in the vector representation is defined as the product of the term's frequency in the document and its inverted document frequency. The average number of clusters for a topic was 3.5.

For the task of presenting a concise description of clusters, we explored three different approaches. We first tried to use top 10 highly weighted terms in the cluster as the cluster representative, but were not satisfied with the outcome when we tried it on the training data.

We then tried another approach based on the genre of the documents. Most of the documents in the HARD collection are news articles. It has been shown that titles and first sentences of news domain articles contain enough information to represent the main topic of the document (Dorr et al., 2003a). Our second approach, therefore, was to select the titles or the first sentences, in the absence of the titles, from each document in a cluster as candidates for the cluster representative. These candidates were ranked based on the normalized sum of the weights of the terms in them, and the one at the top was selected as the cluster representative. An example of the cluster representative generated by this method is *"Middle East Economic Briefs"*.

The third approach we tried was to use a multi-document headline generation tool called GOSP. It was developed at Information Science Institute, University of Southern California (Zhou and Hovy, 2003). Before the documents could be processed GOSP, we tokenized and POS-tagged the documents using MXPOST and MXTERMINATOR software (Ratnaparkhi, 1996). An example of the cluster representative generated by this method is *"MIDDLE EAST COUNTRIES BEIRUT LEBANONS NATIONAL/1998 LEBANESE MINISTER OF PUBLIC WORKS AND TRANSPORTATION NAJIB MIKATI/CENTRAL BANK BE PRIVATIZED"*.

3.1.2 Users' recent experience with the search topic

The second type of questions in our clarification forms was about user's characteristics. In this year's HARD experiment, we concentrated on the user's recent experience with searching on the subject area. This information is important because the user's information need on a topic could be evolving over time, and the answer to this question can help us determine the current status of the user's need on the topic. In addition, the answers to this question provide the necessary information to perform query expansion when the user does not select any clusters as relevant.

The question directly asks for the terms related to the user's recent search on the topic. Our question starts with an inquiry about whether or not the user had seen any relevant documents recently. If the answer is positive, we ask for the key words that would best describe the document. In the question about the key words, we specifically ask the user to provide highly representative content words, person or organization names, and terms related to locations.

3.1.3 Users' preference to sub-collections

The HARD collection contains news articles and US government documents. Among news articles, there are documents from news agencies inside US, and those from Xinhua news agency. A user who has particular information need usually does not necessarily have the same preference for the documents from different sources. For example, a user who is interested in international response to an event will be more willing to read articles from Xinhua news agency than US government documents. Therefore, knowing these preferences would help the retrieval effectiveness since the retrieval system can pull the documents from the preferred sources to higher ranks. However, these preferences are usually user dependent and topic dependent, so asking the user to provide such information is much easier and effective than letting the system automatically infer user preferences.

Our question about the user's preference to sub-collections is designed to facilitate quick response from the user. Users were asked to rank their preferences on a scale from 1 to 5, with 1 as the most preferred one.

3.1.4 Users' anticipation of result formats

Although retrieval systems usually return documents as the default format of the results, different users under different information needs may prefer different formats, such as documents, passages, sentences, or even straight answers. For example, this year's HARD experiment guidelines contain a specification to identify the user's preferred result format. Our question to this type of information was designed to be a straight selection of the format the user wants for this particular search topic.

3.2 Applying Answers to Clarification Forms questions

Our automated process utilized the information obtained from the answers to the clarification forms in three ways, namely, term extraction for query expansion, preference extraction for document re-ranking in a ranked list, and evidence combination for ranked lists merging. To test the effectiveness of these three ways in isolation or combination, we designed several runs to include either only one of them or combinations of them. In the remainder of this section, we are going to present our methods for query expansion, document re-ranking and ranked list combination.

3.2.1 Query Expansion

The information used in query expansion is from two sources. The first source is a set of texts that includes the description and narrative sections of the topic statement (since we only used the title part of each topic statement in our baseline run); the documents belonging to the preferred clusters marked by the user; and the relevant documents provided in the meta data part, if meta data were used in the run. Because texts from different sources were rather different, we extracted terms from them separately, and combined the terms only when using them to expand the query. Terms were extracted based on the combination of their term frequencies in the documents and their inverted document frequencies.

The second source is the set of terms provided by the user when he/she answered the clarification questions about their recent experience with the search topic. Since the answers to these questions were already a set of terms, and they were provided directly by the user, we did not perform any further term extraction before expanding the query using these terms.

The expanded queries were constructed by including the terms from the original baseline query, and the terms from the two sources above. Besides the weights calculated during the term extraction (i.e., the weights associated with the terms from the first source), we also assigned a predefined weight to each term based on its origin. We assigned equal weight (i.e., 20) to terms from the original query, and those from the second source above. The rationale is that both sets of terms were directly provided by the user. We then normalized the weights of the terms from the first source so that the highest weight among them is only half the weight of the terms from the original query (i.e., the weight is 10), and all the other weights from the first source were mapped proportionally. This reflected our view that we trust more the terms directly issued by the user, and less the terms we extracted. If a term appeared in multiple sources, the weights of its several appearances were combined.

3.2.2 Document re-ranking

Document re-ranking in our approach referred to boosting or suppressing documents with certain features based on the user's preference. The goal of this method was to improve the retrieval effectiveness by rearranging the ranks of documents.

The information used in helping us re-rank the documents included the answers about the user's preference to sub-collections, and the answers about the user's preference to a time period covered by the HARD collection. When the meta data were included in the refined run, we also used the information about the user's preference to the genre of the documents to help us in re-ranking.

Two approaches can be applied in document re-ranking. There could be aggressive re-ranking, in which the documents possessing certain features are boosted or suppressed to the maximum. For example, if we know a user prefers documents from one sub-collection, aggressive re-ranking would put all the retrieved documents from this sub-collection at the top of the ranked list. This approach sometimes makes sense. For example, if we knew that the user is really interested only in government documents, we could achieve best results putting all retrieved government documents at the top of the ranked list.

The other approach is conservative re-ranking, that is, boosting or suppressing some documents only to some degree. For example, using a small predefined boost or suppress factor to perform re-ranking. This approach is appropriate when there is not much training information to be used in the development, which means that we could not give too much trust to the re-ranking algorithm.

Our re-ranking algorithm in this year's HARD experiment was the mixture of aggressive and conservative approaches. On the one hand, there seemed to be a clear indication of user's preference when the user marked that he/she wanted certain genre of documents (e.g., government documents), which encouraged us to use aggressive re-ranking approach. But on the other hand, there were not much training data for us to really test our re-ranking algorithm, which indicated the conservative re-ranking approach to be more appropriate. At the end, we adopted the aggressive re-ranking algorithm when we used the genre meta data, but kept conservative approach in re-ranking when we used other information.

3.2.3 Ranked list combination

People have demonstrated that, if designed carefully, merging ranked lists from different resources could improve the retrieval effectiveness (Kamps et al., 2002). In addition, it is also a safer approach to merge a ranked list that we do not know much about with the ranked list that we trust to certain extent. In this year's HARD experiment, we performed ranked list combination for both of the reasons above.

We adopted a linear combination approach where the scores of documents in two lists were first normalized proportionally to the highest score in the list, and then combined linearly by applying a predefined list-specific weight factor λ. During the training stage, we noticed that we can achieve even better performance if we micro-adjust the list weight λ with the difference between the highest scores of the lists.

We combined the ranked list of results of one of the refined runs with our baseline run. The chosen refined run utilized our query expansion method and document re-ranking based on the answers from clarification forms and meta data.

3.2.4 Passage and sentence retrieval

Our passage retrieval module assumes that the relevance of a passage is related to the frequency of the query terms in the passage, the importance of these query terms (i.e. their weights), and the relevance of the document that contains the passage to the query. Among the three factors, we gave more emphasis to the document containing the passage. All passages were ranked according to their relevance, with the restriction that only three passages were allowed from the same document. The final result is top 1000 passages for a given query.

4 Results and Discussion

We conducted several runs during the experiment, taking different combinations of the techniques presented above. As shown in Figure 1, most of the runs outperformed the submitted baseline. Among the techniques we applied to improve over the low baseline runs, query expansion worked well, and the improvement was statistically significant (based on T-test). However, comparing to blind relevance feedback, which was our high baseline, the interactive query expansion approach performed slightly better, but the difference was not statistically significant.

Document re-ranking worked, but not as effective as query expansion. In addition, it seems that our approach of asking the user to rank their preference to the source of HARD sub-collections in general hurts performance (see the decrease of mean average precision in the run "interactive query expansion + strong source re-ranking" in Figure 1). We need further analysis to know the exact reason of this adverse effect. One possibility could be that the users actually did not know much about the documents in the sub-collection, so their preferences provided to us were not reliable in this case.

However, using the genre preference provided in meta data did help the retrieval performance a little, although there was no statistically significant difference between using and not using it, probably due to sparsity of genre preferences in the HARD topics. The potential advantage of genre preferences provided in meta data over the information about source preferences might have its explanation in that the genre information is easier for the user to determine, and recollect during the relevant assessment. Again, we need further study of the effect of using the genre information.

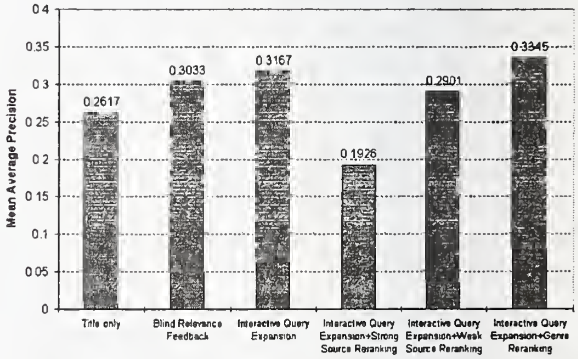


Figure 1: The comparison between different runs on document retrieval against two baseline document retrievals.

We performed failure analysis on the interactive query expansion run. Figure 2 shows the difference of average precisions between interactive query expansion and blind relevance feedback based on individual topics. The topics are arranged in the increasing order of the number of relevant documents in the top 10 retrieved documents. In general, interactive query expansion performed better in the topics at the left end of the X axis, where these topics have none or very low number of relevant documents in the top 10 retrieved documents (e.g., topics 84 to 77). The improvement about 80% is statistically significant. The blind relevance feedback approach performed relatively better when the number of relevant documents in the top 10 retrieved documents increased, especially when most of the top 10 retrieved documents were relevant (i.e., topics 53 to 229).

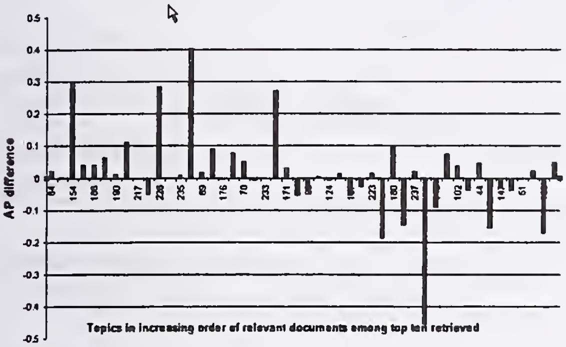


Figure 2: The comparison between interactive query expansion and blind relevance feedback on individual topic level. The upper half of the Y axis indicates that the interactive query expansion is better, whereas the lower half of the Y axis indicates that the blind relevance feedback is better.

It seems that it is helpful to have users' involvement when there is no or few relevant documents at the top of the ranked list, which is the exact place where blind relevance feedback could not perform well. In this case, the users' selection of clusters, even the non-relevant clusters, actually helps to remove some noise that could affect the retrieval effectiveness.

Therefore, a hybrid approach to query expansion is probably ideal. Interactions with users can be used for topics that have none or few relevant documents at the top, and blind relevance feedback should be used when most top ranked documents are relevant, which saves the trouble of having users make relevance judgments. The key here is whether or not we can somehow predict the number of relevant documents in the top ranked documents.

Our passage retrieval module performed reasonably well. In our passage retrieval run, 33 of 42 topics that required passages to be retrieved had the R-precisions above the median, and 6 topics had the best scores.

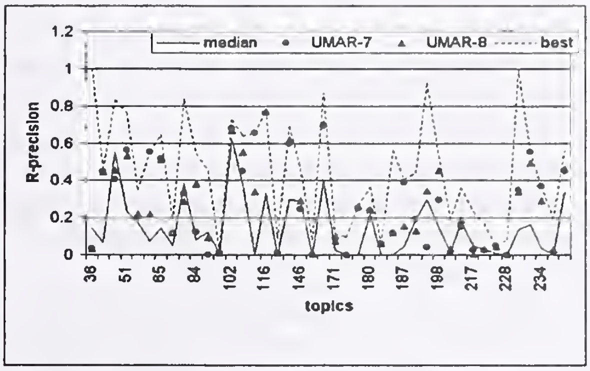


Figure 3: The performance of our passage retrieval runs among all submitted runs.

The run that tested ranked list merging was based on passage retrieval. UMAR-8 was the run that combined the ranked list of our passage retrieval (UMAR-7) and the passage retrieval version of our baseline run. Compared to the result of UMAR-7, UMAR-8 had two more topics with R-precision higher than the medians, but had much less topic among the best (only 1 topic for UMAR-8). One possible reason could be the suboptimal parameter setting used in the merging. Due to lack of training data, the parameters were chosen ad-hoc.

5 Conclusion

In this paper, we talked about our participation in the HARD track. Our approach in this year experiment was to explore the possibility of developing an automated HARD retrieval model by leveraging existing models and theories about information need negotiation in information science. Our initial analysis of the results indicates that this is a promising approach.

Although we have not finished analyzing our results yet, there are already some interesting lessons learned. The first one is about user involvement in the process. At least at the query expansion part, it is not the case that user's involvement would always improve retrieval effectiveness. When there are not many relevant documents at the top of the ranked list, asking user to perform cluster selection is a good idea. Actually in this situation, there is no risk in asking for user's help. However, it is definitely not a good idea to ask the user select clusters if about half of the documents are relevant, and the other half is noise. Designing a mechanism to automatically identify when to ask the user, and when not to, will be one of the foci of our further exploration of the HARD retrieval model.

The second lesson learned is about the questions to the users. Our experience with the questions regarding users' preference to sub-collections indicates the importance of asking the right questions. The questions should be about the type of information that the users know, and also can easily express in their answers. If the questions actually force the users to make decisions that they do not fully understand, it probably is harmful to use the collected answers in the automated process.

Overall it was fun to participate in the HARD experiment, and we are looking forward to HARD 2004.

Acknowledgements

We thank Doug Oard and Eileen Abels for many long discussions about ideas, problems, and results; Zhou Liang and Ed Hovy at ISI University of Southern California for providing the GOSP Tool; James Allan for organizing the HARD track, and folks at NIST and LDC for providing topics and relevance judgments. This work has been supported in part by DARPA cooperative agreement N66001-00-2-8910

References

Eileen Abels. 1996. The Email Reference Interview. *RQ*, 35(3):345–358.

Bonnie Dorr, David Zajic, and Richard Schwartz. 2003a. Cross Language Headline Generation for Hindi. *This volume.*

Bonnie J. Dorr, Daqing He, Jun Luo, Douglas W. Oard, Richard Schwartz, Jianqiang Wang, and David Zajic. 2003b. iCLEF 2003 at Maryland: Translation Selection and Document Selection. In *Proceedings of CLEF'03.*

Daqing He, Jianqiang Wang, Douglas W. Oard, and Michael Nossal. 2002. Comparing user-assisted and Automatic Query Translation. In *Proceedings of CLEF'02.*

Marti A. Hearst and J. O. Pedersen. 1996. Reexamining the Cluster Hypothesis: Scatter/Gatherer on Retrieval Results. In *Proceedings of ACM SIGIR'96*, pages 76–84.

Jaap Kamps, Christof Monz, and Maarten de Rijke. 2002. Combining Evidence for Cross-Language Information Retrieval. In *Proceedings of CLEF'02.*

Anton Leuski and James Allan. 2000. Lighthouse: showing the way to relevant information. In Steven F. Roth and Daniel A. Keim, editors, *Proceedings of IEEE Symposium on Information Visualization (InfoVis'00)*, pages 125 – 130. IEEE Computer Society, October.

Gary Marchionini. 1995. *Information seeking in electronic environments.* Cambridge Series on Human-Computer Interaction. Cambridge University Press.

Adwait Ratnaparkhi. 1996. A Maximum Entropy Part-Of-Speech Tagger . In *Porceedings of Empirical Methods in Natural Language Processing Conference.*

R. S. Taylor. 1968. Question-negotiation and information seeking in libraries. *College & Research Libraries*, 29:178–94.

L. Zhou and E. Hovy. 2003. A Web-Trained Extraction Summarization System. In *Proceedings of the HLT-NAACL conference*, May.

UMass at TREC 2003: HARD and QA

Nasreen AbdulJaleel, Andres Corrada-Emmanuel, Qi Li,
Xiaoyong Liu, Courtney Wade, and James Allan
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts Amherst

The Center for Intelligent Information Retrieval (CIIR) at UMass Amherst participated in two tracks for TREC 2003: High Accuracy Retrieval from Documents (HARD) and Question Answering (QA).

- In the HARD track, we developed document metadata to correspond to query metadata requirements; implemented clarification forms based on query expansion, passage retrieval, and clustering; and retrieved variable length passages deemed most likely to be relevant. This work is discussed at length in Section 1.
- In the QA track, we focused on retrieving passages that were likely to contain the answer to the question.

1 HARD track

1.1 Overview

The goal of the High Accuracy Retrieval from Documents track was to explore techniques for improving the accuracy of the top-ranked documents in response to a query. We participated in all three aspects of the problem:

- We mapped query metadata values to document metadata values that we assigned. We then adjusted the ranking of documents depending on whether their metadata matched the query metadata.
- We generated clarification forms to tease more information out of the searcher. We tried several types of clarification forms, including providing a list of keywords that might appear in relevant documents, a list of top-ranking clusters that might contain relevant documents, and a list of passages that might appear in relevant documents.
- We explored passage-level retrieval of documents to see if we could pinpoint the relevant portions of documents.

In the final analysis, all runs using metadata or clarification forms failed to outperform our best baseline run. We interpret this as an indictment of the track and of our effort. As with most new TREC tracks, the HARD track was slow to get started, had problems being clearly defined, and had poor training data. In addition, several engineering bottlenecks delayed our initial work and prevented us from moving as rapidly as we had originally intended.

The rest of this section discusses what we did. We first describe the baseline runs that we generated for comparison. The same section presents the mechanism behind our passage retrieval runs. In Section 1.3 we discuss the types of clarification forms that we used and, in Section 1.4, how we used the responses. We outline how we used query and document metadata in Section 1.5 and how it was incorporated into the ranking in Section 1.6. We discuss our results in Section 1.8.

1.2 Baseline Runs

1.2.1 Standard Document Retrieval Baseline

Two of our document retrieval baseline runs (ciirtbas and ciirtdbas) used relevance modeling (method 2) [10] as implemented in Lemur.[1] This relevance model was built using the top 75 terms from the top 50 documents. Ciirtbas uses only the topic title as the query and ciirtdbas uses the topic title and description.

1.2.2 Passage-Inspired Document Retrieval Baseline

The two other baseline runs (ciirtpsgbas and ciirtdpsgbas) re-ranked documents from the standard document retrieval baselines based on the best passage from each document.

Passage retrieval was performed using passage language models [1]. The passages were ranked according to query-likelihood. $P(q|P) = \prod_{i=1}^{n} P(q_i|P)$ where P is a passage. The results were smoothed using interpolation with a collection model and a Dirichlet prior.

Each document was divided into passages of 150 words, with an overlap of 75 words. The best passage for every document in the initial ranked list (ciirtbas and ciirtdbas) was determined. These "top" passages were then ranked by their log-likelihood. In the final list(ciirtpsgbas or ciirtdpsgbas), each passage was replaced with its corresponding document.

1.2.3 Passage Retrieval Baseline

Our final baseline (ciirtp) was a ranked list of passages. Passages were retrieved using the method described in the previous section. However for this run, the top 10 passages from each document were considered, as opposed to only one in the previous case.

1.3 Clarification Forms

1.3.1 Top-term clarification form

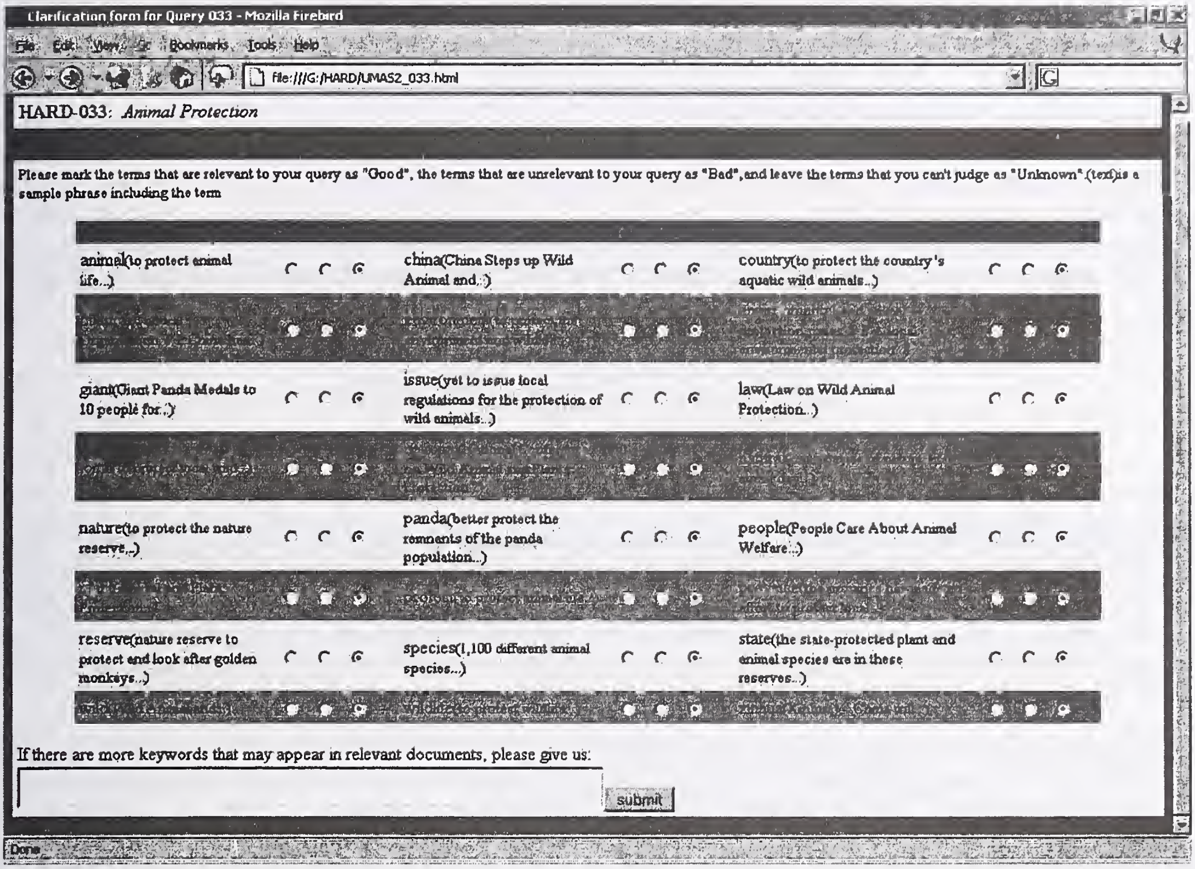
The system first employs a simple language modeling approach [13] to retrieve the top 1000 documents, and then constructs a relevance model [10] from those documents. The top 30 terms selected by the relevance model were used for the top-term clarification form. A snapshot of the form is shown in Figure 1. In this form, we ask the LDC annotators to mark the terms that are relevant to the query as "Good", the terms that are non-relevant to the query as "Bad", and leave the terms that they can't judge as "Unknown" which is the default option. The text in the parenthesis after each term is a sample context in which the term appears in the retrieved documents. In addition, we also ask the annotators to suggest if there are any terms other than the ones already shown in the form that they think might occur in the relevant documents.

In the responses we have obtained from LDC on this clarification form, there is an average of 12.2 good terms marked among the 30 terms provided per test topic, and the maximum number of good terms marked for any topic is 22 while the minimum is 3. Table 1 summarizes the term statistics per topic. Only one test topic had suggested terms other than the 30 provided. In our submission runs that made use of this clarification form, we expanded the original query with the good terms (suggested terms are also considered as good terms) and performed retrieval. Retrieval results are discussed in section 1.7.

1.3.2 Other clarification forms (CF)

Besides the top-term clarification form, we also generated other clarification forms, namely cluster-by-size CF, cluster-by-distribution CF, and passage CF.

[1] http://www-2.cs.cmu.edu/ lemur/


Clarification form for Query 033 - Mozilla Firebird
File Edit View Go Bookmarks Tools Help
file:///G:/HARD/UMASS2_033.html
Done

HARD-033: *Animal Protection*

Please mark the terms that are relevent to your query as "Good", the terms that are unrelevent to your query as "Bad",and leave the terms that you can't judge as "Unknown".(text)is a sample phrase including the term

animal(to protect animal life...)	○ ○ ●	china(China Steps up Wild Animal and...)	○ ○ ●	country(to protect the country's aquatic wild animals...)	○ ○ ●
[illegible]	○ ○ ●	[illegible]	○ ○ ●	[illegible]	○ ○ ●
giant(Giant Panda Medals to 10 people for...)	○ ○ ●	issue(yet to issue local regulations for the protection of wild animals...)	○ ○ ●	law(Law on Wild Animal Protection...)	○ ○ ●
[illegible]	○ ○ ●	[illegible]	○ ○ ●	[illegible]	○ ○ ●
nature(to protect the nature reserve,..)	○ ○ ●	panda(better protect the remnents of the panda population...)	○ ○ ●	people(People Care About Animal Welfare...)	○ ○ ●
[illegible]	○ ○ ●	[illegible]	○ ○ ●	[illegible]	○ ○ ●
reserve(nature reserve to protect and look after golden monkeys...)	○ ○ ●	species(1,100 different animal species...)	○ ○ ●	state(the state-protected plant and animal species are in these reserves...)	○ ○ ●
[illegible]	○ ○ ●	[illegible]	○ ○ ●	[illegible]	○ ○ ●

If there are more keywords that may appear in relevant documents, please give us:

Figure 1: A snapshot of the top-term clarification form for test topic 033.

Table 1: Statistics of the terms that are marked "Good", "Bad", and "Unknown".

	Average	Max.	Min.
Good terms	12.2	22	3
Bad terms	5.4	20	0
Unknown terms	6.4	21	0

- *Cluster-by-size* CF. The system first retrieves the top 200 documents using a simple language modeling approach [13]. The group-average hierarchic clustering algorithm [4] is then applied to the retrieved documents set to obtain clusters. The similarity between documents is determined by the cosine similarity between their term vectors, and the threshold is set to 0.6. Clusters are ranked by their size with the largest cluster at the top of the ranked list. Top 15 clusters are provided, and both the headline of the centroid document and the top 10 terms for each cluster are shown in the CF form. The LDC annotators are asked to mark whether each cluster is good, bad, or unknown. If none of clusters are marked good, the annotators are encouraged to suggest some terms that they think might occur in the relevant documents. In the responses we obtained from LDC on this form, on average, there are 5.5 clusters that are marked "good" , 5.1 marked "bad", and 4.4 marked "unknown", out of the total of 15 clusters provided per test topic. Among the 50 test topics, there are two topics for which all 15 provided clusters are judged "good", four topics for which all 15 clusters are judged "bad", and four topics that have only "unknown" clusters. Only one topic had suggested terms other than the ones displayed in the form. Table 2 shows the statistics on clusters per topic.

Table 2. Statistics of the clusters that are marked "Good", "Bad", and "Unknown".

	Average	Max.	Min.
Good clusters	5.5	15	0
Bad clusters	5.1	15	0
Unknown clusters	4.4	15	0

- *Cluster-by-distribution* CF. This CF is generated in a similar fashion to the *cluster-by-size* CF but with a different cluster ranking mechanism. After the clusters are formed by the group-average clustering algorithm, a simple cluster language model is constructed from the clusters and a query language model is constructed using information from the query. A Kullback-Leibler (KL) divergence score is computed between each cluster model and the query model, and is used as the basis for the ranking of clusters. Clusters and related information are displayed in the same format as the *cluster-by-size* CF.

- *Passage* CF. The system uses the relevance model [10] for retrieval of top 1000 documents, and then segments the documents into passages and ranks the passages. The top 10 passages are presented in the clarification form, again to be marked as "good", "bad", or "unknown" based on their relevance to the query.

However, due to the tight schedule of the LDC annotators, only the top-term CF and the cluster-by-size CF were filled out for all test topics. After comparing retrieval performance on the training data using top-term CF and cluster-by-size CF, we selected the top-term CF to be used in our final submission runs.

1.4 Incorporating Clarification Forms

One of our runs (ciirtcftt) used information gathered from the top-term clarification form. We issued the original query with all of the terms marked "good" added, and with all of the additional suggested terms. In our training experiments, we found that this method actually harmed results, although we did not have enough data to determine whether this could be expected to hold in general. We experimented with several methods for using the terms marked "bad" to improve retrieval, but none produced promising results on the training data.

1.5 Metadata

1.5.1 Statistics on the metadata of the test topics

We summarize in the following tables the statistics on the metadata of the 50 test topics. The first column of each table stands for the type of metadata and its occurring values and the second columns gives the number of test topics with each particular metadata value. Most topics have more than one (often long) snippet of related text.

Purpose	Num. topics
Details	29
Answer	13
Background	8
Any	0

Granularity	Num. topics
Sentence	3
Passage	15
Document	20
Any	12

Genre	Num. topics
Reaction	10
International Reaction	2
Overview	19
Administrative	2
Any	17

Familiarity	Num. topics
1	1
2	25
3	16
4	3
5	0
Unknown	5

Figure 2. Statistics on the metadata of the test topics.

1.5.2 Document Metadata—Annotation and Classification

Metadata annotation. A group of five students was hired to provide relevance judgments and metadata values for the top retrieved documents for the training topics. A web interface was developed for this purpose. An html form was generated for each top-ranked document. It displayed the document and the corresponding training query. The annotators were asked to assign values for the following metadata categories:

Metadata Category	Values
Relevance	Relevant, Related, Non-relevant
Time (for purpose)	Historical, Current, Future, Other
Expertise (for familiarity)	Child, Amateur, Novice, Expert
Granularity	Document, Passage, Sentence, Phrase
Genre	Overview, Administrative, I-Reaction, Other

Some of the above document metadata categories are different from the query metadata categories. It was not clear how to assign values to some of the query metadata categories, when starting from a top-ranked document for a query. Two such categories were "Purpose" and "Familiarity". Instead, we chose metadata categories "Time" and "Expertise" as indicators of "Purpose" and "Familiarity", respectively.

Familiarity mapping. Initially we built a familiarity classifier using support vector machines (SVM) [7], to classify documents into one of the four categories: expert, amateur, novice, and child. However, by using ten-fold cross validation on 629 instances, the classifier only yielded an average accuracy of 71%. To insure the availability of sufficient number of documents for retrieval for each familiarity level, we instead developed the following method.

To determine which document should map to which familiarity metadata value, we compute the readability indices and then map the scores. Readability/reading level describes the ease with which a document can be read or understood [3]. While familiarity and reading level are different, we made an assumption that materials that are more difficult to understand would be more appropriate for a user that is more familiar with a topic. We have computed five readability indices for each document, namely, Dale-Chall [2], FOG [5], Holquist [6], Flesch-Kincaid [8, 9], and SMOG (the Simplified Measure of Gobbledygook) [12]. The statistics of document readability scores across the whole collection is given in table 3. By manually checking how well each of the readability indices is able to differentiate various documents of known levels, we selected SMOG as the final measure.

Table 3. Statistics of document readability scores.

	Min.	Max.	Median	Mean	Std. Dev.
SMOG	3	153.00	10.94	11.99	6.39
Holquist	7.12	5509.0	31.77	47.20	105.64
Dale-Chall	3.69	978.26	4.55	5.48	5.99
Flesch-Kincaid	0	7665.9	9.04	15.95	47.23
FOG	0.03	7867.0	13.03	20.35	48.38

From figure 2, we see that a majority of the test topics specify familiarity values 2 and 3. To insure the quality of retrieval, we made a corresponding mapping that allowed most documents to fall under those two categories. The mapping scheme is shown in table 4. For example, if a document has a SMOG readability score of 9, it is mapped to familiarity=2. If a test topic also specifies the same familiarity value, this document will then be included in the pool of 155,372 documents to perform retrieval for that topic. There are a total of 372,219 documents in the collection out of which 128 documents have no familiarity score because they have no actual contents.

Table 4. Mapping between document SMOG score and familiarity metadata value

Familiarity value	1	2	3	4	5
Range of SMOG score	<=7	(7, 10]	(10, 14]	(14, 19]	>19
Num. of docs mapped	35185	155372	143053	61418	20725

Genre classification. We built a genre classifier using support vector machines. There are total of 615 training documents for this metadata from annotation, out of which 373 were annotated as "overview", 172 as "administrative," 31 as "i-reaction," 21 as "reaction," and 17 as "other." We applied ten-fold cross validation and obtained an average accuracy of 90%. However, the number of training instances is small thus the quality of the classifier can not be reliably determined. When we applied this classifier on the 10 NIST-provided training topics, it was found that 89% of the retrieved documents were classified as "overview" and the remaining 11% were classified as "administrative". There were no instances that were classified as "reaction" or "i-reaction." Because of the potential impact that the classification results might have on the final retrieval, we decided not to use it in our final submission. Instead, we resort to more conservative measures discussed in section 1.6.1.

Purpose classification. A purpose classifier was built, again using SVMs, to classify documents into either "background" or "details," because the third value "answer" can be handled together with the granularity metadata. The documents that were tagged as "current" or "future" for metadata Time used in annotation are considered having a Purpose metadata value of "details", and those tagged "historical" for Time map to "background" for Purpose. Out of the 567 annotated documents for this metadata only 6 were given the value "background." A classifier trained on this data is clearly not reliable. Due to time constraints, we did not use this metadata in our submission runs.

1.6 Incorporating Metadata

1.6.1 Genre

As discussed above, attempts at more sophisticated learning-based methods of assigning genres to documents were unsuccessful so our submitted runs rely on ad hoc rules for re-ranking documents. In the runs that used the genre metadata field (ciirtmda and ciirtmdap), documents were ranked using baseline methodology and then re-ranked according to the following rules:

1. If the query metadata field was **overview**, any document from the Federal Register or Congressional Record was moved five places down the ranked list.

2. If the query metadata field was **reaction**, any document from Xinghua English or the Federal Register was moved five places down the ranked list.

3. If the query metadata field was **i-reaction**, any document *not* from Xinghua English was moved five places down the ranked list.
4. If the query metadata field was **administrative**, any document not from the Congressional Record or Federal Register was moved to the end of the ranked list. Because of the size of the CR and FR document collections, this means that only CR and FR documents were ever returned in the top 1000 documents.

These rules were derived largely from our own experience working with the documents and due to the limited training data, we were generally unable to test their effectiveness.

1.6.2 Familiarity

For all runs incorporating familiarity metadata (ciirtmda and ciirtmdap), we tagged every document with an integer familiarity score in the range 1 to 5 as explained in section 1.5.2. After each document was ranked using the baseline methodology, the rankings were shifted slightly by subtracting a δ value from the rank of each document. We set $\delta = 2 - |(\text{query familiarity score}) - (\text{document familiarity score})|$.

1.7 Granularity

Two of our runs (ciirtmdgp and ciirtmdap) incorporated query granularity by retrieving passages of appropriate size. For queries with Granularity value "Sentence" and " Passage", passages of length 50 words and 150 words, respectively, were retrieved. For the value "Any", ranked lists for passages of size 50 and 150 were merged, after normalization, with the ranked list of documents from ciirtmda. A simple measure of "novelty" was used to reduce overlap between different sized passages from the same document. If a passage or document had more than a 75higher up in the ranked list, it was removed from the ranking. The resulting list was submitted as ciirtmdgp.

The entries in the merged list were reranked using the familiarity and genre information as indicated in the previous section. The "novelty" measure was also applied and the reranked list was submitted as ciirtmdap.

1.7.1 Related Text

Only one of our runs (ciirtrt) used the related text query metadata. Because the related text included relatively large amounts of text (often entire articles), we were concerned that simply appending the related text to the original query might dwarf the query terms. To compensate for this our new query was set to the original query repeated 100 times with the related text appended. Again, we did not have sufficient data to test this model so the number 100 is arbitrary.

1.7.2 Summary of Submitted Runs

We submitted 10 runs in all, including five baseline runs and five other runs that incorporated different combinations of metadata and clarification form data. Table 4 summarizes each of the runs submitted. The first five lines give some information about the baseline runs and the last five lines show what techniques each of the other runs used.

Table 4. Summary of submitted runs.

RUNID	QUERY		METADATA					CLAR. FORM	RETURNS	
	title	descrip.	gran.	purp.	genre	famil.	rltd. text	top terms	docs	psgs.
ciirtbas	*								*	
ciirtdbas	*	*							*	
ciirtpsgbas	*								*	
ciirtdpsgbas	*	*							*	
ciirtp	*									*
ciirtmda	*				*	*			*	
ciirtmdap	*		*		*	*			*	*
ciirtmdgp	*		*						*	*
ciirtrt	*						*		*	
ciirtcftt	*							*	*	

1.8 Results

1.8.1 Document-Level Evaluation

Table 5 shows document-level evaluation results for all ten runs submitted. The top line shows the results for ciirtbas, our best-performing baseline run. Boldface numbers indicate runs that were significantly different (using the Student's t-test at the .05 significance level). Each significance test was performed against the ciirtbas entry in the same column. None of our runs performed significantly better than our ciirtbas baseline. One run, ciirtmda, had better hard average precision and hard precision at 20 documents than the baseline. However, the improvement was not statistically significant.

For 20 of the 48 queries, adding the "good" terms from the clarification form to the original query improved average precision and for 19 of the queries precision at 20 documents retrieved improved. Using genre and familiarity improved the average precision of 16 queries but only improved precision at 20 documents retrieved for two of the queries. We should note here that 4 of the 48 queries had genre of "any" and familiarity "unknown" so they stood no chance of improving when techniques to incorporate genre and familiarity metadata were incorporated. Using the related text metadata to augment the query resulted in better average precision for 16 queries and better precision at 20 documents retrieved for 11 queries. This means that the use of the chosen metadata and our clarification form did help improve retrieval in some cases but not in general.

Advance discussion on the track mailing list led us to believe that only Congressional Record and Federal Register could be considered relevant using HARD relevance evaluation criteria on queries with genre equal to "Administrative." To take advantage of this known fact we designed our system only to retrieve those two types of documents for "Administrative" queries. Interestingly, however, for the two "Administrative" queries in the testing set (HARD-069 and HARD-176), there were several documents from the New York Times and Xinghua English collections that were marked relevant. As a result, our system did worse on both of these queries than it did using the baseline method.

Table 5. Document-level evaluation of all runs. Standard deviations are shown in parenthesis.

RUNID	SOFT AVG. PREC.	HARD AVG. PREC.	SOFT PREC. @ 20	HARD PREC. @ 20
ciirtbas	0.3518 (0.2777)	0.3091 (0.2737)	0.5365 (0.3681)	0.4250 (0.3341)
ciirtdbas	0.3314 (0.2508)	0.2969 (0.2452)	0.5198 (0.3462)	0.4156 (0.3239)
ciirtpsgbas	**0.3052** (0.2371)	**0.2529** (0.2279)	**0.4719** (0.3334)	**0.3438** (0.2822)
ciirtdpsgbas	0.3029 (0.2106)	0.2640 (0.2073)	**0.4708** (0.2988)	**0.3708** (0.2837)
ciirtp	**0.3049** (0.2367)	**0.2526** (0.2275)	**0.4719** (0.3334)	**0.3438** (0.2822)
ciirtmda	0.3500 (0.2811)	0.3136 (0.2815)	0.5344 (0.3689)	0.4260 (0.3361)
ciirtmdap	**0.3056** (0.2540)	0.2682 (0.2497)	**0.5031** (0.3593)	0.3844 (0.3094)
ciirtmdgp	**0.3036** (0.2519)	0.2662 (0.2472)	**0.5031** (0.3593)	0.3844 (0.3094)
ciirtrt	0.3430 (0.2644)	0.3016 (0.2644)	0.5469 (0.3809)	0.4250 (0.3639)
ciirtcftt	0.3146 (0.2669)	0.2942 (0.2668)	0.5448 (0.3900)	0.4469 (0.3810)

1.8.2 Passage-Level Evaluation

Table 6 shows the results of passage-level evaluation for each of the submitted runs. As in Table 5, boldface numbers indicate runs that were significantly different (using the Student's t-test at the .05 significance level). Each significance test was performed against the ciirtdbas entry in the same column.

Results are similar to those in the previous section. None of the runs performed significantly better than the baseline. The baseline run with highest R-Precision is ciirtdbas. Three of the submitted runs, ciirtmdap, ciirtmdgp and ciirtrt, gave higher R-Precision and Passage Precision @ 20 docs than the best baseline run, but the differences were not statistically significant.

Using granularity information to retrieve passages of different size improved R-precision for 12 queries. Of the remaining queries, 19 had the granularity value "Document". The R-precision for these queries was unaffected by incorporating granularity. For 6 other queries, no relevant documents were retrieved in the baseline run, ciirtbas. Since this list was the starting point for passage retrieval, R-Precision remained at 0. For 6 queries, R-precision was adversely affected by the use of granularity values. Therefore, using metadata values helps some queries, but overall, it does not significantly affect performance.

Table 6. Passage-level evaluation of all runs. Standard deviations are shown in parenthesis.

RUNID	R-PRECISION	PASSAGE PREC. @ 20	F-MEASURE
ciirtdbas	0.2301 (0.221)	0.282 (0.3441)	0.209 (0.2061)
ciirtbas	0.2261 (0.2326)	0.2801 (0.3585)	0.2063 (0.2001)
ciirtpsgbas	**0.1853** (0.2076)	0.226 (0.2984)	0.1576 (0.1622)
ciirtdpsgbas	**0.1942** (0.1871)	0.2474 (0.3234)	0.1627 (0.1522)
ciirtp	0.2093 (0.1913)	0.2563 (0.3279)	**0.0929** (0.0878)
ciirtmda	0.2261 (0.2306)	0.2805 (0.3434)	0.211 (0.2102)
ciirtmdap	0.2542 (0.2367)	0.292 (0.3686)	0.1943 (0.2035)
ciirtmdgp	0.2541 (0.2367)	0.2917 (0.3687)	0.1943 (0.2035)
ciirtrt	0.2327 (0.2371)	0.2842 (0.3312)	0.2105 (0.2127)
ciirtcftt	0.2229 (0.2235)	0.3144 (0.4012)	0.1974 (0.2038)

2 Question Answering track

In the QA track, we developed a dynamic passaging retrieval system to identify passages likely to contain answers.

CIIR last participated in the QA task in TREC 9 (2000). At that time we fielded the Marsha system [11]. This system was based on an INQUERY document retrieval engine followed by the application of a series of heuristics rules to identify 250-byte long passages in the retrieved documents that were likely to contain the desired answers.

This year's passage sub-task in the QA track has allowed us to participate once again utilizing our current approach to passage retrieval with language models. We developed a dynamic passaging system that retrieved document passages based on the simplest implementation of language models: cross-entropy between bag-of-word models for a question and a candidate passage.

2.1 Theoretical QA model

Our system used a simple approach to passage retrieval for a question. We constructed a MLE model for the question by treating the words in the question independently, the so-called bag-of-words (BOW) model. A collection-smoothed, with a Dirichlet-prior was used to create a BOW model for a candidate passage. The candidate passage was then ranked by the cross-entropy between its model and the unsmoothed model of the question:

$$H(q|p) = -\sum_{i=1}^{n} P_i(q) \ln P_i(p)$$

where q denotes the question model, p the passage model and n is the number of unique words in the question. This measure is rank-equivalent to the more familiar query-likelihood formula. But we utilize this alternative formulation because it will allow us to build in the future a Bayesian classifier in combination with the use of Relevance Models where the cross-entropy is calculated between a relevance model and the passage model.

2.2 QA System implementation

Passage retrieval was done in two stages. The initial stage consisted of document retrieval using the #sum INQUERY operator. The retrieval used the an index that had INQUERY-stopped and Krovetz-stemmed the AQUAINT collection. Similar stopping and stemming was applied to the questions. We should point out that this step was done solely for the purposes of time savings. As we will comment later on this section, skipping this document retrieval step and going directly to the passage retrieval step produces essentially the same performance.

The top 60 documents (as determined by tuning on TREC 2002 questions) were then selected for the passage retrieval phase. Documents were passaged on-the-fly and sliding windows that moved forward one word at a time while guaranteeing the 250-byte evaluation limit were ranked according to the cross-entropy formula detailed above. Only one-passage per document was allowed to appear on the ranked list.

We tested whether skipping the document retrieval phase would have gained us any performance benefit by looking at the rank-1 measure on the TREC 2002 question set and retrieving 1KB passages. Skipping the document retrieval and doing passage retrieval on the whole AQUAINT collection gave us a rank-1 performance of 38.2%. The two-step retrieval gave us the same rank-1 performance.

The system did no processing to recognize NIL-answer questions so the NIL token was never returned.

2.3 QA Results

The evaluation metric (rank-1 correct) for this year was 20.1%. This compares favourably with our expectations of 19.2% obtained by tuning on the TREC 2002 question set.

3 Conclusion

We participated in the HARD and QA tracks. In both cases, we believe that our results are good, though we had hoped for better. In the case of HARD, we look forward to trying again with a more mature track, based on the lessons learned and with the training data collected this year.

Acknoweldgments

This work was supported in part by the Center for Intelligent Information Retrieval, in part by SPAWARSYSCEN-SD grant number N66001-02-1-8903, and in part by Advanced Research and Development Activity under contract number MDA904-01-C-0984. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsors.

References

[1] Corrada-Emmanuel, A., Croft, W.B., and Murdock, V. Answer passage retrieval for question answering. CIIR Technical Report, 2003.

[2] Dale, E. and J. S. Chall. (1948). A formula for predicting readability, *Education Research Bulletin*, 27, 11-20, 37-54.

[3] Gilliland, J., (1972). *Readability*. University of London Press, 1972.

[4] Griths, A., Robinson, L.A., & Willett, P. (1984). Hierarchic agglomerative clustering methods for automatic document classification*Journal of Documentation*, 40, 175-205.

[5] Gunning, R. (1952). *The Technique of Clear Writing*. McGraw-Hill.

[6] Holquist, J.B. (1968). *A determination of whether the Dale-Chall readability formula may be revised to evaluate more validly the readability of high school science materials*. Ph.D. thesis, Colorado State University.

[7] Joachims, T. (2002). *Learning to Classify Text Using Support Vector Machines*. Ph.D dissertation, Dept. of Computer Science, Cornell University. Kluwer.

[8] Kincaid, J.P., Fishburn, R.P., Jr. Rogers, R.L., and Chissom, B.S. (1975). Derivation of new readability formulas for Navy enlisted personnel. *Research Branch Report 8-75*, Naval Air Station Memphis, Millington, Tennessee, 40 pages.

[9] Kincaid, J.P., Aagard, J.A., O'Hara, J.W., and Cottrell, L.K. (1981). Computer readability editing system. *IEEE Transactions on Professional Communications*, Vol. PC-24, No. 1, pp. 12-22.

[10] Lavrenko, V. and Croft, W.B. (2001). Relevance-based language models. In W.B. Croft, D.J. Harper, D.H. Kraft, & J. Zobel (Eds.), *Proceedings of the 24th annual international ACM-SIGIR conference on research and development in information retrieval*, New Orleans, Louisiana (pp.120-127), New York: ACM.

[11] Li, X. and Croft, W.B., Evaluating Question Answering Techniques in Chinese. *Proceedings of the Human Language Technology Conference*, pp. 201-206, 2001.

[12] McLaughlin, H. (1969). SMOG grading - a new readability formula, *Journal of Reading*, 1969, 22, 639-646.

[13] Miller, D., Leek, T., & Schwartz, R. (1999). A hidden Markov model information retrieval system. In *Proceedings of the 22nd annual international ACM SIGIR conference*, pp. 214-221.

Robust and Web Retrieval with Document-Centric Integral Impacts

Vo Ngoc Anh *Alistair Moffat*

Department of Computer Science and Software Engineering
The University of Melbourne
Victoria 3010, Australia

{vo,alistair}@cs.mu.oz.au

Abstract: *This paper reports the experiments done at The University of Melbourne for the Robust and Web tracks of* TREC-2003. *We explore the idea of determining the impact of a term locally within the document and in a qualitative manner instead of a quantitative one. The impact of each distinct term in a document or query text is defined to be a small integer. The scalar product of the impact vector for a document and the impact vector for a query is taken to be the similarity score between them, an arrangement that allows very fast query evaluation.*

1 Document-Centric Integral Impacts

Consider a document collection with N documents and n distinct terms. Use is often made of the TF-IDF rule to assess the similarity degree between a query q and any document d of the collection. An n-dimensional vector space is constructed, with each dimension representing a term that appears in the collection. In the space each document d is represented as

$$d = (w_{d,t_1}, w_{d,t_2}, \ldots, w_{d,t_n}),$$

and the query q as

$$q = (w_{q,t_1}, w_{q,t_2}, \ldots, w_{q,t_n}).$$

In this framework, the t_i are the distinct terms of the collection, and $w_{x,t}$ is the projection of document or query x in dimension t. That is, $w_{x,t}$ is the "importance" of t in x, and can be calculated by any formulation obeying the TF-IDF requirement. A similarity score $S(d,q)$ between d and q calculated by the cosine measure is of the form:

$$S(d,q) = \frac{\sum(w_{d,t} \cdot w_{q,t})}{\sqrt{\sum w_{d,t}^2} \cdot \sqrt{\sum w_{q,t}^2}}, \qquad (1)$$

where the three summations are over all n terms.

Anh et al. [Anh et al., 2001, Anh and Moffat, 2002a] consider $w_{d,t}/\sqrt{\sum w_{d,t}^2}$ as the *impact* of the term t in d and propose a transformation that maps the impacts into small integers. The transformation is done in the context of the whole document collection. They obtained better effectiveness than using non-transformed impacts even when only $k = 32$ different values are used for transformed impacts.

In our TREC-2003 experiments we define the same mapping locally within each document rather than globally in the whole collection. The rationale for this change is to give all the documents an equal spread of high impact terms and low impact ones.

Because the transformation is done locally within each document, we do not need to rely on the document length factor in Equation 1. We can choose a step further by not to be bound to any particular formulation of $w_{d,t}$, which is often very diverse. In its original form, TF-IDF is not a precise definition – it is a philosophy. It is our human desire for precision in computation that has led to overly precise formulations, with their various tuning factors. Adopting a philosophy that "rank is more important than value" allows us to reduce the effect that numerical constants have on the effectiveness of the ranking [Anh and Moffat, 2002b].

Ideally, we would like to create the sorted list of terms corresponding to each document without doing any detailed computation. The surrogate weights of the n_d terms in document d are then computed for use in the query processing regime, without any further recourse to collection or document statistics.

To achieve this independence, we focus on one document at a time, and divide the process of determining the transformed impacts of terms in a document d into two phases. The first *sorting* phase orders the list of n_d terms d in decreasing order of term contribution. In the second *mapping* phase, each value in the term list is converted to an integer in range 1 to k. As the end of the process, these integer values serve as document term impacts.

The same process can be done to define query term impacts for the terms in any query q.

Finally, the similarity score between d and q is computed as the sum of the products of document term impact and query term impact of the terms shared by d and q.

The question to be faced is how to specify the sorting and the mapping phases. We sorted the terms in decreasing term frequency $f_{d,t}$, with ties broken using increasing f_t as a secondary key. This two-part sort key implicitly imports the TF-IDF rule into our scoring system. Note that the reverse sorting order, where f_t is used as the primary key, and $f_{d,t}$ as the secondary can not be expected to work as well, as the large number of different f_t values means that the TF factor would rarely be brought into play.

In the mapping phase, we follow the main idea of Fibonacci that in a structure, the number of "important" elements should be less than the number of less important ones. Hence we choose the mapping so that the number of elements in a document corresponding to each surrogate weight, in decreasing order of the weights, forms a geometric subsequence. That is, $x_i \approx x_{i+1} \cdot B$, where $B = k^{1/k}$ and $x_0 \approx n_d \cdot (B-1)/(k-1)$ is the number of elements with the highest impact value.

As an example, when a document contains $n_d = 250$ distinct terms and $k = 10$, the most frequent (within that document) 7 terms are assigned the highest impact of ten, and the least frequent 57 terms in that document are assigned the lowest impact value of one.

2 Task Setting and Performance

Robust Track

In the robust track, the collection TREC45-CR (that is, disk 4 and disk 5 of the TREC corpus, minus the *Congressional Record*) is employed with 50 old topics and 50 new topics. The opportunity of using the 50 old topics for training was not exploited.

Our goal in this track is to measure the effectiveness of the retrieval methods respective a range of query length. The following type of queries, that are automatically generated from the original topics, are used as input: title-only queries, set T; description-only, D; title-and-description, TD; and the whole topic, query set TDN.

The standard settings described in the previous section were applied directly for the Robust track. The number of different impacts was set to $k = 10$, a value found to be appropriate based upon experiments carried out on the TREC-2002 data [Anh and Moffat, 2002b].

Our preliminary experiments with other collection show that using document meta data to compute quasi term frequencies, and using them as surrogate to the raw term frequencies in ordering, leads to improvement on effectiveness. The quasi frequencies can be calculated, for example, as weighted sum

Query type	Precision at 10	No of topics not found in top 10	Area underneath MAP(X) for worst 25 topics
For 50 old topics			
T	0.2800	**8**	0.0029
D	0.2680	**9**	0.0041
TD	**0.3340**	**7**	**0.0093**
TDN	0.2980	8	0.0079
For 50 new topics			
T	**0.4200**	**4**	0.0365
D	0.4480	**5**	**0.0205**
TD	**0.5060**	**1**	**0.0456**
TDN	0.4880	4	0.0383
For all 100 topics			
T	**0.3500**	**12**	0.0087
D	0.3580	**14**	**0.0090**
TD	**0.4200**	**8**	**0.0175**
TDN	0.3930	12	0.0155

Table 1: Effectiveness performance on different type of queries. Figures in bold represent values that are not inferior to the median performance of the 2003 TREC runs for the same category of query.

of frequencies of the terms in different text components. Unfortunately, the different text structures in the sub collections of TREC45-CR meant that we did not assign a weight to each component, and only raw term frequencies were used in our runs.

Table 1 presents the performance of our system with respect to different types of queries. Overall, the TD queries obtain the best performance, and TDN is the worst. The poor performance of the TDN queries is presumed to be a consequence of the noise information in the narrative fields of the original topics. The same observation explains the relative weakness of the D in comparison with the query set T. Another possible reason for the relative good performance of the short queries is that we do not apply any pruning technique to prevent the low impact terms in long queries adding their contributions to the final score.

Web Track

In the Web track, the corpus .GOV is used, with 300 topics for the named page finding task, and another 50 for the topic distillation task.

Meta data is used consistently in .GOV, and we made use of its availability for ordering terms in documents. Incoming and outgoing anchor text is also employed. The way to use these extra information is to count the frequency of terms separately in different component of the text, and use a combination of the partial frequencies to determine a primary key for sorting, prior to impact assignment. The second sorting key is either not used or bound to the IDF factor as in the standard settings.

As the first step, we divide each text into the following components:

- *URL text*. The words in the URL can be valuable for both content search and web search, as it is indicated by many of the last year participants. One possible problem is that in many cases acronyms (such as nist) are used. Content management systems that emit alphanumeric strings of gibberish are also an issue. As an initial remedy to this problem, we preprocess the URL text before indexing. The details of this preprocessing are given in the next subsection.

Run name	Description
W3	Weights of $(1, 2, 2, 4, 4, 8)$ are assigned to components (content, meta text, outgoing anchor, incoming anchor, title, URL); then the weighted sum of these components is used as the sorting key. The IDF factor is not employed.
W4	As for W2, but the component weight vector is $(1, 2, 2, 2, 4, 16)$; and IDF is used as a secondary sort key.
S5	The primary sort key is defined as the complex key that represents the term frequency in the sequence (URL, incoming anchor, outgoing anchor, title, meta, content); with IDF used as a secondary sort key.
W1	As for W3, but only entry pages are indexed. Non-entry pages are ignored.

Table 2: Different sorting methods used to define impacts. The left column is a method identification, with the digit being used as the last digit in the names of our two sets of official runs.

- *Titles.* The title field represent a concise summary of the document, and is usually provided by the author of the document. Other visible markup – for example, headings and bold terms – can also be used with some confidence.
- *Incoming anchor text.* Incoming anchor text can be regarded as being an assessment of the content of this document that is provided by a reader of it, rather than the author of it. The ubiquitous "click here" is not helpful in this regard, and nor are clickable images.
- *Outgoing anchor text.* Outgoing anchor text (provided in links in this document that lead to other documents) do not normally reflect the content of this document. It can, however, be valuable in inferring site structure and page connectivities for the topic distillation task.
- *Metadata.* Other meta text, such as keywords, subject fields, and explicit metadata, may also be of assistance. But there is also a risk that it will have been inherited from a previous document and not edited when the content was revised. Even if it has been edited, it will not have been proofread with the same care as the visible text is.
- *Content text.* The plain text of the page is also a reflection of its content, but not of its importance. Content is the baseline resource of all document retrieval systems.

For each document, after counting the term frequency in each text component, different methods can be used to order the terms in a document based on the number of occurrences of each term in each of these categories. The methods used for the TREC runs are listed in Table 2. In all of the methods, the number of different impact values is set to $k = 10$ (as was also the case for the Robust track).

Manipulating URL text

To facilitate use of the URL text, we have employed a simple process that attempts to de-acronymize it. The two parts considered for expansion are the host name, and the last component of the path name of the document (not counting the default page specification such as `index.html`).

The expansion is based on the title of the page and incoming anchor text, to look for possible variants or abbreviations that might give rise to the words in the URL for that page. If the metadata field "Subject" is available, it is also used as part of this process.

Method	Topic Distillation		Home Page Finding		
	P10	AVP	MRR	F10	NF
W3	0.0660	0.0537	**0.508**	**76.7%**	10.3%
W4	0.0400	0.0559	**0.530**	**75.0%**	10.7%
S5	0.0580	0.0728	**0.527**	**76.0%**	11.0%
W1	**0.0920**	**0.1096**	0.288	40.0%	58.0%

Table 3: Effectiveness of the configurations described in Table 2 for the topic distillation and home page finding tasks of the TREC-2003 Web track. The metric P10 is the precision at 10 documents retrieved; AVP is the average precision over all relevant documents; MRR is the mean reciprocal rank of the first relevant document; F10 is the percentage of topics for which the named page found in top 10; and NF is the percentage of topics for which no named page was found in the top 50 documents retrieved. Figures in bold represent values that are higher than the median performance of the 2003 TREC runs for the same task. Comparative figures are not available for the last column.

For example, document G00-00-0000000 in .GOV was crawled from the URL http://www.aspe.hhs.gov. Many of the incoming links that lead to it use variants of "Office Assistant Secretary Planning Evaluation" as their anchor text. The algorithm assigns the words "aspe assistant secretary evaluation planning evaluation hhs" by sorting the set of title, subject, and incoming anchor text words into decreasing frequency, and then choosing for inclusion one matching word for each letter in the URL. Similarly, with http://www.cs.mu.oz.au, "computer science cs mu oz" should be identified from anchor text and title text, and used subsequently as URL-related terms.

For document G00-00-0114013, the original URL was http://wwwcalfed.water.ca.gov, and the URL-based terms are just the words "wwwcalfed water ca". This example is not based upon an acronym, and is correspondingly more complex. It is not satisfactorily handled by the ad-hoc process, and the small number of anchors containing "CALFED Bay Delta Program" were overlooked. The URL www.unimelb.edu.au is also hard to handle using the simple mechanism.

As a final example, incoming anchor text for document G00-00-0497574 from the URL http://wrf.fsl.noaa.gov is all "WRF", but the title words "Weather Research and Forecasting (WRF) Model" contribute to the expansion, and allow "wrf weather forecasting research fsl noaa" as URL-derived index terms.

Topic Distillation and Home Page Finding Tasks

This year's topic distillation task seems biased in favor of entry pages. While this is generally true, there are also exceptions – for example, a person with a particular hobby might build an excellent resource page which can be the target of a topic distillation query, but is probably within that person's web site. Consequently, we treat all pages equally in three of our four runs for this task. In the other run only entry pages are indexed, as listed in Table 2. The same configurations were used for the home page finding task. The performance for both of the Web tasks is shown in Table 3.

As can be expected, for the topic distillation, W1 is the best, since it indexes only entry pages. However, this policy dramatically degrades the effectiveness of the named page finding task. A clear conclusion from here is that, if the topic distillation keeps its present definition we need to find another way to address it in order to get a compromise with the home page finding task.

For both of the tasks, all of the three remaining methods give about the same level of effectiveness. That means that the weighting scheme we applied is not effective enough. There should be another, better, way to combine the term frequencies in different components.

3 Practical Considerations

Our experiments were conducted in a 933 MHz Intel Pentium III with 1 GB RAM, a 9 GB SCSI disk for system needs, and four 36 GB SCSI disks in a RAID-5 configuration for data. The computer is also a server for a Beowulf cluster of 16 nodes, each an 800 MHz Intel Pentium III with 256 MB RAM and local hard disk of 40 GB. There is a link with the capacity of 100 Mbits/second to and from each node as well as the server with a network switch. The system runs Debian GNU Linux, and all the programs are written on ANSI C, compiled with gcc, with the optimization flag -O3. For the Robust task, the experiments are done in the server, while all the Web tasks are done by the cluster in a parallel manner.

Except for the methods used determining the impacts, all the index structure and query processing are as described by Anh et al. [2001], and are based on the *MG* software (available at `http://www.cs.mu.oz.au/mg/`, and described by Witten et al. [1999]).

Index construction using the integer impacts is straight forward. After an additional pass to link the incoming anchor text to the target documents (this pass also computes the document frequencies for each term, if needed) the process of assigning impacts can be done locally within a document.

Parallelism is implemented as follow. The data collection is roughly divided between the nodes of the cluster, and each part transferred to the local disk of the corresponding node. The server plays the role of interface – it receives requests for indexing or querying, and sends them to all the nodes. In the case of indexing, the nodes work independently. In the case of querying, after receiving a query, each node produces a list of answers of its own based on its sub-collection, then the results are merged between pairs of node until one of them finally has the complete results. That node then send the results to the server.

Acknowledgement This work was supported by the Australian Research Council.

References

V. N. Anh, O. de Kretser, and A. Moffat. Vector-space ranking with effective early termination. In W. B. Croft, D. J. Harper, D. H. Kraft, and J. Zobel, editors, *Proc. 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 35–42, New Orleans, LA, September 2001. ACM Press, New York.

V. N. Anh and A. Moffat. Impact transformation: Effective and efficient web retrieval. In M. Beaulieu, R. Baeza-Yates, S. H. Myaeng, and K. Järvelin, editors, *Proc. 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–10, Tampere, Finland, August 2002a. ACM Press, New York.

V. N. Anh and A. Moffat. Vector space ranking: Can we keep it simple? In J. Kay, editor, *Proc. Australian Document Computing Symposium*, pages 7–12, Sydney, December 2002b. University of Sydney, Australia.

I. H. Witten, A. Moffat, and T. C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann, San Francisco, second edition, 1999.

The University of Michigan at TREC 2003

Jahna Otterbacher, Hong Qi, Ali Hakim and Dragomir Radev
University of Michigan
Ann Arbor, MI 48109
{jahna, hqi, alihakim, radev}@umich.edu

February 2, 2004

1 Introduction

This year the University of Michigan team participated in all four tasks of the Novelty track. Our basic approach for all the tasks involved using our multi-document summarizer, MEAD [1], as well as Maxent 2.1.0 software for training maximum entropy classifiers[1]. We submitted five runs for each of the four novelty tasks.

2 General Approach

2.1 Data

We created training and dev/test data sets for training models for each of the four tasks. In tasks 1 and 2, we used the Novelty 2002 test data for training. 25 clusters were assigned to our training set, while the remaining 25 were used for our dev/test data. For tasks 3 and 4, we were given some information about the 2003 test data, so our training and dev/test data were based on the information provided. Table 1 describes our training, dev/test and test data for each of the four tasks.

2.2 Features used in classification

We experimented with six sentence features in building classifiers to detect novel and relevant sentences. Three were standard sentence features in the MEAD package while the other three used the topic queries provided for each cluster in the data.

- **Centroid** The centroid score quantifies the extent to which the sentence contains lexical items that are key to the overall cluster of documents.

[1] *http : //maxent.sf.net*

Task	Description	Training data	Dev/Test	Test
1	Find all novel and relevant sentences	25 clusters from 2002 data	25 clusters from 2002 data	50 clusters from 2003 data
2	Given the list of relevant sentences for all documents in the test data, find all novel sentences	Relevant sentences from 25 clusters in the 2002 data	Relevant sentences from 25 clusters in the 2002 data	50 clusters from 2003 data
3	Given the list of relevant and novel sentences in the first 5 documents of all 2003 clusters, find the relevant/novel sentences for the last 20 documents	Relevant and novel sentences for the first 5 documents in clusters N1-N25	Relevant and novel sentences for the first 5 documents in clusters N26-N50	Last 20 documents of the 50 clusters in the 2003 test data
4	Given the list of relevant sentences in all documents in the test data and the list of novel sentences in ·the first five documents, find the novel sentences in the last 20 documents	First 5 documents in clusters N1-N25	First 5 documents in clusters N26-N50	Relevant sentences in the last 20 documents of the 50 test clusters

Figure 1: Data used in each of the four tasks

- **Length** The length of the sentence by number of words.
- **Position** The position of the sentence in its original document.
- **QueryTitleCosine** The cosine o the vectors representing the title portion of the query for the cluster and the sentence.
- **QueryTitleWordOverlap** The word overlap between the title portion of the query for the cluster and the sentence.
- **QueryNarrativeWordOverlap** The word overlap between the narrative portion of the query of the cluster and the sentence.

For each task, the features were discretized. The cosine feature was broken down into three categories (high, medium and low scores), while the remaining five features were converted to be binary. The threshold values for discretizing the feature scores were tuned using the training data for each task.

2.3 Model

As mentioned, we built maximum entropy classifiers to find relevant and novel sentences in the four tasks. In other words, for each task we found the best conditional exponential model to determine the probability of a sentence being relevant (or novel), given the values of its six features.

- The probability of a sentence being labeled (as novel or relevant, depending on the task) is

$$Prob(label|f) = Z * e^{\sum_i \lambda_i * fd_i(lf)}$$

where w_i are the model parameters (weights), Z(f) is a normalizing factor independent of l, and $\lambda_i = log(w_i)$.

For each task, we trained the above model in order to find the probability of all of the sentences in the data being assigned a label of relevant or novel. We then ranked the sentences by their respective probabilities. Finally, we experimented with our threshold values (the percentage of top ranked sentences to submit for the run). To summarize, the approach we used this year involves four steps for each task:

Step 1 Using the appropriate training data for the task, tune the threshold values in our discretization process for the sentence features.

Step 2 With the discretized data set, train the maximum entropy classifier for finding relevant/new sentences as appropriate to the task.

Step 3 Use the model to predict the probabilities of the sentences in the test data being assigned relevant/new labels. Rank the sentences by their probabilities.

Step 4 Determine the optimal cut-off value for submitting sentences in their ranked order, to the list of relevant or new sentences.

3 Task 1

In the first task, our goal was to identify the relevant and novel sentences in the 50 test clusters, given no additional information. Therefore, we used 25 clusters from the Novelty 2002 test data for our training and the other 25 for development. We developed one discretization procedure for this task, and focused on tuning our threshold for the percentage of top-ranked sentences to submit.

Table 2 shows the five runs we submitted, the performance (F-measure) we obtained on our dev/test data set, as well as the official F-measure from NIST. Note that the F-measure is reported as the average score over all clusters evaluated.

4 Task 2

For task 2, in which the list of relevant sentences was given for all 25 documents in each cluster, the goal was to identify all of the novel sentences in all documents. Therefore, a naive baseline is to submit all of the relevant sentences as novel. This naive approach achieved an F-measure of 0.6174, which seemed rather high. Therefore, we experimented both with our feature discretization procedure as well as with tuning our cut-off threshold. Our submitted runs include different combinations of discretizations and cut-off thresholds. They are described in Table 3.

Run	Description	F-measure on dev/test data	Official F from NIST
Umich11	top 3% of ranked sentences	0.113 (rel) 0.117 (new)	0.192 (rel) 0.182 (new)
Umich12	top 3.5% of ranked sentences	0.126 (rel) 0.119 (new)	0.086 (rel) 0.093 (new)
Umich13	top 4% of ranked sentences	0.129 (rel) 0.121 (new)	0.110 (rel) 0.115 (new)
Umich14	top 4.5% of ranked sentences	0.127 (rel) 0.119 (new)	0.134 (rel) 0.136 (new)
Umich15	top 5% of ranked sentences	0.125 (rel) 0.118 (new)	0.156 (rel) 0.155 (new)

Figure 2: Task 1 - Runs submitted, F-measure on dev/test data, and official F-measure from NIST

Run	Description	F-measure on dev/test data	Official F from NIST
Umich21	Top 95 % of ranked sentences Discretization 2	0.617	0.394
Umich22	Top 90% of ranked sentences Discretization 2	0.592	0.377
Umich23	Top 85% of ranked sentences Discretization 2	0.545	0.361
Umich24	Top 95% of ranked sentences Discretization 1	0.599	0.209
Umich25	Top 90% of ranked sentences Discretization 1	0.596	0.212

Figure 3: Task 2 - Runs submitted, F-measure on dev/test data, and official F-measure from NIST

5 Task 3

In task 3, we were given the lists of relevant and novel sentences from the first 5 documents of each of the 50 clusters in the 2003 test data. Using no additional information (such as the list of all relevant sentences we were given in task 2), the goal was to identify all relevant and novel sentences in the remaining 20 documents of each of the 50 clusters. We decided to use the first 5 documents in clusters N1-N25 of the 2003 data as our training data set, and the first 5 documents in clusters N26-N50 as our dev/test set for this task. In retrospect, we should have also included data from the 2002 competition in our training and dev/test sets. This is because we didn't know if there were any significant differences between the first 5 and last 20 documents in the 2003 clusters that might have influenced the prior probabilities of a sentence taking one of the two (relevant/new) labels.

The prior probability of being a relevant or novel sentence was quite high in our training data set. Therefore, we considered the naive baseline of submitting all sentences as being relevant and novel. On our dev/test set, this naive approach achieved an F-measure of 0.54 for novel sentences and 0.67 for relevant sentences. By using one discretization procedure and focusing on tuning our cut-off threshold on the ranked sentences list, we found that we could beat these baselines. Table 4 shows our results.

Run	Description	F-measure on dev/test data	Official F from NIST
Umich31	top 80% of ranked sentences top 65% of ranked sentences	0.72 (rel) 0.61 (new)	0.560 (rel) 0.409 (new)
Umich32	top 75% of ranked sentences top 75% of ranked sentences	0.71 (rel) 0.59 (new)	0.565 (rel) 0.405 (new)
Umich33	top 70% of ranked sentences top 80% of ranked sentences	0.71 (rel) 0.59 (new)	0.569 (rel) 0.399 (new)
Umich34	top 65% of ranked sentences top 90% of ranked sentences	0.71 (rel) 0.56 (new)	0.566 (rel) 0.385 (new)
Umich35	top 90% of ranked sentences top 60% of ranked sentences	0.68 (rel) 0.58 (new)	0.543 (rel) 0.408 (new)

Figure 4: Task 3 - Runs submitted, F-measure on dev/test data, and official F-measure from NIST

Run	Description	F-measure on dev/test data	Official F from NIST
Umich41	all relevant sentences in last 20 documents	0.848	0.747
Umich42	top 99.5% of ranked sentences	0.848	0.747
Umich43	top 99% of ranked sentences	0.847	0.745
Umich44	top 98% of ranked sentences	0.845	0.742
Umich45	top 97% of ranked sentences	0.843	0.740

Figure 5: Task 4 - Runs submitted, F-measure on dev/test data, and official F-measure from NIST

6 Task 4

For the final task, we were given the list of all relevant sentences from all 25 documents in the 50 clusters as well as the list of novel sentences from the first 5 documents in all the clusters. Our goal was then to find the novel sentences in the last 20 documents of each cluster. Therefore, we used the first five sentences of clusters N1-N25 of the 2003 data for training and the first five sentences of clusters N26-N50 for dev/test data.

For this task, the naive baseline is to submit all of the sentences known to be relevant in the last 20 documents of each cluster. Since the F-measure of this baseline was quite high, 0.848, we were unable to beat it using any other method. Therefore, as shown in Table 5, we submitted the naive approach as one of our runs. We experimented with our feature discretization procedure as well as with using different cut-off values on our ranked list of sentences, but as expected, we did not manage to beat the baseline.

7 Conclusions

This was our second year participating in the Novelty track. While last year, we focused on using the MEAD summarizer for detecting new and relevant sentences, this year we experimented with using MEAD sentence features in building a maximum entropy classifier. Our group is still quite new to the area of novelty detection and we have again learned a lot from participating in the TREC evaluation. We found the three new tasks for this year to be challenging, particularly since naive baselines perform rather well in many cases.

For the immediate future, we plan to focus on a number of issues related to this year's novelty evaluation:

- In analyzing our algorithms' performance by cluster, we see that the F-measure can vary widely across clusters. Clearly not all clusters should be treated the same when it comes to novelty detection, and we would like to investigate how the clusters differ and how we might be able to exploit such differences in the future.

- Obviously, by using only six sentence features in our maximum entropy classifiers, we did not take advantage of the fact that this method can easily handle the addition of many features into the model. Therefore, we need to continue to develop sentence features to be used. In particular, we should think about developing features at the cluster or intra-sentence level in addition to sentence level features.

- Finally, we are interested in how the TREC notion of novelty compares to that of others (such as in [2] and [3]). For example, in the TREC novelty track, we consider novelty at the sentence level, whereas the TDT initiative can be thought of as addressing novelty at the document level. We are especially interested in considering how analyzing novelty at different levels of granularity could benefit text summarization and question-answering systems.

8 Acknowledgments

The authors of the Maxent 2.1.0 package we used are Jason Baldridge, Tom Morton, Gann Bierner and Eric Friedman.

This work was partially supported by the National Science Foundation's Information Technology Research program (ITR) under grant IIS-0082884. All opinions, findings, conclusions and recommendations in any material resulting from this workshop are those of the participants, and do not necessarily reflect the views of the National Science Foundation.

References

[1] Dragomir R. Radev, Hongyan Jing, and Malgorzata Budzikowska. Centroid-Based Summarization of Multiple Documents: Sentence Extraction, Utility-Based Evaluation, and User Studies. In *Proceedings of the Workshop on Automatic Summarization at the 6th Applied Natural Language Processing Conference and the 1st Conference of the North American Chapter of the Association for Computational Linguistics*, Seattle, WA, April 2000.

[2] James Allan, Rahul Gupta, and Vikas Khandelwal. Temporal Summaries of News Topics. In *Proceedings of the 24th Annual International ACM SIGIR*

Conference on Research and Development in Information Retrieval, pages 10–18, New Orleans, LA, 2001.

[3] James Allan, Jaime Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. Topic detection and tracking pilot study final report. In *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, February 1998.

Report on the TREC-2003 Experiment: Genomic and Web Searches

Jacques Savoy, Yves Rasolofo, Laura Perret

Institut interfacultaire d'informatique, University of Neuchatel (Switzerland)

Summary

This year we took part in the genomic information retrieval and information extraction tasks, as well as the named page and topic distillation searches. In carrying out the last two tasks, we made use of link anchor information and document content in order to construct Web page representatives. This type of document representation uses multi-vectors to highlight the importance of both hyperlink and document content.

Introduction

As a part of the TREC-2003 evaluation campaign, the UniNE group is taking part in the genomics and Web tracks. The first section of this paper describes the IR models we used in the genomics information retrieval. Section 2 describes our various approaches to automatically extracting gene descriptions from a given scientific paper, using only its title and its abstract. Section 3 describes our procedures for indexing and retrieving Web pages, based on three document representations, and our distributed indexing framework based on the Okapi probabilistic model. Section 4 explains the IR approach we used to combine both Web page content and anchor information when searching for specific named pages and homepages. Finally, Section 5 describes how our IR scheme can be used within the context of topic distillation tasks.

In order to evaluate our hypothesis, we used the SMART system as a testbed, implementing various vector-space IR schemes and probabilistic models. This year our experiments were conducted on an Intel Pentium III/600 (memory: 1 GB, swap: 2 GB, disk: 6 x 35 GB) and all experiments were fully automated.

1. Genomics Information Retrieval

In carrying out our genomic ad hoc search task, we worked with a Medline collection subset containing 525,938 documents, or more precisely, bibliographic records mainly containing an article title, an abstract and some manually assigned descriptors. The available queries consisted of a gene name (or more precisely, its official name and symbol, together with various alias, associated products and proteins).

This information retrieval task is comparable to the Amaryllis corpus composed of 148,688 scientific bibliographic records written in French. Each of the Amaryllis collection records mainly consisted of an article title, an abstract and some manually assigned descriptors. This collection was included in the CLEF-2002 evaluation campaign (Savoy, 2003).

The indexing procedure we used in both genomic tracks is described in Section 1.1, while Section 1.2 provides an overall description of various IR models. Section 1.3 describes the pseudo-relevance feedback (or blind query expansion) used in our experiments. Section 1.4 shows how we combine the various result lists generated, using different indexing and searching schemes in order to process the same document collection (data fusion). The last section evaluates the retrieval effectiveness achieved by diverse IR models and also that of various combined approaches.

1.1. Indexing Procedure

We chose the SMART system as effective means of searching the Medline collection subset. From the original documents and during the indexing process, we retained only the following logical sections: TI (title), TT (transliterated title from non-Roman alphabet language), AB (abstract), MH (MeSH terms based on the NLM's controlled vocabulary), GS (Gene symbol or abbreviated gene names), and RN (EC/RN numbers are assigned by the Enzyme Commission to designate a particular enzyme).

Only the available descriptions were used to formulate the queries. The "official symbol" field was repeated three times however in order to assign more importance to this particular field.

To form an indexing word, our system takes letter and digit sequences into account, or when preceded and followed by a letter, the following character is used: '.@!_ (the default SMART system list). Thus the strings "IBM360", "U.S", or "sym_name" are viewed as single indexing terms. To this set, we might add / and – characters (a set labeled as "Separator+") in order to view the sequence "DEAD/H" as one indexing term.

Moreover, a stemming procedure was often used in order to reduce to the same root (or stem) inflectional and derivational variants of words. For example, the words "thinking", "thinkers" or "thinks" would be reduced to the stem "think". To achieve this, we employed the Lovins' stemmer (Lovins, 1968) based on a list of over 260 suffixes (the default stemming approach in SMART). On the other hand, we sometimes preferred using a light stemming approach, wherein only the plural form of English words were removed. To evaluate this stemming approach, we adopted the "S stemmer" (Harman, 1991) based on the following rules:

1. If a word ends in «-ies», but not «-eies» or «-aies» then replace «-ies» by «-y»;
2. If a word ends in «-es», but not «-aes», «-ees» or «-oes» then replace «-es» by «-e»;
3. If a word ends in «-s», but not «-us» or «-ss» then remove the «-s».

1.2. Search Models

In order to define a retrieval model, we must specify how the documents and the requests are to be represented (indexing procedure) and how the similarity between document and query surrogates are to be computed. To achieve this and in order to obtain a broader view of the relative merit of various retrieval models, we evaluated the genomic corpus using 11 search models.

As a first approach, we adopted a binary indexing scheme within which each document (or request) was represented by a set of keywords, without any weight. To measure similarity between documents and requests, we counted the number of common terms, computed according to the inner product (retrieval model denoted "doc=bnn, query=bnn" or "bnn-bnn"). For document and query indexing however binary logical restrictions are often too limiting. In order to weight the presence of each indexing term in a document surrogate (or in a query), we sometimes took account of the term occurrence frequency, thus allowing for better term distinction and increasing indexing flexibility (model denoted "doc=nnn, query=nnn" or "nnn-nnn").

Those terms however that did occur very frequently in the collection were very helpful in distinguishing between relevant and non-relevant items. Thus we could count their frequency in the collection, or more precisely the inverse document frequency (denoted by idf$_j$), resulting in more weight for sparse words and less weight for more frequent ones. This idf value is usually computed as ln(n/df$_j$) where n indicates the number of documents in the collection. Moreover, a cosine normalization could prove beneficial and each indexing weight could vary within the range of 0 to 1 (retrieval model notation: "ntc-ntc"). Table A.1 in the Appendix depicts the exact weighting formulation.

Other variants were also created, especially when considering the occurrence of a given term in a document as a rare event. Thus, it could be good practice to assign more importance to the first occurrence of this word as compared to any successive or repeating occurrences. Therefore, the tf component could be computed as 0.5 + 0.5 · [tf / max tf in a document] (retrieval model denoted "doc=atn"). Moreover, we should consider that a term's presence in a shorter document provides stronger evidence than it does in a longer document. To account for this, we integrated document length within the weighting formula, leading to more complex IR models; for example, the IR model denoted by "doc=Lnu" (Buckley *et al.*, 1996), "doc=dtu" (Singhal *et al.*, 1999).

In addition to previous models based on the vector-space approach, we also considered probabilistic models, an example being the Okapi probabilistic model (Robertson *et al.*, 2000). As a second probabilistic approach, we implemented the Prosit (PRObabilistic Sift of Information Terms) approach (Amati & van Rijsbergen, 2002; Amati *et al.*, 2003), based on the following indexing formula:

$$w_{ij} = \mathrm{Inf}1_{ij} \cdot \mathrm{Inf}2_{ij} = (1 - \mathrm{Prob}1_{ij}) \cdot \mathrm{Inf}2_{ij} \quad \text{with}$$
$$\mathrm{Prob}1_{ij} = tfn_{ij} / (tfn_{ij} + 1)$$
$$tfn_{ij} = tf_{ij} \cdot \log_2[1 + ((C \cdot \text{mean dl}) / l_j)]$$

$$\mathrm{Inf}2_{ij} = -\log_2[1 / (1+l_j)] - tfn_{ij} \cdot \log_2[l_j / (1+l_j)]$$
$$\text{with } l_j = tc_j / n$$

in which w_{ij} reflects the importance of each single-term t_j in a document D_i, tf_j the frequency of occurrence of term t_j in a document D_i, tc_j indicates the number of occurrences of term t_j in the collection, n the number of documents, and C and mean dl are constants. In this model, the query terms are weighted according to a term occurrence frequency (denoted "nnn").

1.3. Pseudo-Relevance Feedback

It was observed that pseudo-relevance feedback (or blind-query expansion) seemed to be a useful technique for enhancing retrieval effectiveness. In our evaluations, we adopted Rocchio's approach (Rocchio, 1971), (Buckley *et al.*, 1996) in which the newly expanded query Q' was composed as follows:

$$Q' = \alpha \cdot Q + \beta \cdot \frac{1}{r} \cdot \sum_{j=1}^{r} w_{ij}$$

within which Q denoted the previous request, α= 0.75, β = 0.75 and the system was allowed to add s terms extracted from the original query's r best ranked documents.

1.4. Data Fusion

Until now, we used a single search model (or engine) when searching document collections. We might however suggest sending the request to various

	Mean average precision			
	Lovins' stemmer		S stemmer	
IR model	Default list	Separator+	Default list	Separator+
Okapi-npn	15.10	15.39	15.53	15.41
OkaR-npn	14.74	15.13	15.39	15.45
Prosit	15.07	14.60	15.31	14.91
dtu-dtn	**16.80**	**17.44**	**18.25**	**18.67**
atn-ntc	16.47	16.30	16.71	16.86
Lnu-ltc	16.19	16.28	16.62	16.41
ltn-ntc	16.32	16.39	16.24	16.00
ltc-ltc	15.64	15.95	16.62	16.43
lnc-ltc	14.86	14.00	16.00	14.64
ntc-ntc	14.48	12.78	15.04	13.30
bnn-bnn	7.02	7.48	7.31	8.54
nnn-nnn	3.90	3.54	3.48	3.84

Table 3. Mean average precision for various IR models using the Genomics corpus (50 queries)

search engines that would handle the same document collection but that use different indexing or search schemes. Once we have obtained result lists from these various search engines, we would need to merge them in an effective manner (data fusion). Thus, even though certain degrees of retrieval effectiveness may be attributed to each search approach, combining the result lists might provides better average precision. If we were to use RSV_k to denote the retrieval status value (or document score) for a given document retrieved by the *kth* search engine, Fox & Shaw (1994) suggested using various operators (see Table 1) and showed that the best performance could be achieved using "combSUM".

combMAX	MAX (RSV_k)
combMIN	MIN (RSV_k)
combSUM	SUM (RSV_k)
combANZ	SUM (RSV_k) / # of nonzero (RSV_k)
combNBZ	SUM (RSV_k) * (# of nonzero (RSV_k))
combRSV%	SUM (RSV_k / maxRSV)
combRSVn	SUM[(RSV_k-minRSV)/(maxRSV-minRSV)]

Table 1. Data fusion strategies

Of course we might also employ the round-robin merging strategy, taking the first retrieved item from the first result list, then the first retrieved document from the second list, etc., and finally the first item from the last result list and then back again to the first result list. Duplicates encountered in this process are simply ignored.

1.4. Evaluation

To evaluate various IR models using the genomic collection, we used 50 queries and various statistics on relevance assessments. As shown in Table 2, the number of pertinent items per request is relatively small (mean = 11.32). The mean average precision for nine vector-space schemes together with the Okapi and Prosit probabilistic models are depicted in Table 3. This table shows the results of evaluating two stemmers and two word delimiting strategies.

Number of queries	50
Number of relevant doc.	566
Mean rel. doc. / request	11.32
Standard deviation	13.15
Median	7
Maximum	66 (Query #32)
Minimum	2 (e.g., Query #4)

Table 2. Relevance judgment statistics (Genomics)

An examination of Table 3 shows that the best retrieval effectiveness was obtained when using the vector-space model "dtu-dtn," while second best results were usually obtained using the "atn-ntc" scheme. Ranking third was the "Lnu-ltc" model (using the S stemmer), or the "ltn-ntc" model (using Lovins' stemmer). In these experiments, the simple "tf-idf" approach (denoted "ntc-ntc") did not appear to perform very well. To our surprise, we noted that the Okapi or Prosit probabilistic model did not perform very well in this task, contrarily to our previous experiments (for example those based on the Amaryllis corpus, also composed of bibliographic records (Savoy, 2003)).

From Table 3 we might also infer that the extended word delimiter (labeled "Separator+") usually enhanced performance only slightly. Moreover, the light S stemmer resulted in better retrieval effectiveness than did Lovins' algorithm.

	Mean average precision							
	Lovins' stemmer				S stemmer			
	Default list		Separator+		Default list		Separator+	
IR model	Prosit	dtu-dtn	Prosit	dtu-dtn	Prosit	dtu-dtn	Prosit	dtu-dtn
#doc/#term	15.07	16.80	14.60	17.44	15.31	18.25	14.91	**18.67**
3 / 10	15.60	16.77	15.55	17.53	16.25	18.57	16.43	16.27
3 / 15	15.62	16.45	15.63	17.50	16.26	18.40	16.44	15.88
3 / 20	15.58	16.59	15.80	17.42	15.93	18.47	16.55	16.03
3 / 30	16.10	16.51	15.41	17.46	16.41	18.36	16.92	15.97
5 / 10	16.14	16.75	16.42	17.55	16.79	18.57	16.78	16.27
5 / 15	16.43	16.78	15.65	17.60	16.56	18.61	16.46	16.47
5 / 20	16.30	16.74	15.69	17.62	16.03	18.57	16.87	16.14
5 / 30	15.50	16.76	15.31	17.61	15.87	18.58	16.29	16.05
10 / 10	**17.17**	16.74	17.20	17.67	**17.90**	18.50	17.37	15.93
10 / 15	16.53	16.75	16.93	17.67	16.97	18.57	**17.53**	16.05
10 / 20	15.96	16.76	**17.38**	17.67	16.79	**18.62**	17.29	16.20
10 / 30	15.97	**16.79**	16.37	**17.68**	16.10	18.42	16.81	16.35

Table 4. Mean average precision for various relevance feedback parameter settings (Genomics corpus, 50 queries)

	Mean average precision							
	Lovins' stemmer				S stemmer			
	Default list		Separator+		Default list		Separator+	
IR model		+ Q expand		+ Q expand		+ Q expand		+ Q expand
Prosit	15.07	17.17	14.60	17.38	15.31	17.90	14.91	17.53
dtu-dtn	16.80	16.79	**17.44**	17.68	18.25	18.62	**18.67**	16.35
combMAX	15.07	17.17	14.60	17.38	15.31	17.90	14.91	17.53
combMIN	2.48	1.46	5.02	1.26	3.06	1.34	5.16	2.05
combSUM	15.51	17.08	15.47	17.43	15.72	16.89	16.11	17.52
combANZ	12.16	14.00	12.33	13.39	13.79	15.76	14.15	15.27
combNBZ	15.51	17.07	15.48	17.42	15.72	17.90	16.11	17.39
combRSV%	17.43	17.52	16.52	17.63	18.38	18.72	17.82	16.97
combRSVn	**17.54**	17.51	16.78	17.72	**18.48**	**18.74**	17.83	17.07
round-robin	15.66	**17.53**	16.05	**18.21**	16.67	18.59	16.97	**17.59**

Table 5. Mean average precision of various data fusion approaches (Genomics corpus, 50 queries)

From the data depicted in Table 4, we can conclude that pseudo-relevance feedback usually increases mean average precision. When taking the r=10 best ranked documents into account, performance is usually enhanced compared to r=3 or r=5. This improvement is however rather small, particularly for the "dtu-dtn" vector-space model. On the other hand from previous experiments with the Prosit model, there is evidence that blind query expansion usually improves mean average precision significantly (Savoy, 2003). Our current test-collection seems to confirm this. Finally, we evaluated various data fusion strategies that might be employed to improve retrieval effectiveness. In our case we submitted the same request to two search engines (Prosit and "dtu-dtn") with and without blind query expansion (using the best parameter setting). Based on the data shown in Table 5, it appears that data fusion based on combRSVn or a simple round-robin scheme performs better. Moreover, various data fusion strategies (combMIN, combMAX, combANZ, and combNBZ) degraded the system's overall performance.

Prosit	17.90 (S)	17.38 (Lov+)	17.38 (Lov+)
dtu-dtn	18.67 (S+)	18.62 (S)	18.67 (S+)
combMAX	18.49	17.38	16.98
combMIN	15.97	1.21	15.00
combSUM	19.17	17.51	18.69
combANZ	18.73	13.01	18.38
combNBZ	19.09	17.49	18.63
combRSV%	**19.45**	18.81	**19.18**
combRSVn	19.44	**18.91**	19.15
round-robin	19.09	18.81	17.16

Table 6. Evaluation of various data fusion strategies

Table 6 shows the results of combining the Prosit and "dtu-dtn" search models, using both stemmers (denoted "Lov" or "S") and separator characters lists (our separator list "Separator+" is denoted "+"). From this data, we can conclude that it seemed better to combine retrieval schemes based on a variety of indexing strategies (e.g., using the different separator lists shown in the second column, or different stemming algorithms as depicted in the third and forth columns). Finally, Table 7 lists the specifications for our official runs.

Run name	MAP	Description
UniNEg1	18.52	Okapi+dtu-dtn, def., combRSVn
UniNEg2	18.02	Okapi+dtu-dtn, def., combRSV%
UniNEg4	16.23	dtu-dtn, Lovins, default list
UniNEg5	16.35	Lnu-ltc, S-stem, separator+

Table 7. Description of our official runs (all with blind query expansion)

2. Genomic Information Extraction

The main purpose of the genomic secondary task was to address the bioinformatic community's information extraction needs. More precisely, the goal was to reproduce the GeneRIF (Gene Reference into Function used in the LocusLink[1] database), either from a Medline record or from the entire article. GeneRIF snippets sometimes contain direct quotations from article abstracts but they might also include or paraphrase certain texts extracted from article titles or abstracts.

The data used for this task consisted of 139 GeneRIFs, representing all articles appearing in five journals (*Journal of Biological Chemistry*, *Journal of Cell Biology*, *Science*, *Nucleic Acids Research* and *Proceedings of the National Academy of Sciences*), during the latter half of 2002.

2.1. Models

From the beginning, we decided to use only the article titles and abstracts for this task. As the title was supposed to be a good candidate for the GeneRIF annotation, we tried selecting it systematically and using it as a baseline performance measure for our task.

Then for each GeneRIF we tried selecting each GeneRIF term also contained in the corresponding abstract. This method provided us with a theoretical maximum that could be reached, using only articles titles and abstracts.

2.1.1. Dummy (UniNEie5)

First of all, we established the term frequencies for the words contained in the GeneRIFs. Then, we ranked them and selected in descending order, those terms having frequencies greater or equal to 9. The words selected by using this simple strategy were:

cell role protein expression gene receptor activation regulate human apoptosis alpha sp1 signaling domain regulation kinase suggest pathway

We then supplied this fixed sequence of words as the GeneRIF for each query.

2.1.2. Random (UniNEie4)

For each query, we segmented the corresponding abstract into sentences. Then we considered all sentences, including the title. Each sentence having 10 to 14 words was repeated once into our set of candidates. Each sentence within this set thus had an equal probability of being selected. Finally we randomly chose a sentence that was returned as the GeneRIF.

2.1.3. Logarithm of Term Frequency (UniNEie3)

As the GeneRIFs were provided, we computed the term frequencies for all words contained therein. Then, for each query, we segmented the corresponding abstract into sentences. For each sentence, including the title, we removed the stopwords and then stemmed the remaining words, using the SMART stopword list (571 entries) and the S stemmer (see Section 1.1). We then computed a sentence score as follows:

$$\text{score} = \sum_{j=1}^{\text{len}} \ln(tf_j) \Big/ \text{len}$$

where j is the term index, tf_j the term frequency in GeneRIFs and len the length of the sentence without stopwords. Finally, we returned the sentence having the highest score as the GeneRIF.

2.1.4. Term Frequency and Logistic Regression

We again used the above process except that the score was computed as follows:

$$\text{score} = \sum_{j=1}^{\text{len}} w(tf_j) \Big/ \text{len}$$

where j is the term index, tf_j the term frequency in GeneRIFs, and len is the length of the sentence without stopwords. Table 8 lists the resultant $w(tf_j)$ values.

We then selected the sentence having the highest score as a GeneRIF candidate and applied a logistic regression model (Hosmer & Lemeshow, 2000), using the statistical Fisher method to predict when the system should return the chosen sentence or the title. We tried

[1] Available at www.ncbi.nlm.nih.gov/LocusLink/

two variants, corresponding to different sets of variables.

tf_j	$w(tf_j)$
$9 < tf_j$	4
$4 < tf_j \leq 9$	3
$2 < tf_{ij} \leq 4$	2
$1 < tf_{ij} \leq 2$	1
$tf_{ij} \leq 1$	0

Table 8. Terms weights

Model A (UniNEie2)

The following example provides an explanation of how our model works. Looking at Query #30, we must chose between the title and the candidate shown in Table 9. Table 10 lists the sentence results after removing stopwords and applying the stemming procedure.

Title	Comparative surface accessibility of a pore-lining threonine residue (T6') in the glycine and GABA(A) receptors.
Candidate	This action was not induced by oxidizing agents in either receptor.

Table 9. Competing sentences for Query #30

Title	Comparative surface accessibility pore-lining threonine residue (T6') glycine GABA(A) receptor
Candidate	action induced oxidizing agent either receptor

Table 10. Competing sentences (stemmed, without stopwords)

Variable	Meaning	Candidate	Title	Diff
Len	length	6	10	-4
Abrv	#acronyms	0	1	-1
Terms	#terms	5	10	-5
Max2Idf	2nd max idf	3.44	9.01	-5.58
MinIdf	min idf	2.25	2.35	-0.11
Min2Idf	2nd min idf	2.65	2.65	0.0

Table 11. Variables used for the regression

For each candidate, we could compute certain statistics, such as its length ("Len"), the number of acronyms ("Abrv"), the number of indexing terms ("Terms"), etc. as shown in Table 11. Since however we knew the title can usually be viewed as a suitable GeneRIF, we also computed certain statistics concerning the difference between a given candidate and the article title, as shown in Table 12. These values were then uscd as predictors in our logistic regression model to compute the probability that the corresponding candidate would be a suitable GeneRIF. The last column in Table 12 lists the estimated value of these corresponding statistics. For example, the estimate for the variable "d.Len" is negative, indicating that when the candidate length is greater than the title length, this fact decreases the probability that this candidate would be a suitable GeneRIF.

Variable	Meaning	Estimate
Intercept		-3.9502
d.Len	length candidate – title	-3.3939
d.Abrv	# acronyms candidate - title	2.5182
d.Terms	# terms candidate - title	2.5645
d.Max2Idf	2nd max idf candidate - title	1.1829
d.MinIdf	min idf candidate - title	6.1737
d.Min2Idf	2nd min idf candidate - title	-5.1732

Table 12. First set of variables and estimates

Using the result of the logistic regression, we returned the complete title 126 times and the candidate 13 times, 7 of them forming a part of the title.

Model B (UniNEie1)

As a variant of the previous model, we changed the set of explanatory variables, as depicted in Table 13.

Using the logistic regression results, we returned the complete title 129 times and our candidate 10 times, 6 of them forming a part of the title.

2.2. Evaluation

The Dice coefficient, measuring the degree of overlap of two sentences was used for evaluation purposes. Given two sentences A and B, we defined |A| as the number of words in A, |B| as the number of words in B, and |A∩B| as the number of words occurring in both A and B. The Dice coefficient was measured by:

$$\text{Dice (A, B)} = \left(2 * |A \cap B|\right) \Big/ \left(|A| + |B|\right)$$

Four variants of this measure were used for evaluation (more details can be found at the Web site[1])

- Dice 1 is the classical Dice
- Dice 2 is the modified unigram Dice
- Dice 3 is the bigram Dice
- Dice 4 is the bigram Phrases

[1] See medir.ohsu.edu/~genomics/protocol.html

Variable	Meaning	Estimate
Intercept		68.199
Terms	# indexing terms in the candidate	-19.867
Min2Idf	2nd max idf candidate	-36.733
nb.Art	# common terms in candidate and abstract	18.999
d.Len	length candidate –title	-57.029
d.Abrv	# acronyms candidate - title	17.141
d.Terms	# indexing terms candidate - title	46.910
d.Max2Idf	2nd max idf candidate - title	30.926
d.MinIdf	min idf candidate - title	22.121

Table 13. Second set of variables and estimates

Table 14 shows an evaluation of our runs. The second row forms our baseline, representing the article title, a scheme within which the title is always returned. On the other hand, the third row ("Generifs ∩ abst.") represents the maximum value that could be achieved when selecting the most appropriate sentence, using only the article title and abstract.

	Dice 1	Dice 2	Dice 3	Dice 4
Title (min)	50.47%	52.60%	34.82%	37.91%
Generifs ∩ abst. (max)	59.53%	83.26%	61.66%	52.76%
UniNEie5	9.42%	14.20%	0.15%	0.17%
UniNEie4	25.88%	25.29%	12.03%	13.61%
UniNEie3	49.46%	51.42%	33.62%	36.99%
UniNEie2	51.72%	54.27%	36.62%	39.71%
UniNEie1	**52.28%**	**54.78%**	**37.43%**	**40.35%**

Table 14. Evaluation of our official runs

Using this data, we hoped to improve our extraction of the suitable GeneRIFs from the title scheme, through using one of our logistic regression models. In this case, it seemed that Model B (or UniNEie1) performed slightly better, even though both models returned the title many times. We attempted to improve our system's performance through incorporating additional data, such as full text articles or gene names, together with the selection of the explanatory variable set for the logistic regression.

3. Our Okapi Search Model

Based on our previous work (Savoy & Picard, 2001; (Savoy & Rasolofo, 2003), the Okapi search model provided significantly greater retrieval effectiveness. However, in order to manage the Web collection (1,247,753 documents that were extracted from the .GOV domain, or about 18.1 GB of data), we needed to modify this search model for two reasons. Firstly, we wanted to incorporate three document representatives for each Web page, and secondly we needed to distribute the inverted file in order to respect the 2 GB limit.

When processing three document representations, we estimated the degree of similarity between document D_i and the current query would be a linear combination of the inner product of the three document representations, to be given as:

$$\begin{aligned} RSV(D_i) &= \alpha \cdot \sum_{j=1}^{m} w_{ij}^{(1)} \cdot qw_j \\ &+ \beta \cdot \sum_{j=1}^{m} w_{ij}^{(2)} \cdot qw_j + \gamma \cdot \sum_{j=1}^{m} w_{ij}^{(3)} \cdot qw_j \end{aligned} \qquad (1)$$

where $w^{(1)}_{ij}$ indicates the weight attached to the term t_j in the document D_i in the first document representation ($w^{(2)}_{ij}$ and $w^{(3)}_{ij}$ for the second, respectively, the third document surrogate), and the parameters α, β, γ are used to assign a comparative importance to each document representative.

Creating a single inverted file from a collection of approximately 18 GB might be impossible using a 32-bit system (e.g., Linux). To overcome this limit, we will follow the approach described in Rasolofo & Savoy (2003), whereby each sub-collection would be indexed using the tf component. When merging the result lists obtained from searching into these different sub-collections, we computed the idf component and applied the normalization. Following this step, we could then merge the result lists according to the new document scores.

Knowing that both Web tasks required high precision searches, we decided to enhance our Okapi model by implementing the term proximity scoring function (see Rasolofo & Savoy (2003) for more details). The main premise was that if all search keywords appear in a document representative, our search model would increase the corresponding document score. On the other hand, if only a part of these search terms appeared in a given Web page, the final retrieval status value would remain unchanged (see Eq. 1). While the term proximity function would have a greater value if the search keywords appear close to each other, they may occur more than once within a given sentence or tag. In our system the constant δ denoted the impact of these proximity scores. Of course, setting $\delta = 0$ means that the proximity score is not computed.

4. Named and Home Pages Finding

The following considerations formed the basis of our first Web task. When submitting a request to a search engine, users will sometimes not want a ranked list of Web pages concerning a particular topic, but rather they would prefer the location of an underlying service or known-item (usually presented within a short list of the most probable locations). For example, the appropriate response to a query on "state department", "Secret Service jobs", "Navajo Nation", or "barbara mikulski bio" (and even with a spelling error such as "US Volcano Oservatories") would not be a ranked list of documents covering these subjects but rather those site(s) that contain the required form/information/list. To accomplish this we needed to implement an IR system that could retrieve a limited number of pages (one at the very least) in response to the user's request.

4.1. Search Models

As a basis for our search model we used the Okapi model as described in Section 3. Our first document representative was based on information found in the Web page, including the corresponding <TITLE> and <META> tags ("keywords" and "description"). Of course Web pages might also contain links and their associated anchor texts (or anchor texts for outgoing links). Our second document surrogate was based on the <TITLE> tag and the anchor texts for those Web pages pointing to the current document. The third document representative was built by concatenating the <TITLE>, <H1>, and <BIG> tags from pages pointing to the current Web page. This third aspect was used to reinforce the importance of those Web pages pointing to the current page. Since we know that end tags (e.g., </H1> or </BIG>) are sometimes missing, we only considered the first 64 words following any given tag.

This indexing strategy was based on previous studies (Craswell *et al.*, 2001), (Westerveld *et al.*, 2002), (Kraaij *et al.*, 2002), showing that anchor texts from other Web pages pointing to the current page may provide compact and often accurate descriptions of the current page's content. For this reason, we extracted link anchor texts from all Web pages pointing to the current page and concatenated them to form our second document representative. Finally, we also considered URL content (or more precisely, the similarity between the URL text and the current request, or URL lengths). In our current search models, these additional sources of information had not taken into account.

When high precision results are required for indexing documents or requests, it is usually not a good idea to include a stemming procedure. We could however adopt a light stemming such as the "S-stemmer" (see Section 1.1 (Harman, 1991)). In this case, the words "house" and "houses" would be reduced to the same root while the term "housing" would be treated as a different indexing unit. Based on our experiments from last year (Savoy & Rasolofo, 2003), we decided to ignore the stemming approach for this task due to the fact that even light stemming was usually found to diminish the system's overall performance (Savoy & Rasolofo, 2003).

4.2. Evaluation

In this IR search model, based on three document representatives and a proximity scoring function, we first needed to determine the relative importance assigned to each document representative (based on internal Web page content for the first surrogate), as compared to the weight attached to the second and third document representatives (based mainly on link anchor texts from those Web pages pointing to the current one). The relative importance for each surrogate was controlled through using the parameters α, β, γ (see Section 3) while the proximity score was weighted using the constant δ.

Number of queries	300
Number of relevant doc.	352
Mean rel. doc. / request	1.173
Standard deviation	0.609
Median	1
Maximum	6 (Query #244)
Minimum	1

Table 15. Relevance judgment statistics (named and home page searches, TREC-2003)

Our evaluation was based mainly on the mean reciprocal rank (MRR) of the first correct answer found by the system. Table 15 depicts statistics on the relevance assessments of this test-collection, clearly showing that we usually obtain one correct answer per topic. For each of the 300 queries, we considered only the first 100 retrieved items. As seen in Table 16, the best value for our parameters seems to be around α=0.6, β=0.4, γ=0.05, and δ=0.1, thus assigning a little more weight to internal representation (parameter α) than to the anchor texts of all Web pages pointing to the current document (parameter β). The third representation does not seem to have a great impact on system's performance. The underlined parameters in this table represent the settings used for our official runs.

Finally, Table 17 provides a summary description of our four official runs. Usually, we did not attach much importance (γ = 0 or very small) to the third document representative (<TITLE>, <H1>, and <BIG> texts from pages pointing to the current Web page). The difference between UniNEnp1 and UniNEnp3 represented the inclusion of the term proximity scoring function within UniNEnp3, seemingly a useful technique for improving retrieval effectiveness. Taking account for this third surrogate enhanced the system's performance (see Table 16). The performance differences between UniNEnp2, UniNEnp5 were due to the various parameter settings used for the Okapi model.

Parameters	MRR	# in top 10
Okapi b=0.5		
α=0.6, β=0.4, γ=0, δ=0	0.666	252 (84.0%)
α=0.6, β=0.4, γ=0, δ=0.1	0.691	251 (83.7%)
α=0.6, β=0.4, γ=0, δ=0.2	0.692	251 (83.7%)
α=0.6, β=0.4, γ=0.05, δ=0	0.707	258 (86.0%)
α=0.6, β=0.4, γ=0.05, δ=0.1	**0.720**	259 (86.3%)
α=0.7, β=0.3, γ=0.05, δ=0.1	0.700	258 (86.0%)
α=0.8, β=0.2, γ=0.05, δ=0.1	0.676	252 (84.0%)
α=0.8, β=0.2, γ=0.05, δ=0.2	0.682	254 (84.7%)
Okapi b=0.6		
α=0.6, β=0.4, γ=0, δ=0	0.667	250 (83.3%)
α=0.6, β=0.4, γ=0, δ=0.1	0.690	250 (83.3%)
α=0.6, β=0.4, γ=0, δ=0.2	0.689	250 (83.3%)
α=0.6, β=0.4, γ=0.05, δ=0	0.700	258 (86.0%)
α=0.6, β=0.4, γ=0.05, δ=0.1	**0.713**	259 (86.3%)
α=0.7, β=0.3, γ=0, δ=0	0.626	247 (82.3%)
α=0.7, β=0.3, γ=0, δ=0.1	0.658	251 (83.7%)
α=0.7, β=0.3, γ=0.05, δ=0.1	0.699	257 (85.7%)
α=0.8, β=0.2, γ=0.05, δ=0.1	0.686	254 (84.7%)
Okapi b=0.7		
α=0.6, β=0.4, γ=0, δ=0	0.654	246 (82.0%)
α=0.6, β=0.4, γ=0, δ=0.1	0.677	247 (82.3%)
α=0.6, β=0.4, γ=0, δ=0.2	0.676	248 (82.7%)
α=0.6, β=0.4, γ=0.05, δ=0	0.691	256 (85.3%)
α=0.6, β=0.4, γ=0.05, δ=0.1	0.701	256 (85.3%)
α=0.6, β=0.4, γ=0.05, δ=0.2	**0.706**	257 (85.7%)
α=0.7, β=0.3, γ=0.05, δ=0.1	0.688	254 (84.7%)
α=0.8, β=0.2, γ=0.05, δ=0.1	0.683	253 (84.3%)

Table 16. IR model evaluation for various combinations of our three document representatives

5. Topic Distillation

The basic purpose of the topic distillation task was to return a list of key resources on a given topic (e.g., "pest control safety", "computer viruses" or "children's literature"). Explicitly defining what does or does not constitute a suitable resource was however difficult, and each definition seemed to become more and more ambiguous. While Web pages with appropriate content might be considered as useful key resources and we could have retrieved them using a classic IR model, key resources may also be good hubs (or Web pages pointing to various pages containing pertinent content with respect to the submitted request). Moreover, if a Web page is linked to two, three or more children having a high degree of similarity with the request, it seems more appropriate to return this parent page rather than the two, three of more children. More generally however returning many pages extracted from the same Web site would not be viewed as a wise strategy. Thus to suggest a proper solution for this specific task, we decided to employ various strategies that would point to reliable browsing starting points rather than simply retrieving Web pages with suitable content.

Run name	MRR	Parameter settings
UniNEnp1	0.626	Okapi b=0.6 α=0.7, β=0.3, γ=0.0, δ=0.0
UniNEnp2	0.676	Okapi b=0.5 α=0.8, β=0.2, γ=0.05, δ=0.1
UniNEnp3	0.658	Okapi b=0.6 α=0.7, β=0.3, γ=0.0, δ=0.1
UniNEnp4	**0.688**	Okapi b=0.7 α=0.7, β=0.3, γ=0.05, δ=0.1
UniNEnp5	0.686	Okapi b=0.6 α=0.8, β=0.2, γ=0.05, δ=0.1

Table 17. Description of official named-page & homepage runs

5.1. Search Models

As for the named page and homepage search task, we built three document representatives for each Web page contained in the .GOV collection. The first representative accounted for Web page content along with its <TITLE> and <META> tags ("keywords" and "description") and the anchor texts contained in the page. The second document surrogate was built from the text delimited by the <TITLE> tag together with link anchor texts from all outgoing and all incoming links. The third document representative was composed of all <TITLE> and <H1> tags provided by all pointed pages (or pages accessible within a one-click distance from the current page).

Once the pages were retrieved, we followed hyperlinks coming into them in order to define proper starting points for browsing (in this case we followed existing hyperlinks in the reverse direction). To retrieve these starting points we used our spreading activation (SA) search scheme (Crestani & Lee, 2000), (Savoy & Picard, 2001). Using this method, document scores initially computed by the IR system (denoted $RSV(D_i)$) were propagated to the linked documents through a certain number of cycles, based on a propagation factor. We used a simplified version with only one cycle and a fixed propagation factor λ for all links. As a result, the final retrieval status value for a document D_i linked to k documents was computed using the following equation:

$$RSV'(D_i) = RSV(D_i) + \lambda \cdot \sum_{j=1}^{k} RSV(D_j) \qquad (3)$$

When in our experiments we tried to extract the proper starting sites for browsing, we only considered all incoming links for each of the k best-ranked documents.

As other possibilities, we might consider the PageRank algorithm (Brin & Page, 1998), the HITS algorithm (Kleinberg, 1998) or probabilistic argumentation systems (Picard, 1998). During the evaluation campaign of last year however, we did obtain poor performance when employing the HITS algorithm (Savoy & Rasolofo, 2003).

5.2. Evaluation

In order to evaluate the performance of a topic distillation IR scheme, we could use the precision achieved after retrieving 5 or 10 documents (under the labels "Prec@5" or "Prec@10") together with the number of relevant items retrieved (out of a total of 516 for the 50 queries included in the .GOV collection). Each request would be composed of a short title and a descriptive part.

Number of queries	50
Number of relevant doc.	516
Mean rel. doc. / request	10.32
Standard deviation	13.38
Median	8
Maximum	86 (Query: #32)
Minimum	1 (Query: #13)
Number of distinct roots / query	
Mean	8.38
Standard deviation	11.641
Median	6
Maximum	77 (Query: #32)
Minimum	1 (Query: #13)
URL length 1	79
length 2	93
length 3	171
length 4	108
length 5	44
length 6	13
length 7 and more	8

Table 18. Relevance judgment statistics (topic distillation searching task, TREC-2003)

Table 18 shows various statistics based on relevance assessments. The mean number of relevant items (or key resources) per request is 10.32. From considering the number of distinct roots (e.g., the first part of an URL, e.g., "trec.nist.gov"), we found that in mean, there were 8.38 different roots per query (for Query# 13, the unique relevant item is coming from the Web site "nimh.nih.gov"). On the other hand, for Query# 48, we found 9 relevant pages (over a total of 10) extracted from the root page "prime.jsc.nasa.gov".

In our first set of experiments, we evaluated our extended Okapi IR model (see Section 3). By varying the value attached to the parameters α, β, γ, we assigned more or less weight to each document representation. For example, when we set α to 0, and β to 0, we accounted for text delimited by the <TITLE> and <H1> tags provided by all pointed pages. In other words, we viewed the page as a good starting point for browsing (limited however to a one-click distance). On the other hand, with α = 1, β = 0, and γ = 0, our search model was based only on Web page content.

Run name	Prec@5	Prec@10
α=1, β=0, γ=0, δ=0	8.00	6.20
α=1, β=0, γ=0.03, δ=0.1	11.60	7.60
α=1, β=0, γ=0.03, δ=0.3	12.40	8.00
α=0, β=0, γ=1, δ=0	7.20	4.60
α=0, β=0, γ=1, δ=0.1	8.00	4.60
α=0.5, β=0.5, γ=0, δ=0	16.40	10.80
α=0.5, β=0.5, γ=0, δ=0.1	16.00	11.00
α=0.5, β=0.5, γ=0.03, δ=0	16.40	10.80
α=0.5, β=0.5, γ=0.03,δ=0.1	16.00	11.00
α=0.5, β=0.5, γ=0.1, δ=0	15.60	11.40
α=0.5, β=0.5, γ=0.1, δ=0.1	16.00	**11.60**
α=0.7, β=0.3, γ=0, δ=0	15.20	10.20
α=0.7, β=0.3, γ=0, δ=0.1	16.00	10.20
α=0.7, β=0.3, γ=0.1, δ=0	14.00	11.40
α=0.7, β=0.3, γ=0.1, δ=0.1	14.00	11.40
α=0.8, β=0.2, γ=0, δ=0	14.40	9.80
α=0.8, β=0.2, γ=0, δ=0.1	14.80	9.40
α=0.8, β=0.2, γ=0, δ=0.3	15.20	9.00
α=0.8, β=0.2, γ=0.03,δ=0.3	15.20	9.00
α=0.8, β=0.2, γ=0.1, δ=0.3	14.00	10.60

Table 19. Evaluation of various document surrogates combinations

Table 19 displays the various results produced by our IR model (without stemming) when varying the relative importance of each document representative. From this data, the best parameter values seemed to be: α = 0.5, β = 0.5, and γ = 0. The third document representative does not seem to improve retrieval effectiveness. As such, our second representation (<TITLE> tags & anchor link texts from all pointed and pointing pages) seemed to be more valuable for this specific IR task. Usually, the term proximity scoring function seems to improve the ranking of pertinent items (e.g., precision after 5). The underlined parameters in this table represent the settings used for our official runs.

Table 20 provides a summary description of our five official runs, all of which were created without a stemming procedure. For both UniNEdi2 and UniNEdi5, we only accounted for a single document representative (content-oriented only, based on the good performance of such indexing schemes last year). For UniNEdi3 and UniNEdi4, we accounted for two document surrogates. Our best run was UniNEdi1, which accounted for three document representatives. For UniNEdi4, after retrieving content-based Web pages using our extended Okapi model, we applied a spreading activation with λ = 0.02, for the first k = 50 top-ranked items.

Run name	Prec@10	description
UniNEtd1	**9.80**	α=0.8, β=0.2, γ=0.03, δ=0
UniNEtd2	7.60	α=1, β=0, γ=0, δ=0
UniNEtd3	7.60	α=1, β=0, γ=0.03, δ=0.1
UniNEtd4	8.80	α=1, β=0, γ=0.03, δ=0.3 & SA, k=50, λ=0.02
UniNEtd5	9.60	α=1, β=0, γ=0, δ=0 & data fusion

Table 20. Description of our official topic distillation runs

Parameters	Prec@5	Prec@10
α=0.5, β=0.5, γ=0.1, δ=0.1	16.00	11.60
λ = 0.02, k = 50	17.60	12.00
λ = 0.05, k = 50	**17.60**	12.20
λ = 0.1, k = 50	13.60	12.00
λ = 0.05, k = 100	18.80	12.80
λ = 0.05, k = 200	19.20	12.40
λ = 0.05, k = 300	16.40	**14.00**
λ = 0.05, k = 400	16.00	13.80
α=1, β=0, γ=0, δ=0	8.00	6.20
λ = 0.02, k = 50	10.00	7.20
λ = 0.05, k = 50	10.40	7.80
λ = 0.1, k = 50	10.00	7.60
λ = 0.05, k = 100	12.80	8.60
λ = 0.05, k = 200	12.80	9.20
λ = 0.05, k = 300	12.80	9.80
λ = 0.05, k = 400	13.20	10.20

Table 21. Evaluation of various parameter settings for the spreading activation approach

When evaluating our spreading activation (SA) method, we only take account for hyperlinks in reverse orientation. In this case, a λ fraction of the score attached to the children is propagated to the parent page (see Eq. 3). From data depicted in Table 21, it seems that the propagation factor λ must be around 0.02, and the SA must be limited to the first k = 300 or first k = 400 best-ranked items.

When using the best-run shown in Table 19, we tried various parameter settings as depicted in top part of Table 21. In this case, we may enhance the precision after 10 documents from 11.8% to 14.0% (leading to +20% improvement). On the other hand, when the starting point is based only on the Web page content (as depicted in the bottom part of Table 21, with α= 1, β = 0, γ = 0, δ = 0), our SA may also improve the precision at 10 retrieved item from 6.2% to 10.2% (+64% improvement).

Acknowledgments

This research was supported by the SNSF (grant 21-66'742.01).

References

Amati, G. & van Rijsbergen, C.J. (2002). Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM TOIS*, 20(4), 357-389.

Amati, G., Carpineto, C. & Romano, G. (2003). Italian monolingual information retrieval with PROSIT. In Cross-Language Information Retrieval and Evaluation. Springer, Berlin.

Brin, S. & Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. Proceedings WWW8, 107-117.

Buckley, C., Singhal, A., Mitra, M. & Salton, G. (1996). New retrieval approaches using SMART. TREC-4, 25-48.

Craswell, N., Hawking, D. & Robertson, S. (2001). Effective site finding using link anchor information. ACM-SIGIR'2001, 250-257.

Crestani, F. & Lee, P.L. (2000). Searching the Web by constrained spreading activation. *IPM*, 36(4), 585-605.

Fox, E.A. & Shaw, J.A. (1994). Combination of multiple searches. TREC-2, 243-249.

Harman, D. (1991). How effective is suffixing? *JASIS*, 42(1), 7-15.

Hosmer, D.W. & Lemeshow, S. (2000). Applied Logistic Regression. 2nd edn., John Wiley.

Kleinberg, J. (1998). Authoritative sources in a hyperlinked environment. SIAM Symposium on Discrete Algorithms, 668-677.

Kraaij, W., Westerveld, T. & Hiemstra, D. (2002). The importance of prior probabilities for entry page search. ACM-SIGIR'2002, 27-34.

Lovins, J.B. (1968). Development of a stemming algorithm. *Mechanical Translation and Computational Linguistics*, 11(1), 22-31.

Picard, J. (1998). Modeling and combining evidence provided by document relationships using PAS systems. ACM-SIGIR'1998, 182-189.

Rasolofo, Y. & Savoy, J. (2003). Term proximity scoring for keyword-based retrieval systems. ECIR-03, Springer, Berlin, 207-218.

Robertson, S.E., Walker, S. & Beaulieu, M. (2000). Experimentation as a way of life: Okapi at TREC. *IPM*, 36(1), 95-108.

Rocchio, J.J. Jr. (1971). Relevance Feedback in Information Retrieval. In *The SMART Retrieval System - Experiments in Automatic Document Processing*, G. Salton Ed., Prentice-Hall, 313-323.

Savoy, J. & Picard, J. (2001). Retrieval effectiveness on the Web. *IPM*, 37(4), 543-569.

Savoy, J. & Rasolofo, Y. (2003). Report on the TREC-11 Experiment: Arabic, Named Page and Topic Distillation Searches. TREC-11, 765-774.

Savoy, J. & Rasolofo, Y. (2001). Report on the TREC-9 experiment: Link-based retrieval and distributed collections. TREC-9, 579-588.

Savoy, J. (2003). Report on CLEF-2002 Experiments: Combining Multiple Sources of Evidence. In Cross-Language Information Retrieval and Evaluation. Springer, Berlin, to appear.

Singhal, A., Choi, J., Hindle, D., Lewis, D.D., & Pereira, F. (1999). AT&T at TREC-7. TREC-7, 239-251.

Westerveld, T., Kraaij, W. & Hiemstra, D. (2002). Retrieving Web pages using content, links, URLs and anchors. TREC-10, 663-672.

Appendix 1. Weighting schemes

To assign an indexing weight w_{ij} reflecting the importance of each single-term t_j in a document D_i, the formula shown in Table A.1 may be used, where document length (the number of indexing terms) for document D_i is denoted by nt_i, and n indicates the number of documents in the collection. For the Okapi weighting scheme, K represents the ratio between the length of document D_i measured by l_i (sum of tf_{ij}) and the collection's mean is noted by avdl or more precisely

$$K = k_1 \cdot \left[\left(1-b\right) + b \cdot \frac{l_i}{avdl}\right]$$

For the Genomic corpus, the constant avdl was fixed at 300, b at 0.55, k_1 at 1.2, C at 3, mean dl at 73, pivot at 50 and slope at 0.05. For both Web searching tasks, we set avdl at 900, b at 0.75, k_1 at 1.2, pivot at 125 and the constant slope at 0.1.

bnn	$w_{ij} = 1$	nnn	$w_{ij} = tf_{ij}$
ltn	$w_{ij} = (\ln(tf_{ij}) + 1) \cdot idf_j$	atn	$w_{ij} = idf_j \cdot [0.5 + 0.5 \cdot tf_{ij} / \max tf_{i.}]$
lnc	$w_{ij} = \frac{\ln(tf_{ij}) + 1}{\sqrt{\sum_{k=1}^{t} \left(\left(\ln(tf_{ik}) + 1\right)\right)^2}}$	npn	$w_{ij} = tf_{ij} \cdot \ln\left[\left(n - df_j\right) / df_j\right]$
Okapi	$w_{ij} = \left((k_1 + 1) \cdot tf_{ij}\right) / \left(K + tf_{ij}\right)$	dtn	$w_{ij} = \left(1 + \ln\left(1 + \ln(tf_{ij})\right)\right) \cdot idf_j$
ntc	$w_{ij} = \frac{tf_{ij} \cdot idf_j}{\sqrt{\sum_{k=1}^{t} \left(tf_{ik} \cdot idf_k\right)^2}}$	dtu	$w_{ij} = \frac{\left(1 + \ln\left(1 + \ln(tf_{ij})\right)\right) \cdot idf_j}{(1 - slope) \cdot pivot + slope \cdot nt_i}$
ltc	$w_{ij} = \frac{\left(\ln(tf_{ij}) + 1\right) \cdot idf_j}{\sqrt{\sum_{k=1}^{t} \left(\left(\ln(tf_{ik}) + 1\right) \cdot idf_k\right)^2}}$	Lnu	$w_{ij} = \frac{\left(\frac{\ln(tf_{ij}) + 1}{\ln\left(l_i / nt_i\right) + 1}\right)}{(1 - slope) \cdot pivot + slope \cdot nt_i}$

Table A.1: Weighting schemes

UB at TREC-12: HARD and Genomics Tracks

Munirathnam Srikanth
Computer Science and Engineering
SUNY at Buffalo
Buffalo, NY, 14228
srikanth@cedar.buffalo.edu

Miguel E. Ruiz
Library and Information Studies
SUNY at Buffalo
Buffalo, NY, 14228
meruiz@buffalo.edu

Rohini Srihari
Computer Science and Engineering
SUNY at Buffalo
Buffalo, NY, 14228
rohini@cedar.buffalo.edu

1. INTRODUCTION

University at Buffalo (UB) participated in TREC-12 in Genomics and High Accuracy Retrieval from Documents (HARD) tracks. We explored some techniques that combine Information Retrieval and Information Extraction to perform the TREC tasks. We used an Information Extraction engine - InfoXtract [3] from Cymfony Inc.[1] - to enhance retrieval results.

For the Genomics primary task, documents retrieved using a vector space model with relevance feedback are reweighted based on the biomedical named entities discovered by InfoXtract. For the secondary task, extracted information along with cue words for text snippets that describe functionality is used for generating GeneRIFs for given Gene name and PubMed abstract. A language modeling approach that incorporates keyword and non-keyword features are used for the HARD task. Features extracted by InfoXtract from the HARD corpus are used to rank documents and/or passages as answers to the HARD queries.

Cymfony's InfoXtract [3] is a customizable Information Extraction engine that performs syntactic and semantic parsing of a document to identify features like named entities, relationships and events in them. The baseline InfoXtract engine has been trained for the general English and news domain, It can be customized to recognize new named entities like Gene Names and Gene Function. Biomedical Customization of InfoXtract is briefly presented in Section 2.2.

2. GENOMICS TRACK

UB participated in both primary and secondary task of TREC 12 Genomics track. Our efforts concentrated on trying to apply Information extraction to improve retrieval performance in task 1 and to combine support vector machines and information extraction for selecting sentences as GeneRIFs annotations.

2.1 Relevance Feedback Model for Genomics Retrieval

We used the SMART system as a baseline as the search engine for the Genomics track. Documents are represented using title (TI), abstract (AB), MeSH terms (MH) and EC/RN Numbers (RN). We tried several models using separate *ctypes* for each of the previously mentioned fields and measure the contribution of each part to retrieval performance. We also experimented with a version that used all the information in a single *ctype*. Table 1 shows the contribution of each part to the retrieval performance on the training topics. It is interesting to note that these runs show that the MeSH terms have a very small contribution in the retrieval results of the training topics (0.0326 Avg-Prec). We believe that this is due to two factors: a) low coverage of domain specific Genomics concepts in MeSH, and b) we did not attempt to do a mapping of topics against the MeSH vocabulary (only single word matching is used for this particular run). We were also surprised to see that the contribution of EC/RN numbers does contribute significantly to retrieval performance. The combination of all fields into a single *ctype* outperforms all runs that use a single field. We also tried two different stemming algorithms since we were not sure whether a simple stemming algorithm that takes care of plurals only would work better for our experiments (as reported by Jacques Savoy in his preliminary experiments on this collection). Additionally we used a heuristic method to try to capture phrases and proper nouns. For this purpose we preprocessed the documents to identify fragments delimited by punctuation symbols and extract bigrams (groups of two consecutive words) that don't include stopwords. The heuristic process takes into account exceptions that allow a limited number of stop words to be part of the bigram term ("of", and "for"), i.e. "alignment of proteins". The best results in the baseline runs were obtained using this heuristic method (0.3200 Avg-Prec).

Topics were processed by extracting the information available in each field and then representing it in the corresponding *ctype*. For runs that use bigrams we added the corresponding bigrams and phrases to each training topic using the previously described heuristic. We also tried several weighting schemes (*atn, atc, ann,* and *Lnu* for documents and *ntc, ltc, lnc, atc, atn, ann, anc,* and *ltu* for queries).

We also performed pseudo-relevance feedback using the top n documents as relevant and selecting the top m terms to expand each query. The results of our experiments on the training set are presented in Table 2. Pseudo Relevance feedback combined with bigrams is the one that yields the best performance in the training topics (0.3702 Avg-Prec with a 16% improvement over the corresponding baseline).

2.2 Information Extraction in BioMedical Domain

The InfoXtract engine is customized for the BioMedical domain before using it to process the Genomics document collection. Domain knowledge is essential for an Information Extraction engine to tag documents with named entities in the domain of interest. We used part of UMLS hierarchy

[1] www.cymfony.com

	weights	P@10	Avg. Prec.	R-Prec.
TI	*atn.ntc*	0.1040	0.1961	0.1757
AB	*atn.atc*	0.1380	0.2604	0.2115
MH	*atn.lnc*	0.0180	0.0326	0.0183
RN	*atn.ntc*	0.0620	0.1777	0.1615
TI+AB+ MH+RN (one ctype) Lovins' Stemmer	*atn.ltu*	0.1320	0.3028	0.2683
TI+AB+ MH+RN (one ctype) Rem-s Stemmer	*atn.ltu*	0.1340	0.3044	0.2706
TI+AB+ MH+RN+ Bigrams (one ctype) Rem-s Stemmer	*atn.ltc*	0.1320	0.3200	0.2743

Table 1: Baseline runs on Training Topics

	weights	P@10	Avg. Prec.	R-Prec.
TI+AB+ MH+RN ($\alpha = 8; \beta = 64;$ $\gamma = 16$) ($n = 3; m = 10$) Lovins' Stemmer	*atn.ltc*	0.1420	0.3316	0.3044
TI+AB+ MH+RN ($\alpha = 32; \beta = 16;$ $\gamma = 8$) ($n = 3; m = 5$) Rem-s Stemmer	*atn.ltc*	0.1380	0.3300	0.2907
TI+AB+ MH+RN+ Bigrams ($\alpha = 32; \beta = 16;$ $\gamma = 8$) ($n = 3; m = 5$) Rem-s Stemmer	*atn.ltc*	0.1380	0.3702	0.3291

Table 2: Pseudo-Relevance Feedback runs on Training Collection

as domain knowledge for InfoXtract. We restricted our customization effort to identifying domain-specific terminology through lexicons.

We selected several subtrees from the UMLS that are most related to the Genomics sub domain. For this purpose we selected the following concepts:

- C1136351: Genetic Phenomena.
- C1136308: Genetic Processes
- C1136352: Genetic Structures
- C0017398: Science of Genetics
- C0002526: Amino acids, Peptides and Proteins
- C0019934: Hormones, Hormone Substitutes and Hormone Antagonists
- C0018285: Growth Substances
- C0014443: Enzymes, Co-enzymes and Enzymes Inhibitors

We use the parent–child relationship in the UMLS metathesaurus to select all terms related (as a narrower concept) for each of these general concepts. This produced a set of 21,070 concepts and 51,571 unique terms (after normalization). We also add the related terms to the species of interest in this task (Homo sapiens, Mus musculus, Rattus norvegicus, and Drosophila melanogaster) for 31 extra terms.

In addition, InfoXtract is customized to identify and tag named entities of type *Gene Name* and *Gene Functionality*. Gene names and their synonyms are collected from LocusLink and assigned a unique name (typically the preferred product name). Automatic candidate selection followed by manual truthing is adopted for generating the Gene Functionality lexicon. The InfoXtract engine is customized to detect Gene Names. Some training documents are processed by InfoXtract and term statistics is used to generate candidate functionality terms. Some heuristics like ignoring terms that appear in a named entity like Gene Name are used to filter out terms and construct a candidate set for Gene Functionality. Researchers in Biology were asked to manually go through the functionality term list and the context of their usage in the training documents subset to identify Gene functionality terms.

Genomics document collection is processed by the customized InfoXtract engine. It extracts the 12 different named entities mentioned above.

2.3 Using IE in Genomics IR

Natural Language Processing (NLP) techniques have been used in document retrieval to select index terms. Prior use of Information Extraction output has been restricted to narrow search problems like question answering. We used Information Extraction as a filter to improve or re-rank the retrieval results. Documents are processed by InfoXtract customized for the biomedical domain. The output of InfoXtract for a document, in addition tagged biomedical named-entities, includes part-of-speech, shallow parsing results and relations between named entities. A subset of the extracted information along with the terms in a document are indexed using the TAPIR toolkit. TAPIR toolkit is a library of software tools that facilitate a number of IR tasks and supports different IR models including language models. The position of the index term in a document is also used as the position of tags representing the extracted information. This index is used to re-rank the document retrieval results corresponding to the run *UBgenomRFB1*.

Given a query, each document deemed relevant in the run *UBgenomRFB1* to the query is weighted for the co-occurrence of query terms with named entities in the biomedical domain. The co-occurrence frequency of tags with query terms is used to reweight documents. Ad-hoc weights are assigned to different named entity tags with the highest weight given to co-occurrence with gene function words, terms related to genetic process and other gene name. Reweighted documents set is normalized and re-ranked to generate the result named *UBgenomBGNE*.

2.4 GeneRIF Extraction

As noted in the guidelines for this task Mork and Aronson have found that 95% of the GeneRIF snippets contain text from the title and abstract of the articles. For this reason we decided to concentrate our approach on selecting sentences from the title and abstract as a first approach. For GeneRIF extraction we propose a solution that uses text categorization to select the sentence from the MEDLINE document (Title and Abstract) that is the best candidate to be a GeneRIF descriptor.

Documents are processed using InfoXtract to detect sentences, as well as important information such as the name of the gene of interest, or description of the functionality related to the gene of interest.

Our baseline system for this task is a trivial procedure that assigns the title as the candidate for GeneRIF description. We also tried to find what would be the upper bound of performance for a method based on sentence selection. For this purpose we found the sentence that would give the highest Dice coefficient value for each GeneRIF.

The approach that we use for this task uses a support vector machine (SVM-light) to learn the sentences that should be selected as GeneRIF using as input features the vector representation of the document using the smoothed unigram language model. We also included other features such as the position of then sentence in the document, whether the gene of interest is mentioned in the sentence, and whether biomedical terms (that were extracted by InfoXtract) appear in the sentence. For training the SVM we collected 5496 GeneRIFs annotations from LocusLink and gather the respective MEDLINE documents (making sure that these GeneRIFs where not in the test set for the secondary task). This set was randomly divided into 3, 676 documents for training and 1, 820 documents for validation.We also determined the sentence that had the best Dice score in the document to be the "correct GeneRIFs" and mark it as a positive example while the rest of the sentences in the document were marked as negative examples. This process gave us a total of 19, 658 sentences in the training set and 9, 947 sentences in the validation set. The sentence with the highest SVM classification score was selected as the GeneRIF for each document.

	Baseline	Upper bound	Features only	Feature+ cues+ position
Classic Dice	57.02	76.18	45.54	56.94
Mod unigram Dice	57.70	76.51	45.77	57.68
Mod bigram Dice	42.79	67.59	30.87	43.23
Mod bigram Dice phrases	46.00	69.93	34.01	46.52

Table 3: Results in the Validation Set.

Table 3 shows the results of our experiments on the validation set of 1, 820 GeneRIFs. The upper-bound indicates that the best we can performance of a method that selects full sentences from the MEDLINE article would be at 76.18. This corroborates that most of the GeneRIFs come from "cut and paste" text from the Title and abstract. Our baseline system that selects the title as the GeneRIF annotation performs at 57.02% which indicates that the simplest algorithm for selecting the GeneRIFs annotations is obtaining about 73% of the Upper-bound performance. This is not surprising since a significant number of GeneRIFs are just the title of the article. The first attempt to use SVM only used the features extracted from the unigram language model and performs significantly below our baseline (45.54% Classical Dice and about 20% below the baseline). when we added the cues (gene name, and functionality, words, etc) and the relative position of the sentence in the document the SVM was able to achieve a performance that is about the same as the baseline (56.94% Classical Dice). We were disappointed to realize that after all the training process and information extraction our system wasn't better than our simple baseline.

2.5 Results and Analysis

Our official results in task 1 are presented in Table 4. *UBgenomRFB1* uses pseudo-relevance feedback, *atn.ltc* weighting scheme, and the top 3 retrieved documents from which we get the top 5 terms for query expansion, and $\alpha = 32.\beta = 16.\gamma = 8$ as the parameter in Rocchio's formula. *UBgenomRFB2* uses pseudo-relevance feedback, *atn.ltu* weighting scheme, and the top 3 retrieved documents from which we get the top 5 terms for query expansion, and $\alpha = 8, \beta = 64, \gamma = 16$ as the parameter in Rocchio's formula. *UBgenomeBGNE* is the re-ranked output of *UBgenomRFB1* using the filtering process explained previously. In general our results are slightly below the average performance. We suspect that this could be a consequence of the way topics were converted into queries in the system but we still need to do a more detailed analysis.

	P@10	Avg. Prec.	R-Prec.
UBgenomRFB1	0.1160	0.1511	0.1232
UBgenomRFB2	0.1120	0.1493	0.1141
UBgenomeBGNE	0.1440	0.1867	0.1603

Table 4: Pseudo-Relevance Feedback runs on Training Collection

	Best	≥Median	<Median	Worst
UBgenomRFB1	1	15	34	6
UBgenomRFB2	1	14	35	7
UBgenomeBGNE	1	18	31	0

Table 5: Relative performance of Official Task 1 Genomics runs

Results for the secondary task in Genomics are presented in Table 6. As suspected from our results in the training and validation set the SVM based approach did not perform significantly better than our base line system. All our runs are fairly close to the median system performance (49.31% Classic dice coefficient) in this task.

	Baseline	SVM 1	SVM 2
Classic Dice	49.28	49.03	49.40
Mod unigram Dice	51.25	51.16	51.30
Mod bigram Dice	33.59	33.94	33.59
Mod bigram Dice phrases	36.99	37.35	36.99

Table 6: Officail results for Secondary Task.

3. HARD TRACK

In the HARD track user queries are qualified by metadata that provide additional information on the user's information need such as purpose, genre and granularity of query. Of the three steps in the track - baseline retrieval, clarification forms and final run to use all the information available about the user query. The second step is optional and we did not generate any clarification forms for the queries. We used the metadata provided along with the query in the final run. Language Modeling approach [1] is adopted for both the baseline and final HARD tasks.

HARD document corpus is processed by Cymfony's InfoXtract engine which tags part-of-speech, named entities and associated profiles to the entities and events discovered in the document. A subset of information extracted by InfoXtract is used in our HARD solution. However, the text and all extracted information from a document are indexed by TAPIR - an IR toolkit that supports different IR models including Language Modeling approach to IR. A tag of a term in a document is indexed as though it is embedded in the document at the position(offset) corresponding to the term. This method of indexing has been explored for question answering in earlier TREC.

3.1 Baseline System

We submitted three different runs for baseline. Our baseline solution is based on the textual part of the user query - title and description - and perform document retrieval. No effort is made to predict any metadata for the user queries as well as no query expansion or relevance feedback is experimented with. A brief description of the three runs are given below:

- *ub03sugT* Run based on smoothed unigram language model that uses Dirichlet smoothing. The Dirichlet parameter is set to 1000. This run used only the title of the query.

- *ub03cugTD* This run is based on the Concept Unigram Language Models (CULM) [4] that have been shown to perform better than smoothed unigram and bigram language models. In Concept Language Model a query is viewed as a sequence of concepts and concepts, in turn, are viewed as a sequence of terms. Consecutive terms typically constitute concepts that can be single terms, bigrams or n-grams. In CULM, concept independence is assumed and query probability is computed as a unigram model on concepts. Concept probability is approximated to smoothed bigram probabilities. The InfoXtract question parser [2] and its shallow parsing results are used to identify concepts of interest in the query.

- *ub03ugTcugTD* This run corresponds to a linear combination of the above two methods. It provided slight improvement in mean average precision on the training set.

The results of *ub03ugTcugTD* are used for the final run (*ub03smfugTD*) as the document retrieval system. Its results are further refined to satisfy the metadata values of the user query.

3.2 Passage Retrieval

Except for the queries with granularity value of document, passage retrieval is performed for other queries. Relevant passages are selected based on the query keywords. The granularity of the query determines the length of the passage. Relevant sentences (i.e. passages of length 1) are identified for queries with granularity of sentence or phrase. The granularity of passage resulted in the system selecting text snippets with 3 to 6 sentences. The passages are not overlapped. The coverage of query terms is used as a criteria to determine the passage length. All candidate passages thus selected are considered for final ranking.

3.3 MetaData Modeling and HARD

We modeled Purpose and Genre metadata for the HARD task. For a given query, text snippets are short-listed based on their satisfying the user's query and granularity requirements. A number of keyword- and non-keyword based features are used to weight the snippets and rank them. A text snippet is viewed as a sequence of keyword and non-keyword features and a model is associated with it in the language modeling sense. Text snippets are weighted based on the probability of generating a given feature. Smoothed probabilities are estimated for keyword features and empirical probabilities are used for non-keyword features. Ad-hoc query weights are assigned for these features based on the metadata values.

Genre metadata is handled as follows: *Reaction* or *I-Reaction* typically involve entities that we group as *actors*. This includes persons, organizations, Government entities. The occurrence of such entities triggers Reaction or I-Reaction genre type. If the origin of such an actor is a US location, the text snippet is most likely to be a Reaction than I-Reaction. InfoXtract engine tags associations or relations that can link entities of one type with another. To determine if the text snippet is Reaction or I-Reaction, entities associated with *actors* are searched to see if they include location entities. Location names are checked against a lexicon of US cities, counties and states to determine if the origin of the *actor*. A number of US cities share names with non-US locations - e.g. *Moscow* and *China*. In such instances, the context of the location name is checked. If the next word/phrase is a US location or the phrase "United States" or its variations, the location is classified as a US location. Document-source based features are used to weight text snippets against genre value of *Administration*. We did not eliminate all non-government sourced documents but assigned lower weights than government sourced documents.

The location of the text snippet in a document, presence or absence of *details* (described below) information and the query term coverage were used as triggers for the different values of purpose metadata. We assumed that a text snippet provides details on a particular topic of interest if it contains numeric, percentage, frequency and time information. Such information, extracted by InfoXtract, are grouped together as *details* information and snippets are weighted based on the frequency of such information. The absence of such information is also used as a feature. The snippet location feature gives more weights to snippets closer to the center of the document. It is based on the heuristic that details and background information are typically found in the middle rather than at the start or end of the document. Queries with metadata background or details assign more weights to this feature than queries with metadata or answer or any.

With the absence of significant training data for the different metadata values, we used ad-hoc query weighting. Expectation Maximization or maximum entropy models can be used, in the presence of training data, to weight these features against the metadata values in a query.

3.4 Results and Analysis

Table 7 presents the performance of the different runs submitted for the HARD track. These performance are judged for soft document relevance. Documents that satisfy the query and not necessarily the metadata requirements are identified to be soft relevant to the query. The Best, Worst and Median values for these measures are also given. Concept unigram language model performs better than the rest of our submissions. The use of syntactic information in query modeling provides around 13% improvement over smoothed unigram language model.

	Rel. Ret. @10	Avg. Prec.	R-Prec.
ub03sugT	5.375	0.3126	0.3335
ub03cugTD	5.854	0.3540	0.3784
ub03ugTcugTD	5.729	0.3495	0.3663
ub03smfugTD	3.583	0.2073	0.2562
Best	6.5	0.4069	0.4250
Median	4.729	0.2841	0.2994
Worst	0.417	0.0026	0.0038

Table 7: Soft Document Relevance Comparison

Hard document relevance corresponds to documents that are relevant to the query as well as satisfy the metadata requirements of the query. The Hard-relevance performance comparison is given in Table 8.

	Rel. Ret. @10	Avg. Prec.	R-Prec.
ub03sugT	4.042	0.2543	0.2764
ub03cugTD	4.542	0.2981	0.3286
ub03ugTcugTD	4.479	0.2896	0.3075
ub03smfugTD	2.771	0.1726	0.2078
Best	5.271	0.3875	0.3604
Median	3.792	0.2673	0.2490
Worst	0.312	0.0038	0.0024

Table 8: Hard Document Relevance Comparison

The *ub03ugTcugTD* run on training topics performed better than the other two baseline runs. While the combination of language models did improve the average precision values for 28 topics over the CULM, the average performance over all topics of CULM is better than the combination model. The performance reduction is significant - as much as 300% - for some topics. The *ub03smfugTD* run that used the metadata to re-rank the results of *ub03ugTcugTD* did not results in any improvements at document level performance measures. All text snippets from a document were selected and ranked for HARD retrieval. The ad-hoc weighting of the features and no cutoff on the number of snippets selected from a relevant document are possible reasons for the decrease in performance.

	Psg. Prec. @10	R-Prec.	F @ 30
ub03sugT	0.2399	0.1594	0.1178
ub03cugTD	0.2987	0.1997	0.1528
ub03ugTcugTD	0.2763	0.1825	0.1368
ub03smfugTD	0.2313	0.1699	0.0798
Best	0.3973	0.3195	0.1738
Median	0.2574	0.1794	0.1000
Worst	0.0136	0.0046	0.0010

Table 9: Passage Relevance Comparison

Table 9 gives the performance measures for passage relevance. The first three runs were evaluated with the whole document being the retrieved passage. *Ub03smfugTD* included passage level results corresponding to the granularity metadata.

4. CONCLUSION

This section discusses future directions of our work for the two tracks we participated in. Better query representation, the use of all extracted information and incorporating more domain knowledge in document and query processing are some of the avenues of improvement for the Genomics track. For the HARD track, we have only used a subset of information extracted by the InfoXtract engine to represent queries and model metadata. Ad-hoc weights were assigned to extracted features in ranking documents for a given query. This was partly due to the absence of sufficient training data. We plan to explore some formal methods for modeling metadata - specifically identifying and weighting features that satisfy the metadata requirements of queries.

REFERENCES

[1] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of SIGIR'98*, pages 275–281. ACM, New York, 1998.

[2] Rohini Srihari and Wei Li. A Question Answering System Supported by Information Extraction. In *Proceedings of ANLP'00*, pages 166–172, 2000.

[3] Rohini K Srihari, W Li, C Niu, and T Cornell. Infoxtract: A customizable intermediate level information extraction engine. In *Proceedings of the NAACL 2003 Workshop on Software Engineering and Architecture of Language Technology Systems (SEALTS)*, Edmonton, Canada, 2003.

[4] M. Srikanth and R. Srihari. Exploiting Syntactic Structure of Queries in a Language Modeling Approach to IR. In *to appear in Proceedings of CIKM'03*, 2003.

Report on the TREC 2003 Experiment: Genomic Track

Patrick Ruch[1ab], Christine Chichester[c], Gilles Cohen[a], Giovanni Coray[b]
Frédéric Ehrler[ad], Hatem Ghorbel[b], Henning Müller[a], Vincenzo Pallotta[b]

[a]*SIM, University Hospital of Geneva, Geneva*
[b]*Theoretical Computing Laboratory, Swiss Federal Institute of Technology, Lausanne*
[c]*GeneBio SA, Geneva*
[d]*AI Laboratory, University of Geneva, Geneva*

Summary

For our first participation to the TREC evaluation campaign, our efforts concentrated on the genomic track. Because we joined the competition at the end of June, we were not able to submit runs for the ad hoc retrieval task (task I), and therefore this report mostly focuses on the information extraction task (task II).

Task I. Our approach uses thesaural resources (from the UMLS) together with a variant of the Porter stemmer for string normalization. Gene and Protein Entities (GPE) of the collection (525,938 MedLine citations) were simply marked up by dictionary look up during the indexing in order to avoid erroneous conflation: strings not found in the UMLS Specialist lexicon (augmented with various English lexical resources) were considered as GPE and were moderately overweighted. In the same spirit like other TREC competitors [23] for task I, an overweighting factor was also applied to features belonging to Medical Subject Headings (MeSH) and found in MedLine citations using a MeSH mapping tool [1]. A standard vector space IR engine with tf-idf parameters was used for indexing the Genomic collection: article's titles, MeSH and RN terms, and abstact fields were selected. Best average precisions were obtained with atc.ntn (using the SMART notation) schemes: 16.71 (standard) vs. 17.02 (using UMLS resources and GPE tagging). Studies made after the competition and inspired by results reported by other groups [13][14] confirmed that narrowing the search to those species appearing in the query provides a very effective improvement of the average precision. The species are detected based on their listing in a dictionnary (extracted from the MeSH terminology). The refinement strategy consists in filtering out documents when the targetted species is not found in the abstract. After retrieval, this simple strategy yield to an important improvement of the average precision: from 17.02 up to 35.80.

Task II. Our approach is based on argumentative structuring, i.e. classification of textual segments into argumentative classes. We see the task as a question-answering task using always the very same question. We take advantage of a classifier likely to predict the argumentative class of any textual segment with high accuracy (F-score = 85%). We observe that when not taken from the title, GeneRIFs are found -ranked by decreasing order- in: 1) CONCLUSION, 2) PURPOSE, 3) RESULTS, 4) METHODS. After sentence splitting, sentences are classified and ranked according to these four categories. On that basis, a second ranking is made based on the similarity with the title (45% of GeneRIFs; Dice baseline = 50.47%). Then, we compute a combined score for each of these features, setting a Dice-like threshold to decide whether we use the title or the best scored sentence as GeneRIF. Finally, a last step consists in narrowing segment boundaries to shorten the length of the candidate GeneRIF. A set of ad hoc and argumentative filters are applied in order to remove irrelevant pieces at the end/beginning of the selected segment. Examples of phrases that are removed at the beginning are "in this paper, finally...". Then, sentence endings (up to 7 words) classified as METHODS (such as "in contrast to current models", "by the...") are also removed. Using the complete article instead of the abstract did not result in any improvement. Our best performances are obtained by using 14 (Dice = 52.78%) and 23 (Dice = 52.41%) segments from the abstract, while the reamining originates from the title. The use of argumentative features is encouraging, however more complex features combination will have to be explored in the future.

Introduction

Systems for text mining are becoming increasingly important in biomedicine because of the exponential growth of knowledge. The mass of scientific literature needs to be filtered and categorized to provide for the most efficient use of the data. The problem of accessing this increasing volume

[1] Contact author. Email: patrick.ruch@epfl.ch

of data demands the development of systems that first, can retrieve pertinent information from unstructured texts and second, can help professional curators to annotate high-quality DBs in the biomedical domain (as in SwissProt with Gene Ontology annotations [7] [24] or in MedLine with MeSH annotations [2]). The former task as been largely addressed in previous TREC studies, at least from a general point of view, however it is the first time TREC investigates *ad hoc* retrieval in genomics. The second task of the TREC 2003 Genomic track aims at extracting the Gene Reference Into Function (GeneRIF), as provided in LocusLink, within a corpus of MedLine citations. For this last task, full-text articles are also available.

Input

Locus - ABCA1: ATP-binding cassette, sub-family A (ABC1), member 1

MedLine record - PMID - 12804586

TI - Dynamic regulation of alternative ATP-binding cassette transporter A1 transcripts.

AB – [...] The longest (class 1) transcripts were abundant in adult brain and fetal tissues. Class 2 transcripts predominated in most other tissues. The shortest (class 3) transcripts were present mainly in adult liver and lung. To study the biochemical significance of changes in transcript distribution, two cell models were compared. In primary human fibroblasts, upregulation of mRNA levels by oxysterols and retinoic acid increased the relative proportion of class 2 transcript compared to class 1. Phorbol ester stimulated human macrophage-derived THP-1 cells increased the abundance of class 1 transcripts relative to class 2. In both cell lines class 3 transcript levels were minimal and unchanged. It is shown here for the first time that the **regulation of ABCA1 mRNA levels exploits the use of alternative transcription start sites.**

Output

GeneRIF - **regulation of ABCA1 mRNA levels exploits the use of alternative transcription start sites**

Figure 1. Example of a record in LocusLink and the corresponding GeneRIF.

In order to evaluate our hypothesis, we uses the *easyIR* system (http://lithwww.epfl.ch/~ruch/softs/softs.html), which implements standard vector space IR schemes. The extraction of the gene function is seen as a sentence selection task [3][9][10][11][12] and is conducted using an argumentative classifier (called *LASt*, cf. the same link). Our experiments were conducted on an Intel Pentium IV/2.0, with 2GB of memory and 2 x 240 GB of (external USB-2) disks[2]. All experiments were fully automatic.

Background

In order to have an overall view of the underlying problems in generating the most appropriate GeneRIF during the last TREC genomic evaluation campaign [9], an example of a record in LocusLink is given in Figure 1. In the top part of this figure, we find the locus ("ABCA1") and the MedLine record identifier ("PMID – 12804586"). After the label "TI", we have reported the article's title and the abstract is given after the label "AB". In this case, we can see that the corresponding GeneRIF is extracted from the abstract. A typical GeneRIF extraction task is defined as follows: given a PMID (a PubMed reference to a MedLine citation), find the function of a given gene (Figure 1).

Preliminary studies [19] [20] showed that around 95% of the GeneRIF snippets are extracted from the title or from the abstract of the linked scientific paper. Moreover, from this set, around 42% were direct "cut and paste" from either the title or the abstract (Figure 1 is such an example) while another 25% contained significant portions of the title or abstract.

Latent Argumentative Structuring

In MedLine citations, abstracts are sometimes provided with explicit argumentative moves, such as "BACKGROUND", "AIM AND BACKGROUND", "PURPOSE", "METHODS", "RESULTS", "DISCUSSION", "CONCLUSION"... Unfortunately these explicit structural markers are neither stable, nor mandatory; therefore it is difficult to rely on such explicit markers. Although the labels that are used to express these moves are unstable, the hypothesis supporting this study is that conclusion sentences would be good candidates for identifying key/novelty facts in scientific texts; thus supporting gene functions in genomic corpora. Indeed, as stated in professional guidelines (ANSI/NISO Z39. 14-1979), articles in experimental sciences tend to respect strict argumentative patterns with at least 4 sections: PURPOSE-METHODS-RESULTS-CONCLUSION. Several studies confirm that at least the above 4 moves –leaving aside minor variation of labels– are reported to be very stable across different scientific genres (chemistry, anthropology, computer sciences, linguistics...) [4], and are confirmed in biomedical [5] [6].

[2] The indexing of the MedLine collection took more than 250 hours, and we encountered some problems with the first indexing, so that we had to run the process twice.

With position of sentences				
	PURP	METH	RESU	CONC
PURP	80.65%	0%	3.23%	**16%**
METH	10%	70%	10%	10%
RESU	18.58%	5.31%	23.89%	52.21%
CONC	**18.18%**	0%	2.27%	79.55%
Without position of sentences				
	PURP	METH	RESU	CONC
PURP	93.55%	0%	3.23%	**3%**
METH	30%	70%	0%	0%
RESU	27.43%	5.31%	23.01%	44.25%
CONC	**2.27%**	0%	2.27%	95.45%

Table 1. Confusion matrices for the argumentative classifier. The position is useful to separate between PURPOSES and CONCLUSION classes.

In table 1, we give the confusion matrix of the argumentative classifier. The F-score for the overall classification task is about 85%, but important variations are observed depending on the considered binary classification: if CONCLUSION and PURPOSES classes are well classified, the RESULTS class is mostly ill defined and cannot be accurately separated from CONCLUSION.

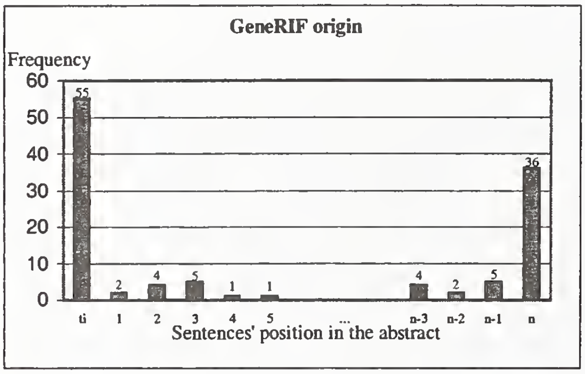


Figure 2. Distribution[3] of the GeneRIF position in the title (ti) and abstract.

We have also analyzed the distribution of sentences location used to produce the GeneRIF. In this case, we consider together the title (the first column labeled "ti" in Figure 2) and sentences from the abstract. From the 139 GeneRIFs used in our experiments, 55 are extracted from the title as depicted in Figure 2. The second most frequent source of GeneRIF is the last sentence of the abstract (last column of Figure 2 with the label "n") which provides 36 geneRIFs. Between these two extreme positions, the distribution of the GeneRIF location is rather flat. It is interesting to observe that the argumentative distribution (Table 3) does not fully match the positional ditribution in the abstract, but the two distributions tend to correlate (Table 3).

Genomics Information Extraction

A sentence splitting module was designed for the task in order to take into account specific and frequent usages of the dot character, such as in decimals and acronyms. In table 2, we separate between GeneRIFs found by using only the title of the abstract and other GeneRIFs.

Origin	Number	Query ID
Title	9	9, 10, 16, 74, 88, 123, 126, 133
Other	131	Other queries

Table 2: Distribution between GeneRIFs found in titles (with classic Dice = 100%, when using the title) and those found elsewhere.

To identify where human GeneRIFs originates from, regarding argumentative criteria, we apply the LASt system to analyse the distribution of 139 (i.e. the complete data set) and 131 (i.e. excluding queries, whose GeneRIF originates from the title) GeneRIFs into each of the argumentative classes: results are reported in table 3. A sample of the output data in given at the end of the report in Annex 1.

CONCLUSION	72 (51.8%)	68 (51.9%)
PURPOSE	59 (43.9%)	58 (44.3%)
RESULTS	3 (2.2%)	3 (2.3%)
METHODS	3 (2.2%)	2 (1.5%)
Total	139 (100%)	131 (100%)

Table 3. Distribution of argumentative classes among GeneRIFs.

These results are consistent with the confusion matrix given in Table 1: PURPOSES and CONCLUSION hard hardly separable regarding strict lexical information (confusion between 16% amd 18.18% in table 1), therefore positional information becomes important. This information is already combined in the LASt classification and time was too short to redesign the classifier. Unlike for argumentative classification, it is observed that confusion between RESULTS and CONCLUSION classes is not a major issue for GeneRIF extraction.

Combining argumentative classes with titles

Unfortunately, our first submitted run (run 0) computed by selecting the best CONCLUSION sentence as GeneRIF results in performances below the baseline, as shown in Table 4. The baseline, in Table 5, is calculated by simply selecting the title of the abstract and we can notice that more than half of the submitted runs were below this baseline.

[3] Courtesy of Jacques Savoy.

Classic Dice	35.20%
Modified unigram Dice	34.57%
Modified bigram Dice	20.04%
Modified bigram Dice phrases	21.58%

Table 4. Results when selecting GeneRIFs considering only sentences classified as CONCLUSION (Run 0).

Classic Dice[4] (Dice1)	50.47%
Modified unigram Dice (Dice2)	52.60%
Modified bigram Dice (Dice3)	34.82%
Modified bigram Dice phrases (Dice4)	37.91%

Table 5. Baseline measures: obtained by selecting the article's title as GeneRIF.

Because of these poor results, we attempt to combine both argumentative and title features together. In addition, we also try to shorten the candidate GeneRIF by removing a few words at the beginning or at the end of the candidate GeneRIF segment.

Features fusion

In this second approach, we decide to use the title as default GeneRIF. Then, we compute a Dice-like distance between candidates GeneRIF (which were classified as PURPOSE or CONCLUSION by the LASt system) and titles, as provided in MedLine citations. Again, in this approach, GPE tokens are overweighted in the Dice calculus.

Run	Sent.	Dice1	Dice2	Dice3	Dice4
1	14	52.78	54.33	37.72	40.65
2	23	52.41	54.22	37.61	40.44
3	31	51.06	52.84	35.43	38.70
Without using the string length of candidates GeneRIFs					
4	14	51.98	53.91	36.82	39.60

Table 6. Final results: combining features from the title and argumentative features for 3 different thresholds (run 1 to 3). We also observe that shortening strategies results in a modest improvement (run 1 vs. 4).

For GPE identification, we extracted a list of synonyms from LocusLink for each of the targeted Locus: thus, for the query 11 (Locus ID =7066), the list of synonyms (or related terms) is the following: *THPO, thrombopoietin (myeloproliferative leukemia virus oncogene ligand, megakaryocyte growth and development factor), TPO, MGDF, MKCSF, MPLLG.* We ran the system with different threshold. Varying this threshold results in changing the proportion of candidate GeneRIF extracted from the abstract vs. the article's title. In table 6, three of the most interesting results are reported. The top performing run, run 1, is the one we submitted. Although very empirical and so data-driven, these thresholds were found particularly stable, and calculating the threshold for run 1 on half of the data did not result in any degradation of performances for the information extraction task.

Reducing GeneRIFs' Length

The sentence compression step can be seen as a word removal process: it combines syntactic features (based on an hybrid part-of-speech tagger [8]), a set of ad hoc triggers (such as "In this paper…") and argumentative structuring. The basic syntactic removal attempts to remove non-content bearing clauses: phrases introducing relative clauses ("data suggest that", "these data indicate that", "in this report, we provide the first direct evidence that"…) and other introduction/adverbial clauses ("in addition", "surprisingly", "finally"…) are thus removed. Finally, clauses (from 3 to 7 words) expressing methods are also filtered out when found at the extremities of the candidate segment: for example, phrases such as "…using this method…" are removed. Serializing these compression strategies results in removing clauses as long as "These data indicate that, in contrast to current models…". It is to be noted that clauses containing GPE do not follow the length reduction process because such segments are potentially relevant: for instance, the segment "ARH is a modular adaptor protein that…" is not removed, because ARH is identified as a GPE.

Conclusion

In conclusion, our preliminary observations suggest that structural features (those stemming from the explicit structure of MedLine citations, such as the title, as well as those extracted from the latent structure, such as the argumentative structure) must be seen as a reasonable step in direction of automatic GeneRIF extraction. However, the task will require additional materials, as well as more powerful fusion's strategies, as explored in Savoy and Perret [15]. It is to be noted that for the secondary task, it was not clear whether other GeneRIF were to be used as training instances or not; however it is to be noted that apart from our experiments, other competitive approaches [15] [16] [17] were based on classifiers trained on GeneRIF. The use of the "function" axe in the Gene Ontology [7] together with using better gene and protein names recognition tools [21] [22] could also help identifying gene functions in MedLine abstracts.

Finally, we would like to remark that the chosen metrics were sufficient to compare the different approaches, but that more elaborated -and unfortunately more human-intensive-measures should be investigated in order to take into account the lexical variation of the biomedical language in general and of gene and protein names in particular [18].

Acknowledgments

The study reported in this paper has been partially supported by the Swiss National Foundation within the IM2 NCCR (http://www.im2.ch/) and by a separate SNF grant (MedTag, Grant 3200-065228.01). We used the CMU Statistical

[4] Best = 57.83; Median = 49.31, among all submitted runs and systems for the TREC genomic track (task II).

Language Modeling toolkit to compute the n-grams frequency of the argumentative classifier. We would like to thank Frédérique Lisacek, Arnaud Gaudinat, Johann Marty and Thomas de la Charrière for the comments they provided on this study.

References

[1] P Ruch, R Baud, and A Geissbühler. Learning-free Text Categorization, In *AIME* 2003, Springer LNCS/LNAI 2780, Dojat, M; Keravnou, E; Barahona, P (Eds.).

[2] 1a A Aronson, O Bodenreider, H Chang, S Humphrey, J Mork, S Nelson, T Rindflesch, W Wilbur, The NLM Indexing Initiative. Proc AMIA Symp. 2000. p. 17-21.

[3] H Edmundson, New methods in automatic abstracting. Journal of The Association for Computing Machinery, 16(2), pp. 264-285. Reprinted in Mani, I., and Maybury, M., eds., Advances in Automatic Text Summarization, MIT Press, p. 21-42. 1969.

[4] C Orasan, Patterns in scientific abstracts, in Proceedings of Corpus Linguistics 2001 Conference, Lancaster University, Lancaster, UK, pp. 433 - 443, 2001.

[5] J Swales, Genre Analysis: English in academic and research settings, Cambridge University Press, 1990.

[6] F Salanger-Meyer, Discoursal movements in medical English abstracts and their linguistic exponents: a genre analysis study, INTERFACE: Journal of Applied Linguistics 4(2), 1990, pp. 107 - 124

[7] F Ehrler and P Ruch. Report on the BioCreative Experiment: Task Presentation, System Description and Preliminary Results. In *Notebook of BioCreative 2004*, Granada, March 2004. (to appear)

[8] P Ruch, R Baud, P Bouillon, and G Robert. Minimal Commitment and Full Lexical Disambiguation: Balancing Rules and Hidden Markov Models. *CoNLL-2000 (ACL-SIGNLL) Proceedings*, p. 111-115, 2000.

[9] J Kupiec, J Pedersen and F Chen: A Trainable Document Summarizer, SIGIR 1995, p. 55-60.

[10] S Teufel and M Moens, Argumentative classification of extracted sentences as a first step towards flexible abstracting. In: I. Mani, M. Maybury (eds.), Advances in automatic text summarization, MIT Press, 1999.

[11] S Teufel and M Moens, Sentence Extraction and rhetorical classification for flexible abstracts, AAAI Spring Symposium on Intelligent Text summarization, 1998, p. 89-97.

[12] C Nedellec, M Ould Abdel Vetah, Ph Bessières: Sentence Filtering for Information Extraction in Genomics, a Classification Problem. PKDD 2001: 326-337

[13] B de Bruijn and J Martin, Finding Gene Function using LitMiner. In *Notebook of the TREC-2003*, p. 486-494.

[14] M Kayaalp, AR Aronson, SM Humphrey, N ide, L Tanabe, LH Smith, R Looane, JG Mork and O Bodenreider. Methods for Accurate Retrieval of MedLine Citations in Functional Genomics. In *Notebook of the TREC-2003*, p. 175-184.

[15] J Savoy, Y Rasolofo and L Perret. Report on the TREC 2003 Experiment: Genomics and Web Searches. In *Notebook of the TREC-2003*, Gaithersburg, p. 686-697.

[16] G Bhalotia, P Nakov, A Schwartz and M Hearst. Biotext Team Report for the TREC2003 Genomics Track. In *Notebook of the TREC-2003*, p. 158-166.

[17] R Jeller, M Schuemie, C vander Eijk, M Weeber, E van Mulligen, B Schijvenaars, B Mons and K Kors. Searching for GeneGIFs: Concept-based Query Expansion and Bayesian Cassification. In *Notebook of the TREC-2003*, 167-174.

[18] S Albert, S Gaudan, H Knigge, A Raetsch, A Delgado, B Huhse, H Kirsch, M Albers, D Rebholz-Schuhmann and M Koegl. Computer-assisted generation of a protein-interaction database for nuclear receptors. *Mol Endocrinol*. 2003 Aug;17(8):1555-67.

[19] W Hersh and RT Bhupatiraju. TREC genomics track overview. In *Notebook of the TREC-2003*, 148-157.

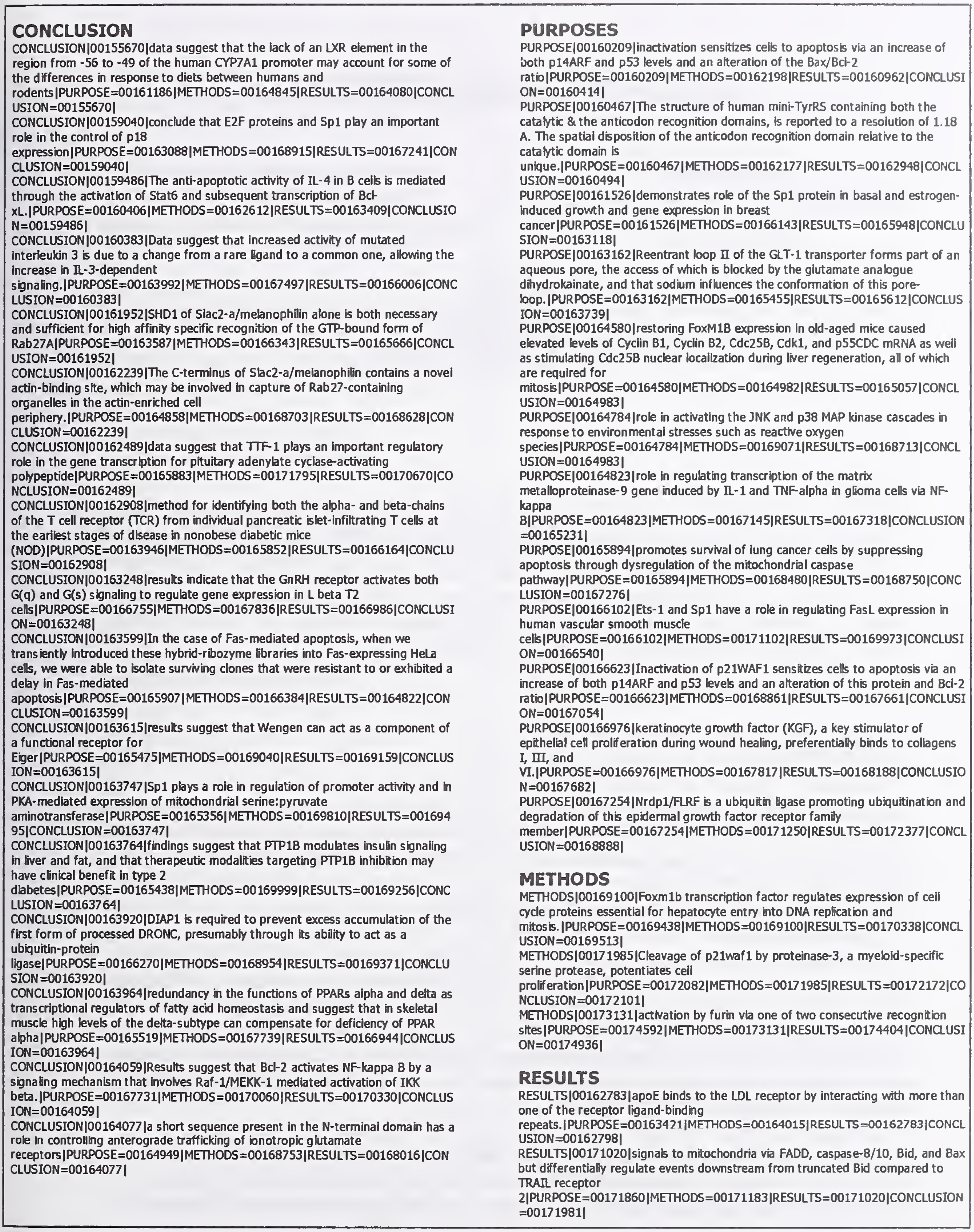
[20] Mitchell, J.A., Aronson, A.R., Mork, J.G., Folk, L.C., Humphrey, S.M., and Ward, J.M. Gene indexing: characterization and analysis of NLM's GeneRIFs. In *Proceedings of the AMIA 2003 Annual Symposium. 2003.*

[21] Collier, N., Nobata, C., and Tsujii, J. "Extracting the Names of Genes and Gene Products with a Hidden Markov Model", In COLING, pp. 201-207, August, 2000.

[22] Blaschke, C., Andrade, M.A., Ouzounis, C.A., and Valencia, A. Automatic extraction of biological information from scientific text: protein-protein interactions. In *Proceedings of the 7th International Conference on Intelligent Systems for Molecular Biology*, 1999, p. 60-67.

[23] S Blott, C Gurrin, GJF Jones, AF Smeaton and T Sodring. On the Use of MeSH Headings to Improve Retrieval Effectiveness. In *Notebook of the TREC-2003*, Gaithersburg, p. 355-364.

[24] PB Dobrokhotov, C Goutte, AL Veuthey, E Gaussier. A Probabilistic Information Retrieval Approach to Medical Annotation in SWISS-PROT. *Stud Health Technol Inform*.2003;95:421-6.



CONCLUSION

```
CONCLUSION|00155670|data suggest that the lack of an LXR element in the
region from -56 to -49 of the human CYP7A1 promoter may account for some of
the differences in response to diets between humans and
rodents|PURPOSE=00161186|METHODS=00164845|RESULTS=00164080|CONCL
USION=00155670|
CONCLUSION|00159040|conclude that E2F proteins and Sp1 play an important
role in the control of p18
expression|PURPOSE=00163088|METHODS=00168915|RESULTS=00167241|CON
CLUSION=00159040|
CONCLUSION|00159486|The anti-apoptotic activity of IL-4 in B cells is mediated
through the activation of Stat6 and subsequent transcription of Bcl-
xL.|PURPOSE=00160406|METHODS=00162612|RESULTS=00163409|CONCLUSIO
N=00159486|
CONCLUSION|00160383|Data suggest that increased activity of mutated
interleukin 3 is due to a change from a rare ligand to a common one, allowing the
increase in IL-3-dependent
signaling.|PURPOSE=00163992|METHODS=00167497|RESULTS=00166006|CONC
LUSION=00160383|
CONCLUSION|00161952|SHD1 of Slac2-a/melanophilin alone is both necessary
and sufficient for high affinity specific recognition of the GTP-bound form of
Rab27A|PURPOSE=00163587|METHODS=00166343|RESULTS=00165666|CONCL
USION=00161952|
CONCLUSION|00162239|The C-terminus of Slac2-a/melanophilin contains a novel
actin-binding site, which may be involved in capture of Rab27-containing
organelles in the actin-enriched cell
periphery.|PURPOSE=00164858|METHODS=00168703|RESULTS=00168628|CON
CLUSION=00162239|
CONCLUSION|00162489|data suggest that TTF-1 plays an important regulatory
role in the gene transcription for pituitary adenylate cyclase-activating
polypeptide|PURPOSE=00165883|METHODS=00171795|RESULTS=00170670|CO
NCLUSION=00162489|
CONCLUSION|00162908|method for identifying both the alpha- and beta-chains
of the T cell receptor (TCR) from individual pancreatic islet-infiltrating T cells at
the earliest stages of disease in nonobese diabetic mice
(NOD)|PURPOSE=00163946|METHODS=00165852|RESULTS=00166164|CONCLU
SION=00162908|
CONCLUSION|00163248|results indicate that the GnRH receptor activates both
G(q) and G(s) signaling to regulate gene expression in L beta T2
cells|PURPOSE=00166755|METHODS=00167836|RESULTS=00166986|CONCLUSI
ON=00163248|
CONCLUSION|00163599|In the case of Fas-mediated apoptosis, when we
transiently introduced these hybrid-ribozyme libraries into Fas-expressing HeLa
cells, we were able to isolate surviving clones that were resistant to or exhibited a
delay in Fas-mediated
apoptosis|PURPOSE=00165907|METHODS=00166384|RESULTS=00164822|CON
CLUSION=00163599|
CONCLUSION|00163615|results suggest that Wengen can act as a component of
a functional receptor for
Eiger|PURPOSE=00165475|METHODS=00169040|RESULTS=00169159|CONCLUS
ION=00163615|
CONCLUSION|00163747|Sp1 plays a role in regulation of promoter activity and in
PKA-mediated expression of mitochondrial serine:pyruvate
aminotransferase|PURPOSE=00165356|METHODS=00169810|RESULTS=001694
95|CONCLUSION=00163747|
CONCLUSION|00163764|findings suggest that PTP1B modulates insulin signaling
in liver and fat, and that therapeutic modalities targeting PTP1B inhibition may
have clinical benefit in type 2
diabetes|PURPOSE=00165438|METHODS=00169999|RESULTS=00169256|CONC
LUSION=00163764|
CONCLUSION|00163920|DIAP1 is required to prevent excess accumulation of the
first form of processed DRONC, presumably through its ability to act as a
ubiquitin-protein
ligase|PURPOSE=00166270|METHODS=00168954|RESULTS=00169371|CONCLU
SION=00163920|
CONCLUSION|00163964|redundancy in the functions of PPARs alpha and delta as
transcriptional regulators of fatty acid homeostasis and suggest that in skeletal
muscle high levels of the delta-subtype can compensate for deficiency of PPAR
alpha|PURPOSE=00165519|METHODS=00167739|RESULTS=00166944|CONCLUS
ION=00163964|
CONCLUSION|00164059|Results suggest that Bcl-2 activates NF-kappa B by a
signaling mechanism that involves Raf-1/MEKK-1 mediated activation of IKK
beta.|PURPOSE=00167731|METHODS=00170060|RESULTS=00170330|CONCLUS
ION=00164059|
CONCLUSION|00164077|a short sequence present in the N-terminal domain has a
role in controlling anterograde trafficking of ionotropic glutamate
receptors|PURPOSE=00164949|METHODS=00168753|RESULTS=00168016|CON
CLUSION=00164077|
```

PURPOSES

```
PURPOSE|00160209|inactivation sensitizes cells to apoptosis via an increase of
both p14ARF and p53 levels and an alteration of the Bax/Bcl-2
ratio|PURPOSE=00160209|METHODS=00162198|RESULTS=00160962|CONCLUSI
ON=00160414|
PURPOSE|00160467|The structure of human mini-TyrRS containing both the
catalytic & the anticodon recognition domains, is reported to a resolution of 1.18
A. The spatial disposition of the anticodon recognition domain relative to the
catalytic domain is
unique.|PURPOSE=00160467|METHODS=00162177|RESULTS=00162948|CONCL
USION=00160494|
PURPOSE|00161526|demonstrates role of the Sp1 protein in basal and estrogen-
induced growth and gene expression in breast
cancer|PURPOSE=00161526|METHODS=00166143|RESULTS=00165948|CONCLU
SION=00163118|
PURPOSE|00163162|Reentrant loop II of the GLT-1 transporter forms part of an
aqueous pore, the access of which is blocked by the glutamate analogue
dihydrokainate, and that sodium influences the conformation of this pore-
loop.|PURPOSE=00163162|METHODS=00165455|RESULTS=00165612|CONCLUS
ION=00163739|
PURPOSE|00164580|restoring FoxM1B expression in old-aged mice caused
elevated levels of Cyclin B1, Cyclin B2, Cdc25B, Cdk1, and p55CDC mRNA as well
as stimulating Cdc25B nuclear localization during liver regeneration, all of which
are required for
mitosis|PURPOSE=00164580|METHODS=00164982|RESULTS=00165057|CONCL
USION=00164983|
PURPOSE|00164784|role in activating the JNK and p38 MAP kinase cascades in
response to environmental stresses such as reactive oxygen
species|PURPOSE=00164784|METHODS=00169071|RESULTS=00168713|CONCL
USION=00164983|
PURPOSE|00164823|role in regulating transcription of the matrix
metalloproteinase-9 gene induced by IL-1 and TNF-alpha in glioma cells via NF-
kappa
B|PURPOSE=00164823|METHODS=00167145|RESULTS=00167318|CONCLUSION
=00165231|
PURPOSE|00165894|promotes survival of lung cancer cells by suppressing
apoptosis through dysregulation of the mitochondrial caspase
pathway|PURPOSE=00165894|METHODS=00168480|RESULTS=00168750|CONC
LUSION=00167276|
PURPOSE|00166102|Ets-1 and Sp1 have a role in regulating FasL expression in
human vascular smooth muscle
cells|PURPOSE=00166102|METHODS=00171102|RESULTS=00169973|CONCLUSI
ON=00166540|
PURPOSE|00166623|Inactivation of p21WAF1 sensitizes cells to apoptosis via an
increase of both p14ARF and p53 levels and an alteration of this protein and Bcl-2
ratio|PURPOSE=00166623|METHODS=00168861|RESULTS=00167661|CONCLUSI
ON=00167054|
PURPOSE|00166976|keratinocyte growth factor (KGF), a key stimulator of
epithelial cell proliferation during wound healing, preferentially binds to collagens
I, III, and
VI.|PURPOSE=00166976|METHODS=00167817|RESULTS=00168188|CONCLUSIO
N=00167682|
PURPOSE|00167254|Nrdp1/FLRF is a ubiquitin ligase promoting ubiquitination and
degradation of this epidermal growth factor receptor family
member|PURPOSE=00167254|METHODS=00171250|RESULTS=00172377|CONCL
USION=00168888|
```

METHODS

```
METHODS|00169100|Foxm1b transcription factor regulates expression of cell
cycle proteins essential for hepatocyte entry into DNA replication and
mitosis.|PURPOSE=00169438|METHODS=00169100|RESULTS=00170338|CONCL
USION=00169513|
METHODS|00171985|Cleavage of p21waf1 by proteinase-3, a myeloid-specific
serine protease, potentiates cell
proliferation|PURPOSE=00172082|METHODS=00171985|RESULTS=00172172|CO
NCLUSION=00172101|
METHODS|00173131|activation by furin via one of two consecutive recognition
sites|PURPOSE=00174592|METHODS=00173131|RESULTS=00174404|CONCLUSI
ON=00174936|
```

RESULTS

```
RESULTS|00162783|apoE binds to the LDL receptor by interacting with more than
one of the receptor ligand-binding
repeats.|PURPOSE=00163421|METHODS=00164015|RESULTS=00162783|CONCL
USION=00162798|
RESULTS|00171020|signals to mitochondria via FADD, caspase-8/10, Bid, and Bax
but differentially regulate events downstream from truncated Bid compared to
TRAIL receptor
2|PURPOSE=00171860|METHODS=00171183|RESULTS=00171020|CONCLUSION
=00171981|
```

Annex 1. Samples of GeneRIFs after argumentative classification: the first row gives the class; the second row gives the score of the selected class; then the textual segments is given; finally the score of the other classes is indicated.

QA UdG-UPC System at TREC-12

Marc Massot
Dept. Informàtica i Matemàtica Aplicada
Universitat de Girona
marc@ima.udg.es

Horacio Rodríguez
TALP Researh Center
Universitat Politècnica de Catalunya
dferres@lsi.upc.es

Daniel Ferrés
TALP Researh Center
Universitat Politècnica de Catalunya
dferres@lsi.upc.es

Abstract

This paper describes a prototype multilingual Q&A system that we have designed to participate in the Q&A Track of TREC-12. The system answer concrete responses, then we participate in the Q&A main task for factoid questions. The main areas of our system are: (1) Inductive Logic Programming to learn the question type, (2) Clustering of Named Entities to improve Information Retrieval and (3) Semantic relations and EuroWordNet synsets to perform a language-independent answer extraction.

1. Introduction

This paper describes a prototype Q&A system we have designed to participate in the Q&A Track of TREC-12. Our aim has been to build a system as much as possible language independent, where language dependent modules could be changed for allowing the system to be applied to different languages. In this way we have developed in parallel two different Q&A systems, one for English and another for Spanish.

As our research group has mainly focused in building resources and tools for NLP in Spanish, we have directly applied these tools and resources in our system. For English system we have used, when possible, publicly available resources or adapted our own tools.

In this paper we present the overall architecture of the system, we describe briefly its main parts, focusing on the language independent ones, and we present a preliminary evaluation of the prototype presented at the TREC-12 competition.

Our system was designed to participate in the Q&A main task for factoid questions. Thus, we develop a system to answer questions with a concrete response. We structure the remaining part of the paper as follows. In Section 2, we first give an overview of the system and then focusing on every subsystem. Finally, in section 3, we evaluate the results of this participation and we detail some conclusions.

2. The System

Overview

The system architecture follows the most commonly used schema, splitting the process into three phases that are performed in turn. Several iterations can be carried out within these phases to achieve their goals but once one phase is finished there is no possibility to return to previous phases. There are three main subsystems, one corresponding to each phase:

1. Question processing (QP)
2. Passage retrieval (PR)
3. Answer extraction (AE)

These subsystems are described below. Some pre-processing has been done on the document collection (the Acquaint corpus in this case). We will describe this issue when we present the PR subsystem.

Language dependent components are included only within the QP and AE subsystems.

Question Processing

The main goal of this subsystem is to classify the question regarding the kind of expected answer and to attach the information needed for the following subsystems. For PR the information needed is basically lexical (lists of keywords) and for AE lexical, syntactic and semantic. We have tried to represent all these kind of information using a language independent formalism. In particular we use the same semantic primitives and relations for the two languages (English and Spanish) involved in our system.

This subsystem uses a great amount of linguistic resources for performing its task. As our goal is processing questions in Spanish and English, both with independent linguistic resources and tools, we need mapping tools for providing information for the following subsystems in an uniform representation.

The tools used for the Spanish version are those of the group of NLP of the UPC (see [Atseries et al, 1998] for a description of these tools). The question is analyzed with a pipe including the following processors:

- **ms-analyze**, that performs tokenization, morphological analysis (including identification of quantities, dates, multiword terms, etc.), and POS tagging. As a result we obtain a sequence of tokens with POS and lemma.
- **tacat**, a partial parser that obtains nominal, prepositional and verbal phrases.

- **NERC**, a Named Entity Recognizer and Classifier that identifies the NE occurring in the question and classifies them in basic categories (person, place, organization,...). See [Carreras et al, 2002]
- Finally we obtain and attach semantic information using EWN[1]. The information obtained and used for further processing consists of the list of synsets (with no attempt to Word Sense Disambiguation), the list of hyperonyms of each synset (up to the top of each hyperonymy chain), the EWN's Top Concept Ontology, TCO class [Rodriguez et al,1998],and the Magnini's Domain Code [Magnini, Cavaglià, 2000].

For English version, we have adapted some of our tools or used publicly available ones for getting the same information using the same representation formalism.

- **TnT** [Brants, 2000], for the morphologic information, As TnT does not provide the lemma we have used the lemmatizer included in Princeton's WordNet software for covering this functionality.
- **MINIPAR** [Lin, 1998], to perform full parsing. A post-process has been needed for representing the output in a way compatible with tacat's output.
- The same NERC used for Spanish has been trained for English.
- A finer grained classifier for Geographic NE (those of type location) was developed, [Ferres et al, 2003a], devising a set of gazetteers with binary classifiers learned using an ILP learner, FOIL, [Quinlan, 1993].
- A gazetteer of acronyms obtained using a Decision Tree learning approach [Ferres et al, 2003b].
- A list of relations actor-action obtained through an analysis of the glosses of WordNet.

The result of applying these linguistic resources and tools, obviously language dependent, to the text of the question is represented in two structures:

- **Sint**, composed by two lists, one recording the information related to the syntactic components of the question (basically nominal, prepositional and verbal phrases) and the other collecting the information of dependencies and other relations between these components.
- **Sent**, that provides us with information for each lexical unit: the word form, the lemma, the POS, the semantic class (and subclass if available) of NE, the list of EWN synsets, the information associated to these synsets (TCO and DC) and, finally, whenever possible the verbs associated to the actor and the relations between locations and their gentile.

Once this information is obtained we are able to get the information relevant to the following tasks:

[1] http://www.illc.uva.nl/EuroWordNet/

- **Question type**. The most important information we need to extract from the question text is the Question Type, QT, because all the work the system has to perform for searching the answer is based on this issue. A failure on identifying QT practically disables the correct extraction of the answer. Currently we are working with about 50 QT. The QT tries to focus the type of the expected answer providing as well additional constraints on it. For instance, when the expected type of the answer is a person, two types of questions are considered, *Who_action*, which indicates that we are looking for a person who performs a certain action and *Who_person_quality*, that indicates that we are looking for a person having the desired quality. The action and the quality are the parameter of the corresponding QT. The following are examples of questions classified as *Who_action* type:

  - What is the name of the managing director of Apricot Computer?
  - Who won the Nobel Peace Prize in 1991?
  - Who is the writer of the book, "The Hobbit"?

  In order to determine the QT our system uses an Inductive Logic Programming (ILP) learner that learns, from a set of positive and negative examples, a set of weighted rules. We have used as learner the FOIL system [Quinlan, 1993]. We use FOIL for learning a different classifier (i.e. a set of rules) for each QT. As training set we have used the set of questions of TREC 8 and 9 (~900 questions) manually tagged and as test set the 500 questions of TREC 11. With these rules we have obtained an overall precision 68.54% and a recall of 60.00%. But the most of the errors are in similar QT categories, i.e. the 50% of errors for a generated when_begins are when_action questions, thus impact of these errors is minimum.
  For each classifier we have used as negative examples the questions belonging to the other classes. The features used for classifying are the following:

  - Word form
  - Word position in the question
  - Lemma
  - POS
  - Semantic class of NE, without subclassing
  - Synsets of the lemma
  - All hyperonyms for each synset without the information about the distance
  - TCO for each synset
  - Domain Codes for each synset
  - Main syntactic relations, subject and object relations

  The set of rules for each class has been manually revised and completed with a set of manually built rules (with lower weights) in order to assure a complete coverage. See below a couple of such rules:

  - A learned rule:

```
regla(non_human_actor_of_action,A,1) :-
  first_position(A,B),
  next_position(B,C),
      is_tco(cObject,C),
      is_domain(dTransport,C).
```

- The same rule after transformation(performed for the sake of efficiency):

```
    regla(non_human_actor_of_action,A,1,[],TT) :-
      sent(A,_,TT), TT=[_,W2|_],
      has_tco(W2,cObject),
      has_domain(W2,dTransport).
```

- A manual rule:

```
    regla(non_human_actor_of_action,A,994,[T1,T3],T) :-
      sent(A,_,[T1|T]),
      the_lema(T1,lema("which")),
      has_chunk_with_hyperonym(_,T,[T2|TT],
        [sArtifact,sObject,sAnimal],T3),
      the_pos(T2,pos("IN")),
      not(has_term_with_pos(TT,pos("JJS"),_)).
```

- **Environment.** Under this term we collect the semantic relations that hold between the different components identified in the question text. These relations are organized into an ontology of about 100 semantic classes and 25 relations (mostly binary) between them. Both classes and relations are related by taxonomic links. The ontology has been manually built and tries to reflect what is needed for an appropriate representation of the semantic environment of the question (and the expected answer). For instance, *Action* is a class and *Human_action* another class related to *Action* by an *is_a relation*. In the same way, *Human* is a subclass of *Entity*. *Actor_of_action* is a binary relation (between a *Human_action* and a *Human*). When a question is classified as *Who_action* an instance of the class *Human_action* has to be located in the question text. Later, in the AE phase, an instance of *Human_action* has to be located in the selected passages and an instance of *Human* related to it by the *Actor_of_action* relation has to be extracted as candidate to be the answer.

  The environment of the question is obtained from the syntactic information **(sint)** and the semantic information included in **sent**. For performing this task a set of about 150 rules has been manually built.

  For instance, for the question:

  Who won the Nobel Peace Prize in 1991?

  the following environment was obtained:

  action(A, won),
  time_of_event(A, T),
  year(T, 1991),
  theme_of_event(A, U),
  neothers(U, Nobel Peace Prize)

- **Semantic Constraints**. The environment tries to represent the whole semantic content of the question. Not all the items belonging to the environment are useful, however, for extracting the answer. So, depending on the QT, a subset of the environment has to be extracted. Sometimes additional relations, not present in the environment, are used and sometimes the relations extracted from the environment are extended, refined or modified. We define in this way the set of relations (the semantic constraints) that are supposed to be found in the answer. These relations are classified as mandatory or optional. Following the preceding example:
    - Mandatory Constraints:
        - actor_of_action(A, X)
        - action(A, won)
        - theme_of_event(A, U)
        - neothers(U, Nobel Peace Prize)
    - Optional Constraints:
        - time_of_event(A, T)
        - year(T, 1991)
- **Question Keywords.** The terms extracted from the question text that have to be used for performing the PR task.

Passage retrieval

In order to perform the PR task we have used MG [Witten et al, 1999]. Before the TREC-12 competition we indexed into MG the whole Acquaint collection. We built two indexes:

- Textual, i.e. indexing the documents from their textual content, as simple bag of words, with no pre-process
- Named Entities: We carried out a NERC process of the whole collection, we clustered these NE into clusters trying to group the different variants of the same entity, including acronyms for NE of type organization (see [Ferres et al, 2003b] for details of this process), and we indexed the documents using as key words the terms representative of the clusters.

At PR phase the process was the following:

With the Question Keywords obtained in the previous subsystem for each question, we looked for relevant documents in the two collections. We use a ranked retrieval for the textual collection with a threshold of 10% of the first retrieved document and a limit of 200 documents. For the Named Entities collection, we use Boolean retrieval in order to covers all the Named Entities of the query terms. The union of both sets of documents was indexed again into MG, this time only with the textual form but at level of passage. We consider a passage a sequence of 8 consecutive sentences of the original document allowing an overlapping of one sentence. Then, we retrieved the candidate passages with the same keywords, again with a threshold of 10% of the first retrieved passage but this time using a limit of 50 passages.

Answer extraction

The process of extraction of the answer is carried out on the set of passages obtained from the previous subsystem. These passages are segmented into sentences. Each sentence is then scored according to its semantic content regarding the question. We build a general semantic representation of the concepts that occurring in the question in order to overcome the term-based approach limits for the sentence selection. The semantic content of a term is represented using a weighted vector, the weight of each term is computed using the *idf* of the term, synonyms, hyponyms and hyperonyms. The semantic content of a concept is then built from the semantic content of its terms. [Vicedo, 2002].

The linguistic process of extraction, quite expensive, is carried out on the sentences best scored.

This process is similar to the process carried out on questions and leads to the construction of the environment of each candidate sentence. The rest is a mapping between the semantic relations contained in this environment and the Semantic Constraints extracted from the question. The mandatory restrictions must be satisfied to take in consideration the sentence, the satisfaction of the optional constraints simply increases the score of the candidate.

The final extraction process is carried out on the sentences satisfying this filter.

The Knowledge Source used for this process is a set of extraction rules owning a credibility score. Each QT has its own subset of extraction rules that leads to the selection of the answer.

The process of application of the rules follows an iterative approach. In the first iteration all the semantic constraints have to be satisfied by at least one of the candidate sentences. If no sentence has satisfied the constraints, the set of semantic constraint is relaxed by means of structural or semantic relaxation rules,· using the semantic ontology. If no candidate sentence occurs when all possible relaxations have been performed the question is assumed to have no answer.

Each candidate to solution comes weighted by diverse factors (sentence score, confidence of the used rules, satisfied optional restrictions, etc.). In the case more than one candidate is detected, a final process of weighted voting is carried out to select the preferred answer.

3. Evaluation and Conclusions

As we have said in the introduction, this paper describes the system we have built for our first participation in TREC competitions. Our main goal on attempting to participate in TREC-12 has been to acquire some experience on the kind of problems that have to faced in Q&A tasks. Although some of these problems have been foreseen by analysing other systems and previous competitions it is necessary to face the real problems (in real time) for taking the appropriate conclusions.

We have participate in TREC-12 with a prototype, not with a complete Q&A system. The different components of the system have got different levels of development (and, obviously, different level of accuracy). The first two subsystems, QP and PR, are the most completed and present the best results but the last one, AE, was only sketched and is currently under construction. Only a few rules for each Question Type were developed and no sufficient experimentation of the performance of these rules was carried out at the time of the competition.

With these constraints, the results obtained by our system are not good, but we think that they are a good starting point for further improvements of our system.

Some initial analysis of the results has been made and some comments follow.

As has been pointed above our participation was constrained to the factoid questions. So we provided answer to 413 questions. From them we gave the exact answer to only 22 questions (2 other were considered wrong). So the global accuracy of our system was 5.3%. The precision of recognising questions with no answer was of 9.2%, the recall was in this case of 43.3% (this figure was due to the fact that when our system was unable no find an answer the response was NIL, this was the case of 141 questions).

The classification of the question was rather good. The accuracy of our system was in this issue of 69%, quite close to the scores measured in our tests on previous TREC. Taking into account the fine granularity of our Types of Questions we think that it is a good result.

There 383 questions for which an answer exists in the collection. From these, only 157 were in the passages (50 per question) retrieved from the PR subsystem. This means that only 38% of the questions could be correctly answered by the AE subsystem.

As has been noted above, the AE module was only sketched for the competition. The accuracy of these components for questions which the answer occurred in our selected passages was of 8.3%.

Obviously, considered in isolation, the figures of AE are the worst of the three components. Part of the reason could be the accumulation of errors from previous components but the component itself has to be improved heavily.

There is of course enough room for improvement in every component and work is currently being done in all these components. In the next future, however, we plan to concentrate on getting better extraction rules by means of applying Machine Learning techniques, in the same line we have applied to the classification of questions problem.

References

[Atseries et al, 1998] J. Atseries, J. Carmona, I. Castellón, S. Cervell, M. Civil, L. Màrquez, M.A. Martí, L. Padró, R. Placer, H. Rodríguez, M. Taules, J. Turmo
Morphosyntactic Análisis and Parking of Unrestricted Spanish Text.
Proceedings of First International Conference on Language Resources and Evaluation. LREC-98, Granada, Spain.

[Brants, 2000] Brant, T.
TNT, A Statistical Part-of-Speech Tagger,
Proceedings of the 6th ANLP-NAACL, Seattle, USA. 2000

[Carreras et al, 2002] Carreras, X., Màrquez, L. and Padró L.
Named Entity Extraction Using Adaboost.
Proceedings of the 6th conference on Computational Natural Language Learning (CoNLL 2002). Shared Task Contribution. Taipei, Taiwan. September 2002.

[Ferres et al, 2003a] D. Ferrés, M. Massot, M. Padro, H. Rodríguez, J. Turmo
Automatic Classification of Geographic Named Entities
4th LREC, 2004, Lisboa, Portugal. Submittted

[Ferres et al, 2003b] D. Ferrés, M. Massot, M. Padro, H. Rodríguez, J. Turmo
Automatic Building Gazetteers of Co-referring Named Entities
4th LREC, 2004. Lisboa, Portugal. Submittted

[Lin, 1998] Lin, D.
Dependency-based evaluation of MINIPAR
Proceedings of the Workshop on the Evaluation of Parsing Systems, LREC 1998, Granada, Spain

[Magnini, Cavaglià, 2000] B. Magnini, G. Cavaglià
Integrating Subject Field Codes into WordNet.
Proceedings LREC, Athens, Greece, 2000.

[Quinlan, 1993] Quinlan, J. R.
FOIL: A midterm report. In Proc. of the sixth European Conf. on Machine
Learning, Springer-Verlag, 1993.

[Rodriguez et al,1998] H. Rodriguez, S. Climent, P. Vassen, L. Bloksma, W. Peters, A. Alonge, F. Bertanga, A. Roventini
The Top-Down Strategy for Bulding EuroWordNet: Vocabulary Coverage, Base Conceps and Top Ontology.
Computer and Humanities 32. 1998, Kluwer Academic Publishers.

[Vicedo, 2002] J.L. Vicedo
Un modelo semántico aplicado a los sistemas de B´squeda de Respuestas.
Ph.D. thesis, Dept. LSI. Universidad de Alicante, May 2002.

[Witten et al, 1999] I. Witten, A. Moffat, T. Bell
Managing Gygabytes.
1999, Morgan Kauffman, San Francisco, second edition.

Multiple-Engine Question Answering in TextMap

Abdessamad Echihabi, Ulf Hermjakob, Eduard Hovy,
Daniel Marcu, Eric Melz, Deepak Ravichandran
Information Sciences Institute
University of Southern California
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292
{echihabi,ulf,hovy,marcu,melz,ravichan}@isi.edu

1. Introduction

At TREC-2003, TextMap participated in the Main task, which encompassed answering the following types of questions:

- factoid questions;
- list questions;
- definition questions.

In this paper, we overview the architecture of the TextMap system and report its performance, as evaluated by the NIST assessors, on each of these tracks.

2. System architecture

2.1 Answering factoid and list questions

The TextMap system that answers factoid questions implements the following pipeline:

- A question analyzer identifies the expected answer type for the question given as input (see Section 3.1 for details).
- A query generator produces Web- and TREC-specific queries. The query generator exploits a database of paraphrases (see Section 3.2).
- Web queries are submitted to Google and TREC queries are submitted to the IR engine Inquery (Callan et al., 1995), to retrieve respectively 100 Web and 100 TREC documents.
- A sentence retrieval module selects 100 sentences each from the retrieved Web and TREC documents that are most likely to contain a correct answer.
- Three distinct answer selection modules (knowledge-, pattern-, and statistical-based) each pinpoint the correct answers in the resulting 200 sentences and assign them a score (see Section 3.3).
- A maximum-entropy-based re-ranker is used to combine the outputs of the three answer selection modules into a single ranked list (Section 3.4).
- If the top answer comes from a TREC sentence, the answer is presented to the user. If the top answer comes from a Web document, it is retrofitted to the TREC collection. When retrofitting fails, the system returns NIL.

List questions are answered by a system having the same architecture as the system used to answer factoid question. The only difference pertains to a simple backend algorithm that is used to output not one, but all answers whose score is higher than a given threshold.

2.2 Answering definition questions

Definition questions are answered by a system that uses many of the components described above and some additional ones. It scores answer candidates using the following resources:

- WordNet glosses
- Collection of 14,414 biographies from biography.com
- Mike Fleischman's corpus of 966,557 descriptors of proper people
- Set of subject-verb, object-verb, subject-copula-object relations

More details of the QA system that answers definition questions are given in Section 3.5.

In what follows, we describe in more detail each of the individual components and report the performance of our system.

3. Main Components

3.1 Question analysis

After parsing a question, TextMap determines its answer type, or "Qtarget", such as *PROPER-PERSON*, *PHONE-NUMBER*, or *NP*. We have built a typology of currently 185 different types, organized into several classes (Abstract, Semantic, Relational, Syntactic, etc.).

3.2 Query generation via reformulation

To bridge the gap between question and answer sentence wordings, TextMap uses paraphrasing. For any given question, TextMap generates a set of high-precision meaning-preserving reformulations to increase the likelihood of finding correct answers in texts. These reformulations are used to generate more focused TREC and Web queries. For example, the question below produces the following reformulations:

Question: How did Mahatma Gandhi die?

Reformulation patterns:

- Mahatma Gandhi died <how>?
- Mahatma Gandhi died of <what>?
- Mahatma Gandhi lost his life in <what>?
- Mahatma Gandhi was assassinated?
- Mahatma Gandhi committed suicide?
- … plus 40 other reformulations …

The fourth reformulation will easily match "Mahatma Gandhi was assassinated by a young Hindu extremist," preferring it over alternatives such as "Mahatma Gandhi died in 1948."

The reformulation collection in TextMap currently contains 550 assertions grouped into about 105 equivalence blocks. At run-time, the number of reformulations produced by our current system varies from one reformulation (which might just rephrase a question into a declarative form) to more than 40. On the TREC-2003 questions, we produced, on average, 5.03 reformulations per question.

3.3 Answer selection

3.3.1 Knowledge-based answer selection

The knowledge-based answer selection module uses the CONTEX parser (Hermjakob, 1997; 2001), a decision tree based deterministic parser, which has been enhanced for question answering by an additional treebank of 1,200 questions, named entity tagging that among other components uses BBN's IdentiFinder (Bikel et al., 1999), and a semantically motivated parse tree structure that facilitates matching for paraphrasing and of question/answer pairs. Answer selection is guided by the degree of matching at the syntactic/semantic level between question and answer parse trees and several additional heuristics that penalize answers for a variety of reasons, including:

- Qtarget match factor: Q: How long did the Manson trial last? Semantic mismatch: 20 miles
- Vagueness penalty: Q: Where is Luxor? Too vague: on the other side
- Negation penalty: Q: Who invented the electric guitar? Negation: Fender did **not** invent the electric guitar.

The knowledge-based answer selection module uses a limited amount of external knowledge to enhance performance: the WordNet hierarchy; internal quantity and calendar conversion routines; and abbreviation routines. See (Echihabi et al., forthcoming) for a detailed description.

3.3.2 Pattern-based answer selection

The pattern-based answer selection module uses a set of automatically learned patterns (Ravichandran and Hovy, 2002). The learning proceeds in two steps:

1. Given a Qtarget from the TextMap ontology (a relation such as BIRTHYEAR), and a few instances of <Question; Answer> pairs such as (NAME_OF_PERSON, BIRTHYEAR), extract from the web all the different patterns (TEMPLATEs) that contain such a pair.
2. Calculate the precision of each pattern and keep the patterns of high precision.

For example, for BIRTHYEAR, some of the patterns learned by the system and the precision of those patterns are:

Prec.	**#Correct**	**#Found**	**Pattern**
1	122	122	<NAME> (<BD>- <DD>
1	15	15	<NAME> (<BD> - <DD>) ,
1	13	13	, <NAME> (<BD> - <DD>)

0.9166	11	12	<NAME> was born on <BD> in
0.9090	10	11	<NAME> : <BD> - <TIME>
0.6944	25	36	<NAME> was born on <BD>

Once a set of patterns is learned, the pattern-based answer selection system uses them in order to find possible answer candidates. The set of potential answers is sorted into a ranked list using a maximum-entropy-based framework, in the style of Ittycheriah (2002). The features used by the system are the type of pattern; the frequency of an answer; the expected answer type; the word-overlap between the question and the answer sentence. The pattern-based ranker was trained using the 1192 questions in the TREC 9 and TREC 10 data sets for training and the 500 questions in the TREC 11 data set for cross-validation.

3.3.3 Statistics-based answer selection

The statistics-based answer selection module implements a noisy-channel model for answer selection - see (Echihabi and Marcu, 2003) for details. This model explains how a given sentence S_A that contains an answer sub-string A to a question Q can be rewritten into Q through a sequence of stochastic operations. Given a corpus of question-answer pairs (Q, S_A), one can train a probabilistic model for estimating the conditional probability $P(Q \mid S_A)$. Once the parameters of this model are learned, given a question Q and the set of sentences Σ returned by the IR engine, we find the sentence $S_i \in \Sigma$ and an answer in it $A_{i,j}$ by searching for the $S_{i,A_{i,j}}$ that maximizes the conditional probability $P(Q \mid S_{i,A_{i,j}})$. The probability model is trained using off-the-shelf parameter estimation package that was developed for statistical machine translation.

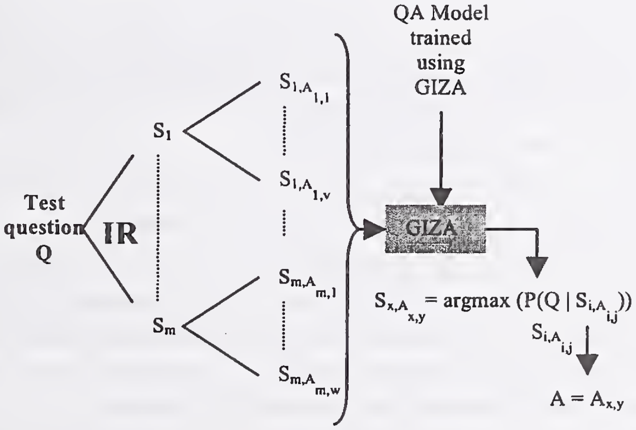


Figure 1: The noisy-channel approach to question answering.

For training, we use TREC 9 and TREC 10 questions (1091) with answer sentences (18618) automatically generated from the corresponding judgment sets. We also use questions (2000) from http://www.quiz-zone.co.uk with answer sentences (6516) semi-

automatically collected from the web and annotated for correctness by a linguist[1]. To estimate the parameters of our model, we use GIZA, a publicly available statistical machine translation package (http://www.clsp.jhu.edu/ws99/projects/mt/).
Figure 1 depicts graphically the noisy-channel based answer selection.

3.4 Combining the output of multiple answer-selection modules

Simple inspection of the outputs produced by our individual components on a development corpus of questions from the TREC-2002 collection revealed several consistent patterns of error:

- The pattern-based answer selection module performs well on questions whose Qtargets are recognizable named entities (NAMES, ORGANIZATIONS, LOCATIONS). However, this module performs less well on questions with more general QTargets (NPs, for example).
- The statistics-based answer selection module does not restrict the types of the answers it expects for a question Qtargets: it assumes that any semantic constituent can be an answer. As a consequence, many of the answers produced by this module may not be exact.
- Overall, all modules made some blatant mistakes. The pattern-based and statistics-based modules in particular sometimes select as top answers strings like "he", "she", and "it", which are unlikely to be good answers for any factoid question one may imagine.

To address these problems, we decided to use the maximum entropy framework to re-rank the answers produced by the answer selection modules and root out the blatant errors. For this purpose, we have used a set of 48 feature functions, which can be classified into the following categories:

1. Component-specific: Scores from the individual answer selection module and associated features like the rank of the answer and the presence/absence of the answers produced by the individual answer selection module. We also add features based on the scores produced by the IR module and word overlap between question and answers.
2. Redundancy-specific: Count of candidate answers in the collection. We also add additional features for the logarithm and the square root of the counts.
3. Qtarget-specific: It was observed that some of the answer selection modules answer certain Qtargets better than others. We therefore model a set of features wherein we combine certain classes of Qtarget with the score of the individual answer selection module. This enables the maximum entropy model to assign different model parameters to different types of questions and answer selection module.
4. Blatant-error-specific: We model a set of features based on the development set in an iterative process. These include (negative) features such as "answers usually do not have personal pronouns" and "WHEN questions usually do not contain day of the week as answer", among others.

[1] We are grateful to Miruna Ticrea for annotating the question-answer pairs.

3.5 Special modules/resources for answering definition questions

The question-answering track for TREC-2003 included a new definition subtask, in which definition questions such as "Who is Aaron Copland?", "What is a golden parachute?", and "What is Bausch&Lomb?" had to be answered by a set of relevant information nuggets.

Given a question like "Who is Aaron Copland?", the challenge here is to identify relevant sentences such as

- Aaron Copland's death comes a definitive biography of America's most important composer
- Copland was born in 1900, the son of Russian Jewish immigrants (original name: Kaplan)

in a larger set of sentences that the IR module extracted as potential candidates:

- So she took me to meet Aaron Copland, who was then in his early thirties.
- Each recipient of the honor, known as the Aaron Copland Awards, will get the run of the place with a spouse or partner, but no children or pets are allowed.

We built and used several resources to identify relevant information

3.5.1 Collection of Biographies

We were able to obtain 14,414 biographical entries from http://www.biography.com, which we used to extract core biographical information for specific people as well as identify words that are indicative of biographical information.

We built a list of 6,640 words that occur at least five times in biographies and occur much more frequently in biographical text (biography.com) than in standard text (Wall Street Journal). The top 20 terms (along with their "strength") are:

494.0	Nobel
467.5	Oxford
406.0	poems
384.0	knighted
290.0	Info
278.0	Ballet
257.0	Broadway
251.4	studied
247.0	travelled
209.0	poem
206.0	Labour
204.0	Composer
194.5	St
188.7	poetry
188.3	edited
187.5	Painter
183.0	Angeles
181.7	Physicist
171.9	War
169.2	commanded

Ten terms of lesser strength that nevertheless boost the score in the Copland examples:

49.5	Symphony
47.4	teacher
42.5	Music
34.7	Oscar
22.3	son
18.7	achievements
16.5	immigrant
13.2	established
12.9	advocate
10.1	Jewish

The second use of the biography collection was to boost answer candidates with words from core biography terms for the specific person in question:

- Copland , Aaron 1900 -- 1990 is a Composer; born in New York City.

3.5.2 Collection of Descriptors for Proper People

TextMap uses a slightly cleaned-up version of Michael Fleischman's list of 966,557 descriptor entries for proper people (Fleischman et al., 2003) such as

10 composer , composer , composer , Aaron Copland
2 witness , witness , witness , Aaron Copland
1 musician , musician , Pulitzer Prize winning musician , Aaron Copland
1 musician , musician , musician , Aaron Copland
1 composer , composer , late composer , Aaron Copland
1 composer , composer , classical composer , Aaron Copland

boosting the score of sentences containing descriptors such as "composer" or "Pulitzer Prize winning musician".

3.5.3 WordNet

If the anchor term of the definition question (e.g. "Aaron Copland") has a WordNet gloss (Miller and Fellbaum, 1998), TextMap gives credit for sentences that contain words and expressions that overlap with that WordNet gloss (as underlined in the example below):

WordNet gloss: Copland, Aaron Copland -- United States composer (1900-1990)

3.5.4 Semantic-Relationship Patterns

TextMap gives credit when the anchor of a question is found to match one of currently 110 semantic relations, for example when the question anchor is the logical subject of verbs like "to compose", "to write", "to teach"; when it is the logical object of verbs like "to bear" (to be born); when it is in a subject-copula-object relationship with a head noun like "composer", "advocate", "son" (or any other relative).

- Aaron Copland composed "Fanfare for the Common Man."
- Aaron Copland was born in 1900, the son of Russian Jewish immigrants

3.5.5 Duplicate Removal and Answer Cutoff

Duplicates are avoided by giving credit only for those words, expressions and relations that have not yet been observed in higher-scoring information nuggets.

To determine the number of answer nuggets for a given question, we start with a very low threshold score and then linearly increase the threshold for each answer, eliminating answers with a score lower than the rising threshold. This mechanism keeps the number of answers in a reasonable range while allowing more answers for questions with many high-scoring answer candidates. This allowed our system to provide over 20 nuggets for questions such as "Who is Bill Bradley/Aaron Copland/Andrew Carnegie?" while limiting itself to as few as 3 answers questions such as "What is Iqra/El Shaddai/ Restorative Justice?" for which the system found fewer promising information nuggets.

For the definition questions in TREC-2003, our system returned an average of 12.2 answers and 1663.1 bytes per question.

4. Evaluation Results

In the official TREC evaluation, TextMap scored 33.7% for factoid questions, 11.8% for list questions, and 46.1% for definition questions, resulting in a composite score of 31.3%

We also performed the following three sets of experiments with the Web as our corpus.

1. Maximum Entropy re-ranking performed on individual answer selection modules: In this experiment, we turn off all the features in any answer selection module that depends on another answer selection module and we use for re-ranking only the top 50 answers supplied by one answer selection component. Thus this experiment enables us to assess the impact separately on each individual answer selection module of redundancy-, Qtarget-, and blatant-error-specific features, as well as improved weighting of the features specific to each module.
2. Maximum Entropy re-ranking on all answer selection modules: In this experiment we add all the features described in Section 5.3 and test the performance of the combined system after re-ranking. Re-ranking is performed on the answer set that results from combining the top 50 answers returned by each individual answer selection module. This experiment enables us to assess whether the ME framework enables us to better exploit strengths specific to each answer selection module.
3. Feature Selection: Since we have only 200 training examples and almost 50 features, the training is likely to be highly prone to over-fitting. To avoid this problem we apply feature selection as described in (Della Pietra et al., 1996). This reduces the number of features from 48 to 31.

Metric	Knowledge-Based		Pattern-Based		Statistical-Based		Base from all systems followed by ME re-ranking (no feature selection)	Base from all systems followed by ME re-ranking (with feature selection)
	Base	Base followed by ME re-ranking	Base	Base followed by ME re-ranking	Base	Base followed by ME re-ranking		
Top answer	35.83%	45.03%	25.18%	30.50%	21.30%	32.20%	46.37%	47.21%
Top 5	57.38%	56.41%	35.59%	43.09%	31.23%	40.92%	57.62%	57.62%
MRR	43.88	49.36	28.57	35.37	24.83	35.51	51.07	51.27

Table 1. Performance of various answer selection modules in TextMap, an end-to-end QA system.

Table 1 summarizes the results: it shows the percentage of correct, exact answers returned by each answer selection module with and without ME-based re-ranking, as well as the percentage of correct, exact answers returned by an end-to-end QA system that uses all three answer selection modules together. Table 1 also shows the performance of these systems in terms of percentage of correct answers ranked in the top 5 answers and the corresponding MRR scores.

The results in Table 1 show that appropriate weighting of the features used by each answer selection module as well as the ability to capitalize on global features, such as the counts associated with each answer, are extremely important means for increasing the overall performance of a QA system. ME re-ranking led to significant increases in performance for each answer selection module individually. When measured with respect to the ability of our system to find correct, exact answers when returning only the top answer, it accounted for 14.33% reduction in the error rate of the knowledge-based answer selection module; 7.1% reduction in the error rate of the pattern-based answer selection module; and 13.85% reduction in the error rate of the statistical-based answer selection module. Combining the outputs of all systems yields an additional increase in performance; when over-fitting is avoided through feature selection, the overall reduction in error rate is 3.96%. This corresponds to an increase in performance from 45.03% to 47.21% on the top-answer accuracy metric.
The inspection of the weights computed by the ME-based feature selection algorithm shows that our log-linear model was able to lock and exploit strengths and weaknesses of the various modules. For example, the weight learned for a feature that assesses the likelihood of the pattern-based module to find correct answers for questions that have QUANTITY as a Qtarget is negative, which suggests that the pattern-based module is not good at answering QUANTITY-type questions. The weight learned for a feature that assesses the likelihood of the statistics-based module to find correct answers for questions that have an UNKNOWN or NP Qtarget is large, which suggests that the statistical module is better suited for answering these types of questions. The knowledge-based module is better than the others in answering QUANTITY and DEFINITION questions. These results are quite intuitive: The pattern-based module did not have enough QUANTITY-type questions in the training corpus to learn any useful patterns. The Statistics-based module implemented here does not explicitly use the Qtarget in answer selection and does not model the source probability of a given string being a correct answer. As a consequence, it explores a much larger space of choices when determining an answer compared to the other two modules.
At the moment, the knowledge-based module yields the best individual results. This is not surprising given that it is a module that was engineered carefully, over a period of three years, to accommodate various types of Qtargets and knowledge resources. Many of the resources used by the knowledge-based module are not currently exploited by the other modules: the semantic relations identified by CONTEX; the ability to exploit the paraphrase patterns and advanced forms of reformulation for answer pinpointing; the external sources of knowledge (WordNet; abbreviation lists; etc.); and a significant set of heuristics (see Section 2.4).
Our results suggest that in order to build good answer selection modules, one needs to both exploit as many sources of knowledge as possible and have good methods for

integrating them. The sources of knowledge used only by the knowledge-based answer selection module proved to have a stronger impact on the overall performance of our answer selection systems than the ability to automatically train parameters in the pattern- and statistics-based systems, which use poorer representations. Yet, the ability to properly weight the contribution of various knowledge resources was equally important. For example, Maximum Entropy naturally integrated additional features into the knowledge-based answer selection module; a significant part of the 9.2% increase in correct answers reported in Table 1 can be attributed to the addition of redundancy features, a source of knowledge that was unexploited by the base system.

References

Bikel, D., R. Schwartz, and R. Weischedel. 1999. An Algorithm that Learns What's in a Name. *Machine Learning—Special Issue on NL Learning*, 34, 1–3.

Callan, J.P., W.B. Croft, and J. Broglio. 1995. "TREC and Tipster Experiments with Inquery. *Information Processing and Management* 31(3), 327-343.

Echihabi, A., U. Hermjakob, E. Hovy, D. Marcu, E. Melz, D. Ravichandran. 2003. How to Select an Answer String. *Forthcoming.*

Echihabi, A., and D. Marcu. 2003. A Noisy-Channel Approach to Question Answering. *Proceedings of the 41st Meeting of the Association for Computational Linguistics (ACL)*, July 7-12, Sapporo, Japan.

Fellbaum, C. (Editor); G. Miller (Preface). 1998. WordNet: An Electronic Lexical Database. MIT Press. http://www.cogsci.princeton.edu

Fleischman, M., E. Hovy and A. Echihabi. 2003. Offline Strategies for Online Question Answering: Answering Questions Before They Are Asked. *Proceedings of the 41st Meeting of the Association for Computational Linguistics (ACL)*, July 7-12, Sapporo, Japan.

Hermjakob, U. 1997. *Learning Parse and Translation Decisions from Examples with Rich Context.* Ph.D. dissertation, University of Texas Austin. file://ftp.cs.utexas.edu/pub/mooney/papers/hermjakob-dissertation-97.ps.gz.

Hermjakob, U., A. Echihabi, and D. Marcu. 2002. Natural Language Based Reformulation Resource and Web Exploitation for Question Answering. *Proceedings of the TREC-11.* NIST, Gaithersburg, MD.

Hovy, E.H., U. Hermjakob, C.-Y. Lin, and D. Ravichandran. 2002. Using Knowledge to Facilitate Pinpointing of Factoid Answers. *Proceedings of the COLING-2002 Conference.* Taipei, Taiwan.

Ittycheriah, A. 2001. *Trainable Question Answering System*. Ph.D. Dissertation, Rutgers, The State University of New Jersey, New Brunswick, NJ.

Ravichandran, D. and E.H. Hovy. 2002. Learning Surface Text Patterns for a Question Answering System. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL).* Philadelphia, PA 41–47.

The University of Sheffield's TREC 2003 Q&A Experiments

Robert Gaizauskas, Mark A. Greenwood, Mark Hepple,
Ian Roberts, Horacio Saggion and Matthew Sargaison

r.gaizauskas,m.greenwood,
m.hepple,i.roberts,saggion } @dcs.shef.ac.uk

Department of Computer Science
University of Sheffield

1 Introduction

The systems entered by the University of Sheffield in the question answering track of previous TRECs have been developments of the system first entered in TREC 8 (Humphreys et al., 1999). Although a range of improvements have been made to the system over the last four years (Scott and Gaizauskas, 2000; Greenwood et al., 2002), none has resulted in a significant performance increase. For this reason it was decided to approach the TREC 2003 evaluation more as a learning experience than as a forum in which to promote a particular approach to QA. We view this as the beginning of a process that will lead to much fuller appreciation of how to build more effective QA systems.

Our efforts this year were focussed on a number of objectives:

1. to understand better how certain key components of our system architecture, specifically the information retrieval (IR) component that feeds the answer extraction component, were performing and how modifying their behaviour might affect overall system performance;

2. to implement a simple baseline system that would allow for comparison with the more linguistically motivated system we have entered in the past;

3. to build a dedicated subsystem for the definition question subtask that was introduced this year;

4. to preprocess the entire AQUAINT corpus to extract useful information for use by a QA system, in particular named entities in specific classes.

In the following section we first briefly summarise our overall system architecture and approach to QA – largely unchanged from previous years – and then discuss our work with respect to each of the above four objectives. With this detail as background, we then describe the three runs we submitted and review their results. While the final scores obtained are not high, they do not reflect the value of what was learned in the course of preparing for the evaluation, most of which was learned too late to feed through into in the system we entered.

2 System Development and Experimentation

As detailed in our previous years' TREC submissions, our core QA system consists of an IR system coupled to a natural language analysis system. The essence of the approach is to pass the question unmodified to the information retrieval (IR) system which uses it as a query to do passage retrieval against the text collection. The top ranked passages output from the IR system are then passed to a modified information extraction (IE) system. This system first carries out partial, robust syntactic and semantic analysis of these passages and of the question (in which a specific *"sought entity"* is determined), transducing them both into a predicate-argument or quasi-logical form (QLF) representation. In this representation the predicates are, for the most part, either the unary predicates formed from the morphological roots of nominal or verbal forms in the text or binary predicates from a closed set of grammatical relations (e.g. `object`, `subject`) or of prepositions (e.g. `in`, `after`).

Given these sentence level "semantic" representations of candidate answer-bearing passages and of the question, a discourse interpretation step then creates a discourse model of each retrieved passage by running a coreference algorithm against the semantic representation of successive sentences in the passage, in order to unify them with the discourse model built for the passage so far. This results in multiple references to the same entity across the passage being merged into a single unified instance. Next, coreference is computed again between the QLF of the question and the discourse model of the passage, in order to unify common references.

In these passage+question models, possible answer entities are identified and scored as follows. First each sentence in each passage is given a score based on counting matches of entity types (unary predicates) between the sentence QLF and the question QLF (similar to counting noun and verb overlap in word-overlap approaches). Next each entity from a passage not so matched with an entity in the question (and hence remaining a possible answer) gets a preliminary score according to (1) its semantic proximity (in Wordnet) to the type of the entity sought by the question and (2) whether or not it stands in a relation R to some other entity in the sentence in which it occurs which is itself matched with an entity in the question which stands in relation R to the sought entity (e.g. an entity in a candidate answer passage which is the subject of a verb that matches a verb in the question whose subject is the sought entity will have its score boosted). An overall score is computed for each entity as a function of its preliminary score and the score of the sentence in which it occurs.

Finally, the ranked entity list is post-processed to merge and boost the scores of multiple occurrences of the same answer found in multiple passages and the top scoring answer is then proposed as the answer to the question.

Overall the intention is that the matching of candidate answer entities to the sought entity of the question be guided primarily by semantic type similarity (so "who" questions should have persons proposed as answers), then by lexeme overlap between question and answer-bearing sentence, and finally by sharing of grammatical relations where they can be identified. Redundancy of the answer across the candidate answer bearing passages (and optionally across external resources such as the Web) is also taken into account.

2.1 Coupling IR and QA

Using an IR system as the first component in a QA system to retrieve relevant candidate answer-bearing passages is an approach widely adopted by TREC participants. It makes sense as a way of narrowing the collection size down to something manageable for the more detailed, and processor intensive, analysis required for answer extraction. However, in such an architecture the performance of the IR component cleary bounds the performance of the overall system.

For the past two TREC QA evaluations in which we have participated we have used the Okapi system (Robertson and Walker, 1999), a probabilistic IR system which has been evaluated very favourably in the TREC mainstream IR evaluations. Despite these favourable evaluations, our analysis of its performance in the context of QA has shown up a serious problem. Experiments detailed in (Roberts and Gaizauskas, 2004) suggest that when using the top 20 relevant passages, answer bearing passages are found for only 66% of the questions. This figure increases to about 80% if the top 100 passages are considered and only to 82% for the top 200 passages. Similar results have been found by Hovy et al (2000), who report that this figure rises only to 92% considering the top 1000 documents returned per question by their IR system.

Document Rank	*Coverage*	*Percentage Correct*
0	0.0%	0.000%
1	18.6%	11.549%
2	25.4%	14.930%
5	37.0%	16.056%
10	46.0%	17.746%
20	54.0%	18.592%
30	57.4%	16.620%
50	61.4%	17.183%
100	66.8%	16.056%
150	68.0%	13.803%
200	69.2%	12.676%

Table 1: Coverage and end-to-end performance when using Okapi.

Since in the past we had been considering only the top 20 passages per question, we were placing an upper bound of 60% of questions being answered correctly on our overall system before answer extraction even began. To address this major issue we pursued two lines of enquiry this year:

1. How does supplying more documents from lower down the IR system ranking affect overall QA performance?

2. Can better IR performance, as viewed from a QA perspective, and better combined IR/QA performance be obtained by adopting a more hands-on approach to IR, e.g. implementing an inverted index for boolean retrieval, and experimenting with query formulation from natural language questions?

2.1.1 Descending the Ranking

Experiments were carried out with the aim of finding if the increase in coverage (the percentage of questions for which at least one answer bearing document is returned) obtained by considering more documents improves the end-to-end performance of the question answering system. The top 200 relevant passages were retrieved from AQUAINT for 350 of the main track questions from TREC 2002[1]. Coverage and end-to-end performance was then determined at a number of document ranks giving the results shown in Table 1.

It should be clear from the results in Table 1 that although the best coverage is achieved by considering

[1] All of these questions were known to have at least one correct answer within the collection.

the top 200 relevant passages this does not translate to better end-to-end performance of the system as this is clearly achieved using only the top 20 relevant passages.

Although these results show that using Okapi as the IR engine yields a relatively low coverage and end-to-end performance it at least shows that the best results are obtained when using just twenty relevant passages. On the one hand these results are pleasing as this is the configuration we were and will continue to use, now with some justification. On the other hand, since this places an upper bound of 60% of less on the correctness of our QA system, it reveals that we must either change the approach to IR, to get more answer-bearing passages in the top ranks, or reduce the answer extraction component's tendency to get distracted by noise as it descends the ranking.

2.1.2 Boolean IR with MadCow

To experiment with a more hands-on approach to IR we implemented a boolean search engine called MadCow which does word indexing at the sentence level. That is, each document in the AQUAINT collection is sentence-split and an inverted index of all words, minus those in a stoplist, is created which associates with each word a list of its document plus sentence offset occurrences and a count of how many documents it occurs in across the collection (so-called document frequency). A basic boolean query language was implemented which supports queries of arbitrarily deeply nested conjunctions and disjunctions of search terms (words); negation is not supported at this point as there has not appeared to be a need for it. When a query is issued all sentences which satisfy the query are returned. Note that this means that, e.g., a document or even two adjacent sentences which satisfy the query will not be returned: the query must be satisfied by a single sentence. This constraint may prove to be too severe, but it is a principled starting point. Relaxing the matching conditions can straightforwardly be done in further work.

Given the availability of a boolean search engine, how can questions best be mapped into queries to maximise the number of answer-bearing sentences returned from the collection? Where a conventional ranked retrieval engine is used in a QA system, for example engines using a probabilistic model (like Okapi) or a vector-space model, a question can be used directly as an IR query to retrieve the documents that are to be analysed for possible answers. More complicated schemes can be employed, of course, but this simple approach is both possible and widely used.

For boolean retrieval engines such as MadCow, however, the task of formulating queries to retrieve documents relevant to answering some question is non-trivial. A query formed as a conjunction of the words in the question (omitting stoplist words, perhaps) will commonly be too restrictive, returning few or no documents, whereas a query that is a disjunction of the same words will commonly retrieve too many. The best approach for formulating boolean retrieval queries is an open research topic. In what follows, we will describe the particular query formulation approach we implemented.

This query formulation approach places particular importance on the recognition of names within questions, and knowledge of variant forms for names of the same person/company, as acquired from the corpus by the methods described in Sec. 2.4.1. Query formulation operates in two phases, of which the first phase works with only names recognised within questions, whilst the second phase uses other, non-name, words from the question. A preprocessing component is applied to each question, returning the names identified, plus any known variants of these names. These alternate forms are used to create a 'strong' boolean condition, which succeeds if all the words of any one of the name variants are found in a document. A 'weak' condition is also computed for each name, which is usually just the final word of the name. For example, the system might recognise the name *Bill Clinton* and suggest the variant *President Clinton*, giving rise to the strong condition: `(( Bill & Clinton ) | ( President & Clinton ))`, with the weak condition being just `( Clinton )`. A range of search expressions are generated for each question by conjoining over conditions from the identified names, where each name may be represented by either its strong or weak condition, or might be omitted. For example, for two names, we will have conditions such as $(str_1 \& str_2)$ and $(wk_1 \& str_2)$, but also just (str_1) and (wk_2) (giving eight in total, as the entirely null condition is excluded). Retrieval is done for all of these search expressions, whose 'specificity' is reflected in the size of the passage set returned, i.e. the less passages retrieved, the more specific the query. Working through these results in order of decreasing specificity, the system collects the result passages together (deleting duplicates) until either all results sets have been included or adding passages for the next set would make the collection exceed a specified upper limit of size (we used a limit of 250 in these experiments). At the end phase 1, the overall process may terminate if the collected passages number more than some specified lower limit of size (we used a limit of 100 in these experiments). If not, phase 2 is initiated.

If phase 1 ended because the passage set for some name condition `N` is too large, then phase 2 serves to elaborate this condition by constructing search conditions W built from non-name words in the question, i.e. constraints (N & W) are used to documents for addition to the collection. Otherwise, conditions W built from non-name words in the question are used on their own. The conditions W are built by taking non-stoplist words from the question, which are known to exist somewhere in the corpus (i.e. have document frequency > 0). For each such word, its known variant forms are accessed (i.e. inflectional variants, e.g. *decided/decide*, and related nouns/verbs, e.g. *decision/decide*) are disjoined together, and then the expressions that result for the different words in the question are conjoined together. If the search condition that results fails to retrieve enough documents, it is weakened by deleting the sub-expression for the question word having the highest document frequency in the corpus (which is thereby deemed to be the 'least informative' term). This process iterates until either enough documents have been collected, or the condition cannot be further weakened (i.e. when it is derived from only a single word of the question).

At the end of the two phases, the system has a collection of documents from which a maximum of 50 are returned, with a preference being given firstly to those extracted in phase 1, and secondly those having greatest overlap with the question. If the system's collected document set is empty (which might happen if all searches have retrieved either no documents or more than the specified upper limit), then the system will fall back to just using 50 documents from some oversized set produced at an earlier stage.

Clearly there are many ways in whic this particular query formulation approach might be varied or refined and empirical work is required for the further development of the approach.

2.2 A Simple Baseline System

For TREC-8 and 9, which required 50 or 250 byte answers, we were able to implement a simple baseline using Okapi only in which the central 50 or 250 bytes in the top-ranked passages were returned as an answer. This is not a sensible baseline for returning single, exact answers, however, so for the past few years we have not had a baseline system against which we could compare our more complex, linguistically motivated system.

To rectify this situation we decided to implement a simple system that works as follows. First, the approach starts with the limiting assumption that all questions can be answered by one or more entities from a fixed set of semantic types. The entity types which can be recognised include the standard named entity categories from the MUC competitions (persons, organizations, locations, times and dates, monetary amounts) plus a wider class of measures (space, mass, duration, etc.) and types frequently seen in previous TRECs, as discussed below in 2.5.1 Clearly this assumption is not always warranted as in the question *"How did Patsy Cline die?"* for which the correct answer is *"in a plane crash"* which is an event type and not an entity.

The operation of this simple baseline system is as follows:

- Question Typing: The expected answer type is determined using a set of hand coded rules which operate over the words in the sentence. For example a question containing the word "who" suggests that the answer will be of type Person.
- Information Retrieval: Assuming the question can be typed then the IR approach outlined in Section 2.1.1 is used to find the top 20 most relevant passages; if the question cannot be typed, then the system simply returns no answer for this question.
- Answer Extraction: All named entities of the correct type are extracted from the relevant documents and retained as possible answers unless they fail one of the following conditions:
  - The document from which the current entity is drawn must contain all the named entities found in the question, and
  - The current entity must have no overlap with the question.

The remaining entities are then grouped together using the following equivalence test (Brill et al., 2001): *two answers are said to be equivalent if all of the non-stopwords in one are present in the other or vice versa.* The most frequently occurring answer group is then proposed as the answer to the question or if the question requires multiple answers then all answers located are proposed in order of frequency of occurrence[2].

2.3 Answering Definition Questions Through Query Expansion

Definition questions require a different approach to that used by QA-LaSIE for answering factoid questions mainly because the questions contain very little information useful for finding definition-bearing documents. For example there are 1108 sentences in the AQUAINT collection which contain the word *"aspirin"* most of which do not include a definition. However, if we can determine from external sources that *"analgesic"* occurs frequently in sentences defining aspirin

[2] The longest realisation of the answer within the group is used along with the highest ranked document identifier, which can, on occasions, lead to unsupported answers – 15 of the 450 questions in this evaluation.

then we can reduce the search space to only eight sentences in AQUAINT. The following method is used to determine the secondary terms to help locate definition bearing sentences:

- Definiendum Extraction: As the definiendum can be a complex linguistic unit we rely on a chart parser to produce a syntactic representation of the question. The definiendum is then assumed to be the right most noun phrase (note we assume that all definiendums will be nouns).
- Pattern Generation: We generate definition phrases, such as *"aspirin is a"* and *"such as aspirin"* from a set of fifty "seed" patterns ready for later processing.
- Secondary Term Extraction: We rely on three external sources for secondary term extraction:
  - WordNet (Miller, 1995): All adjectives, nouns and verbs found in the glosses of the definiendum are extracted as are the related hypernyms of the definiendum.
  - Britannica: All pages containing the definiendum are retrieved and the nouns, verbs and adjectives found in sentences containing the definiendum are extracted.
  - Web: The definition phrases generated earlier are used to locate unique relevant documents on the web. The nouns, verbs and adjectives are then extracted from the sentences within these documents which contain the definiendum.

  A combined list of secondary terms, up to a maximum of n elements, is then generated as follows:

  - all secondary terms found in WordNet (m terms, $m \leq n$),
  - a maximum of $(n-m)/2$ terms from Britannica with a frequency greater than one,
  - web terms with a frequency greater than one are then appended to the list until the maximum of n is reached.

  The order of the list reflects a degree of confidence we have in the source and also the fact that the more a term is used in a particular "definition" context the more we believe it is associated with the definiendum.
- Query Generation and Passage Retrieval: We have tried two different approaches to passage retrieval which necessitate different approaches to query generation. The approaches are as follows:
  - When using the probabilistic IR engine, Okapi, the search query is composed of the tokens in the question and the full list of secondary terms found in the previous stage. The twenty most relevant passages within AQUAINT are then retrieved using this query.
  - In the case of the MadCow boolean search engine an iterative procedure is used to generate the search query. Suppose the term sought is composed of tokens $\{m_i\}$ $(1 \leq i \leq k)$ and the list of secondary terms consists of tokens $\{s_j\}$ $(0 \leq j \leq l)$, then during iteration p the following boolean search is tried (term conjunction is represented by & and term disjunction by |):

    $$(m_1 \& m_2 \& ... \& m_k \& (s_1|s_2|...|s_p))$$

    If the number of passages returned in iteration p is greater than 20, then the searching procedure stops. This procedure on the one hand limits the scope of the main term at each step by conjoining it with a secondary term and on the other hand broadens its scope by disjoining it with additional secondary terms. In a sense this approach is similar to the narrowing and broadening techniques used in the MURAX system (Kupiec, 1993). In our approach, the order in which the secondary terms are used to expand the query reflects their association with the main term; thus, this search procedure is expected to retrieve good passages.
- Definition Extraction: More than one definition can be extracted at this stage not only because different fragments cover different aspects of the definition (e.g. *"aspirin is a drug"* vs. *"aspirin is a blood thinner"*) but also because the definiendum can be ambiguous (i.e. there are seven senses of "battery" in WordNet). We restrict our analysis of definitions to the sentence level, as a sentence is considered definition-bearing if it matches one of the previously generated patterns or if it contains the definiendum and at least three secondary terms (this threshold was arrived at after several experiments on a small training set).

  Instead of returning the full sentence as a definition we return the suffix which contains the definiendum and all secondary terms appearing in the sentence. This is in a crude attempt to remove some of the unnecessary information the sentence may contain.

  In an attempt to return the same definition only once, a vector representation of the definition, consisting of its terms and term frequencies, is created. If the current definition is too similar to any of the previously extracted defintions then it is discarded. Similarity here is quantified as the cosine between two vector representations, i.e. two vectors v_1 and v_2 are considered similar if $cosine(v_1, v_2) > threshold$ (the $threshold$ was determined through experimentation over a small training set).

2.4 Corpus Pre-Processing

In previous years QA tracks we brought language processing modules to the QA task that were not tailored in any particular way to the evaluation corpus. Clearly it makes sense to ensure your tools are able to work effectively on the evaluation corpus and so to address this issue we carried out two activities, one to preprocess the corpus to attempt to extract information about all named entities in the corpus, the other to extend our POS tagger lexicon by attempting to assign tags to all words in the corpus not in the lexicon.

2.4.1 Automatically Acquiring Named Entity Information from AQUAINT

All groupings of two or more consecutive capitalised word (plus full-stops and the lower-case word 'of') were extracted from the corpus. This list, comprising 14,919,780 multi-word expressions in 2,615,767 distinct forms, ranging from "UnitedStates" to "Prime Minister Benjamin Netanyahu of Israel", was used in developing an experimental automated Named Entity catalog.

Because of the size of the corpus and the fact that these groups are lacking in complex grammar, the process used was one of simple pattern matching against key-word list including Titles ('Mr', 'President', etc), Company markers ('Ltd', 'AG', etc), Place markers ('Road', 'Bridge', etc), Country and City names ('Israel', 'New York') and common First names ('Benjamin'). This initial process identified nearly 800,000 distinct people names, 160,000 institution names and around 75,000 company names. Random sample analysis of the results suggest that the name identification precision and recall are in the region of 0.92 and 0.88 respectively.

A second phase in the process was to automatically group together similarly named people into equivalence classes. On the basis of probable gender, nickname information (e.g. Bill$\Longleftrightarrow$William), middle-initial (where present), title similarity (e.g. Gen$\Longleftrightarrow$General, but not Actress$\Longleftrightarrow$Congressman) and simple spelling mistakes/deviations in long names ('Gennady Zyuganov'$\Longleftrightarrow$'Gennadi Zyugavov') the 800,000 people names were clustered down to 674,000 distinct individuals. 25 distinct instances of Bill Clinton's name were found to be equivalent, ranging from "Billy Boy Clinton" to "William Milhous Clinton", with the titles "President", "Governor", "Gov" and (incorrectly) "Senator".

One beneficial side affect of this approach was the amount of 'extra' information that was gathered about individuals on the basis of frequency of associated title and country keywords. Without further analysis of the corpus, the list of named people already contains the correct country of origin and main title/profession of at least 30,000 members.

The ultimate aim in this work is to arrive at an improved indexing of the corpus in which every distinct variation on a person or company name is recognised as the correct unique equivalence class. For this year's TREC, however, the process was not fully developed and so only a simple query expansion was implented for the MadCow system, as described in Sec. 2.1.2.

2.4.2 Lexicon Improvement for Part-of-Speech Tagging

During development we noticed that there were a large number of word types (over 180,000) found in the AQUAINT corpus which were not in the lexicon used by our part-of-speech tagger. As a majority of the further processing relies, at least in part, on the POS tags assigned to the words any improvement should be beneficial to the system as a whole as well as improving the tagger for use in other projects.

The approach taken to deal with these unknown words was as follows:

- The entire AQUAINT corpus was tokenized and sentence boundaries were determined.
- The tagger was adapted to tag unknown words with UNK instead of a default tag, and then the tagger was run over the sentence split corpus.
- All non-upper case words which were tagged as UNK were then extracted from the corpus along with their frequency of occurance.
- The POS tag for these unknown words was determined from the tag assigned in the British National Corpus (BNC). Unfortunately the BNC uses a different tag set to the tagger and so the tags had to be mapped to PTB tags using a mapping derived by Steve Abney (1997).

The above approach provides tags for 63633 of the 183648 tokens previously assigned UNK and doubles the size of the lexicon used by the POS tagger.

2.5 Miscellaneous Further Improvements

2.5.1 Improved Named Entity Recognition

Examination of questions used in the previous TREC question answering evaluations, as well as questions from various freely available sources on the web, showed that the answers to some questions can be drawn from a closed set of possibilities. For example the question ”What is the state flower of Hawaii?” should have a flower as the answer, however, we can reduce this to the set of known state flowers (which is finite) increasing the chance that a question answering system will choose the right flower as answer. To this end numerous gazetteer lists of (almost) closed sets of entities were built including: languages, birthstones, Greek, Roman and Egyptian Gods, national anthems, planets, spacecraft, and state flowers, birds, mottos, trees and nicknames

Further examination of our existing named entity recogniser showed that it provided very little support for recognising measurements (especially compound measurements such as speed - meters per second). This was rectified and the system can now recognise measurements of distance, mass, time, speed, temperature and currency plus combinations of these (e.g. 6 foot 6 inches, ten meters per second).

Combining these improvements with those of the previous section results in named entity recognition that is better suited to answering questions especially via the AQUAINT corpus.

2.5.2 Updates to QA-LaSIE

The only major change from the version of QA-LaSIE described in (Greenwood et al., 2002) was the way in which questions requiring multiple answers are handled. The list questions asked in TREC 2002 explicitly stated the number of answers expected whereas the questions used for this evaluation do not. As QA-LaSIE draws its answers from two locations, AQUAINT and the web, using Google, it simply returns the overlap between the two sets of answers. Unfortunately this can lead to the situation in which no answers are returned for a question, but the guidelines for the evaluation state that at least one answer must be given for each list question. In these instances we return the string UNKNOWN ANSWER so as to abide by the guidelines while still allowing us to easily analyse the output from the system.

3 Final Evaluation Results

From the development and experimentation detailed above, we configured three evaluation runs as follows.

shef12okapi This run consisted of using Okapi to retrieve the relevant documents (Section 2.1.1) and then using either QA-LaSIE, with the extended named entity transducer and gazetteer lists, (Sections 2.4.1 to 2.5.2) or the definition system (Section 2.3) to extract the possible answers.

shef12madcow This run is identical to shef12okapi other than the MadCow boolean search engine (Section 2.1.2) was used to retrieve the relevant documents instead of Okapi.

shef12simple This run consisted of using Okapi to retrieve the relevant documents (Section 2.1.1) and then using either the simple baseline system (Section 2.2) or the definition system (Section 2.3) to locate the possible answers.

The results from these three runs can be seen in Table 2. We discuss aspects of each in turn.

3.1 shef12okapi

Of the 413 factoid questions Okapi was able to find answer bearing passages[3] for 198 of them. As 30 of the questions have NIL as the correct answer then the system should be able to answer these as well giving a maximum attainable score of 0.552 (228/413). Unfortunately the official score for this run was only 0.046 (19/413). Part of the reason for such a low score is the way in which the system selects the answer text, always favouring the longest realisation, which causes quite a few correct answers to be classed as inexact – in this case 27 answer were marked as inexact.

For definition questions the system retrieved response sets from the AQUAINT collection for 28 of the 50 definition questions. Of these 28 questions, 22 contained at least one definition nugget all of which contained at least one essential nugget giving a score of 0.230.

3.2 shef12madcow

Of the 413 factoid questions MadCow was able to find answer bearing passages for 173 of them. As 30 of the questions have NIL as the correct answer then the system should be able to answer these as well giving a maximum attainable score of 0.492 (203/413). Unfortunately the official score for this run was only 0.063 (26/413). This run suffers the same problem as shef12okapi with 32 answers being marked as inexact.

For definition questions, the system retrieved response sets from the AQUAINT collection for 28 of the 50 definition questions. Of these 28 questions, 20 contained at least one definition nugget all but one of which contained at least one essential nugget.

3.3 shef12simple

If we analyse the output of the intermediate stages (question typing and information retrieval) then we can

[3] A passage is classed as answer bearing if it comes from a document known to contain the answer and one of the Perl patterns, kindly supplied by Ken Litkowski, matches the text.

Run Tag	*Factoid*	*List*	*Definition*	*Combined*
shef12madcow	0.063	0.015	0.171	0.078
shef12okapi	0.046	0.033	0.230	0.089
shef12simple	0.138	0.029	0.236	0.135

Table 2: Summary results from our three main task entries.

arrive at a maximum attainable score for the final answer extraction stage.

Of the 413 factoid questions the first stage assigned an incorrect type to 53 questions, 27 of these were typed as UNKNOWN so only 26 answers of the wrong type could actually be returned. In total 146 of the questions were assigned the UNKNOWN type, so 267 questions were typed – 241 correctly. The system returned NIL for 191 of the questions so the system was unable to find an answer for 45 of the typed questions. Of the remaining 241 questions 18 have no known answer leaving 223 documents to which the system could return a correct non-NIL answer.

Unfortunately the IR stage was only able to locate answer bearing passages for 131 of the 223 questions correctly processed by the previous stage which means that the maximum obtainable score for the whole system is 0.317 (131/413). The official score for this run is 0.138 (57/413) but this contains fifteen correct NIL responses so we only provided a correct answer for 42 questions giving a score of 0.102 which is 32.2% of the maximum score.

As the answers proposed by this system are named entities, very few are marked as inexact (six are marked as inexact of which only two are missing information) so counting these as correct answers does very little to the overall score increasing it to only 0.153 (63/413).

Of the three runs we submitted this run found the most distinct answers for the list questions – 20 distinct answers compared with 12 for shef12okapi and 7 for shef12madcow. Unfortuntaely the ability of the system to locate many distinct answers was offset by the fact that for each question many answers are proposed dramatically lowering the precision scores and hence the average F score for the run. For example there are seven known answers to question 2346, *"What countries have won the men's World Cup for soccer?"* for which this run returned 32 answers only two of which were correct giving a recall of 0.286 but a precision of only 0.062. Clearly more work needs to be done to limit the number of answers returned for questions of this type.

Although the score given in Table 2 for the definition section of this run is slightly different from that for the shef12okapi run they are actually identical submissions so for more details refer to Section 3.1.

4 Conclusions and Future Work

In comparison to our results from previous years, the performance scores achieved by the three Sheffield systems this year are somewhat disappointing, although it is difficult to have a clear view of their merit without knowing how other systems have also performed this year. Even so, our experiments have thrown up some interesting results, which are suggestive of valuable directions for future work, and we shall close with some comments in this regard.

Firstly, it is striking (perhaps even distressing) to observe that the 'baseline' shef12simple system has performed better overall, and on factoid questions in particular, than either shef12madcow or shef12okapi, the systems which incorporate QA-LaSIE. Looking more closely at the answers produced by the systems, we have found that, in contrast to the short named entity expressions that shef12simple typically returns, QA-LaSIE tends to return longer answers, corresponding to the longest amongst the alternative descriptions of the same entity brought together by co-reference resolution. A consequence of this has been that many more of the answers produced by QA-LaSIE have been judged inexact than for shef12simple. If we ignore the distinction between answers judged inexact and correct, the performance of the three systems come much more closely into line, and so we expect that the actual performance of the shef12madcow or shef12okapi systems could be improved by modifying QA-LaSIE's behaviour in terms of its preference for long or short answers. Even given these observations, the simple system has performed sufficiently well that the approach appears worthy of further development/refinement.

Secondly, in regard to the issue of retrieval models, the results obtained using MadCow are better than using Okapi, even given the highly preliminary version of query formulation used. Interestingly, the performance with MadCow is better despite the fact that it returns answer bearing passages for fewer of the questions than Okapi. This result may relate to the fact that MadCow passages are always single sentences, so that the volume of irrelevant text presented to the answer extraction system as 'noise' may be less. The results suggest that boolean retrieval is a worthwhile direction for further research, both in relation to better question analysis and query formulation, and also regarding what is the best size of passage for indexation and retrieval.

Thirdly, the results for the definition system look promising. In future work, we hope to address the following: general improvement of the definition patterns; relaxation of the filters used for the identification of answer-bearing passages; and development of a syntactic-based technique that prunes a parse tree in order to extract definition phases from answer-bearing sentences.

Finally, the work on automatically identifying name expressions within AQUAINT, and gathering these expressions into equivalence classes, has considerably improved our capabilities for processing name expressions. So far, the results of this work have only been used in the query formulation process for boolean retrieval. However, using this approach, we could instead seek to pre-identify all named entities within AQUAINT, and fold the results of this analysis into the inverted index created for boolean retrieval. Then, for example, given a *who* question, we might restrict retrieval to address only those sentences which are known in advance to contain a `person` entity. Likewise, having identified a name such as *President Clinton* in the question, we might use the index to directly access all sentences known to contain an occurrence of any variant of this name, rather than just searching for sentences that contain the words of this name and its variants,

References

Steve Abney, 1997. *The SCOL Manual*. University of Tübingen.

Eric Brill, Jimmy Lin, Michele Banko, Susan Dumais, and Andrew Ng. 2001. Data-Intensive Question Answering. In *Proceedings of the Tenth Text REtrieval Conference*.

Mark A. Greenwood, Ian Roberts, and Robert Gaizauskas. 2002. The University of Sheffield TREC 2002 Q&A System. In *Proceedings of the 11th Text REtrieval Conference*.

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Michael Junk, and Chin-Yew Lin. 2000. Question Answering in Webclopedia. In *Proceedings of the 9th Text REtrieval Conference*.

Kevin Humphreys, Robert Gaizauskas, Mark Hepple, and Mark Sanderson. 1999. University of Sheffield TREC-8 Q & A System. In *Proceedings of the 8th Text REtrieval Conference*.

Julian Kupiec. 1993. MURAX: A Robust Linguistic Approach for Question Answering Using an On-Line Encyclopedia. In *Research and Development in Information Retrieval*, pages 181–190.

George A. Miller. 1995. WordNet: A Lexical Database. *Communications of the ACM*, 38(11):39–41, November.

Ian Roberts and Robert Gaizauskas. 2004. Evaluating passage retrieval approaches for question answering. In *Proceedings of 26th European Conference on Information Retrieval*. To Appear.

S. Robertson and S. Walker. 1999. Okapi/Keenbow at TREC-8. In *Proceedings of the 8th Text REtrieval Conference*.

Sam Scott and Robert Gaizauskas. 2000. University of Sheffield TREC-9 Q & A System. In *Proceedings of the 9th Text REtrieval Conference*.

Towards a Sense Based Document Representation for Internet Information Retrieval

Christopher Stokoe
University of Sunderland
Informatics Centre
St Peters Campus
+44 (0)191 515 3291
Christopher.Stokoe@sund.ac.uk

John Tait
University of Sunderland
Informatics Centre
St Peters Campus
+44 (0)191 515 2712
John.Tait@sund.ac.uk

Abstract

We describe an attempt to use word sense as an alternate text representation within an information retrieval system in order to enhance retrieval effectiveness. A performance comparison between a term and sense based system was carried out indicating increased retrieval effectiveness using a sense based representation. These increases come about by using a retrieval strategy designed to down rank documents containing query terms identified as being used in an infrequent sense.

1. Introduction.

Lexical ambiguity has long been considered as having a negative impact on the performance of information retrieval (IR) systems. Despite a number of studies [10,5,8,7,9] into ambiguity and IR to date only two have demonstrated significant performance increases. In the first, Shütze and Pederson [8] used the computationally expensive approach of clustering co-occurrences within the collection. Each of the clusters in which a given word was found was considered a unique "word use" with each word use arguably representing an individual sense of the word. The second study by Stokoe, Oakes and Tait [9] took advantage of the skewed frequency distribution in test collections observed by Krovetz and Croft[5] to create a sense based retrieval strategy that down-ranked documents which contained infrequent senses of a word. An additional property of this approach was that in cases of inaccurate disambiguation the technique degrades gracefully to at worst the baseline performance of a term model.

One perceived failing of both of these studies was their evaluation setting. Both showed a comparable performance increase when contrasting TF*IDF ranking with those achieved using sense frequency (SF*IDF). Although this is a first step to demonstrating the worth of a sense based document representation it is clear that most modern information systems rely on a combination of techniques to assign rank. This is most clearly demonstrated when we compare the performance of the Stokoe, Oakes and Tait work against the Web Track submissions for TREC 9. A baseline ranking produced using only TF*IDF was considerably below the average performance achieved by systems at the evaluation. Given this we must question whether the performance increases demonstrated by this technique are simply an artefact of the low performance of the baseline retrieval method.

2. Hypothesis.

Given that, at a low level we see a sense based representation outperforming a term based model, it is our belief that this increased performance can carry over to a modern web retrieval system. In order to demonstrate this we undertook to construct a "full featured" term based topic distillation system and to compare its performance against an identical system which used a sense based model. Our aims were as follows:

1) Produce a term based system with average or above performance.
2) Produce a corresponding sense based system.
3) Compare and contrast the performance of the term based vs. sense based system.

For our disambiguation we used the Sunderland University Disambiguation System (SUDS) running in the configuration described in Stokoe, Oakes and Tait for comparability. In terms of the topic distillation techniques to be used we selected a number of common ranking algorithms that were utilised in the 2002 evaluation.

3. Experimental Methodology.

All the experimental work was carried out using a 1 GHz Pentium 3 with 398Mb of memory running Linux. All documents were striped of their headers and HTML tags and an initial term based inverted index of the .GOV collection was produced reducing the collection from 21GB to 5.3GB on disk. Total processing time for the production of this index was 11hrs 23 mins. The full text of each document in the collection was also made available for subsequent processing. For each query all the documents containing the query terms were identified from the index and were then subsequently ranked in accordance with our retrieval strategy. These same documents were subsequently disambiguated and re-ranked using a sense based representation.

4. Automated Disambiguation.

The disambiguation system we used (SUDS) is based on a statistical language model constructed from the manually sense tagged Brown1 part of the Semcor corpus. Briefly, it uses a statistical analysis of collocation, co-occurrence and occurrence frequency in order to assign sense. A more detailed explanation of SUDS can be found in Stokoe, Oakes and Tait. SUDS normally work's using a context window consisting of the sentence encapsulating the target word. However in cases where this information is not available E.g. queries, SUDS relies on occurrence frequency stats to perform sense tagging. Therefore one of the underlying assumptions behind SUDS use in IR is that query terms will rarely be seen as examples of a term being used in an infrequent sense.

SUDS overall accuracy is reported at 62.1% when evaluated using the Brown2 part of SemCor, this is representative of the current state of the art systems[2]. However more notably it outperforms bare frequency tagging by 8.2%. Given that frequency only tagging treat's all terms in the collection as being non-polysemous, and in turn represents the best possible accuracy you could achieve by ignoring sense. This indicates that SUDS can provide a more accurate representation of a collection than simply ignoring sense given that it is more accurate than frequency only tagging.

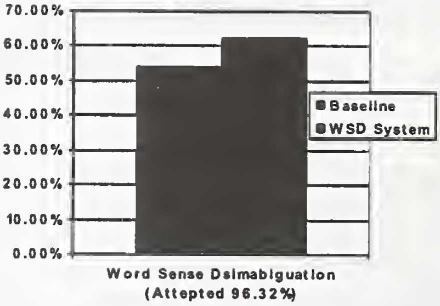


Figure 1: Comparison between the precision of our WSD algorithm compared to baseline frequency

5. Topic Distillation.

Our strategy for topic distillation was based on the increasingly popular link analysis theory initially proposed by Kleinberg [3]. This approach uses the notion that a key resource can either be an authority or a hub. Authorities are considered to be highly relevant documents of the type often in-linked by hubs which are inversely documents that contain significant out-links to authorities. Kleinberg proposed that by exploring the link topology of the WWW using a connectivity analysis mechanism one could make inferences on the relevance of a given document based on its linkages. Despite a number of key studies into the performance of link analysis as a ranking mechanism there remain some questions as to its effectiveness. In general the technique has demonstrated comparable performance to traditional statistical retrieval models. However in some cases reduced performance has occurred. These performance drops are generally perceived to be as a result of evaluation using a static collection with a high number of documents that contain out-links to documents not contained in the collection.

Given that traditional statistical ranking had performed reasonable favourably at TREC 2002 [1] we used a combination of ranking algorithms to identify authorities. The linkages between documents in our result set were then analysed in order to inflate the rankings of hubs by identifying those pages that had a significant number of out-links to other pages that were judged relevant by our system. Although we collected information about in-links to a given document this was not used in our eventual ranking algorithm as we were unable to identify a way to use it which demonstrated increased retrieval effectiveness. Additionally our system tracked the number of pages form each unique domain which appeared in our final rankings allowing us to manipulate the number of results from each site that the system returned. This was eventually used in the "UNIQUE" runs in order to evaluate system performance where only the highest ranking page from a given site was returned. This was a common strategy at TREC 2002 where several groups [1] manipulated the number of unique sites that appeared in the top 10 retrieved results. Our sense based retrieval experiments followed exactly the same algorithm (see section 6) however terms were replaced by WordNet sense tags. Therefore each of our sense-based runs has a corresponding term-based baseline for comparison.

6. Retrieval Algorithm.

Our retrieval strategy used a number of common techniques associated with the vector space model of retrieval. Each query was stop worded to leave just the content terms and for each document which contained one or more of these terms we calculated an authority_rank using the following features:

1) TF*IDF

Using the well known ranking algorithm (1) presented by Salton and McGill[6]. With rank being assigned based on the sum of the weights of each term in the query.

$$\begin{aligned} W(w_i, d) &= TF(w_i, d) * IDF(w_i) \\ &= N(W_{id}) * LOG\left({}^{Nf}\!\big/_{Nf(w_i)} \right) \end{aligned} \quad (1)$$

2) Cosine similarity (title)

A vector based comparison of the document title against the query. This was carried out using a cosine similarity measure (2). As seen in Salton and McGill [6].

$$\sigma(D,Q) = \frac{\sum_k (t_k \times q_k)}{\sqrt{\sum_k (t_k)^2} \times \sqrt{\sum_k (q_k)^2}} \quad (2)$$

3) Cosine similarity (body)

A vector based comparison of the document body and the query carried out using the similarity algorithm (2) we previously used on the document title.

4) Boolean Weighting

A weighting modifier applied based on testing whether a document is binary 'AND' complete for a given query. I.e. contains all of the content terms.

The rank assigned by each of these techniques was then normalised between 0..1 using max / min normalisation. Table 1 shows the weightings used in the max / min combination, added bias was given to documents that contained query terms in their *<title></title>* tags.

Feature	Weight
TF*IDF	1
TITLE_SIM	2
BODY_SIM	1
Boolean	1

Table 1: Weighting bias applied to each technique when merging the rankings using max / min combination.

Additionally for each document we calculated a hub_rank based on the sum of the authority_rank's assigned to any outward links contained in that document. In order to calculate the final page rank the hub_rank for a document was normalised and added to the authority_rank.

7. Results.

We submitted a total of four topic distillation runs for evaluation. Runtags beginning with SB indicate sense based runs while those beginning with TB indicate term based ones. Firstly if we examine the performance of the baseline term based runs (Table 2) we can see that TBBASE outperformed TBUNIQUE with regard to P@10. This demonstrates that returning only the highest ranked document from each website reduces overall system performance. Given that one key feature of topic distillation has always been to identify a suitable entry point to a relevant website it is interesting to note that our system gains performance when returning multiple pages from the same site. On several occasions our

Run Tag	R-Precision	Avg. Precision	P@10
TBBASE	0.1333	0.1166	0.1020
TBUNIQUE	0.1278	0.0948	0.0880
SBBASE	0.1283	0.1259	0.1020
SBUNIQUE	0.1407	0.1114	0.0940

Table 2: R-Precision, Avg. Precision, and Precision @ 10 for all runs.

TBBASE run demonstrates increased precision @10 by returning multiple ranked relevant pages from a single domain. If we consider the performance (precision @ 10) of our best baseline run compared with the median of all runs submitted to the evaluation (Figure 2) we can see that this run has average or above performance on all but five of the fifty topics. In addition we showed above average performance on thirteen of the topics.

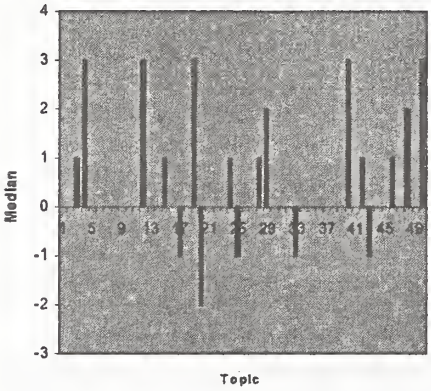


Figure 2: Difference from Median in Precision @ 10 per Topic.

Having established that the TBBASE run is representative of the current state of the art we can contrast its performance with the equivalent sense based run (SBBASE). The graph in Figure 3 shows the precision of both the TBBASE and SBBASE runs plotted for the 11 standard points of recall. We can see that the sense based run demonstrates increased precision compared with the term only model particularly in the mid-recall range. In addition if we compare the average precision of both runs 0.1166 (TBBASE) and 0.1259 (SBBASE) we note a small increase when using sense's rather than terms. There is an insignificant increase in Precision @ 5 (0.004) achieved using senses however the term and sense models performed identically when we compare the official measure of precision @ 10.

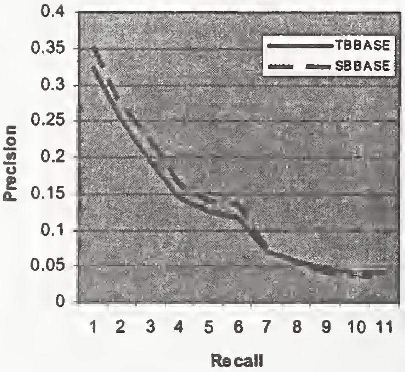


Figure 3: Precision – Recall for TBBASE and SBBASE @ 11 Standard Points of Recall.

8. Conclusion.

The main aim of our participation in TREC was to assess whether automated word sense disambiguation could be used to improve retrieval effectiveness. We developed a combination topic distillation system that used several traditional techniques and merged their rankings. A comparison between our term based system and the median of all runs in the Web Track topic distillation stream was performed. This demonstrated that our system was representative of average system performance levels seen at the evaluation. A comparison between the performance of our algorithm using a term and sense based representation was performed. The results of this evaluation demonstrate a clear performance gain from using sense information to represent documents. Although our sense based system failed to demonstrate increased precision @ 10 significant gains in recall were made without a corresponding drop in precision. In addition we do see increased avg. precision using a sense based representation and increased R-Precision. This increase was notable given that it was achieved using a disambiguation algorithm that has significantly lower levels of accuracy than those commonly associated with humans who perform the same task. One possible explanation for the success of this approach is that it exploits the skewed frequency distribution known to exist in large collections of natural language. The work also offers anecdotal evidence to suggest there is a bias towards queries using polysemous terms in a frequently observed sense.

9. References.

[1] Craswell, N; Hawking, D. "Overview of the TREC 2002 Web Track" in proceedings of the Eleventh Text Retrieval Conference, TREC 2002.

[2] Edmounds, P; Cotton, S. 2002 "SENSEVAL-2: Overview" in proceedings of the Second International workshop on Evaluating Word Sense Disambiguation Systems.

[3] Kleinberg, J.M."Hubs, authorities, and communities" ACM Computing Surveys, 31(4es), 1999.

[4] Korfhage, R. "Information Storage and Retrieval" Wiley, New York, 1997.

[5] Krovetz, R; Croft, W. B. 1992. "Lexical Ambiguity and Information Retrieval" in ACM Transactions on Information Systems Vol 10 Issue 1.

[6] Salton G; McGill, M.J. 1983 "Introduction to Modern Information Retrieval" New York: McGraw & Hill.

[7] Sanderson, M. 2000. "Retrieving with good sense" in Information Retrieval Vol 2, No 1 Pp 49 – 69.

[8] Shütze, H; Pederson, J. O. 1995 "Information Retrieval Based on Word Senses" in proceedings of the Symposium on Document Analysis and Information Retrieval 4 Pp 161 -175.

[9] Stokoe, C; Oakes, M. P; Tait, J. 2003 "Word sense disambiguation in information retrieval revisited." in proceedings of ACM SIGIR Conference (26) Pp 159-166

[10] Voorehees, E. M. (1993). "Using WordNet to disambiguate word sense for text retrieval" in proceedings of ACM SIGIR Conference (16): Pp 171-180.

TREC 2003 Genomics Track Experiments at UTA: Query Expansion with Predefined High Frequency Terms

Ari Pirkola and Erkka Leppänen
University of Tampere (UTA), Finland
Department of Information Studies
pirkola@cc.jyu.fi

Abstract We studied the effects of query expansion and query structure on retrieval performance. Two sets of words frequent in relevant documents for Genomics Track's training topics were collected, the first manually and the second automatically. The high frequency words collected and the names of organisms designated in the test topics, were used as expansion keys in gene name queries formed from the final test topics. The results indicated that Boolean structured queries expanded with automatically collected high frequency words and names of organisms performed considerably better than queries containing gene names only as keys. In the Boolean queries the expansion keys were categorized based on the aspects they represent in the documents discussing gene function. All the structured queries performed better than unstructured queries where each key contributed equally to document weights. In the structured queries gene names were assigned more weight than the expansion keys.

1. Introduction

TREC Genomics Track's primary task was defined as follows: *For gene X, find all Medline references that focus on the basic biology of the gene or its protein products from the designated organism. Basic biology includes isolation, structure, genetics and function of genes/proteins in normal and disease states.* Documents for which there were GeneRIFs were considered relevant. As the focus of GeneRIFs is to provide information on the functions of genes (and proteins), basically documents discussing gene function were searched for in the primary task.

In this research we studied, first, which types of words other than gene names are useful in searching for documents for the given task and, second, how queries using these words should be formulated for the best possible retrieval performance (the question of query structure). For the first research problem we identified two sets of words (manual and automatic identification) whose frequencies were higher in documents relevant for Genomics Track's *training topics* than in the rest of the documents in the Medline test collection. The power of these words to actually discriminate between gene function and other documents was evaluated in experiments where the words were used as expansion keys in the final test queries. The performance of the expanded queries was compared to the performance of baseline queries containing gene (and protein) names only as query keys. For the runs that were submitted to NIST the expansion keys were selected manually. Most of the runs we did were additional runs, and for these the expansion keys were selected automatically. In the additional tests names of organisms were also used as expansion keys.

The second research problem considered the effects of query structure on retrieval performance. The test system of this study was the *InQuery* retrieval system, a probabilistic retrieval system. In InQuery queries can be structured using a variety of query operators. The assumption was that for good performance gene names should be weighted more than other keys, and that in addition to gene names other aspects of relevant documents should also be included in queries.

These factors were taken into account in the *Boolean structured queries* tested in the study. The results showed that performance was improved considerably when gene name queries were expanded with names of organisms. The best queries, however, were extensively expanded Boolean structured queries where gene names were weighted structurally more than the other keys, and where expansion keys were categorized based on the aspects they represent in gene function documents. The categories were: (1) names of organisms (given in the test topics), (2) high frequency special terms related to genes and proteins, and (3) high frequency general biological terms. Expansion key categories 2 and 3 were selected automatically from documents relevant for Genomics Track's training topics. The Boolean structured queries performed considerably better than queries containing gene names only as keys (baseline) and unstructured queries. In the unstructured queries gene names and expansion keys were weighted equally.

2. Methods and data

2.1 Indexing and query keys

We used the following approaches and techniques in indexing and for query keys:

- Genomics Track's test collection for the primary task was a subset of the Medline collection. Each record consisted of several fields. We indexed the text fields TI (title), AB (abstract), and MH (MeSH headings). (And PMID, PubMed Identifier.)
- Letters were normalized to lower case.
- Query keys and the words of documents were normalized using the morphological analyzer *Kstem*, which is part of InQuery.
- Only letters (a-z) and numbers (0-9) were indexed. Hyphens, slashes and other characters in strings than letters and numbers were replaced by spaces, and were not searchable.
- Strings containing both letters and numbers were decomposed into separate alphabetical and numerical strings. For example, the string *AW986256* was converted into *aw 986256*. In searching the decomposed strings were combined by means of a proximity operator of #od4. For the string *AW986256* the proximity statement was as *#od4(aw 986256).*

2.2 Retrieval system and query operators

The test system was the *InQuery* retrieval system (Allan et al., 2000; Broglio et al., 1994). InQuery is a probabilistic retrieval system based on the Bayesian inference net model. Queries can be presented as a bag of word queries, or they can be structured using a variety of query operators. In this study we considered structured queries. For the structured queries high frequency words were selected as expansion keys (Section 2.3). The expansion keys were grouped into categories (subqueries) based on the aspects they represent in documents. In these types of queries different aspects of documents discussing the functions of genes contribute to retrieval. Query operators were applied in such a way that more weight was assigned to gene names than to other keys.

Boolean conjunction ("and") formed the basis of queries. In InQuery Boolean conjunction is denoted using the operators of #band or #filreq (and for phrases #odn, see below). All the argument keys of the *#band-operator* must occur in a document in order for the operator to contribute to the weight computed for that document. Otherwise #band contributes to the document score like the #and-operator. For the #band-operator, see Pirkola and Järvelin (2001). The weight of the *#and-operator* is computed as the product of the weights of its arguments. The syntax of *#filreq* (filter require) operator is #filreq(arg1 arg2). Documents for the first argument are returned and ranked if and only if the second argument would return documents.

Other InQuery operators used in the queries of this study were #sum, #syn, #odn, and #phrase. For the *#sum-operator*, the system computes an average weight of query keys (subqueries). The *#syn-operator* treats its operand query keys as instances of the same query key. The benefits of combining keys by the #syn-operator is discussed in Darwish and Oard (2003). Phrases were searched for using the proximity operators of #odn and #phrase. The *#odn*-operator is a Boolean conjunction operator. The operator retrieves documents where all the arguments of the operator appear in a specified order. The window size *n* refers to the number of spaces between words in the text. In the queries of this study n=4 was used as a window size. The *#phrase*-operator is treated as #odn-operator, if the terms within the phrase-operator occur together in a collection. If the terms do not co-occur in the collection the phrase operator is turned into a #sum-operator.

2.3 Selecting expansion keys

We tested in this study whether words typical in gene function documents are helpful for the given retrieval task when used as expansion keys in gene name queries. Two lists of expansion keys were collected, the first manually and the second automatically. The basic assumption behind this approach was that those words whose frequencies were high in relevant documents for the Genomics Track training topics with respect to their frequencies in other Medline documents are good discriminators and good keys in queries, designating generally documents discussing gene function.

In manual word selection a list of words was collected that in an intellectual analysis seemed to be frequent in the documents relevant for the training topics. The words were divided into two categories: (a) *gene function terms*, and (b) other high frequency terms, for example, *cell*, *mrna*, and *sequence*. The latter words are called *general terms* in Section 3.1. The two categories formed two different subqueries in the structured queries.

The automatic expansion key selection was done as follows. For each training topic at most two relevant documents were chosen for statistical analysis. The total number of 91 relevant documents and approximately 24 000 words were analysed. The same number of randomly selected documents (approximately the same number of words) was analysed. In both cases words were normalized using the Kstem morphological analyser, and stopwords were removed. For each remaining word the frequency of a word was computed. Those words whose frequencies were low (freq. < 10) were removed.

A word i was chosen as an expansion key based on the following criteria:

(1) Its term frequency in the relevant documents ($tf_{i, rel}$) was higher than a given threshold; the threshold was $tf_{i, rel} = 30$ (in other words, word's term frequency on the average was higher than 30/91).

(2) Its relative frequency was higher than a given threshold. The relative frequency of word i is its frequency in the relevant documents ($tf_{i, rel}$) divided by its frequency in the random documents ($tf_{i, ran}$). Three thresholds were tested:

$tf_{i, rel} / tf_{i, ran} = 3.1$	(No of expansion keys = 32)
$tf_{i, rel} / tf_{i, ran} = 3.7$	(No of expansion keys = 20)
$tf_{i, rel} / tf_{i, ran} = 4.6$	(No of expansion keys = 11)

The numbers in parentheses show the number of words kept for expansion as the threshold was applied.

The automatically selected word list with the threshold 3.1 is presented in the Appendix.

In both cases (manual and automatic selection of expansion keys) the selected words were added to gene name queries (baseline) to yield expanded queries. The performance of the expanded queries was compared to that of baseline queries. Manually collected expansion keys were used in the runs submitted to NIST while the automatically selected words were used in the additional runs.

3. Experiments

In summary, we tested in this study the effects of (1) manually and (2) automatically selected high frequency words as expansion keys, and (3) different query structures. Different types of expansion keys were put into different subqueries in queries. In the experiments 2 and 3 we also tested the effects of organism name keys. Next, the query types used in the three experiments are presented.

3.1 Manually selected expansion keys

Baseline

The baseline queries were formulated on the basis of Genomics Track's test topics, and they contained all the gene and protein names occurring in the topics.

Phrases were searched for using the proximity operator of #od4, e.g., *#od4(hematopoietic growth factor)*. As there is much variation how phrasal expressions are written in texts, phrase components were also repeated as single keys in queries, and for phrases consisting of more than 3 phrase components several proximity statements were constructed, with consecutive phrase components appearing in 1-3 proximity statements, as in the example below:

```
#od4(camp responsive element) #od4(responsive element binding) #od4(element binding protein)
#od4(binding protein 2)
```

An example of a baseline query is presented below (query 31):

```
#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin releasing
hormone receptor)
```

Test queries (runs submitted to NIST)

In the training experiments we tested various combinations of query operators. The most effective query types turned out to be those based on InQuery's *band*- and *filreq*-operators. They were named *utaband* and *utafil*. In the *utaband* queries in addition to the #od4-operator the #phrase-operator was used for phrases. The utaband queries contained 74 manually selected expansion keys, and the utafil queries 58 expansion keys. The structure of *utaband* is schematically as follows:

```
#band(
#and(#sum(gene names) #syn( gene function terms))
#and (#sum(gene names) #syn(general terms))
)
```

The structure of *utafil* is schematically as follows:

```
#filreq(
#sum(gene names #syn(gene function terms) #syn(general  terms))
#syn(gene names [as single keys])
)
```

The last #syn-subquery functions as a filter, basically being a Boolean conjunction restriction.

3.2 Automatically selected expansion keys, and organism names as expansion keys

Baseline

The baseline was the same as in the experiments presented in Section 3.1.

Test queries

The expansion key categories were as follows:

1. Organism names (*on*) designated in the Genomics Track's test topics. However, instead of using the Latin names used in the test topics, we used the MeSH terms *human, drosophila, rats* and *mice.*

2. Automatically selected high frequency terms related to genes and proteins. These are called *special terms* below.

3. Automatically selected high frequency general biological terms, i.e., other terms than those in the cases 1 and 2 frequent in documents discussing the functions of genes. These are called *general terms* below.

Obviously, the boundary between the categories 2 and 3 is diffuse, in some cases causing classification difficulties.

All the special and general terms collected are shown in the example queries below and in the Appendix.

In queries the terms of the category 2, as well as the terms of the the category 3, were combined by the #syn-operator to yield a #syn-subquery. The subqueries were combined with each other as follows (*gn* refers to gene names):

- gn (baseline)
- gn + on (see Section 3.3)
- gn + general terms
- gn + special terms
- gn + on + general terms
- gn + on + special terms
- gn + on + general + special terms

Query 31 with 32 expansion keys of the type *gn + on + general + special* below provides an example of the queries presented in this section:

#band(#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin releasing hormone receptor) rats #syn(cytology genetic membrane cell metabolism region site level receptor) #syn(amino alpha beta kinase tyrosine protein gene mutant mutation sequence activate expressing activation apoptosis binding expression regulate transcription induce pathway signal domain function))

The first #syn-subquery includes the general terms while the second #syn-subquery includes the special terms.

We call the query structure above *#band/#syn structure*, and it was used in all queries presented in this section, only the expansion keys were different in different types of queries. As can be seen in the example, in the #band/#syn structure the expansion keys of a certain category are combined with the #syn-operator.

The gene names are combined with the #sum-operator. The #sum-subquery, the #syn-subqueries, and organism name are combined with the #band-operator.

3.3 Query structures

In the query structure experiments, we studied #band/#syn structured (presented in Section 3.2), #and/#band structured, and unstructured queries. In this section we present the last two query types. The #and/#band structure gave good results in the training tests, and was chosen for the final tests.

The #and/#band structure is schematically as follows:

```
#and(
#band(#sum(gene names) #syn(general terms))
#band(#sum(gene names) #syn(special terms))
#band(#sum(gene names) organism name)
)
```

The first example below represents a #and/#band structured query, and the second one an unstructured query (query 31 with 32 expansion keys).

#and/#band structured query

#and(#band(#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin releasing hormone receptor) #syn(cytology genetic membrane cell metabolism region site level receptor)) #band(#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin releasing hormone receptor) #syn(amino alpha beta kinase tyrosine protein gene mutant mutation sequence activate expressing activation apoptosis binding expression regulate transcription induce pathway signal domain function)) #band(#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin releasing hormone receptor) rats))

Unstructured query

#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin releasing hormone receptor cytology genetic membrane cell metabolism region site level receptor amino alpha beta kinase tyrosine protein gene mutant mutation sequence activate expressing activation apoptosis binding expression regulate transcription induce pathway signal domain function rats)

Moreover, in addition to gn+on/ #band queries (Section 3.2) we tested *gn+on* queries where the gn- and on-subqueries were combined with the #and-operator. Examples of both query types are presented below.

Gn+on / #band

#band(#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin releasing hormone receptor) rats)

Gn+on / #and

#and(#sum(gnrhr #od4(gonadotropin releasing hormone) #od4(releasing hormone receptor) gnrhr gonadotropin releasing hormone receptor) rats)

4. Results

The results of the expansion experiments are presented in Table 1 (manual selection of expansion keys) and Table 2 (automatic selection of expansion keys, and organism names as expansion keys). Table 2 also presents the results of query structure experiments.

Table 1. Retrieval performance of queries with manually selected expansion keys.

Query type	Average precision	% change
Baseline (gn)	0.1896	-
Utaband	0.1927	+1.6
Utafil	0.1931	+1.8

Table 2. Retrieval performance of queries with automatically selected expansion keys, and organism names as expansion keys; the effects of query structure.

Query type	Average precision	% change
Baseline (gn)	0.1896	-
Gn+on / #band	0.2600	+37.1
Gn+on / #and	0.2278	+20.1
Gn+general / #band/#syn	0.2042	+7.7
Gn+ special / #band/#syn	0.1999	+5.4
Gn+on+general / #band/#syn	0.2693	+42.0
Gn+on+special / #band/#syn	0.2606	+37.4
Gn+on+general+special / #band/#syn	0.2648	+39.7
Gn+on+general+special / #and/#band	0.3097	+63.3
Gn+on+general+special / unstructured	0.1692	-10.8

As shown in Table 1 there is only a small difference between the performance of baseline and the performance of queries containing manually selected expansion keys. The difference between *utaband* and *utafil* is also small. It should be noted that the utaband and utafil queries did not contain organism name keys.

The run *gn+on* (organism names as expansion keys) where the two subqueries (gn and on) are combined with the #band-operator performs considerably better than baseline (Table 2). For *gn+on* queries average precision is 0.2600. For the baseline the corresponding number is 0.1896. The performance of *gn+on* is lower in the case of the #and-operator but is still better than baseline's performance.

Table 2 shows the results for the queries where the expansion keys were selected on the basis of the threshold of 3.1 (Section 2.3) as they gave better results than the queries with the thresholds of 3.7 and 4.6.

The use of *special* and *general* terms as expansion keys is useful: the *gn+on+ general+special* / #and/#band queries give the best results among all query types tested in this study (average precision 0.3097). It should be noted that the structure of the *gn+on+general+special* / #and/#band queries and the *gn+on* / #band queries (average precision 0.2600) is the same. In both a gene name subquery is combined to an organism name subquery with the #band-operator. The difference is that the *gn+on* / #band queries do not contain the automatically selected expansion keys.

The unstructured queries containing all the expansion keys perform poorly (Table 2). For them average precision is 0.1692, while the average precision for the #band/#syn and #and/#band queries containing all expansion keys is 0.2648 and 0.3097, respectively.

5. Discussion and conclusions

In this study we explored an approach to concept analysis similar to that used in traditional Boolean searching. Based on the concept analysis the concepts of a request which represent different aspects of the request (conjunctive or restrictive concepts) are combined with the and-operator, and those concepts which represent the same aspect (disjunctive concepts) are combined with the *or*-operator with one another. For experienced users, Boolean and natural language searching have been shown to yield comparable results (Hersh et al., 1998). We analysed documents instead of requests, and formulated Boolean types of queries where terms representing the same aspect were put into the same category (subquery) in a query. The test system of this study was the *InQuery* retrieval system, a probabilistic retrieval system. Probabilistic query operators corresponding to the Boolean *and* and *or* operators were used in the queries. We found that retrieval performance improved remarkably with respect to gene name queries by expanding queries with names of organisms and automatically selected terms whose frequencies were high in gene function documents, and by using Boolean #and/#band structured queries. Extensively expanded queries gave the best results. The benefits of the #and/#band structure are that gene names are weighted more than other keys and that the #and/#band structure captures in a balanced way the different aspects involved in gene function documents. Different documents discussing gene function share the same aspects and the same terms. We identified four aspects (gene names and three aspects for expansion), but this is a crude categorization which should be elaborated and analysed more carefully in terms of biological processes and structures. The results, however, show that it is possible to improve results by using predefined high frequency discriminating words as query keys.

The most effective expansion key type was organism name. The explanation for this seems obvious. Different organisms contain the same genes which are named similarly. Unless organism name is given in a query, documents discussing the gene of interest in another organism may be retrieved.

There are many factors affecting the effectiveness of the expanded queries. Their performance depends in particular on which individual keys are used, the number of keys used, and the way the keys are applied in queries. In this study the latter two questions were investigated. The first is a question for future research. Also, based on these initial experiments on the use of predefined discriminating words as expansion keys, a more sophisticated scheme for the identification of discriminating words might be developed in future research. In addition to the relative term frequency that we used as a basis of the selection of expansion keys, the use of document frequency, for example, in such a scheme might be useful.

Acknowledgements

The Inquery search engine was provided by the Center for Intelligent Information Retrieval at the University of Massachusetts.

This research was supported by the Academy of Finland (funding decision 80142).

References

Allan, J., Connell, M.E., Croft, W.B., Feng, F.-F, Fisher, D. and Li, X. 2000. Inquery and TREC-9. *The Ninth Text REtrieval Conference (TREC-9)*, Gaithesburg, MD. Available at: http://trec.nist.gov/pubs/trec9/t9_proceedings.html

Broglio, J., Callan, J. and Croft, W.B. 1994. Inquery system overview. *Proceedings of the TIPSTER Text Program (Phase I)*, pp. 47-67. San Francisco, CA: Morgan Kaufman Publishers Inc.

Darwish, K. and Oard, D. 2003. Probabilistic structured query methods. *Proceedings ACM SIGIR*, Toronto, Canada, pp. 338-344.

Hersh, W., Price, S., Kraemer, D., Chan, B., Sacherek, L., and Olson, D. 1998. A large-scale comparison of Boolean vs. natural language searching for the TREC-7 Interactive Track. *The Seventh Text REtrieval Conference (TREC-7)*, Gaithesburg, MD. Available at: http://trec.nist.gov/pubs/trec7/t7_proceedings.html

Pirkola, A. and Järvelin, K. 2001. Employing the resolution power of search keys. *Journal of the American Society for Information Science and Technology*, 52(7): 575-583.

Appendix - the automatically selected expansion keys

The automatically selected expansion keys presented below are sorted by the score of word's relative frequency, i.e., word's frequency in a sample of documents relevant for Genomics Track's training topics divided by word's frequency in the same number of randomly selected documents. *Min* after many values show that in that case the real value could be higher (but not lower) than the value shown and used in the experiments as a basis of the selection of expansion keys. This is due to removal of words of low frequency (Section 2.3).

expression	9,3	level	3,9
sequence	9,3 (min)	mutant	3,9 (min)
regulate	9,1 (min)	membrane	3,8 (min)
signal	8,8 (min)	domain	3,7 (min)
induce	7,0 (min)	expressing	3,6 (min)
mutation	5,9	site	3,6 (min)
gene	5,7	receptor	3,6
transcription	5,4 (min)	function	3,5
activation	5,3	genetic	3,3
apoptosis	5,1 (min)	metabolism	3,3
cell	4,6	tyrosine	3,3 (min)
amino	4,5 (min)	beta	3,2
kinase	4,5 (min)	cytology	3,2 (min)
activate	4,4 (min)	region	3,2 (min)
binding	4,2	protein	3,2
pathway	4,1 (min)	alpha	3,1

A method to retrieve papers from MEDLINE: PETER system

Hiroko Ao, Yasunori Yamamoto, Toshihisa Takagi
aohiroko@ims.u-tokyo.ac.jp, yayamamo@ims.u-tokyo.ac.jp, tt@cb.k.u-tokyo.ac.jp
Dept. of Computational Biology, University of Tokyo

We attempted to eliminate non-relevant papers from results of PubMed searches for each topic. The system is called PETER (PubMed Enhancer Toward Efficient Research) and it works as follows.

1. get LocusLink IDs manually.
2. collect information of gene names (AKA synonyms) from public databases.
3. make synonym variations automatically.
4. search papers by PubMed with each synonym.
5. extract titles and abstracts.
6. take another information about synonyms from the extracted titles and abstracts.
7. extract information about abbreviations from the titles and the abstracts.
8. retrieve appropriate papers by using the synonyms and the abbreviations.

Keywords to be used for the PubMed searches were synonyms which were collected from public databases (e.g., SWISS-PROT, LocusLink, etc.). The retrieval method PETER employs is rule-based and rules were constructed from the observations that a potential abstract usually includes a synonym of a query and at least one word from the other synonyms. We call these words "selected words", each of which must have no less than four letters and should not be stopwords we prepared.

The scoring system was designed to evaluate a paper in terms of whether or not it contains a query's abbreviation, another synonym (full spelling), or a selected word. The one-sentence splitter called JASMINE (Just A Sentence-splitter Maximizing Intelligence of kNowledge Extraction) and the abbreviation extractor called ALICE (Abbreviation LIfter using Condition-based Extraction) were also developed for PETER system.

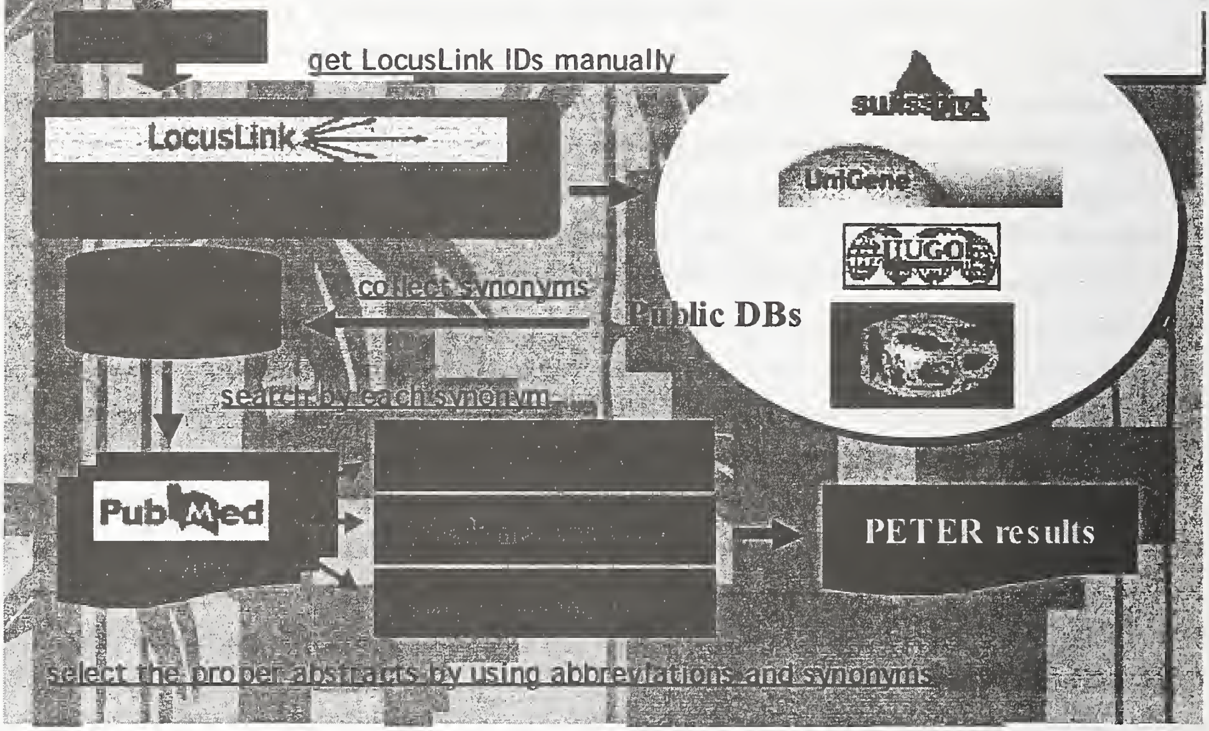
Making out a list of synonyms was the hardest work due to the insufficiency of the databases for gathering appropriate ones. Some entries related to gene names stored in the databases are inappropriate as gene names (e.g., hypothetical protein FLJ20006, Hirschsprung disease and EST), some are not gene specific (e.g., A1, DNA binding protein and tumor necrosis factor), and some are not appeared in real papers. In order to overcame these difficulties and achieve high recall, we made a variant generator to add synonym variations for PubMed searches[1], and collected the other synonyms from the retrieved papers' titles and abstracts after the searches. To get high precision, at the same time, we established empirical selection rules toilsomely.

As a conclusion, we got high precision and recall concerning human UV-regulated genes. The reason is that we have developed PETER for dermatologists in the first place as a joint research with a cosmetic company. It was built to retrieve papers about UV-regulated genes from MEDLINE. Our approach was to make a system which worked as much the same as biologists' do to get papers. While we tried to improve PETER to work well for all genes, it was quite difficult to adjust our method even to general human genes. Through this TREC project, we recognized that, to get better results, it was important for a retrieval system to be able to tune the rules upon biologist's requests. There is no perfect rule, and there is no specialist to establish an all-round method and to predict the results which the system provides. Although it is impossible to create a flawless method for all genes, we want to make an effort to improve PETER as much as we can. We take pleasure in discussing scholars engaged in Bioinformatics, Biology, and Information Retrieval.

[1] For example, hairy and enhancer of split-1, (Drosophila)

⇨ {
hairy and enhancer of split-1,
hairy and enhancer of split 1
hairy and enhancer of split1,
etc.

PETER (overview)



get LocusLink IDs manually

collect synonyms

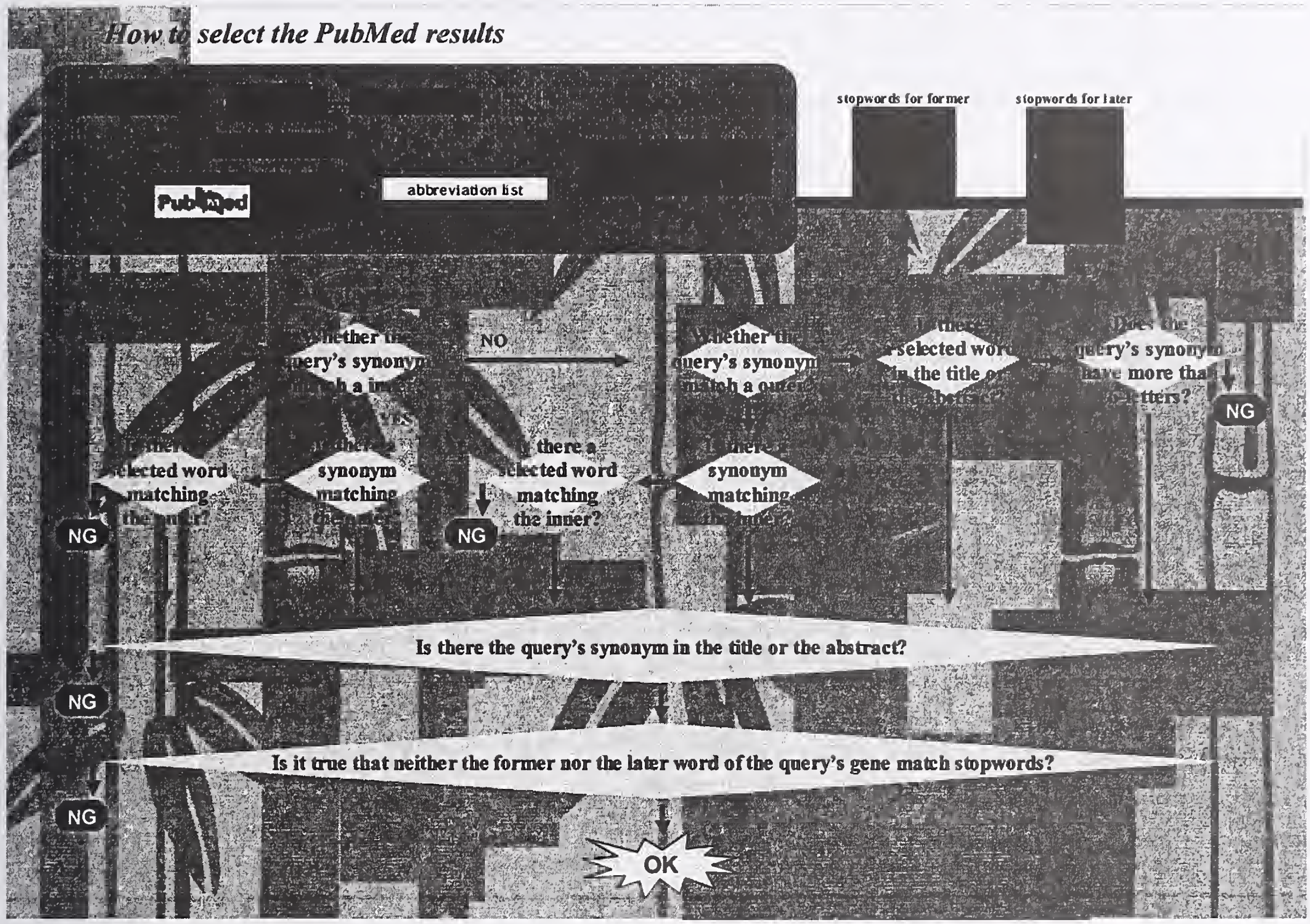
Public DBs

search by each synonym

PETER results

select the proper abstracts by using abbreviations and synonyms

How to select the PubMed results


PubMed
abbreviation list
stopwords for former
stopwords for later
Whether the query's synonym match a inner
NO
YES
Whether the query's synonym match a outer
Is there a selected word in the title or the abstract?
Does the query's synonym have more than letters?
NG
Is there a selected word matching the inner?
Is there a synonym matching the inner?
NG
Is there a selected word matching the inner?
NG
Is there a synonym matching the inner?
Is there the query's synonym in the title or the abstract?
NG
Is it true that neither the former nor the later word of the query's gene match stopwords?
NG
OK

Task-Specific Query Expansion (MultiText Experiments for TREC 2003)

David L. Yeung Charles L. A. Clarke Gordon V. Cormack Thomas R. Lynam Egidio L. Terra

School of Computer Science, University of Waterloo, Canada
mt@plg.uwaterloo.ca

I. Introduction

For TREC 2003 the MultiText Project focused its efforts on the Genomics and Robust tracks. We also submitted passage-retrieval runs for the QA track. For the Genomics Track primary task, we used an amalgamation of retrieval and query expansion techniques, including tiering, term re-writing and pseudo-relevance feedback. For the Robust Track, we examined the impact of pseudo-relevance feedback on retrieval effectiveness under the new robustness measures.

All of our TREC runs were generated by the MultiText System, a collection of tools and techniques for information retrieval, question answering and structured text search. The MultiText Project at the University of Waterloo has been developing this system since 1993 and has participated in TREC annually since TREC-4 in 1995.

In the next section, we briefly review the retrieval methods used in our TREC 2003 runs. Depending on the track, various combinations of these methods were used to generate our runs. The remaining sections describe our activities for the individual tracks, with the bulk of the report covering our Genomics Track results.

II. Retrieval Methods

The MultiText System implements a variety of retrieval methods, three of which were used in our TREC 2003 experiments:

1) *Shortest Substring Ranking* (SSR), a ranked retrieval method for extended boolean queries [1];
2) Qap, a passage-retrieval technique originally developed for question answering [3], [4];
3) Okapi BM25 [8].

Our Genomics runs use a combination of SSR and Okapi methods, generating and executing a number of different term sets and boolean queries, and merging the results to produce a final ranked document set. Our Robust runs use Qap to generate passages for pseudo-relevance feedback and Okapi to evaluate the expanded queries. For our QA track runs, 250-byte answer passages are selected from passages generated by Qap.

In the next few sections, we provide brief overviews of the retrieval methods. Please consult the associated references if further details are required.

A. Shortest Substring Ranking

The MultiText project has used variants of the SSR algorithm in TREC experiments since TREC-4 in 1995. The SSR algorithm operates by locating passages that satisfy a boolean query. A passage may consist of any substring of any document in the target corpus. The algorithm identifies all document substrings that satisfy the query and do not contain shorter substrings that also satisfy the query. This *shortest substring rule* serves to limit the number of passages that must be considered by the SSR algorithm. For ranking purposes, a document's score is computed from the lengths of the passages contained within it.

For example, along with other queries, our system generates the following query for Genomics topic 23:

"c gamma" ^ ("phospholipase"+"phospholipases")

where the + symbol represents boolean OR and the ^ symbol represents boolean AND. Since the algorithm locates the shortest substrings that satisfy the query, a passage located by the algorithm will begin (or end) with the phrase "c gamma" and end (or begin) with one of the words "phospholipase" or "phospholipases". None of these terms will appear elsewhere in the passage, since otherwise the passage would contain a shorter substring that also satisfies the query.

In some cases, structural constraints are applied to the query. For example, the query:

("<NameOfSubstance>".."</NameOfSubstance>")>"cip1"

identifies instances of the NameOfSubstance field that contain the term "cip1", where the > symbol is used to express the CONTAINS relationship.

Assume that document d contains passages $P_1, P_2, ..., P_n$, sorted by increasing length, with each passage satisfying the query under the shortest substring rule. We compute a score for d that rewards shorter passages and documents that contain more passages. For a passage P define:

$$I(P) = \begin{cases} \frac{\mathcal{K}}{l(P)} & \text{if } l(P) \geq \mathcal{K} \\ 1 & \text{if } l(P) \leq \mathcal{K} \end{cases} \qquad (1)$$

where $l(P)$ is the length of P as measured by the number of alphanumeric tokens it contains. For any passage P, we have $0 < I(P) \leq 1$. The score for d is then computed by the formula:

$$\sum_{i=0}^{n} I(P_i)^{\gamma} \tag{2}$$

For our TREC 2003 experiments we use the parameters $\mathcal{K} = 16$ and $\gamma = 0.5$.

SSR is most valuable when a boolean query matches a large number of documents. However, in many cases, a boolean query matches few or no documents. To address this case, we often generate *tiers* of boolean queries, with earlier tiers providing higher precision and later tiers providing higher recall. The tiers are executed in order, with the documents generated by earlier tiers ranked before the documents generated by later tiers. Once a document is generated by a tier, it is eliminated from later tiers. Section III-B describes the tiered boolean queries used in our Genomics Track runs.

A complete discussion and analysis of the SSR algorithm may be found in Clarke and Cormack [1]. That paper also provides an efficient algorithm to implement SSR.

B. Passage Retrieval

The Qap passage retrieval algorithm is related to the SSR algorithm, in that it may return any substring of any document. The algorithm locates "hotspots" within the corpus where query terms cluster in close proximity. The score of a hotspot is based on its length and the weights of the terms occurring within it. A hotspot is usually less than 50 words in length, but may be longer. It may start or end at any word and is not constrained by sentence or paragraph boundaries. At most one hotspot is selected from a document, since additional hotspots from the same document may not exhibit the independence properties assumed by our feedback and QA methods.

Given a query Q, document substring H, and a term set $T \subseteq Q$, we compute a score for H as follows:

$$\sum_{t \in T} \log(N/f_t) \; - \; |T| \log(l(H)) \tag{3}$$

where f_t is the total number of times t appears in the corpus and N is the total length of all documents in the corpus. In effect, the Qap algorithm considers every substring of every document in the corpus and locates the m substrings with the highest score. Details of the Qap algorithm, its efficient implementation and its application to question answering may be found in Clarke et al. [4].

C. Okapi

Our implementation of Okapi BM25 follows the description of Robertson et al. [8] with standard parameters: $k_1 = 1.2$, $b = 0.75$, $k_2 = 0$, $k_3 = \infty$. Specifically, given a term set Q, a document d is assigned the score

$$\sum_{t \in Q} w^{(1)} q_t \frac{(k_1 + 1) d_t}{K + d_t} \tag{4}$$

where

$$\begin{array}{rcl}
w^{(1)} & = & \log\left(\frac{D - D_t + 0.5}{D_t + 0.5}\right) \\
D & = & \text{number of documents in the corpus} \\
D_t & = & \text{number of documents containing } t \\
q_t & = & \text{frequency that } t \text{ occurs in the topic} \\
d_t & = & \text{frequency that } t \text{ occurs in } d \\
K & = & k_1((1 - b) + b \cdot l_d / l_{avg}) \\
l_d & = & \text{length of } d \\
l_{avg} & = & \text{average document length}
\end{array}$$

As an extension, our implementation of BM25 allows phrases (and extended boolean queries) to be used as query terms, a facility used in our Genomic runs to allow multi-token gene names and bigrams extracted from gene names to be treated as individual terms.

III. Genomics Track - Methodology

For the Genomics track we experimented with a number of different retrieval, feedback and fusion techniques. The following sections describe the various aspects of the experimental methodology. Analysis of the results obtained from the training data is found in the Section IV. Section V discusses the results of our official TREC test runs.

In Section III-A, we investigate the effects of query formulation using the Okapi retrieval model. Section III-B deals with our experiments on using tiers of boolean queries to match against the metadata fields in the MEDLINE records. Then, in Section III-C, we explore the idea of merging the document sets retrieved by Okapi and the query tiering techniques. We describe the use of query expansion and feedback in Section III-D. In Section IV, we assemble the techniques into complete runs.

A. Okapi Query Formulation for the Genomics Track

Two important facts were discovered in preliminary experiments which influenced the design of the Okapi experiments. First, the gene name type did not seem to matter. A document discussing a particular gene was as likely to use an official name as an alternate one. Second, spacing and punctuation had a large effect on performance in some cases. The gene name in the original LocusLink-derived query may differ from the gene or protein name as it actually appears in the corpus only by the addition or removal of spaces or dashes. In a model based on term sets, such as Okapi, these slight variations may significantly affect the results.

We investigated the effects of query formulation on IR in the Genomics domain by generating multiple term sets from the original query, and comparing the effects of using these term sets to retrieve documents using the Okapi retrieval model. The three rules used to generate the term sets were:

- *Okapi 1*: Each gene name in the original query, which may consist of multiple alphanumeric tokens, is considered as a phrase and treated as a single term, the only change being the removal of punctuation.
- *Okapi 2*: Heuristics were used to split up gene names containing semi-colons, commas, and brackets. Heuristics were also used to guess "plurals" for some of the terms.
- *Okapi 3*: First, the gene names were separated into two sets, one containing those gene names which were comprised of a single token, and another containing gene names which were comprised of multiple tokens. The Okapi term set was created from these two sets by first concatenating all pairs of single-token gene names together, and adding all the token-bigrams from the multiple-token gene names.

The name of the species was also included in each of the term vectors. The three rules are in decreasing order of strictness. Documents retrieved by Okapi 1 will contain the terms exactly as given in the original query (ignoring punctuation), while those retrieved by Okapi 2 will contain terms which are similar to but not exactly like those in the original query. Documents retrieved by Okapi 3 contain the same bigrams as found in the original query.

Each query formulation has its own advantages and disadvantages. The top documents returned by Okapi 1 are likely to be relevant, since they contain the query exactly, but many relevant documents may be missed because the gene name in the document appears differently than in the query. On the other hand, Okapi 3 retrieves many relevant documents in which the gene name does not appear exactly as in the query. However, it also retrieves many documents that are not relevant. The documents retrieved by Okapi 2 are intermediate between the two.

We found that the document sets retrieved using the term vectors generated by the three rules were quite different. Therefore, it was decided that the document sets produced by Okapi 1, Okapi 2, and Okapi 3 would be fused together. The fusion was accomplished in the following manner:

- *Okapi Fusion*: The document sets retrieved by Okapi 1, Okapi 2, and Okapi 3 are combined by taking the intersection of the three result sets. A document's score is taken to be the product of the three scores. This list is then followed by the remainder of Okapi 3, with the scores appropriately scaled.

The rationale behind the fusion is that a document that scores highly on all three query formulations is very likely to be relevant. Taking the product of the scores allows each of the three document sets to vote on the relative distance between similarity values equally. Since Okapi 3 is the most relaxed of the three query formulations, it retrieves most if not all of the relevant documents retrieved by Okapi 1 and 2. Thus, the intersection of the three document sets likely contains most of the relevant documents in the document sets returned by Okapi 1 and 2, while it might miss relevant documents retrieved by Okapi 3. For that reason, the remainder of the Okapi 3 document set is appended to the end of the combined list.

While there are other fusion techniques, the above seemed to work very well in preliminary trials, and thus was the only technique used in the final completed runs. The performance of Okapi 1 is considered to be the baseline for comparison purposes in the rest of this report.

B. Boolean Query Formulation for the Genomics Track

Preliminary experiments showed that there was a correlation between some of the metadata fields in the MEDLINE record and the relevance of the document. In particular, there was a strong correspondence between the query terms and the terms that appeared in the RN (registry number) field of the MEDLINE record. The RN field contains a list of the chemicals discussed in the document. Many of these chemical names can be matched to the gene names found in the query. The chemical list is a better indicator of a document's relevance than the document's title, which in turn is a better indicator than the abstract. To capture this hierarchical structure among the metadata fields, we experimented with using a number of query tiers. The final tiering system had the following tiers, in decreasing order of relevance:

1) *Tier 1*: The gene name is found in the chemical list, or it is found in the chemical list preceded or followed by the word "protein", optionally followed by the name or description of the species. Spaces and punctuation are ignored for the purposes of comparison. (From training topic 5, "glycine receptor, alpha 1" is considered to be equivalent to "glycine receptor alpha1".)
2) *Tier 2*: This tier is similar to Tier 1, except that the chemical name is allowed to have additional terms. (From training topic 11, "RAC1" retrieves documents in which "rac1 GTP-Binding Protein" appears in the chemical list.)
3) *Tier 3*: An attempt is made to find the conjunction of the terms from the gene name in the chemical list. If the gene name consists of a class name followed by a sequence of letters and numbers that specifies an object of that class, the name is successively weakened until a match is made. A set of heuristics are also used to recognize plurals. (From training topic 32, "estrogen receptor 1" is weakened until the documents retrieved contain "Receptors, Estrogen" in the chemical list.)
4) *Tier 4*: The query is converted into a boolean expression by turning each gene name into the conjunction of its terms, and taking the disjunction of all gene names. The boolean expression is applied to the title.
5) *Tier 5*: The boolean expression is applied to the chemical list.
6) *Tier 6*: The boolean expression is applied to the abstract.

In addition, the documents are restricted to those in which the name of the species appears in the MeSH (Medical Subject Heading) metadata field. This does not completely eliminate documents which are not relevant to the species, since it is possible for the name of the species to appear in the MeSH field even if the focus of the paper is another species. It is quite common for an article about a gene in one species to mention a homologue in a related species. Nevertheless, if the name of the wanted species does not appear in the MeSH heading, then the article is (almost certainly) not relevant. Thus, using species data in the MeSH metadata field may result in false positives but not (or rarely) in false negatives.

Based on the query tiering model described above, we tested three different ways of retrieving documents:

- *All Tiers*: Retrieve documents from all the tiers. Documents retrieved by each tier are ranked ahead of all documents retrieved by the next tier. A document that is retrieved in more than one tier is counted towards only its highest tier.
- *Best Tier*: Retrieve the documents in the first tier that contains a non-zero number of documents. Subsequent tiers are ignored.
- *Exact*: Retrieve only documents in Tier 1. No documents are retrieved if there are no documents in Tier 1.

Note that for some topics, the above techniques may return zero documents. For that reason, the complete runs described in Section IV supplement the document sets retrieved by the tiering techniques with documents retrieved using the Okapi methods.

While the query tiers have a significant effect on performance, further improvement is possible by using fusion and feedback.

C. Genomics Track Fusion

Since the Okapi and tiering experiments retrieved different document sets, we explored merging the results of the two techniques. We tried two different methods of combining the two document sets returned from Okapi and the tiering technique:

- *Interweave*: The two document sets are combined by taking one document from each set successively.
- *Rank Fusion*: First, documents which were retrieved by both methods are merged. The score assigned to a document is a weighted sum of its (reverse) rank in each document set. The combined documents are followed by interweaving the remainder of the two document sets.

We also attempted other types of fusion, but these were the only two which were completely implemented and tested due to time constraints.

D. Genomics Track Feedback

As explained in Subsection III-B, the similarity of the chemical list in the MEDLINE record to the query is a good indicator of a document's relevance. Because a gene name may have many variants, however, it is not always possible to match the gene name to an item in the chemical list even though one of the chemicals may refer to that gene or its product.

One possible solution to this problem is to attempt to recognize these name variants. That is not the approach we took. Instead, we attempted to learn the variant name by using feedback. If the gene name was matched in Tier 1 using the tiering technique, then the chemical list in the top retrieved documents already contains the gene name, and so feedback is unnecessary. Otherwise, we assume the top documents retrieved to be relevant, and find the chemical that has the highest correlation with these documents. The chemical names in the top documents were assigned a score using the formula:

$$w_i = R_i \cdot \left(\log\left(\frac{N}{f_i}\right)\right)^{\alpha}$$

Method Used	Rel. & Ret.	Avg. Precision	R-Precision
Okapi 1	224	0.3273	0.3077
Okapi 2	245	0.3193	0.2917
Okapi 3	261	0.3157	0.2700
Okapi Fusion	261	0.3321	0.3173
AT	282	0.3819	0.3452
ATI	282	0.4394	0.3836
ATIF	289	0.4429	0.3844
ATR	284	0.4519	0.4324
ATRF	291	0.4598	0.4448
BT	279	0.4003	0.3818
BTI	279	0.4528	0.4236
BTIF	286	0.4812	0.4448
BTR	279	0.4452	0.4216
BTRF	286	**0.4821**	**0.4579**
Exact	277	0.3981	0.3820
ExactI	277	0.4246	0.3959

TABLE I

Genomic Track — Summary of Results on Training Data: 50 topics, 1000 retrieved per query, 335 total relevant.

For a chemical i, R_i is the number of times the chemical name appears in the chemical list of the top documents, f_i is the number of times it appears in the corpus, N is the total length of all documents in the corpus, and w_i is the score assigned to i. For our TREC 2003 experiments, we set $\alpha = 3$.

The highest scoring chemical name is then used to retrieve a set of documents containing that chemical name. The three sets of documents retrieved by the Okapi Fusion, query tiers, and feedback are then merged to produce the final document set. The number of top documents assumed to be relevant and the precise mechanism used to merge the final document sets are discussed in the next subsection.

IV. Genomics Track - Training Runs

The parameters of the various runs were optimized for the training data, using the supplied relevance judgments. Thus, the performance of the IR system on the training data is not necessarily reflective of its performance on the test data, especially if the training and test data have different characteristics. In particular, the relative performance of some of the runs that relied on a single retrieval technique may not be necessarily preserved. Nevertheless, the runs involving fusion and feedback do seem to consistently outperform the systems on which they are based. The parameters for these runs were adjusted not only to maximize performance, but to increase stability as well.

Following the TREC standard procedure, 1000 documents were retrieved for each run. We attempted to test a large

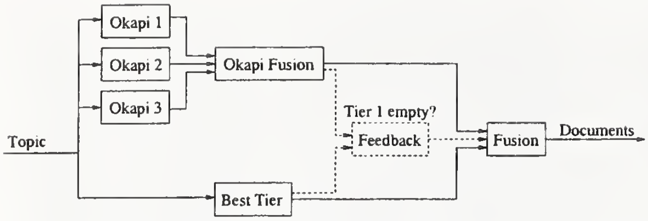


Fig. 1. Flow diagram for the combined system of the BTRF run.

variety of techniques, but unfortunately many tests could not be completed due to time constraints. The results for the runs which we conducted on the training data are shown in Table I. These were:

- *Okapi 1, 2, 3, and Fusion*: These are the document sets retrieved by the procedure described in Section III-A.
- *All Tiers* (AT): This is the set of documents retrieved by using the All Tiers method as described in Section III-B. The documents retrieved by Okapi Fusion are appended to the end.
- *All Tiers Interweave-fusion* (ATI): The set of documents retrieved by All Tiers is interweaved with the document set retrieved by Okapi Fusion.
- *All Tiers Rank-fusion* (ATR): The set of documents retrieved by All Tiers is merged with the Okapi Fusion documents using the weighted rank fusion. It was experimentally determined that good results can be obtained if the Okapi rank was weighted 4 times as heavily as the tiering rank.
- *All Tiers Interweave/Rank-fusion with Feedback* (ATIF, ATRF): These are the same as ATI and ATR, respectively, except that the feedback procedure described in Section III-D and further elaborated below was used if no documents were retrieved in Tier 1.
- *Best Tier* (BT, BTI, BTR, BTIF, BTRF): These are analogous to the above, except that the query tiering subsystem retrieved only documents from the first tier with non-zero documents.
- *Exact*: Instead of all the tiers or the best tier, only Tier 1 was used to retrieve documents. The Okapi Fusion document set was then appended to the end. (If no documents were retrieved in Tier 1 for a topic, then the final set of retrieved documents is just the set retrieved by Okapi Fusion.)
- *ExactI*: The set of documents retrieved by Tier 1 is interweaved with the Okapi Fusion set.

Figure 1 shows the combined system for the BTRF (Best Tier, Rank-fusion, Feedback) run. The topic is sent to both the Okapi and query tiers subsystems, each of which returns a set of documents. If the first tier to retrieve a non-zero number of documents is Tier 1, then the two document sets are fused. Otherwise, a third set of documents is retrieved using feedback, and the three sets of documents are fused. The other runs follow a similar logic flow.

The performance of feedback is dependent on the number of top documents used to determine the most relevant chemical name, and on the type of fusion used to merge the three document sets. These parameters are in turn dependent upon the query tiering technique used. For the All Tiers technique, it was determined that using the top 25–30 documents to determine the most relevant chemical name produced the best performance. (The value of 27 was used in the experiments.) The three document sets are fused using rank fusion with equal weights. For the Best Tier technique, the top 42 documents were used, and the three document sets were merged using weighted rank fusion with a weight of 5 for the query tiers document set, 28 for the feedback document set, and 20 for the Okapi Fusion document set. These numbers were determined experimentally.

The reason for the difference between the feedback parameters of the AT and BT runs is that more of the top documents retrieved by the Best Tier technique are relevant compared to those retrieved by All Tiers. Since feedback is only used when no documents are retrieved in Tier 1, the set of documents retrieved using the top chemical name will be far more likely to be relevant than the documents retrieved by the Best Tier, and slightly more likely to be relevant than those retrieved by Okapi.

As can be seen from Table I, the best average precision belonged to the BTRF run, at 0.4821. This is a 47.3% improvement over the baseline Okapi 1, which had an average precision of 0.3273. The BTIF run had an average precision of 0.4812, a 47.0% improvement, and the ATRF run had an average precision of 0.4598, a 40.5% improvement. The ATRF run retrieved 291 relevant documents, which was the most relevant documents retrieved of all the runs. This is slightly more than the 286 retrieved by BTRF and BTIF, and significantly more than the 224 retrieved by the Okapi 1 run.

Some general trends are discernible from the numbers. Feedback and fusion improved performance in every case, and the systems with the best performance made use of both. It isn't clear which fusion method is better, since ATR outperformed ATI, but BTI did better than BTR. However, when fusion is used with feedback, the rank fusion method outperformed the interweave fusion method in both cases.

There is a high level of correspondence between the metadata fields and the relevance of the documents. This is clear from the fact that retrieval using query tiers based on the information in the metadata fields outperformed the Okapi runs, including the Okapi Fusion run. Before fusion and feedback, the best technique that is based on query tiers is BT, with an average precision of 0.4003, which is a 22% improvement over Okapi 1. The Exact run had an average precision of 0.3981, a 21% improvement, while the AT run had an average precision of 0.3819, which close to 17% over Okapi 1. Note that both Best Tier and Exact had a better average precision than the All Tiers method. It appears that once a match has been found in a tier, it was a better strategy to append the Okapi Fusion list rather than documents from lower tiers. The experimental results suggest that the performance of the Okapi Fusion method was between that of Tier 1 and 2.

Topic	Number of Documents Retrieved						Matches in Best Tier
	T1	T2	T3	T4	T5	T6	
1	**438**	120	0	19	0	482	“cip1 protein”
2	**6**	13	38	4	0	28	“rna dependent atpase”, “protein p68”
3	**19**	31	0	5	0	43	“tel protein”
4	**35**	2	499	2	0	75	“keratinocyte growth factor”, “fibroblast growth factor 7 precursor”, “fibroblast growth factor 7”
5	**16**	0	23	0	0	6	“glycine receptor alpha1”
6	**93**	10	0	2	0	101	“hla dqb1”
7	**56**	3	44	0	0	39	“janus kinase 2”
8	–	–	–	**8**	0	50	((“luteinizing”^“hormone”^“choriogonadotropin”^“receptor”)+“lhcgr”+“lcgr”+“lhr”+ (“luteinizing”^“hormone”^“receptor”)+(“lutropin”^“choriogonadotropin”^“receptor”)+“lcgrs”+ “lhcgrs”+(“luteinizing”^“choriogonadotropin”^“receptor”)+“lgr2”+“lhrs”+(“lutropin”^“receptor”)+ (“choriogonadotropin”^“receptor”))
9	**15**	1	68	12	0	345	“growth inhibitory factor”
10	**161**	360	757	480	0	785	“protein c”
11	–	**80**	0	0	0	117	“rac1”
12	**3**	0	41	0	0	11	“tropomyosin 1”
13	**3**	0	3	7	0	163	“gpcr protein”, “frizzled 4 protein vertebrate”
14	–	–	–	**10**	0	408	((“tyrosyl”^“trna”^“synthetase”)+“tyrrses”+“ytses”+“yts”+ (“tyrosyl”^“trna”^“ligase”)+“yars”+“tyrrs”+“yarses”+“yrses”+“yrs”)
15	**11**	1	0	13	0	109	“major vault protein”
16	**4**	0	80	0	0	0	“adrenergic receptor alpha 1d”, “adrenergic receptor alpha 1a”
17	–	**10**	0	0	0	0	“rhob”
18	**213**	0	205	2	0	73	“cpp32 protein”
19	**6**	0	0	0	0	6	“ctcf protein”
20	**162**	0	979	2	0	68	“fasl protein”
21	–	–	**1**	2	0	44	(((“ig”)))
22	–	–	–	**4**	0	14	(“ihhs”+(“indian”^“hedgehog”)+“ihh”)
23	–	–	**47**	1	0	16	(((“phospholipase”+“phospholipases”))^“c gamma”)
24	–	–	**3**	0	0	0	(((“seven”+“sevens”)^(“absentia”+“absentias”)))
25	–	–	–	**3**	0	112	(“dntts”+“tdt”+“dntt”+(“terminal”^“deoxynucleotidyl”^“transferase”)+ (“deoxynucleotidyltransferase”^“terminal”)+“tdts”)
26	–	–	–	**1**	0	1	((“rho”^“related”^“btb”^“domain”^“containing”^“2”)+“rhobtb2”+“kiaa0717”+“dbc2”)
27	–	–	–	–	–	**19**	((“cholinergic”^“receptor”^“muscarinic”^“3”)+“chrm3”)
28	–	**11**	0	9	0	57	“egr1”, “ngfi”
29	**19**	1	0	0	0	8	“glucokinase”
30	**2**	0	40	0	0	1	“retinoic acid receptor gamma”
31	**149**	4	460	9	0	93	“neurokinin a”, “substance p”, “neuropeptide k”
32	–	–	**186**	4	0	75	(((“estrogen”+“estrogens”)^(“receptor”+“receptors”)))
33	–	–	**70**	0	0	21	(((“guanylate”+“guanylates”)^(“cyclase”+“cyclases”)))
34	**20**	1	0	0	0	2	“cocaine and amphetamine regulated transcript protein”
35	–	–	–	–	–	–	–
36	**5**	0	9	2	0	6	“hop protein”
37	**1**	0	0	0	0	1	“slob protein”
38	**3**	0	0	0	0	0	“eiger protein drosophila”
39	**32**	1	7	1	0	15	“cadherins”
40	**6**	0	0	3	0	2	“stat92e protein”
41	**3**	0	0	0	0	3	“ebony protein”
42	**10**	0	0	0	0	5	“crb protein drosophila”
43	–	–	**3**	11	0	422	(((“calcineurin”+“calcineurins”)))
44	**3**	0	4	0	0	0	“gp73 protein”
45	**5**	1	3	2	0	5	“sh3px1 protein”, “wisp protein”
46	–	**7**	0	5	0	16	“hanks”, “ank”
47	**2**	0	0	0	0	0	“dda3 protein”
48	**10**	0	0	10	0	323	“artemis protein human”
49	–	–	**1000**	67	0	947	(((“transcription”+“transcriptions”)^(“factor”+“factors”)))
50	**1**	0	2	0	0	1	“pax 8 protein”
Total	32	4	8	4	0	1	

TABLE II

GENOMIC TRACK — MATCHES IN THE QUERY TIERS FOR THE TRAINING TOPICS.

Topic	Query Term/Phrase	Feedback Chem. Name	Ret.	R.&R.	MAP	R-P	MAP Fb.	R-P Fb.	Imp.
8	luteinizing hormone/ choriogonadotropin receptor	Receptors, LH	49	7	0.2917	0.4286	0.4305	0.4286	+47%
11	ras-related C3 botulinum toxin substrate 1 (rho family, small GTP binding protein Rac1)	rac1 GTP-Binding Protein	80	13	0.2302	0.4118	0.1977	0.1765	-14%
14	tyrosyl-rRNA synthetase	Tyrosine-rRNA Ligase	10	6	0.5872	0.5000	0.8238	0.6667	+40%
17	ras homolog B (RhoB)	rhoB GTP-Binding Protein	6	2	0.3333	0.3333	0.3889	0.6667	+17%
21	immunoglobulin heavy chain 6 (heavy chain of IgM)	Immunoglobulins, mu-Chain	21	0	–	–	–	–	–
22	Indian hedgehog	hedgehog protein, vertebrate	69	6	0.4703	0.5000	0.6723	0.5000	+43%
23	phospholipase C, gamma 1	phospholipase C gamma	47	9	0.6503	0.5556	0.5262	0.4444	-19%
24	seven in absentia 2	seven in absentia protein	3	2	1.0000	1.0000	1.0000	1.0000	0%
25	terminal deoxynucleotidyl transferase	DNA Nucleotidylexotransferase	8	2	1.0000	1.0000	1.0000	1.0000	0%
26	Rho-related BTB domain containing 2	QM protein, Trypanosoma brucei	0	0	1.0000	1.0000	1.0000	1.0000	0%
27	cholinergic receptor, muscarinic 3	Receptors, Muscarinic	153	2	0.0312	0.0000	0.0747	0.0000	+139%
28	Early growth response 1	Krox-24 protein	40	8	0.0258	0.1250	0.2523	0.1250	+878%
32	estrogen receptor 1	Receptors, Estrogen	163	11	0.1039	0.0909	0.1354	0.0000	+30%
33	guanylate cyclase 1, soluble, beta 3	Guanylate Cyclase	70	1	0.0774	0.0000	0.0569	0.0000	-26%
35	CG3599	Drosophila Proteins	638	0	–	–	–	–	–
43	Calcineurin B	Calcineurin	3	1	0.5000	0.0000	1.0000	1.0000	+100%
46	ankylosis, progressive homolog	ankylosis protein	5	3	0.1595	0.0000	0.7500	0.7500	+370%
49	transcription factor 23	Transcription Factors	1000	0	–	–	–	–	–

TABLE III

GENOMIC TRACK — ANALYSIS OF THE EFFECTS OF FEEDBACK ON PERFORMANCE.

Table II shows the documents retrieved in each tier for the 50 training topics. The topic number is shown in the first column, followed by six columns showing the number of documents retrieved in each of the six tiers. The last column contains the expression or expressions used in the first tier in which a match was made.

In 32 out of 50 topics, the best tier was Tier 1. Of the remaining topics, Tier 2 was the best tier in 4 topics, Tier 3 was best in 8, and Tier 4 was best in 4. No documents were retrieved at all in Tier 5, and Tier 6 was the best tier for 1 topic. In the final arrangement of the query tiers, it happened that every document retrieved by Tier 5 had already been retrieved in a higher tier.

Because Tier 1 had a better performance on its own than Okapi or even feedback, performance can be improved by recognizing relevant chemical names in the chemical list metadata, even in cases where the name of the gene and the relevant chemical name are different.

Table III shows the chemical names produced by the pseudo-relevance feedback for those topics in which no documents were retrieved in Tier 1, for the BTRF run. The first column gives the topic number, and the second column gives a gene name from the query. The third column shows the chemical name that was found using automatic query expansion. The next four columns show the number of documents retrieved, the number retrieved and relevant, the mean average precision, and the interpolated recall-precision, respectively, for that topic without using feedback. The next two columns give the mean average precision and interpolated recall-precision with feedback, and the last column gives the percentage improvement (or degradation) due to using feedback. It is apparent that most of the chemical names are related in some way to the gene name, and a better way of recognizing the relationship between a gene and a chemical name will clearly improve performance.

For topic 28, the top chemical name "Krox-24 protein" was produced for the "Early growth response 1". In fact, "Krox-24 protein" is another name for "Early growth response 1". By searching on "Krox-24 protein", which does not appear in the original query, the average precision was improved by an incredible 878%. Of course, the original performance for this topic was very poor, but there is clearly a lot of potential for improving performance by recognizing the alternate names of a gene or a substance related to a gene.

In some cases, this is relatively simple. For topic 14, for example, the chemical name "Tyrosine-rRNA Ligase" was generated for the gene name "tyrosyl-rRNA synthetase". A system that understood the relationship between "tyrosine" and "tyrosyl" and "ligase" and "synthetase" can determine that the two expressions refer to the same thing (or closely related things), and even assign a score for the degree of similarity. In other cases, this is complicated by the fact that more than one chemical name generated by the automatic expansion might be relevant to the query. For topic 27, searching on the gene name "cholinergic receptor, muscarinic 3" resulted in the top chemical name "Receptors, Muscarinic". However, the chemical name "muscarinic receptor M3", which is clearly more relevant, was overlooked. Choosing this chemical name instead of the more general "Receptors, Muscarinic" would have resulted in an improvement of 534%.

As the table shows, in most cases the performance was improved by using feedback to find the most relevant chemical, though in some cases there was a degradation in performance. Determining the conditions under which feedback improved or

Method Used	Rel. & Ret.	Avg. Precision	R-Precision
Okapi 1	447	0.2060	0.1965
Okapi 2	473	0.2155	0.1948
Okapi 3	524	0.2169	0.2095
Okapi Fusion	524	0.2323	0.2138
AT	550	0.2542	0.1967
ATl	550	0.3334	0.2723
ATlF	559	0.3379	0.2680
ATR	552	0.3425	0.3050
ATRF	**562**	0.3479	0.3013
BT	535	0.2443	0.2010
BTl	535	0.3066	0.2581
BTlF	556	0.3322	0.2745
BTR	535	0.3161	0.2852
BTRF	556	**0.3534**	**0.3113**
Exact	528	0.2500	0.2194
Exactl	528	0.2803	0.2449

TABLE IV

GENOMIC TRACK — SUMMARY OF RESULTS ON TEST DATA: 50 TOPICS, 1000 RETRIEVED PER QUERY, 566 TOTAL RELEVANT.

degraded performance would allow feedback to be used more effectively.

The two runs chosen for official submission to TREC were the ATRF and BTRF runs. Even though BTIF had a better mean average precision than ATRF, it was too similar to the BTRF run in that it differed only in the fusion method used. It was found that by adjusting the fusion weights, it was always possible for the rank-fusion to outperform the interweave fusion. It was also suspected that the ATRF run might be more stable, in the sense that the performance would not be too adversely affected by an incorrect match in Tier 1. ATRF also had the most number of relevant documents retrieved, and it would be interesting to examine the trade-off between retrieving more relevant documents and having a better precision. In addition to the two official runs, we also performed all the various runs using the test data.

V. Genomics Track - Results

The results for various test runs of our system are shown in Table IV. ATRF and BTRF are official runs, submitted to NIST under the run tags "uwmtg03atrf" and "uwmtg03btrf".

Some similarities and differences between the training and test results may be noted. As with the training data, the BTRF run had the best performance on the test data, with an average precision of 0.3534. This is a 71.5% improvement over the Okapi 1 run, which had an average precision of 0.2060. The ATRF run retrieved the most relevant documents, and had the second best average precision at 0.3479, a 68.9% improvement over the Okapi 1 run. Furthermore, ATRF performed better than BTIF, which had an average precision of 0.3322. The distance between ATRF and BTRF was also smaller. On the training data, BTRF had a 4.8% improvement in average precision over ATRF, but on the test data that difference is only 1.6%. This suggests that with the test data, the gene names in the corpus are less like the queries than with the training data.

This conjecture is also supported by the performance of the Okapi runs. While the Okapi Fusion run performed better than any individual Okapi run, the Okapi 3 run had the highest average precision, followed by Okapi 2, and then Okapi 1. This is the reverse of the order with the training data. Using bigrams rather than the original query resulted in better performance on the test data. A more thorough analysis is needed to determine if this conjecture is correct.

VI. Robust Retrieval Track

For the Robust Track, MultiText examined the impact of pseudo-relevance feedback on retrieval effectiveness under the new robustness measures. There are two unusual aspects to our work on this track: 1) the adaptation of techniques from our question answering system to pseudo-relevance feedback, and 2) the expansion of the corpus with a terabyte of Web data for pseudo-relevance feedback. Previous applications of this "collection enrichment" technique have generally used much smaller corpora [6], [9].

A. Robust Track Feedback

For feedback, we adapted the passage-retrieval and term-extraction methods from our QA system, which we have been developing over the past four years. Query processing proceeds as follows:

1) After stopword elimination and stemming, the terms from the topic field(s) are used by the Qap passage-retrieval algorithm (Section II-B) to locate the top m hotspots.
2) Feedback terms are extracted from the hotspots and the text surrounding them. A score is computed for each extracted term. If two terms stem to the same root, the term with the lowest score is eliminated, since stemming will be applied to the expanded query.
3) The top k feedback terms are added to the original term set.
4) Terms in the expanded query are stemmed. The result is executed using our implementation of Okapi BM25 to return the top 1000 documents.

We treat any non-query term appearing within or near a hotspot as a candidate for feedback. For most of our TREC 2003 runs we extracted only single-word terms for feedback; for one run we we also extracted word bigrams. Our term extraction method assigns a score to each candidate term based on its distance from the hotspot, the number of retrieved passages it which it appears, and its relative frequency within the corpus.

Let $H_1, H_2, ..., H_m$ be the hotspots located by Qap. Let $l(H)$ be the length of hotspot H as measured by the number of alphanumeric tokens it contains. We define a function $L(H, t)$ over hotspots H and terms t that measures the "length" of a passage that contains both the hotspot and the term. If t appears in the hotspot, then $L(H, t) = l(H)$. If t appears outside the hotspot, then $L(H, t)$ is the length in tokens of the shortest passage that contains both t and the entire hotspot H. If t does not appear in proximity to the hotspot — if it

		Old Topics			New Topics			Old + New Topics		
Run Tag	Run Type	avgp	norel	bad	avgp	norel	bad	avgp	norel	bad
uwmtCR0	description only, feedback	0.150	14.0%	0.011	0.403	8.0%	0.052	0.276	11.0%	0.018
uwmtCR1	description only, no feedback	0.114	18.0%	0.009	0.355	6.0%	0.035	0.234	12.0%	0.013
uwmtCR2	title only, feedback	0.168	22.0%	0.006	0.370	10.0%	0.053	0.269	16.0%	0.015
uwmtCR3	title only, no feedback	0.102	16.0%	0.007	0.285	8.0%	0.042	0.194	12.0%	0.013
uwmtCR4	description only, feedback, bigrams	0.148	20.0%	0.014	0.404	8.0%	0.054	0.274	14.0%	0.019
-	title + description, feedback	0.175	16.0%	0.017	0.408	8.0%	0.087	0.292	12.0%	0.029
-	title + description, no feedback	0.133	10.0%	0.018	0.369	2.0%	0.066	0.251	6.0%	0.026
-	combMNZ (uwmtCR0, uwmtCR2)	0.174	10.0%	0.020	0.411	4.0%	0.085	0.292	7.0%	0.033
-	combMNZ (uwmtCR1, uwmtCR3)	0.130	10.0%	0.016	0.360	2.0%	0.066	0.245	6.0%	0.024

TABLE V

ROBUST TRACK — SUMMARY OF RESULTS

appears outside a large window surrounding the hotspot or if it does not appear inside the document containing the hotspot — then for simplicity we define $L(H,t) = N/f_t$, where N is the total length of all documents in the corpus and f_t is the total number of times t appears in the corpus. We then compute the feedback score for term t as:

$$w_t = \sum_{1 \leq i \leq m} \log \left(\frac{N}{f_t \cdot L(H_i, t)} \right) \tag{5}$$

We generate an expanded query by combining the top k feedback terms with the original topic terms. We adjust the retrieval weights of the added terms with a scaling factor that takes the feedback score w_t into account, and reflects the fact that terms added through feedback should not be assigned the same importance as the original topic terms.

Let W be the score of the top-ranking feedback term (i.e. the term with the largest feedback score). We define the scaling factor for feedback term t as:

$$S_t = \frac{C \cdot w_t}{W} \tag{6}$$

where $C = 1/3$ in all our experiments. S_t is used to adjust the retrieval weights in the Okapi BM25 formula, modifying Equation 4 to:

$$\sum_{t \in Q} S_t w^{(1)} q_t \frac{(k_1 + 1)d_t}{K + d_t} \tag{7}$$

For original topic terms $S_t = 1$; for feedback terms $S_t \leq 1/3$.

As an example, we examine the end-to-end processing for topic 613, using the description field:

> How were pieces of the Berlin wall disposed of after their removal?

After stopword removal, the Qap algorithm is used to generate a set of passages. A typical passage returned by Qap is:

> Edwina Sandys, whose sculptures are installed at five United Nations centers around the world. One of Winston Churchills 10 grandchildren, her sculpture Breakthrough made of *Berlin Wall pieces* has been called one of the most important monuments constructed on American soil since the Vietnam War Memorial.

The hotspot is in italics. Feedback over these passages generates the expanded term set:

> #1.00 pieces #1.00 berlin #1.00 wall #1.00 disposed #1.00 removal #0.333 war #0.297 freedom #0.293 cold #0.286 dismantling #0.236 souvenirs #0.195 display #0.137 selling #0.110 gift ...

Scaling factors (S_t) precede each term. The average precision for the topic increases from 0.212 to 0.487.

B. Robust Track Experiments

In preparation for the track, we generated a large number of training runs and examined the impact of feedback parameters (e.g. the number of added terms) on average precision and robustness. In general, as feedback parameters are changed, average precision changes slowly in the direction of a single local maxima over a wide range of parameters. In contrast, changes to feedback parameters have small but unpredictable effects on the robustness measures, with many local maxima, making it difficult to tune these parameters specifically for robustness. As a result of our preliminary experiments we used a single set of parameters for all experiments: $m = 20$ passages and $k = 35$ terms.

Using our passage-retrieval algorithm, we executed the original queries against three collections: 1) the AQUAINT corpus, used for the QA Track; 2) a terabyte collection of Web data; and 3) TREC disks 4 and 5 minus the CR documents, which is the target collection for the Robust Track. We then retrieved the top 20 passages from each collection, extracted the top 300 terms from each set of passages, and merged these terms into a single ranked list. A term was only included in this list if it appeared in list of terms extracted from the target collection. Finally, we added the top 35 terms from the merged list to the original query and executed this expanded query against the target collection using our version of Okapi BM25.

Table V provides a summary of our Robust track results. Each line provides results for a single run over various topic sets. The "Old Topics" were taken from the adhoc tasks of previous TREC evaluations; the "New Topics" were created for TREC 2003. For each set of topics the table reports values for the three measures used in the Robust track: 1) average precision ("avgp"), 2) the percentage of topics with no relevant documents in the top ten ("norel"), 3) and the mean average precision over the 25% of the topic set on which the run exhibited its worst performance ("bad").

The first five lines give the results for our five official runs: The pair uwmtCR0 and uwmtCR1 are description-only runs; the pair uwmtCR2 and uwmtCR3 are title-only runs. For the fifth run (uwmtCR4), we extended the feedback process to extract both single words and word bigrams. The last four lines give results for unofficial runs that use the topic title and description. For the sixth and seventh runs, the title and description were merged into a combined query and evaluated using the procedure above. For the last pair of runs, we fused our official title-only and description-only runs with the CombMNZ algorithm [5], [7].

As expected, feedback has a substantial positive impact on average precision. The impact is greatest on the old topics, were average precision increases by 31-65%. Over the new topics, performance improves by 11-30%. Overall, feedback has a positive impact on the "bad" robustness measure, but unfortunately it often has a negative impact on the "norel" measure.

VII. Question Answering Track

While we continue to develop our QA system, we did not submit runs for the main task of the QA track. The number of new requirements for the task, and our interest in other tasks, precluded our full participation.

We did submit runs for the passage retrieval task. These runs combine aspects of our TREC 2001 and 2002 QA systems. Using our TREC 2002 system [2], we extracted exact answers from passages returned by the Qap algorithm. We then used techniques from our TREC 2001 system to locate 250-byte fragments that contain both the exact answers and related terms.

VIII. Conclusion

The MultiText system supports a variety of standard and non-standard IR techniques. Depending on the track, we have combined these techniques in different ways to produce competitive runs. For the Genomics Track we merged the results of structured (boolean) and unstructured (term set) queries. Queries were expanded by re-writing terms and through feedback over the chemical names contained in metadata. For the Robust Track, we utilized a new pseudo-relevance feedback method developed from our existing QA system. Queries were expanded through feedback over an expanded collection that included a terabyte of Web data.

References

[1] Charles L. A. Clarke and Gordon V. Cormack. Shortest substring retrieval and ranking. *ACM Transactions on Information Systems*, 18(1):44–78, January 2000.

[2] Charles L. A Clarke, Gordon V. Cormack, Graeme Kemkes, Michae l Laszlo, Thomas R. Lynam, Egidio L. Terra, and Philip L. Tilker. Statistical selection of exact answers. In *Eleventh Text REtrieval Conference (TREC-2002)*, Gaithersburg, Maryland, November 2002.

[3] Charles L. A. Clarke, Gordon V. Cormack, and Thomas R. Lynam. Exploiting redundancy in question answering. In *24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 358–365, New Orleans, September 2001.

[4] Charles L. A. Clarke, Gordon V. Cormack, Thomas R. Lynam, and Egidio L. Terra. Question answering by passage selection. In *Advances in Open Domain Question Answering*. Kluwer Academic Publishers, 2003. To appear.

[5] E. A. Fox and J. A. Shaw. Combination of multiple searches. In *Second Text REtrieval Conference (TREC-2)*, Gaithersburg, Maryland, November 1994.

[6] K. L. Kwok and M. Chan. Improving two-stage ad-hoc retrieval for short queries. In *21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 250–256, Melbourne, Australia, August 1998.

[7] Joon Ho Lee. Analyses of multiple evidence combination. In *20st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 267–276, Philadelphia, July 1997.

[8] S. E. Robertson, S. Walker, and M. Beaulieu. Okapi at TREC-7. In *Seventh Text REtrieval Conference (TREC-7)*, Gaithersburg, Maryland, November 1998.

[9] Amit Singhal, John Choi, Donald Hindle, David D. Lewis, and Fernando Pereira. At&t at TREC-7. In *Seventh Text REtrieval Conference (TREC-7)*, Gaithersburg, Maryland, November 1998.

Interactive search refinement techniques for HARD tasks

Olga Vechtomova[*] Murat Karamuftuoglu[**] Eric Lam[***]

[*] Department of Management Sciences, University of Waterloo, Waterloo, Canada
ovechtom@engmail.uwaterloo.ca
[**]Department of Computer Engineering, Bilkent University, Ankara, Turkey
hmk@cs.bilkent.edu.tr
[***]Department of Computer Science, University of Waterloo, Waterloo, Canada
ekhlamlo@student.cs.uwaterloo.ca

Abstract

In our entry to the new HARD track, we have investigated two methods of interactively refining user search formulations. One method consists of asking the user to select a number of sentences that may represent relevant documents, and then using the documents, whose sentences were selected for query expansion. The second method is to show to the user a list of noun phrases, extracted from the initial document set, and then expanding the query with the terms from the phrases selected by the user. The results show that the second method is an effective means of interactive query expansion and yields significant performance improvements.

1. Introduction

The main goal of the HARD track this year is to explore what techniques could be used to improve search results by using two types of information: (1) extra-linguistic contextual information about the user and the information need, and (2) information dynamically provided by the user in response to topic clarification questions.

We decided to focus this year on designing techniques for dynamically eliciting additional topic-oriented search criteria from the users, and using their feedback in the second-stage search iterations. We submitted one baseline run, two clarification forms and two final runs. The following subsections describe the system design for each of the runs.

2. System description

2. 1 Baseline run

For all of our submitted runs we used Okapi BSS (Basic Search System). For the baseline run *UWAThard1* we used keywords only from the title fields of topics, as these proved to be most effective in our preliminary experiments described in section 2.2.3. The topic titles were parsed in Okapi, weighted and searched using BM25 function against the HARD track corpus. We used a GSL file, constructed on the basis of past TREC data, and comprised of 222 stopwords, 254 semi-stopwords (terms that are indexed, but not used in blind or relevance feedback), 58 phrases and 432 synonym groups.

2. 2 Query expansion method 1

According to the track specifications, a clarification form for each topic must fit into a 1152 x 900 screen and the user may spend no more than 3 minutes filling out each form. We have evaluated two clarification forms, the first of which – CF1 – is described in this section.

In brief, our first approach to eliciting user feedback was to take *N* top-ranked documents from the baseline run, select one sentence per document, include it in the clarification form CF1, and ask the user to select all sentences that possibly represent relevant documents.

The goal that we aim to achieve with the aid of the clarification form CF 1 is to have the users judge as many relevant documents as possible on the basis of one sentence per document. The main research questions that we explored here were:

RQ1: What is the error rate in selecting relevant documents on the basis of one sentence representation of its content?
RQ2: How does sentence-level relevance feedback affect retrieval performance?

2.2.1 Sentence selection

In more detail the sentence selection algorithm consists of the following steps:
From the baseline run, we take N top-ranked documents. Given the screen space restrictions, we could display only 15 three-line sentences, hence *N*=15. The full-text of each of the documents is split into sentences. For every sentence that contains one or more query terms, i.e. any term from the title field of the query topic, two scores are calculated: S1 and S2.

Sentence selection score 1 (S1) is the sum of *idf* of all query terms present in the sentence.

$$S1 = \sum idf_q \tag{1}$$

Sentence selection score 2 (S2):

$$S2 = \frac{\sum W_i}{f_s} \tag{2}$$

Where: W_i – Weight of the term *i*, see (3);
f_s – length normalisation factor for sentence *s*, see (4).

The weight of each term in the sentence, except stopwords, is calculated as follows:

$$W_i = idf_i(0.5 + (0.5 * \frac{tf_i}{t\max})) \tag{3}$$

Where: idf_i – inverse document frequency of term *i* in the corpus;
tf_i – frequency of term *i* in the document;
tmax – *tf* of the term with the highest frequency in the document.

To normalise the length of the sentence we introduced the sentence length normalisation factor *f*:

$$f_s = \frac{s\max}{slen_s} \tag{4}$$

Where: *smax* – the length of the longest sentence in the document, measured as a number of terms, excluding stopwords;
slen – the length of the current sentence.

All sentences in the document were ranked by S1 as the primary score and S2 as the secondary score. The rationale of our approach to sentence ranking is to pre-select, first, the sentences that contain more query terms, and therefore are more likely to be related to the user's query formulation, and secondly, from this pool of sentences to select the one which is more content-bearing and central to the topic of the document, i.e. which contains a higher proportion of terms with high tf*idf weights.

Next, since we are restricted by the screen space, we reject sentences that exceed 250 characters, i.e. three lines. In addition, to avoid displaying very short, and hence insufficiently informative sentences, we reject sentences with less than 6 non-stopwords. If the top-scoring sentence does not satisfy the length criteria, the next sentence in the ranked list is considered to represent the document.

Finally, since there are a number of almost identical documents in the corpus, we remove the representations of the duplicate documents from the clarification form using pattern matching, and process the necessary number of additional documents from the baseline run sets.

By selecting the sentence with the query terms and the highest proportion of high-weighted terms in the document, we are showing query term instances in their typical context in this document. Usually, but not always, a term is only used in one sense in the same document. Also, in many cases it is sufficient to establish the linguistic sense of a word by looking at its immediate neighbours in the same syntactic unit (i.e. a sentence or a clause). Based on this, we hypothesise that users will be able to filter out those sentences, where the query terms are used in an unrelated linguistic sense. We, however, recognise that it is more difficult, if not impossible, for users to reliably determine the relevance of the document on the basis of one sentence in those cases where the relevance of the document to the query is due to more subtle aspects of the topic, which are not evident from one sentence.

We evaluated CF1 on Financial Times and Los Angeles Times corpora from TREC volumes 4 and 5, and ad hoc topics 301-350. Forms were filled in by the authors. The average precision of user-selected sentences, calculated as the number of relevant sentences selected by the user out of the total number of sentences selected by the user, was 0.70 in our experiments. The average recall, calculated as the number of relevant sentences selected by the user out of the total number of relevant sentences shown in the clarification form, was 0.64. The average number of relevant documents shown was 5.36; average number of relevant selected sentences - 3.44; non-relevant selected – 1.44

2.2.2 Query expansion algorithm

The user's feedback to clarification form 1 is used for obtaining query expansion terms for the final run. Our approach to query expansion is to identify collocates of query terms – words co-occurring within a limited span with query terms. Previous query expansion experiments with long-span collocates of query terms obtained from 5 known relevant documents showed 72-74% improvement over the use of title only query terms on the Financial Times (TREC volume 4) corpus with TREC-5 ad hoc topics [4].

For the HARD track experiments we slightly modified our collocate extraction and selection algorithm. Instead of using fixed-size windows around instances of query terms to extract collocates, we define a window as one or more sentences surrounding the query term occurrence. The span of the window is measured as the number of sentences to the left and right of the sentence containing the instance of the query term. For example, span 0 means that only terms from the same sentence as the query term are considered as collocates, span 1 means that terms from 1 preceding and 1 following sentences are also considered as collocates.

In more detail the collocate extraction and ranking algorithm is as follows:
Each document judged relevant is split into sentences. For each query term we extract all sentences containing its instance, plus *s* sentences to the left and right of these sentences, where *s* is the span size. If *s*>0 we may have overlapping windows and extract the same sentence several times. To avoid this we keep the record of the sentences already extracted, so that each sentence is only extracted once.

After all required sentences are selected we extract stems from them, discarding stopwords. For each unique stem we calculate the Z score to measure the significance of its co-occurrence with the query term as follows:

$$Z = \frac{f_r(x,y) - \frac{f_c(y)}{N} f_r(x) v_x(R)}{\sqrt{\frac{f_c(y)}{N} f_r(x) v_x(R)}} \qquad (5)$$

Where: $f_r(x,y)$ – frequency of x and y occurring in the same windows in the known relevant document set (R);
$f_c(y)$ – frequency of y in the corpus;
$f_r(x)$ – frequency of x in the relevant documents;
$v_x(R)$ – average size of windows around x in the known relevant document set (R);
N – corpus size in words.

The joint frequency of x and y – $f_r(x,y)$ is calculated as the product of $f(x)$ and $f(y)$ in that window.

All collocates with an insignificant degree of association: Z<1.65 are discarded, see [2]. The remaining collocates are sorted by their Z score.

After we obtain sorted lists of collocates of each query term, we select those collocates for query expansion, which co-occur significantly with two or more query terms. First, for each collocate the collocate score (C1) is calculated:

$$C1 = \sum n_i W_i \qquad (6)$$

Where: n_i – rank of the collocate in the z-sorted collocation list for the query term *i;*
W_i – weight of the query term *i*.

The reason why we use the rank of the collocate in the above formula instead of its Z score is because Z scores of collocates of different terms are not comparable.
Finally, we rank collocates by two parameters:

(1) the number of query terms they co-occur with;
(2) C1 score.

Top *k* collocates in the ranked list are added to the original query terms.

2.2.3. Evaluation

We evaluated the algorithm with blind feedback, trying to find the optimal values for *R* - the size of the pseudo-relevant set, *s* – the span size, and *k* – the number of query expansion terms. The results indicate that variations of these parameters do not have significant effect on precision. However, some tendencies were observed, namely: (1) larger *R* values tend to lead to poorer performance in both Title and Title+Desc. runs; (2) larger span sizes also tend to degrade performance in both Title and Title+Desc runs.

The Title-only unexpanded run was 10% better than Title+Description. Similarly, the expansion of Title-only queries performed better than expansion of Title+Description queries. For example, AveP of the worst Title+Desc expansion run (R=50, s=4, k=40) is 23% worse than the baseline, and AveP of the best run (R=5, s=1, k=10) is 8% better than the baseline. AveP of the worst Title expansion run (R=50, s=5, k=20) is 4.5% worse than the baseline, and AveP of the best Title expansion run (R=5, s=1, k=40) is 10.9% better than the baseline.

Based on this data we decided to use terms only from the Title section of the topics for the official runs, and, given that values *k*=40 and s=1 contributed to a somewhat better performance, we used these values in all of our official expansion runs. The question of *R* value is obviously irrelevant here, as we used all documents selected by users in the clarification form.

2.3 Query expansion method 2

The second user feedback mechanism that we evaluated consists of automatically selecting noun phrases from the top-ranked documents retrieved in the baseline run, and asking the users to select all phrases that contain possibly useful query expansion terms. The research questions explored here were:

RQ3: Do noun phrases provide sufficient context for the user to select potentially useful terms for query expansion?

RQ4: How does query expansion based on user-selected phrases affect retrieval performance?

We take top 25 documents from the baseline run, and select 2 sentences per document using the algorithm described above. We then apply Brill's rule-based tagger [1] and BaseNP noun phrase chunker [3] to extract noun phrases from these sentences. The phrases are then parsed in Okapi to obtain their term weights, removing all stopwords and phrases consisting entirely of the original query terms. The remaining phrases are ranked by the sum of weights of their constituent terms. Top 78 phrases are then included in the clarification form for the user to select. This is the maximum number of phrases that could fit into the clarification form.

All user-selected phrases were split into single terms, which were then used to expand the original user query. The expanded query was then searched against the HARD track database using BM25 function for document retrieval and BM250 for passage retrieval.

3 Results

3.1 Document-level evaluation

The document-level results of the three official runs are presented in table 1. UWAThard1 is the baseline run using original query terms from the topic titles. UWAThard2 is an experimental run using query expansion method 1 plus the granularity and known relevant documents metadata. UWAThard3 is an experimental run using query expansion method 2 plus the granularity metadata.

Run	Run description	Soft relevance		Hard relevance	
		Precision @ 10	Average Precision	Precision @ 10	Average Precision
UWAThard1	Original title-only query terms; BM25 used for all topics	0.4875	0.3134	0.3875	0.2638
UWAThard2	Query expansion method 1; granularity and rel.docs metadata	0.5479	0.3150	0.4354	0.2978
UWAThard3	Query expansion method 2; granularity metadata	0.5958	0.3719	0.4854	0.3335

Table 1. Document-level evaluation results

UWAThard2 did not achieve statistically significant improvement over the baseline. This finding did not correspond to our initial test runs with the FT and LA collections, which showed 21% improvement over the original title-only query run. The analysis of the numbers of relevant and non-relevant sentences selected by the annotators showed slightly better results of 0.73 in average precision and 0.69 in average recall, compared to the selections that we made from clarification forms based on the FT and LA collections (see section 2.2.1). On average 7.14 relevant sentences were included in the forms. The annotators on average selected 4.9 relevant and 1.8 non-relevant sentences. Figure 1 shows the number of relevant/non-relevant selected sentences by topic. It is not clear why query expansion method 1 performed worse in the official UWAThard2 run compared to the test run on FT and LA collection, given very similar numbers of relevant sentences selected. Corpus differences could be one reason – HARD corpus contains a large proportion of governmental documents, and we have only evaluated our algorithm on newswire corpora. More experimentation is needed to determine the effect of the governmental documents on our query expansion algorithm.

In addition to clarification forms, we used the known relevant documents metadata for UWAThard2. We need to conduct more experiments without this metadata to determine how much it contributed to the performance of our query expansion method.

Our second query expansion run (UWAThard3) was among the best runs in the track, gaining an 18% improvement over the baseline in average precision in soft-doc evaluation and 26.4% in hard-doc evaluation, both of which are statistically significant (using t-test at .05 significance level). These are the highest improvements over the baseline among the top 50% highest-scoring baseline runs submitted to the track. Query expansion method 2 achieved lower performance gains in our training runs on FT and LA collections, which can be explained by the lower number of phrases selected. LDC annotators selected on average 19 phrases, whereas we selected on average 7 phrases in the training runs. This suggests that selecting more phrases leads to a notably better performance. The reason why we selected fewer phrases than the LDC annotators could be due to the fact that on many occasions we were not sufficiently familiar with the topic, and could not determine how an out-of-context phrase is related or not related to the topic. The LDC annotators are more familiar with the topics, which they have formulated.

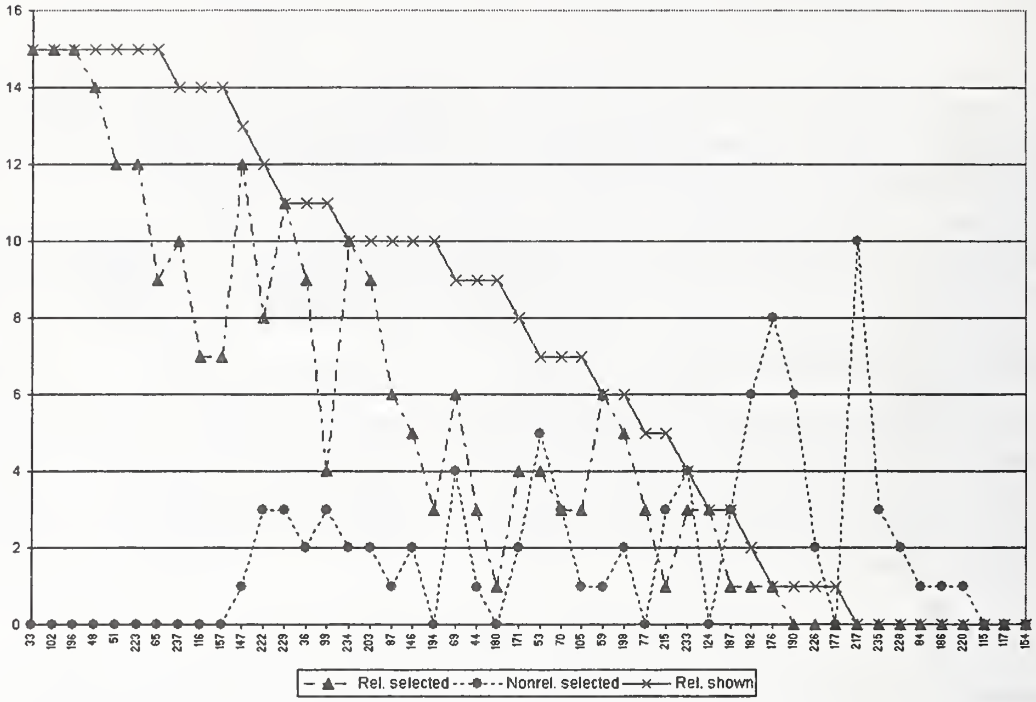


Figure 1. Sentences selected by LDC annotators from the clarification form 1.

3.2 Passage-level evaluation

Passage-level evaluation results are given in table 2. UWAThard3 showed 27% improvement in R-precision over UWAThard1, while UWAThard2 – 23%. Such big difference between the expansion runs and the baseline was expected, since we only did document-level retrieval for the baseline run. All our runs were above the median in all passage-level measures

Run	Passage P@10	R-Precision	F(30)
UWAThard1	0.2668	0.1908	0.1255
UWAThard2	0.3305	0.2359	0.1454
UWAThard3	0.3617	0.2426	0.1559

Table 2. Passage-level evaluation results

4 Conclusions and future work

This year we experimented with two user-assisted search refinement techniques:

(1) inviting the user to select from the clarification form a number of sentences that may represent relevant

documents, and then using the documents whose sentences were selected for query expansion.

(2) showing to the user a list of noun phrases, extracted from the initial document set, and then expanding the query with the terms from the user-selected phrases.

Comparison with other submissions to the HARD track (88 in total) shows that our submitted runs are above the median in all official evaluation measures. The second query expansion method is more promising than the first, and was among the best runs this year, achieving statistically significant improvement of 18% (soft-rel) and 26% (hard-rel) over the baseline.

In this year's entry we focused on utilising the user's feedback to clarification forms plus granularity and known relevant documents metadata. For the next year's entry, we plan to address other metadata: genre, familiarity and purpose.

Acknowledgements

This material is based on work supported in part by Natural Sciences and Engineering Research Council of Canada.

References

[1] Brill. E. Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging. Computational Linguistics, 21(4), 1995, pp. 543-565.

[2] Church K., Gale W., Hanks P., Hindle D. Using statistics in lexical analysis. *In Lexical Acquisition: Using On-line Resources to Build a Lexicon*, ed. U.Zernik, Englewood Cliffs, NJ: Lawrence Elbraum Associates, 1991, pp. 115-164.

[3] Ramshaw L., Marcus M., Text Chunking Using Transformation-Based Learning. Proceedings of the Third ACL Workshop on Very Large Corpora, MIT, June, 1995

[4] Vechtomova O., Robertson S.E., Jones S. Query expansion with long-span collocates. Information Retrieval, 6(2), 2003, pp. 251-273.

Ranking Function Discovery by Genetic Programming for Robust Retrieval

Li Wang[1], Weiguo Fan, Rui Yang, Wensi Xi, Ming Luo, Ye Zhou, Edward A. Fox
Department of Computer Science
Virginia Polytechnic Institute and State University
{liwang5, wfan, yrui, xwensi, lming, yezhou, fox}@vt.edu

Abstract

Ranking functions are instrumental for the success of an information retrieval (search engine) system. However nearly all existing ranking functions are manually designed based on experience, observations and probabilistic theories. This paper tested a novel ranking function discovery technique proposed in [Fan 2003a, Fan2003b] – **ARRANGER** (Automatic gene**R**ation of **RAN**king functions by **GE**netic p**R**ogramming), which uses Genetic Programming (GP) to automatically learn the "best" ranking function, for the robust retrieval task. Ranking function discovery is essentially an optimization problem. As the search space here *is not a coordinate system*, most of the traditional optimization algorithms could not work. However, this ranking discovery problem could be easily tackled by ARRANGER. In our evaluations on 150 queries from the ad-hoc track of TREC 6, 7, and 8, the performance of our system (in average precision) was improved by nearly 16%, after replacing Okapi BM25 function with a function automatically discovered by ARRANGER. By applying pseudo-relevance feedback and ranking fusion on newly discovered functions, we improved the retrieval performance by up to 30%. The results of our experiments showed that our ranking function discovery technique – ARRANGER – is very effective in discovering high-performing ranking functions.

1. Introduction

Text resources in digital format are quickly increasing with the rapid development of the IT industry. This tremendous collection of resources serves as a rich repository for our society in general. However, it also brings challenges to the general public. How to use this repository effectively is one of the biggest challenges. Researchers have developed various information retrieval systems, also known as search engines, to help people quickly and accurately find what they need from this repository. The working process for an information retrieval (search engine) system can be simplified to the process of returning an ordered document list according to a user's information need (expressed as queries). Therefore the most critical part for an IR system is its ranking function, which is used to order documents based on their similarity degrees to a user query. Designing a good ranking function, however, is not an easy task. There are many well-known ranking functions, such as Okapi BM25, TFIDF, and INQUERY. But most of those ranking functions are manually designed by experts based on heuristics, experience, observations, and statistical theories. One novel part of our work is that we use a Genetic Programming (GP) based technique called ARRANGER (Automatic gene**R**ation of **RAN**king functions by **GE**netic p**R**ogramming) to discover ranking functions automatically [Fan 2003a, Fan2003b]. Ranking functions usually could not work consistently well under all situations. Various information retrieval studies have shown that the performance of a ranking function is very context-dependent [Salton & Buckley, 1988; Zobel & Moffat, 1998]. The context may depend on text collections or even properties of queries. Using a static ranking function can not guarantee good performance under all situations. How to find the "optimal" ranking function for a specific context is quite a challenge. The advantage of ARRANGER is that it can learn the "optimal" ranking functions according to different contexts by effectively combining multiple types of evidence in an automatic and systematic way. Using 150 queries from the ad-hoc task of the Robust Track in TREC 6, 7, and 8, we found ranking functions discovered with ARRANGER

[1] Li Wang is now at the University of Michigan (wang@umich.edu).

improved the performance of our baseline system, which uses the Okapi BM25 ranking function, by 8 ~ 16%. Based on those newly discovered ranking functions, we also tried other performance improvement techniques such as pseudo-feedback (query expansion) and information fusion which combines scores from different ranking functions using a regression technique. These techniques altogether help improve our performance by up to 30% in average precision over the baseline Okapi system.

Our paper is organized as follows. Section 2 states our research objectives. Section 3 describes basic data processing steps. Section 4 reviews ARRANGER – a GP-based ranking function discovery technique. Section 5 summarizes other techniques used in our system and gives a detailed description of our final submissions. Section 6 shows the official submission results in comparison with the other TREC teams. We conclude our paper in Section 7.

2. Research objectives

We have two objectives in this year's Robust Track:

1) We want to test the ARRANGER framework proposed in [Fan 2003a, Fan 2003b] to see whether it can work well on more heterogeneous collections.
2) We want to test whether the newly discovered ranking functions can work well with other performance improvement techniques such as query expansion through blind feedback, and ranking fusion using logistic regression.

3. Data processing

All our experiments were run on a two-2.3GHz processor Dell Server running the Linux operating system. Since our concentration in TREC is to test our GP-based ranking function discovery technique, ARRANGER, we didn't take advantage of the document structure. Past TREC results also showed that structure information didn't help in these data. In the parsing process, we simply removed the non-informative content in the collection and kept only the texts in the TEXT field. These texts were indexed into both forward index and inverted index formats for our experimental purposes after removing stop words and stemming. No phrases were used in our experiments.

For query processing, we indexed three different versions of the topic descriptions. The first version is description queries, which are generated based on the Description field only as required by the Robust Track. The second, short queries, are based on the Title and Description fields. The third, long queries, are extracted based on all fields from the topic description.

4. Ranking function discovery based on Genetic Programming

4.1 Genetic Programming

Genetic Programming (GP), an extension of Genetic Algorithms (GA), is an artificial intelligence technique, inspired by Darwin's theory of evolution. "Computer programs that evolve in ways that resemble natural selection can solve complex problems even their creators do not fully understand" [Holland, 1975]. Genetic Programming has been widely used and approved to be effective in solving optimization problems, such as financial forecasting, engineering design, data mining, and operations management. GP makes it possible to solve complex problems for which conventional methods can not find an answer easily.

In Genetic Programming, a large number of individuals, called a population, are maintained at each generation. An individual represents a tentative solution for the target problem. All these solutions form a space, say, Σ. In reality, individuals could be stored using complex data structures, such as a tree, a linked list, or a stack. A tree is the most popular form to store and represent individuals. Figure 1a shows an example of a tree which represents the expression (X + Y)*Z. Now, as our target in TREC is to find an "optimal" ranking function to sort documents in the collection, individuals should represent tentative ranking functions. Figure 1b shows an individual representing a ranking function. A fitness function (f (·): Σ → R) is also needed in Genetic Programming. A fitness function takes the solution space, Σ, as its domain and returns a real number. Hence tentative solutions, represented by individuals, could be measured and ordered according to their return values. The return value of a fitness function must appropriately measure how well an individual, which represents a solution, can solve the target problem.

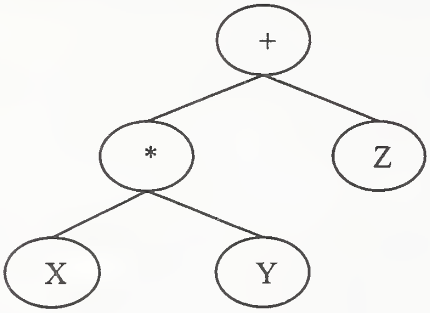


Figure1a. A simple expression represented by a tree

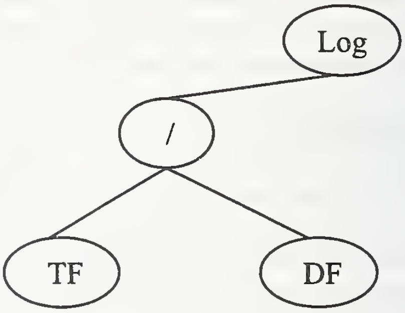


Figure 1b. A simple ranking function

Genetic Programming searches for the "optimal" solution by evolving the population generation after generation. Individuals in the new generation are produced based on those in the current one. Three genetic operators are usually used to produce the new generation. They are Reproduction, Crossover, and Mutation. The reproduction operator directly copies or, in a more appropriate term, clones some individuals into the next generation. The probability for an individual to be selected for Reproduction should be proportional to its fitness. Therefore the better a solution solves the problem, the higher probability it has to enter the next generation. While Reproduction keeps the best individuals in the population, Crossover and Mutation introduce transformation and so provide variations to enter into the new generation. The crossover operator randomly picks two groups of individuals, selects the best individual in each of two groups as parent according to their fitness, exchanges a randomly selected gene fragment of each parent and produces two "children". Thus, a "child" may obtain the good fragments of its excellent parents and may exceed them further, providing a better solution to the problem. Since parents are selected from a "competition", good individuals are more likely to be used to generate offspring. The mutation operator randomly changes a gene code, which could be a function or a parameter in our ranking function discovery task, of an individual. Figure 2 shows how the Crossover operator works. Using these genetic operators, a new generation is produced. The new generation keeps individuals with the best fitness in the last generation and takes in more "fresher air", providing creative solutions to the target problem. Better solutions are obtained either by inheriting and reorganizing old ones or by lucky mutation, simulating Darwinian Evolution. As we can see, Genetic Programming takes a so-called *stochastic search* approach, intelligently, extensively, and "randomly" searching for the optimal point in the entire solution space. It is less likely to be trapped in the local optima, which is the major problem of many other search algorithms. It provides sound solutions to many arduous problems, for which people have not found a theoretical or practical breakthrough.

4.2 Motivation for using Genetic Programming in ranking function discovery

A ranking function plays an essential role in an IR system (or search engine). It evaluates the similarity degree of a document to the query, so documents can be ranked according to its returned value. However, many empirical studies have shown inconsistent performance by existed well-known ranking functions on various collections [Salton & Buckley, 1988; Zobel & Moffat, 1998]. The same ranking function may work well on one collection, but poorly on others. They are collection-sensitive, and sometimes even query-sensitive. Given a specific context, how to select the right one from available ranking functions or how to design a new function for a given context has not been fully studied before.

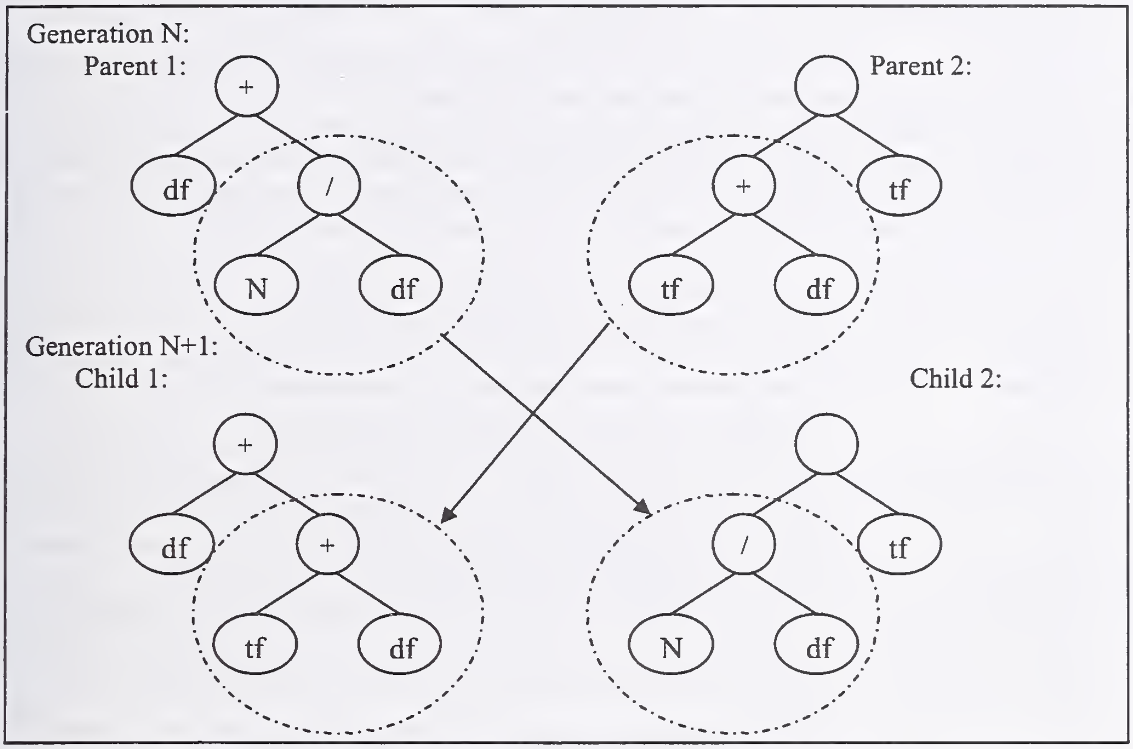


Figure 2. A simple example shows how the Crossover operator works in our ranking function discovery task.

Nearly all the existed ranking functions are manually designed, based on experience, heuristics and probability theory. Some parameters in these functions are usually adjusted to accommodate collection differences. However, these functions should still be categorized as static ranking functions, since the function structure is untouched and the effect of such adjustment is limited. Our GP-based ranking function discovery approach provides a framework which could automatically learn the "optimal" ranking function for the given context. As the structures of discovered ranking functions are not constrained, these customized functions could provide striking performance on the target collection where static ranking functions can not.

Ranking function discovery is essentially an optimization problem. We are looking for the global optimal point in the space, which consists of all the possible ranking functions. However this task is completely different from the traditional high-dimension optimization problem, since the space of ranking functions *is no longer a coordinate system* (As in Abstract, make this clearer. Do you mean a vector space or metric space or measure space?). Conventional approaches for solving optimization problems, such as conjugate gradient, linear programming, nonlinear programming, and simulated annealing, can hardly work here. Also the ranking function space consists of an infinite number of elements, which makes it impossible to get the "optimal" point for random search and exhaustive search. As we showed before, functions could be expressed by trees. We can actually treat the ranking function space as a space consists of all kinds of tree structures. Genetic Programming shows its sharp edge in solving such kind of problems, since its internal tree structure representation for "individuals" can be perfectly used for describing ranking functions. This is the major motivation to choose GP for the ranking function discovery task.

4.3 Outline of our GP-based ranking function discovery system – ARRANGER

In this section, we give a brief introduction to the ARRANGER engine. Please refer to [Fan 2003a, Fan 2003b] for a more detailed introduction and for validation.

Basically a ranking function consists of three parts: variables, constants, and operations (which connect the first two parts). Hence we need to identify all the potential variables that are used in the ranking function by ARRANGER. Some examples for these variables are tf, tf_query, tf_max, length, N, tf_avg, tf_Avg_Col, df_max_Col, df, etc. Table 1 gives the meaning of these variables.

tf	Query term frequency in the document (vector)
tf_query	Query term frequency in the query (vector)
tf_max	The maximum term frequency in a document (scalar)
Length	Document length in the number of words (scalar)
Length_avg	Average document length in the number of words (scalar)
N	Number of documents in the collection (scalar)
tf_avg	Average term frequency in the current document (scalar)
tf_avg_Col	Average term frequency for all the documents in the collection (scalar)
df_max_Col	Maximum document frequency for a word in the collection (scalar)
df	Document frequency for the query words (vector)

Table 1. Definitions for variables

There are two different types of variables, scalar and vector. Some of these predefined variables are summaries calculated for the whole collection or a specific document, such as tf_max, N, tf_Avg_Col, etc. These variables belong to the category of scalar variable. The remaining variables have vector nature, such as tf_doc and tf_query. We defined that when such variables appear in a ranking function, they represent vectors, instead of single numbers. For example, if a query has n words in it, tf_doc could be represented by $(x_1, x_2, \ldots, x_n)$, where x_i ($i = 1,2,\ldots,n$) is the term frequency (tf) of the query's *i*th word in the document. For constants, they are defined to be scalar only. Based on pre-selected variables and constants, we define two types of functions (operations), single-parameter functions (denoted by $\sigma(\cdot)$) and two-parameter functions (denoted by $\circ$). Single-parameter functions include log( ) and sqrt( ). Two-parameter functions include +, -, *, /. Some functions, such as log( ), sqrt( ) and /, need to be protected, since the domain of these functions is not the whole real number space. As a variable could be a scalar or a vector, those functions must take that into consideration. For one-parameter functions, we define $\sigma(x) = y$ and $\sigma((x_1, x_2, \ldots, x_n)) = (\sigma(x_1), \sigma(x_2), \ldots, \sigma(x_n))$, where x, y and x_i represent scalar variables and

$(x_1, x_2, \ldots, x_n)$ is used to represent vectors. For two-parameter functions, we define $x \circ y = z$, $x \circ (x_1, x_2, \ldots, x_n) = (x \circ x_1, x \circ x_2, \ldots, x \circ x_n)$ and $(x_1, x_2, \ldots, x_n) \circ (y_1, y_2, \ldots, y_n) = (x_1 \circ y_1, x_2 \circ y_2, \ldots, x_n \circ y_n)$, where x, y, z, x_i and y_i represent scalars and $(x_1, x_2, \ldots, x_n)$ represent a vector. Following our definitions for variables and functions, when a vector variable appears in the ranking function, the final result also is a vector, where a scalar usually is needed to measure the similarity degree between a document and a query. In this case, we further define that the return value of a ranking function is the summation of all the elements when a vector is finally returned by that function. Based on all those rules defined by us, the ARRANGER could work on discovering ranking functions. Also when we plug in the newly-discovered functions into our search engine, the same rules must be followed.

Queries in the TREC 6, 7, and 8 Ad-Hoc task (topic 301- 450) are used to discover ranking functions. According to the procedure described in section 3, the collection is first processed into dictionary and inverted files, such that our search engine can work on them. For each query, the search engine returns the top 5000 document names using an arbitrary function. Any popular ranking functions, which have been proved effective, could be used for this purpose. We used the Okapi BM25 ranking function for this first scan. On average more than half of all the relevant documents are listed in the top 5000 documents for each query. Therefore those documents have included enough relevant documents, whose properties could be learned later. According to the relevance judgments, those documents are separated into two groups, relevant and nonrelevant. Each group needs to be randomly divided into three parts, called training, validation, and testing data set. Then we randomly combine the relevant and nonrelevant documents associated with each data set. Now the training, validation, and testing sets all include relevant and nonrelevant documents in random order. The fitness value for a ranking function is the average precision we could get in our system when using that function.

The framework of ARRANGER works as follows: First, the best ranking functions learned from the training set are stored and the rest are discarded. Then those functions are tested on the validation set. According to their performance, the functions which do not have consistent performance on both data sets are screened out. Finally, "survived" functions are tested again on the test data set. The same screening rule follows. Only the most robust and consistent functions are selected and they form the ranking function candidate pool. Since an appropriate stopping rule is hard to find for the Genetic Programming approach, over-training is inevitable unless protecting rules are set. By running the ranking functions on two other independent data sets, over-trained functions are filtered out once performance inconsistencies appear.

We used ARRANGER to discover "optimal" functions on the Robust Track collection. We tested the automatically learned functions on three types of queries: description query, short query, and long query as described in the Section 3. Table 2 shows the results on the entire collection. From this table, you can see that significant improvement is achieved by replacing the Okapi BM25 function with our newly-discovered functions.

	Description query (average precision)	Short query (average precision)	Long query (average precision)
Okapi BM25 (baseline)	0.1880	0.2194	0.2375
GP func1	0.2173 (+15.6%)	0.2394 (+9.1%)	0.262 (+10.3%)
GP func2	0.2079 (+ 10.6%)	0.2317 (+5.6%)	0.2607 (+9.8%)
GP func3	0.2047 (+ 8.9%)	0.2282 (+4.0%)	0.259 (+9.1%)
GP func4	0.2036 (+8.3%)	0.2245 (+2.3%)	0.2602 (+9.6%)

Table 2. Performance comparison of Okapi BM25 and GP functions on 150 queries of Ad-Hoc task at TREC 6, 7, and 8.

5. Other performance improvement techniques

5.1 Pseudo-relevant feedback

Pseudo-relevance feedback (automatic query expansion) is the process of adding more terms to a user's query to promote performance of search engines. It is a widely-used and effective technique, especially for very short queries. In pseudo-relevance feedback, a small number of documents are first retrieved according to the user's query and these documents are assumed to be relevant. Words in those documents as well as words in the original query are sorted according to a weighting function. An expanded query is generated by selecting some words from this list. There are many variations in using different weighting functions and strategies to select words for the new query.

We apply various pseudo-relevance feedback techniques, based on new functions discovered by our ARRANGER. They are Rocchio, Ide dec-hi, CHI, KLD, RSV, DRC, and a variation of KLD, which we deduced by probability theory. Those techniques are applied on both description queries and long queries. They provide significant performance improvement on both types of queries. As we expected they improve more on description queries than long queries. For each approach, there are several parameters to be adjusted, for example, the number of documents assumed relevant, the number of terms for the expanded query, and parameters in the weighing function. A factorial design was used to look for the "best" parameter settings, which provides at the same time a high performance mean and low performance variation for ad-hoc tasks in TREC 6, 7, and 8. After comparison, we found Rocchio and Ide dec-hi are the best query expansion schemes on our automatically learned functions. Table 3 gives the performance comparisons.

	Description query (average precision on 150 queries)	Long query (average precision on 150 queries)
GP function 1 without QE (baseline)	0.2173 (+15.6%)	0.2394 (+9.1%)
GP function 1 + Rocchio	0.2422 (+28.9%)	0.2661 (+ 12.0%)
GP function 1 + Ide Dec-Hi	0.2390 (+27.1%)	0.2744 (+15.5%)

Table 3. – The effects of pseudo-relevance feedback on performance

5.2 Rank fusion – combine scores from different ranking functions

Since many high quality ranking functions have been learned, an old saying "two heads are better than one" could be used in our system to further improve performance. In our experiment, three GP-based functions and Okapi BM25 are combined to produce a new ranking function. Because the relevance judgment only provides binary relevant (1) and nonrelevant (0) information, logistic regression is an appropriate tool to find such a relationship. Let p denote the probability that a document is relevant to the query and let gp1, gp2, gp3, and okp represent scores returned by our three GP-based functions and the Okapi BM25 function for this document, respectively. Our initial model is

$$\text{logit}(p) = \beta_0 + \beta_1 * \text{gp1} + \beta_2 * \text{gp2} + \beta_3 * \text{gp3} + \beta_4 * \text{okp} + \text{INT}$$

INT includes all the possible two factor, three factor, and four factor interactions. Only after including interaction terms, the similarity degree between a document and query could be appropriately measured when conflict scores are given by different ranking functions. Otherwise a main-effect-only model can not fit the data well.

For each of 150 queries, the search engine generates names and scores of the top 300 documents returned by these four ranking functions. A union operation is applied on all the returned documents, therefore we generate a huge matrix with 5 columns (gp1 score, gp2 score, gp3 score, okp score, and relevance information). If a document was not listed in the top 1000 list by a function, its score associated with this function is assigned to 0. After model selections, we achieved such a model: logit(p) = β_0 + β_1*gp1 +β_2*gp2+β_3*gp3+β_4*okp +β_5*gp1:gp2 +β_6* gp1:gp3 +β_7* gp2:okp +β_8*gp1:gp3:okp +β_9*gp2:gp3:gp4+ β_{10}*gp1:gp2:gp3:okp, where X:Y represents the interaction between factor X and Y. All β_i's are highly significant (with p-value $< 10^{-5}$) in this model. The combined ranking function is then tested on the whole collection and the result is shown in Table 4.

	150 queries (long)	50 test old queries (long)
Okapi BM25	0.2375	0.1251
GP1 function	0.2620	0.1393
GP2 function	0.2602	0.1334
GP3 function	0.2607	0.1346
Comb function	0.2666	0.1417

Table 4. Performance comparison between combined function and other functions

The performance of the combined function is superior to all other functions from which it is generated. Another appealing property we found in experiments is that the combined function produces the smallest performance variation on TREC 6, 7, and 8 among all the ranking functions. Experiments show that our ranking function fusion approach improves not only the performance but also the consistency of the information retrieval system, although the difference is not statistically significant.

6. Results

We submitted five independent runs for this year's Robust Track. Our submissions do not involve any human intervention, so they are all automatic runs. The first four runs use all the topic fields and the last one only uses the description field of topics. Table 5 gives the detailed description of our submissions. Table 6 summarizes the final evaluation results from TREC for all 5 runs.

Run Number	Description
VTcdhgp1	In this run, we first search long queries (all fields of topics) against Robust collection, using a linearly combined ranking function (combining 3 GP functions we derived from experiments with Okapi). Secondly, we assume the top 6 documents are relevant and use Ide dec-hi approach to "expand" the description field of each query to 22 words. Finally, we search the "expanded query" against the Robust collection again using a GP ranking function, which we derived from previous experiments.
VTgpdhgp2	Same as VTcdhgp1 except that we use GP ranking function for the first search and expand the query to 14 words instead of 22 words.
VTcdhgp3	Same as VTcdhgp1 except that we expand the query to 23 words.
VTgpdhgp4	Same as VTcdhgp2 except that we expand the query to 17 words.
VTDokrcgp5	In this run, we first search description field of queries against Robust collection using Okapi BM25 ranking function. Secondly, we assume the top 8 documents are relevant and use Rocchio method to "expand" the description field of each query to 22 words. Finally, we search the "expanded query" against the Robust collection again using a GP ranking function, which we derived from previous experiments.

Table 5. Description of our five official submissions

Run No.	MAP	P10	#>Median	#Best
VTcdhgp1	0.2649	0.432	62	3
VTgpdhgp2	0.2731	0.449	69	3
VTcdhgp3	0.2637	0.432	61	2
VTgpdhgp4	0.2696	0.448	65	2
VTDokrcgp5	0.2563	0.408	60	4
In total, we contribute 14 queries that have the best performance among 100 queries				

Table 6. Official submission results. The last run is based on the description field only.

As can be seen from Table 6, we contribute 14 queries that have the best performance in 100 topics. Our last run based on the description field performs even better than the median submission run (MAP= 0.2387, P10 = 0.3990). Our best run trails by 12% in MAP and 8% in P10 from the best team. We consider the performance results very satisfactory considering the fact that we had a relatively low baseline system. We are currently in the process of improving the parsing and indexing process to improve the baseline performance.

7. Conclusion

In this paper, we used ARRANGER, a GP-based discovery engine, to discover several ranking functions for the Robust Track. We observed up to 16% performance improvement over our baseline Okapi system. The experimental results show that the automatically learned ranking functions are capable of outperforming expert-designed functions.

In addition, we also tried some other popular performance improvement techniques, such as pseudo-relevance feedback and a ranking fusion technique. Both of them work well with those new functions and help further improve our system performance. Not only do they increase the average precision, but they also make the system more robust and provide less performance variation on different query sets.

References

W. Fan, M.D. Gordon, P. Pathak, "A generic ranking function discovery framework by genetic programming for information retrieval", *Information Processing and Management*, in press, 2003a.

W. Fan, M. D. Gordon, P. Pathak, "Discovery of context-specific ranking functions for effective information retrieval by Genetic Programming", *IEEE Transactions on Knowledge and Data Engineering*, in press, 2003b.

J. H. Holland, *Adaptation in Natural and Artificial Systems*, the University of Michigan Press, Ann Arbor, 1975.

S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford, "Okapi at TREC-4", in D. K. Harman, editor, *Proceedings of the Fourth Text Retrieval Conference*, pages 73–97. NIST Special Publication 500-236, 1996.

G. Salton and C. Buckley, "Term weighting approaches in automatic text retrieval", *Information Processing and Management*, 24(5): 513–523, 1988.

J. Zobel and A. Moffat, "Exploring the similarity space", *SIGIR Forum*, 32(1): 18–34, 1998.

VT at TREC-2003: The Web Track Report

Rui Yang, Li Wang[1], Weiguo Fan, Wensi Xi, Ming Luo, Ye Zhou, Edward A. Fox
Department of Computer Science
Virginia Polytechnic Institute and State University
{yrui, liwang5, wfan, xwensi, lming, yezhou, fox}@vt.edu

1. Introduction

This year, we participated in the Web Track in addition to the Robust Track. We submitted results on both topic distillation and home page/named page finding tasks. As our time and human resources were limited for taking two tasks simultaneously, in this task we only concentrate on testing our ranking function discovery technique, **ARRANGER** (**A**utomatic **R**endering of **RAN**king functions by **GE**netic p**R**ogramming) [Fan 2003a, Fan 2003b], which uses Genetic Programming (GP) to discover the "optimal" ranking functions for various information needs. From Web Track 2002, the training, testing and validation data sets are constructed in the same manner as in Robust Track. Our ARRANGER engine works on those data sets and automatically searches for the "best" ranking functions. The best runs are selected for submission according to their performance on queries in Web track 2002.

Our paper is organized as follows. Section 2 states our research objectives. Section 3 describes basic data processing steps. Section 4 summarizes the GP algorithm used in our system and detailed information about how we use GP to find ranking function. Section 5 shows the official submission results in comparison with the other TREC teams.

2. Research objectives

We have two objectives in this year's Web Track

1) We want to test the effectiveness of our ranking function discovery framework (ARRANGER) for other tasks (topic distillation, named page finding) and new collections. Previously, the framework is tested only on AP news collection.
2) We want to test whether the combing the structure information into the rank function can significantly improve the result. Unfortunately for this objective, due to the lack of time, we have not got any conclusive result yet.

3. Data processing

All our experiments were run on a dual-2.3GHz-processor Server running a Linux (Red-hat 7.3) operating system. Since our concentration in the Web Track is to test the significance of document structure as well as the GP ranking function, we made a lot of effort to parse the structure information of the documents. Those structured information then stored separately to form its own index. For instance, we stored the anchor information in one particular folder and use only the anchor text and properly split phrases of the URL as the text of this field, and index them into both forward index and inverted index format for our experimental purposes after removing stop words and stemming. We also parsed the in-link and out-link graph of the source document and hope to utilize it in the finding of the proper ranking function. No phrases were used in our experiments.

To speed up the process of the parsing, we used the standard HTML parser library from Perl. We modified the parsing codes. We also incorporated the parsing of all tags in one parsing step instead of separated parsing for each tag in the previous approach. This significantly increased the structure parsing processing speed. The parsed texts are then stored separately according the tag they belong to. We also removed the stop word and do stemming at this step. Then the output is sent to our indexer for indexing.

The structural elements we parsed are url (<a>), header (<h1> <h2> < h3>, <h4>, <h5>, <h6>, <th>) , title (<title>), meta (<meta>), anchor (the text in the <a> tag that point to the current document), strong (<strong>, <u>, <b>, <font>, <em>, <i>), list (<ul>, <dl>, <ol>), and abstract (the first hundred non-stop-word from the body part of the document). We also include a plaintext part that is the union of the text parsed from all the tags. Then we separately store and index the parsed texts for each structural element.

[1] Li Wang is now at the University of Michigan (wang@umich.edu).

Although we spent substantial effort on categorizing all the web pages into different collections based on structural elements, in the end we only use plaintext collection to train our ARRANGER system and to prepare for the submissions because of lack of time.

4. Ranking function discovery using Genetic Programming

In Web track this year, we did not take advantage of the structural information of web pages. Instead, we construct a "surrogate" plaintext collection by merging full text content with all the anchor information for a page. Based on the plaintext collection, our ARRANGER engine, a Genetic Programming (GP) based ranking function discovery system, is used to discover the "optimal" ranking functions for the topic distillation task. For home/named page finding task, we simply plug various GP-based functions learned before as well as Okapi BM25 into our search engine and pick the best five runs for submission.

As reported in the Robust track paper, we achieved significant performance improvement by using new ranking functions discovered by our ARRANGER system. In the Web track, our main goal is to test if the same ranking function discovery framework could work well under the Web context for other information needs (named/home page finding and topic distillation needs). There are substantial differences between Robust track and Web track. Besides collection and query property differences between two tracks, the objectives of Web track are totally different from Robust track. In Robust track, the document providing the most sufficient information for a query should be ranked at the top of the returned document list. In Web track, a different strategy must be employed to find the most likely home/named page (for home/named page task) or find the key resources for a topic (for topic distillation task). However for these two tracks, the same ranking function discovery system (ARRANGER) is used and so are the training, testing and validation processes. We want to demonstrate that our ARRANGER system is effective under various contexts and could satisfy distinct information needs, provided that training data are appropriately prepared. As GP can easily over-train the data, we use three independent data sets for training, testing and validation purposes. 150 queries and relevance information are obtained from TREC 2002 topic distillation task for training, testing and validation processes. The details of our system and methodology for Genetic Programming (GP) are discussed in our Robust track paper. Interested readers can reference that paper or [Fan 2003a, Fan 2003b].

5. Results

In the end, we submit ten independent runs for this year's Web Track – five for the topic distillation task, five for the Name/Home page finding task. Our submissions do not involve any human intervention, so they are all automatic runs. Tables 1, 2 give the detailed description of our submissions. Tables 3, 4 summarize the final evaluation results from TREC for all 5 runs.

Run Number	Description
VTnhpgp33	This run is for the name / home page finding task, using the GP discovered function.
VTnhpgp42	This run is for the name / home page finding task, using the GP discovered function.
VTnhpgp55	This run is for the name / home page finding task, using the GP discovered function.
VTnhpgpd4	This run is for the name / home page finding task, using the GP discovered function.
VTnhpokl	This run is for the name / home page finding task, using the OKAPI function

Table 1 - Description of our five official submissions for named/home page finding task

Run Number	Description
VTtdgp33	This run is for the topic distillation task, using the GP discovered function.
VTtdgp41	This run is for the topic distillation task, using the GP discovered function.
VTtdgp5055	This run is for the topic distillation task, using the GP discovered function.
VTtdgp52	This run is for the topic distillation task, using the GP discovered function.
VTtdok4	This run is for the topic distillation task, using the OKAPI function

Table 2 - Description of our five official submissions for topic distillation task

Run No.	P10	#Best	#>= Median	P20	#Bes t	#>= Median	P30	#Best	#>= Median
VTtdgp33	0.0540	6	43	0.0560	8	42	0.0460	3	40
VTtdgp41	0.0620	6	41	0.0550	7	43	0.0473	3	40
VTtdgp5055	0.0760	7	43	0.0550	5	38	0.0520	4	36
VTtdgp52	0.0660	6	42	0.0560	7	42	0.0487	3	40
VTtdok4	0.0620	7	43	0.0530	7	39	0.0447	2	37
Total		**10**			**9**			**11**	

Table 3 - Official submission results for the topic distillation task.

Run No.	#not found Named Page	MRR Named Page	#not found Home Page	MRR Home Page
VTnhpgp33	16	0.5129	59	0.2317
VTnhpgp42	17	0.5084	58	0.2391
VTnhpgp55	18	0.4971	58	0.2216
VTnhpgpd4	18	0.4919	66	0.1660
VTnhpok1	20	0.4929	61	0.2024

Table 4 - Official submission results for the page finding task.

As can be seen from Table 3, we did relatively well in the topic distillation task. Almost 90% of our results are equal or above the median performance. This indicates the relative advantage of ranking function optimization using GP.

Since we did not do any optimization for the named/home page finding task, our results in this task is a little bit disappointing. One of the main reasons is that we use the same ranking strategy for both home page and named page finding tasks. This proves to be wrong if we look at the performance results from Table 4. We did well for named page finding task, but poorly on home page finding task. This indicates that home paging requires some additional evidence such as URL, Link information for effective ranking. Our future strategy is to design a query classification scheme to automatically classify queries into two different types and apply different ranking strategies based on the type of queries.

References

W. Fan, M.D. Gordon, P. Pathak, "A generic ranking function discovery framework by genetic programming for information retrieval", *Information Processing and Management*, in press, 2003a.

W. Fan, M. D. Gordon, P. Pathak, "Discovery of context-specific ranking functions for effective information retrieval by Genetic Programming", *IEEE Transactions on Knowledge and Data Engineering*, in press, 2003b.

TREC 2003 Results

APPENDIX A

This appendix contains the evaluation results for TREC 2003 runs. The initial pages list each of the runs (identified by the run tags) that were included in the different tracks. Associated with each tag is the organization that produced the run and additional information such as whether the queries were produced manually or automatically as appropriate.

Following the run list is a description of the evaluation measures computed by `trec_eval`. Most tracks use variants of (some of) these measures for the evaluation in that track. For more details about the measures used in a particular track, see the overview paper for that track.

The remainder of the appendix contains the evaluation results themselves, in the order given in the run list.

Genomics Track, Primary Task

Tag	Organization	Description	Number of Topics
axon1	Axontologic, Inc.	automatic	50
axon2	Axontologic, Inc.	automatic	50
CSUSM1	California State Univ./San Marcos	automatic	50
CSUSM2	California State Univ./San Marcos	automatic	50
DcuMesh1	Dublin City Univ.	automatic	50
DcuMesh2	Dublin City Univ.	automatic	50
ErasmusMC2	Erasmus MC	automatic	50
ErasmusMC3	Erasmus MC	automatic	50
humG03ns	Hummingbird	automatic	50
humG03ns5	Hummingbird	automatic	50
IBMbt1	IBM TJ Watson Research Ctr.	automatic	50
IBMbt2	IBM TJ Watson Research Ctr.	automatic	50
vvP05mil3	IRIT/SIG	automatic	50
KUBIOIRNE	Korea Univ.	automatic	50
KUBIOIRRAW	Korea Univ.	automatic	50
NLMUMDSE	Nat. Library of Med. & Univ. of Maryland	automatic	50
NLMUMDSRB	Nat. Library of Med. & Univ. of Maryland	automatic	50
nrc1	National Res. Council Canada	automatic	50
nrc2	National Res. Council Canada	automatic	50
balsc2	NTT Comm. Science Labs	automatic	50
balsc3	NTT Comm. Science Labs.	automatic	50
ohsuboost	Oregon Health & Science Univ.	automatic	50
dayrutgers1	Rutgers Univ.	automatic	50
dayrutgers2	Rutgers Univ.	automatic	50
SCAI	SCAI	automatic	50
UBgenomRFB1	State Univ. of NY at Buffalo-CEDAR	automatic	50
UBgenomRFB2	State Univ. of NY at Buffalo-CEDAR	automatic	50
UBgenomeBGNE	State Univ. of NY at Buffalo-CEDAR	automatic	50
StreamSage3	StreamSage, Inc.	automatic	50
StreamSage4	StreamSage, Inc.	automatic	50
tgnBaseline	Tarragon Consulting Corp.	manual	50
tgnVariant1	Tarragon Consulting Corp.	manual	50
biotext0	Univ. of California, Berkeley	automatic	50
biotext1	Univ. of California, Berkeley	automatic	50
edstanprec	Univ. of Edinburgh & Stanford Univ.	automatic	50
edstanrecall	Univ. of Edinburgh & Stanford Univ.	automatic	50
UIUC03Ga	Univ. of Illinois at Urbana-Champaign	automatic	50
UIUC03Gb	Univ. of Illinois at Urbana-Champaign	automatic	50
UIowaGN1	Univ. of Iowa	automatic	50
UniNEg1	Univ. de Neuchatel	automatic	50
UniNEg2	Univ. de Neuchatel	automatic	50
UniNEg4	Univ. de Neuchatel	automatic	50
UniNEg5	Univ. de Neuchatel	automatic	50
utaband	Univ. of Tampere	manual	50
utafil	Univ. of Tampere	manual	50
aoyama	Univ. of Tokyo	automatic	50
aoyama2	Univ. of Tokyo	automatic	50
uwmtg03atrf	Univ. of Waterloo-MultiText	automatic	50
uwmtg03btrf	Univ. of Waterloo-MultiText	automatic	50

Genomics Track, Secondary Task

Tag	Organization	Description
CSUSMcand	California State Univ. San Marcos	automatic
emc4	Erasmus MC	automatic
IBMbtT2	IBM TJ Watson Research Center	automatic
IUB2003	Indiana Univ., Bloomington	automatic
NLMUMDLIN	National Library of Medicine and Univ. Maryland	automatic
nwe	National Taiwan Univ.	automatic
we	National Taiwan Univ.	automatic
balscsec1	NTT Communications Science Laboratories	automatic
UBGenT2BL1	State Univ. of New York at Buffalo-CEDAR	automatic
UBGenT2R1	State Univ. of New York at Buffalo-CEDAR	automatic
UBGenT2R2	State Univ. of New York at Buffalo-CEDAR	automatic
biotextTask2	Univ. of California, Berkeley	automatic
EDISTFruns2	Univ. of Edinburgh & Stanford Univ.	automatic
tg2hug	Univ. Hospital of Geneva	automatic
tgIIhugLASt	Univ. Hospital of Geneva	automatic
UIowaSecCan	Univ. of Iowa	automatic
UniNEie1	Universite de Neuchatel	automatic
UniNEie2	Universite de Neuchatel	automatic
UniNEie3	Universite de Neuchatel	automatic
UniNEie4	Universite de Neuchatel	automatic
UniNEie5	Universite de Neuchatel	automatic
uwb2	Univ. of Wales, Bangor	automatic
uwb3	Univ. of Wales, Bangor	automatic
uwb4	Univ. of Wales, Bangor	automatic

HARD Track

Tag	Organization	Baseline Run
OPEN	Chinese Info. Processing Ctr.	Yes
OPEN1	Chinese Info. Processing Ctr.	No
CLAISTDNG	Clairvoyance Corp.	Yes
CLSTD630	Clairvoyance Corp.	Yes
CLA1G	Clairvoyance Corp.	No
CLAI1NG	Clairvoyance Corp.	No
CLAI2G	Clairvoyance Corp.	No
CLAI2NG	Clairvoyance Corp.	No
CLAI2RTG	Clairvoyance Corp.	No
CLAI2RTNG	Clairvoyance Corp.	No
CLAI2WRTG	Clairvoyance Corp.	No
CLAI2WRTNG	Clairvoyance Corp.	No
CLAISTDG	Clairvoyance Corp.	No
CLAISTDRTG	Clairvoyance Corp.	No
CLAISTDRTNG	Clairvoyance Corp.	No
CLAISTDWRTG	Clairvoyance Corp.	No
CLAISTDWRTNG	Clairvoyance Corp.	No
IITBHDBSLN1	Indian Inst. Of Tech. Bombay	Yes
IITBHDBSLN2	Indian Inst. Of Tech. Bombay	Yes
IITBHDF1	Indian Inst. Of Tech. Bombay	No
IITBHDF2	Indian Inst. Of Tech. Bombay	No
IITBHDF3	Indian Inst. Of Tech. Bombay	No
IITBHDFTEMP	Indian Inst. Of Tech. Bombay	No
MSRCbase	Microsoft Research Ltd.	Yes
MSRCS1e0p0B	Microsoft Research Ltd.	No
MSRCs1e0p0	Microsoft Research Ltd.	No
MSRCs1e0p1B	Microsoft Research Ltd.	No
MSRCs1e0p1	Microsoft Research Ltd.	No
MSRCs1e1p1B	Microsoft Research Ltd.	No
MSRCs1e1p1	Microsoft Research Ltd.	No
MSRCs2e0p1	Microsoft Research Ltd.	No
MSRCs9e1p0	Microsoft Research Ltd.	No
MSRCs9e1p1	Microsoft Research Ltd.	No
pircHDBt1	Queens College, CUNY	Yes
pircHDBtd1	Queens College, CUNY	Yes
pircHDC1t1	Queens College, CUNY	No
pircHDC1tp	Queens College, CUNY	No
pircHDC2t1	Queens College, CUNY	No
pircHDC2t2	Queens College, CUNY	No
pircHDC2tp	Queens College, CUNY	No
pircHDC3td1	Queens College, CUNY	No
pircHDC3td2	Queens College, CUNY	No
pircHDC3tdp	Queens College, CUNY	No
rutbase1	Rutgers Univ.	Yes
rutbase2	Rutgers Univ.	Yes
Rutmeta	Rutgers Univ.	No

HARD Track (continued)

Tag	Organization	Baseline Run
ub03cugTD	State Univ. of NY at Buffalo-CEDAR	Yes
ub03sugT	State Univ. of NY at Buffalo-CEDAR	Yes
ub03ugTcugTD	State Univ. of NY at Buffalo-CEDAR	Yes
ub03smfugTD	State Univ. of NY at Buffalo-CEDAR	No
TUCSHardBR1	Tsinghua Univ. (Ma)	Yes
TUCSHARD1	Tsinghua Univ. (Ma)	No
TUCSHARD2	Tsinghua Univ. (Ma)	No
TUCSHARD3	Tsinghua Univ. (Ma)	No
TUKE1	Tsinghua Univ. (Keg)	Yes
TUKE2	Tsinghua Univ. (Keg)	Yes
TUKE3	Tsinghua Univ. (Keg)	No
TUKE4	Tsinghua Univ. (Keg)	No
HKbas1tni1td	Univ. of Helskinki	Yes
HKIdoc1t1td	Univ. of Helskinki	Yes
HKIdoc1t1t	Univ. of Helskinki	Yes
HKIdoc1ti1td	Univ. of Helskinki	Yes
HKIdoc1tilt	Univ. of Helskinki	Yes
HKupw1tni1td	Univ. of Helskinki	Yes
HKup1tni1td	Univ. of Helskinki	No
uiuchard	Univ. of Illinois at Urbana-Champaign	Yes
UIHARD1	Univ. of Illinois at Urbana-Champaign	No
UIHARD2	Univ. of Illinois at Urbana-Champaign	No
UIHARD3	Univ. of Illinois at Urbana-Champaign	No
UIHARD4	Univ. of Illinois at Urbana-Champaign	No
umdbsne2tit	Univ. of Maryland	Yes
UMARIPVR4	Univ. of Maryland	No
UMARIPVR5	Univ. of Maryland	No
UMARIPVR7	Univ. of Maryland	No
UMARIPVR8	Univ. of Maryland	No
ciirtbas	Univ. of Massachusetts	Yes
ciirtdbas	Univ. of Massachusetts	Yes
ciirtpsgbas	Univ. of Massachusetts	Yes
ciirtp	Univ. of Massachusetts	Yes
ciirtdpsgbas	Univ. of Massachusetts	Yes
ciirtcftt	Univ. of Massachusetts	No
ciirtmdap	Univ. of Massachusetts	No
ciirtmda	Univ. of Massachusetts	No
ciirtmdgp	Univ. of Massachusetts	No
ciirtrt	Univ. of Massachusetts	No
UWAThard1	Univ. of Waterloo	Yes
UWAThard2	Univ. of Waterloo	No
UWAThard3	Univ. of Waterloo	No

Novelty Track, Task 1

Tag	Organization
ICT03NOV1BSL	CAS-ICT
ICT03NOV1DTH	CAS-ICT
ICT03NOV1NAR	CAS-ICT
ICT03NOV1SQR	CAS-ICT
ICT03NOV1XTD	CAS-ICT
NLPR03n1f1	CAS-NLPR
NLPR03n1f2	CAS-NLPR
NLPR03n1w1	CAS-NLPR
NLPR03n1w2	CAS-NLPR
NLPR03n1w3	CAS-NLPR
ccsumlaqr	CCS and Univ. Maryland
ccsummeoqr	CCS and Univ. Maryland
ccsummeosvd	CCS and Univ. Maryland
ccsumrelqr	CCS and Univ. Maryland
ccsumrelsvd	CCS and Univ. Maryland
clr03n1d	CL Research
clr03n1n2	CL Research
clr03n1n3	CL Research
clr02n1t	CL Research
IRITf2bis	IRIT/SIG
IRITfNegR2	IRIT/SIG
IRITfb1MtmIb	IRIT/SIG
IRITnb1MtmI4	IRIT/SIG
IRITnip2bis	IRIT/SIG
lexiclone03	LexiClone, Inc.
MeijiHilF11	Meiji Univ.
MeijiHilF12	Meiji Univ.
MeijiHilF13	Meiji Univ.
MeijiHilF14	Meiji Univ.
MeijiHilF15	Meiji Univ.
NTU11	National Taiwan Univ.
NTU12	National Taiwan Univ.
NTU13	National Taiwan Univ.
NTU14	National Taiwan Univ.
NTU15	National Taiwan Univ.
THUIRnv0311	Tsinghua Univ. (Ma)
THUIRnv0312	Tsinghua Univ. (Ma)
THUIRnv0313	Tsinghua Univ. (Ma)
THUIRnv0314	Tsinghua Univ. (Ma)
THUIRnv0315	Tsinghua Univ. (Ma)
UIowa03Nov01	Univ. of Iowa
UIowa03Nov02	Univ. of Iowa
umbcrun1	Univ. of Maryland Baltimore County
umbcrun2	Univ. of Maryland Baltimore County
umbcrun3	Univ. of Maryland Baltimore County
umich1	Univ. of Michigan

Novelty Track, Task 1 (continued)

Tag	Organization
umich2	Univ. of Michigan
umich3	Univ. of Michigan
umich4	Univ. of Michigan
umich5	Univ. of Michigan
ISIALL03	Univ. of Southern California-ISI
ISIDSCm203	Univ. of Southern California-ISI
ISIDSm203	Univ. of Southern California-ISI
ISINONE03	Univ. of Southern California-ISI
ISIRAND03	Univ. of Southern California-ISI

Novelty Track, Task 2

Tag	Organization
ICT03NOV2CUR	CAS-ICT
ICT03NOV2LPA	CAS-ICT
ICT03NOV2LPP	CAS-ICT
ICT03NOV2PNK	CAS-ICT
ICT03NOV2SQR	CAS-ICT
NLPR03n2d1	CAS-NLPR
NLPR03n2d2	CAS-NLPR
NLPR03n2d3	CAS-NLPR
NLPR03n2s1	CAS-NLPR
NLPR03n2s2	CAS-NLPR
ccsum2svdpqr	CFCS and Univ. Maryland
ccsumt2pqr	CFCS and Univ. Maryland
ccsumt2qr	CFCS and Univ. Maryland
ccsumt2svdqr	CFCS and Univ. Maryland
clr03n2	CL Research
Irit1	IRIT/SIG
Irit5q	IRIT/SIG
IritMtm4	IRIT/SIG
IritMtm5	IRIT/SIG
Irito	IRIT/SIG
MeijiHilf21	Meiji Univ.
MeijiHilf22	Meiji Univ.
MeijiHilf23	Meiji Univ.
MeijiHilF24	Meiji Univ.
NTU21	National Taiwan Univ.
NTU22	National Taiwan Univ.
NTU23	National Taiwan Univ.
NTU24	National Taiwan Univ.
NTU25	National Taiwan Univ.
THUIRnv0321	Tsinghua Univ. (Ma)
THUIRnv0322	Tsinghua Univ. (Ma)
THUIRnv0323	Tsinghua Univ. (Ma)
UIowa03Nov03	Univ. of Iowa
UIowa03Nov04	Univ. of Iowa
UIowa03Nov05	Univ. of Iowa
UIowa03Nov06	Univ. of Iowa
UIowa03Nov07	Univ. of Iowa
umbcnew1	Univ. of Maryland Baltimore County
umbcnew2	Univ. of Maryland Baltimore County
wmbcnew3	Univ. of Maryland Baltimore County
umich21	Univ. of Michigan
umich22	Univ. of Michigan

Novelty Track, Task 2 (continued)

Tag	Organization
umich23	Univ. of Michigan
umich24	Univ. of Michigan
umich25	Univ. of Michigan

Novelty Track, Task 3

Tag	Organization
ICT03NOV3IKK	CAS-ICT
ICT03NOV3KNN	CAS-ICT
ICT03NOV3KNS	CAS-ICT
ICT03NOV3WN3	CAS-ICT
ICT03NOV3WND	CAS-ICT
NLPR03n3d1	CAS-NLPR
NLPR03n3d2	CAS-NLPR
NLPR03n3d3	CAS-NLPR
NLPR03n3s1	CAS-NLPR
NLPR03n3s2	CAS-NLPR
ccsum3pqr	CFCS and Univ. Maryland
ccsum3qr	CFCS and Univ. Maryland
ccsum3svdpqr	CFCS and Univ. Maryland
clr03n3f01	CL Research
clr03n3f02	CL Research
clr03n3f03	CL Research
clr03n3f04	CL Research
clr03n3f05	CL Research
MeijiHilF31	Meiji Univ.
MeijiHilF32	Meiji Univ.
MeijiHilF32	Meiji Univ.
MeijiHilF33	Meiji Univ.
MeijiHilF34	Meiji Univ.
NTU31	National Taiwan Univ.
NTU32	National Taiwan Univ.
NTU33	National Taiwan Univ.
NTU34	National Taiwan Univ.
NTU35	National Taiwan Univ.
THUIRnv0331	Tsinghua Univ.
THUIRnv0332	Tsinghua Univ.
THUIRnv0333	Tsinghua Univ.
THUIRnv0334	Tsinghua Univ.
UIowa03Nov08	Univ. of Iowa
UIowa03Nov09	Univ. of Iowa
umich31	Univ. of Michigan
umich32	Univ. of Michigan
umich33	Univ. of Michigan
umich34	Univ. of Michigan
umich35	Univ. of Michigan

Novelty Track, Task 4

Tag	Organization
ICT03NOV4ALL	CAS-ICT
ICT03NOV4LFF	CAS-ICT
ICT03NOV4OTP	CAS-ICT
ICT03NOV4SQR	CAS-ICT
ICT03NOV4WNW	CAS-ICT
NLPR03n4d1	CAS-NLPR
NLPR03n4d2	CAS-NLPR
NLPR03n4s1	CAS-NLPR
NLPR03n4s2	CAS-NLPR
NLPR03n4s3	CAS-NLPR
ccsum4spq001	CCS and Univ. Maryland
ccsum4svdpqr	CCS and Univ. Maryland
ccsumt4pqr	CCS and Univ. Maryland
ccsumt4qr	CCS and Univ. Maryland
ccsumt4sqr01	CCS and Univ. Maryland
clr03n4	CL Research
IITBN1	Indian Institute of Tech. Bombay
MeijiHilF41	Meiji Univ.
MeijiHilF42	Meiji Univ.
MeijiHilF43	Meiji Univ.
MeijiHilF44	Meiji Univ.
NTU41	National Taiwan Univ.
NTU42	National Taiwan Univ.
NTU43	National Taiwan Univ.
NTU44	National Taiwan Univ.
NTU45	National Taiwan Univ.
THUIRnv0341	Tsinghua Univ. (Ma)
THUIRnv0342	Tsinghua Univ. (Ma)
THUIRnv0343	Tsinghua Univ. (Ma)
THUIRnv0344	Tsinghua Univ. (Ma)
THUIRnv0345	Tsinghua Univ. (Ma)
UIowa03Nov10	Univ. of Iowa
UIowa03Nov11	Univ. of Iowa
UIowa03Nov12	Univ. of Iowa
UIowa03Nov13	Univ. of Iowa
UIowa03Nov14	Univ. of Iowa
umich41	Univ. of Michigan
umich42	Univ. of Michigan
umich43	Univ. of Michigan
umich44	Univ. of Michigan
umich45	Univ. of Michigan

Question Answering Track, Main Task

Tag	Organization
BBN2003A	BBN
BBN2003B	BBN
BBN2003C	BBN
CMUJAV2003	Carnegie Mellon Univ.
ICTQA2003A	CAS-ICT
ICTQA2003B	CAS-ICT
ICTQA2003C	CAS-ICT
clr03m1	CL Research
FDUT12QA1	Fudan Univ.
FDUT12QA2	Fudan Univ.
FDUT12QA3	Fudan Univ.
IBM2003a	IBM T.J. Watson Research Center
IBM2003b	IBM T.J. Watson Research Center
IBM2003c	IBM T.J. Watson Research Center
irstqa2003d	ITC-irst
irstqa2003p	ITC-irst
irstqa2003w	ITC-irst
LCCmainE03	Language Computer Corporation
LCCmainS03	Language Computer Corporation
lexiclone92	Lexiclone, Inc.
MITCSAIL03a	Massachusetts Institute of Technology
MITCSAIL03b	Massachusetts Institute of Technology
MITCSAIL03c	Massachusetts Institute of Technology
MITRE2003A	MITRE Corp.
nusmml03r1	National Univ. of Singapore
nusmml03r2	National Univ. of Singapore
nusmml03r3	National Univ. of Singapore
CRL2003	New Mexico State Univ.
ntt2003qam1	NTT Communication Science Laboratories
Albany03I2	Univ. of Albany
Albany03I3	Univ. of Albany
Albany03I4	Univ. of Albany
UAmsT03M1	Univ. of Amsterdam
UAmsT03M2	Univ. of Amsterdam
UAmsT03M3	Univ. of Amsterdam
cuaqdef2003	Univ. of Colorado & Columbia Univ.
EdinInf2003A	Univ. of Edinburgh
EdinInf2003B	Univ. of Edinburgh
EdinInf2003C	Univ. of Edinburgh
UIowaQA0301	Univ. of Iowa
UIowaQA0302	Univ. of Iowa
UIowaQA0303	Univ. of Iowa

Question Answering Track, Main Task (continued)

Tag	Organization
DLT03QA01	Univ. of Limerick
DLT03QA02	Univ. of Limerick
piq001	Univ. of Pisa
piq002	Univ. of Pisa
UPCUdGsys1	UPC & UdG
shef12madcow	Univ. of Sheffield
shef12okapi	Univ. of Sheffield
shef12simple	Univ. of Sheffield
isi03a	Univ. of Southern California-ISI
isi03b	Univ. of Southern California-ISI
isi03c	Univ. of Southern California-ISI
uwbqitekat03	Univ. of Wales, Bangor

Question Answering Track, Passages Task

Tag	Organization
clr03pl	CL Research
IITBQA	Indian Institute of Tech. Bombay
LCCpass03	Language Computer Corporation
answfind1	Macquarie Univ.
answfind2	Macquarie Univ.
answfind3	Macquarie Univ.
nuslamp03	National Univ. of Singapore
nuslamp03a	National Univ. of Singapore
nuslamp03b	National Univ. of Singapore
pircsqa1	Queens College, CUNY
pircsqa2	Queens College, CUNY
pircsqa3	Queens College, CUNY
Saarland	Saarland Univ.
UAmsT03P1	Univ. of Amsterdam
umassq1	Univ. of Massachusetts
NSIR	Univ. of Michigan
uwmtCQ0	Univ. of Waterloo-MultiText
uwmtCQ1	Univ. of Waterloo-MultiText
uwmtCQ2	Univ. of Waterloo-MultiText

Robust Task

Tag	Organization	Topic Parts Used
NLPR03vb10	Chinese Academy of Sciences	description only
NLPR03vb25	Chinese Academy of Sciences	description only
NLPR03vb50	Chinese Academy of Sciences	description only
NLPR03w16	Chinese Academy of Sciences	description only
NLPR03w49	Chinese Academy of Sciences	description only
fub03IeOLKe3	Fondazione Ugo Bordoni	description only
fub03InB2e3	Fondazione Ugo Bordoni	description only
fub03InOLe3	Fondazione Ugo Bordoni	description only
fub03IneOBu3	Fondazione Ugo Bordoni	description only
fub03IneOLe3	Fondazione Ugo Bordoni	description only
humR03d	Hummingbird	description only
humR03dc	Hummingbird	description only
humR03de	Hummingbird	description only
humR03t	Hummingbird	title only
humR03tc	Hummingbird	title only
aplrob03a	Johns Hopkins U.-APL	title+desc+narr
aplrob03b	Johns Hopkins U. –APL	description only
aplrob03c	Johns Hopkins U. –APL	description only
aplrob03d	Johns Hopkins U. –APL	description only
aplrob03e	Johns Hopkins U. –APL	description only
oce03Xbm	OcE Technologies	title+desc
oce03Xpr	OcE Technologies	title+desc
oce03noXbm	OcE Technologies	title+desc
oce03noXpr	OcE Technologies	title+desc
oce03noXbmD	OcE Technologies	description only
pircRBa1	Queens College, CUNY	title+desc+narr
pircRBa2	Queens College, CUNY	title+desc+narr
pircRBd1	Queens College, CUNY	description only
pircRBd2	Queens College, CUNY	description only
pircRBd3	Queens College, CUNY	description only
rutcor030	Rutgers Univ.	description only
rutcor03100	Rutgers Univ.	description only
rutcor0325	Rutgers Univ.	description only
rutcor0350	Rutgers Univ.	description only
rutcor0375	Rutgers Univ.	description only
SABIR03BASE	Sabir Research, Inc.	description only
SABIR03BF	Sabir Research, Inc.	description only
SABIR03MERGE	Sabir Research, Inc.	description only
THUIRr0301	Tsinghua Univ. (Ma)	title+desc+narr
THUIRr0302	Tsinghua Univ. (Ma)	title+desc+narr
THUIRr0303	Tsinghua Univ. (Ma)	title+desc+narr
THUIRr0304	Tsinghua Univ. (Ma)	title+desc
THUIRr0305	Tsinghua Univ. (Ma)	description only
UAmsT03RDesc	Univ. of Amsterdam	description only
UAmsT03R	Univ. of Amsterdam	title+desc
UAmsT03RFb	Univ. of Amsterdam	title+desc
UAmsT03RSt	Univ. of Amsterdam	title+desc

Robust Task (continued)

Tag	Organization	Topic Parts Used
UAmsT03RStFb	Univ. of Amsterdam	title+desc
InexpC2	Univ. of Glasgow	description only
InexpC2QE	Univ. of Glasgow	description only
Sel50	Univ. of Glasgow	description only
Sel50QE	Univ. of Glasgow	description only
Sel78QE	Univ. of Glasgow	description only
uic0301	Univ. of Illinois at Chicago	title+desc
uic0303	Univ. of Illinois at Chicago	title+desc
uic0302	Univ. of Illinois at Chicago	title only
uic0304	Univ. of Illinois at Chicago	title only
uic0305	Univ. of Illinois at Chicago	title+desc
UIUC03Rd1	Univ. of Illinois at Urbana-Champaign	description only
UIUC03Rd2	Univ. of Illinois at Urbana-Champaign	description only
UIUC03Rd3	Univ. of Illinois at Urbana-Champaign	description only
UIUC03Rt1	Univ. of Illinois at Urbana-Champaign	title only
UIUC03Rtd1	Univ. of Illinois at Urbana-Champaign	title+desc
MU03rob01	Univ. of Melbourne	description only
MU03rob03	Univ. of Melbourne	title only
MU03rob02	Univ. of Melbourne	title+desc
MU03rob05	Univ. of Melbourne	title+desc
MU03rob04	Univ. of Melbourne	title+desc+narr
uwmtCR0	Univ. of Waterloo-MultiText	description only
uwmtCR1	Univ. of Waterloo-MultiText	description only
uwmtCR2	Univ. of Waterloo-MultiText	title only
uwmtCR3	Univ. of Waterloo-MultiText	title only
uwmtCR4	Univ. of Waterloo-MultiText	description only
VTDokrcgp5	Virginia Tech	description only
VTcdhgp1	Virginia Tech	title+desc+narr
VTcdhgp3	Virginia Tech	title+desc+narr
VTgpdhgp2	Virginia Tech	title+desc+narr
VTgpdhgp4	Virginia Tech	title+desc+narr

Web Track, Home Page Finding Task

Tag	Organization	Document Structure	Anchor Text	Link Structure
ajouai0306	Ajou Univ.	Yes	No	No
ajouai0308	Ajou Univ.	Yes	No	Yes
ajouai0309	Ajou Univ.	Yes	No	No
ICTWebKI12A	CAS-ICT	No	No	No
ICTWebKI12B	CAS-ICT	Yes	Yes	No
ICTWebKI12C	CAS-ICT	Yes	Yes	No
LmrEq	Carnegie Mellon Univ.	Yes	Yes	No
LmrEqUrl	Carnegie Mellon Univ.	Yes	Yes	No
LmrEst	Carnegie Mellon Univ.	Yes	Yes	No
LmrEstUrl	Carnegie Mellon Univ.	Yes	Yes	No
copNpRun1	Copernic Research	Yes	No	No
copNpRun2	Copernic Research	Yes	No	No
copNpRun3	Copernic Research	Yes	No	No
copNpRun4	Copernic Research	Yes	No	No
copNpRun5	Copernic Research	Yes	No	No
csiro03ki01	CSIRO	Yes	Yes	Yes
csiro03ki02	CSIRO	Yes	Yes	Yes
csiro03ki03	CSIRO	Yes	Yes	Yes
csiro03ki04	CSIRO	Yes	Yes	Yes
csiro03ki05	CSIRO	Yes	Yes	Yes
humNP031	Hummingbird	No	No	No
humNP03pl	Hummingbird	Yes	No	No
humNP03uhpl	Hummingbird	Yes	No	No
humNP03up	Hummingbird	Yes	No	No
humNP03upl	Hummingbird	Yes	No	No
iit03sa	Illinois Institute of Technology	Yes	Yes	No
iit03sau	Illinois Institute of Technology	Yes	Yes	No
iit03su	Illinois Institute of Technology	Yes	Yes	No
iit03wp75	Illinois Institute of Technology	No	No	No
iit03wtaez	Illinois Institute of Technology	Yes	Yes	No
widitpfb1	Indiana Univ. , Bloomington	No	No	No
widitpff1	Indiana Univ. , Bloomington	No	Yes	No
03wume296	Lehigh Univ.	No	Yes	No
03wume298	Lehigh Univ.	No	Yes	No
MSRANP1	Microsoft Research Asia	Yes	Yes	No
MSRANP2	Microsoft Research Asia	Yes	Yes	No
MSRANP3	Microsoft Research Asia	Yes	Yes	No
UniNEnp1	Universite de Neuchatel	Yes	Yes	No
UniNEnp2	Universite de Neuchatel	Yes	Yes	No
UniNEnp3	Universite de Neuchatel	Yes	Yes	No
UniNEnp4	Universite de Neuchatel	Yes	Yes	No
UniNEnp5	Universite de Neuchatel	Yes	Yes	No
RMITSEG1	RMIT Univ.	Yes	No	No
RMITSEG2	RMIT Univ.	Yes	No	No
RMITSEG3	RMIT Univ.	Yes	No	No
RMITSEG4	RMIT Univ.	Yes	No	No
RMITSEG5	RMIT Univ.	Yes	No	No

Web Track, Home Page Finding Task (continued)

Tag	Organization	Document Structure	Anchor Text	Link Structure
homepages0	Saarland Univ.	No	No	No
THUIRpf0301	Tsinghua Univ. (Ma)	No	Yes	No
THUIRpf0302	Tsinghua Univ. (Ma)	No	Yes	No
THUIRpf0303	Tsinghua Univ. (Ma)	No	Yes	No
THUIRpf0304	Tsinghua Univ. (Ma)	No	Yes	No
THUIRpf0305	Tsinghua Univ. (Ma)	No	Yes	No
irtfgrep	Univ. of Alaska, Fairbanks	No	No	No
irttgrep	Univ. of Alaska, Fairbanks	No	No	No
UAmsT03WnLM	Univ. of Amsterdam	No	No	No
UAmsT03WnLM3	Univ. of Amsterdam	Yes	Yes	No
UAmsT03WnLn3	Univ. of Amsterdam	Yes	Yes	No
UAmsT03WnMSW	Univ. of Amsterdam	No	No	No
UAmsT03WnOWS	Univ. of Amsterdam	No	No	No
uogki1c	Univ. of Glasgow	No	No	No
uogki2ca	Univ. of Glasgow	No	Yes	No
uogki3cah	Univ. of Glasgow	No	Yes	No
uogki4cahs	Univ. of Glasgow	No	Yes	No
MU03np1	Univ. of Melbourne	Yes	Yes	No
MU03np3	Univ. of Melbourne	Yes	Yes	No
MU03np4	Univ. of Melbourne	Yes	Yes	No
MU03np5	Univ. of Melbourne	Yes	Yes	No
VTnhpgp33	Virginia Tech	No	Yes	No
VTnhpgp42	Virginia Tech	No	Yes	No
VTnhpgp55	Virginia Tech	No	Yes	No
VTnhpgpd4	Virginia Tech	No	Yes	No
VTnhpok1	Virginia Tech	No	Yes	No

Web Track, Topic Distillation Task

Tag	Organization	Document Structure	Anchor Text	Link Structure
ajouai0301	Ajou Univ.	No	No	No
ajouai0302	Ajou Univ.	Yes	No	Yes
ajouai0305	Ajou Univ.	Yes	No	No
ICTWebTD12A	CAS-ICT	No	No	No
ICTWebTD12B	CAS-ICT	No	No	No
ICTWebTD12C	CAS-ICT	Yes	Yes	No
copTdRun1	Copernic Research	Yes	No	No
copTdRun2	Copernic Research	Yes	No	No
copTdRun3	Copernic Research	Yes	No	No
copTdRun4	Copernic Research	Yes	No	No
copTdRun5	Copernic Research	Yes	No	No
csiro03td01	CSIRO	Yes	Yes	Yes
csiro03td02	CSIRO	Yes	Yes	Yes
csiro03td03	CSIRO	Yes	Yes	Yes
csiro03td04	CSIRO	Yes	No	Yes
csiro03td05	CSIRO	Yes	Yes	Yes
fub03InBMt	Fondazione Ugo Bordoni	No	No	No
fub03InLBo1t	Fondazione Ugo Bordoni	No	No	No
fub03InLBt	Fondazione Ugo Bordoni	No	No	No
fub03IneBBt	Fondazione Ugo Bordoni	No	No	No
fub03IneBMt	Fondazione Ugo Bordoni	No	No	No
humTD03l	Hummingbird	No	No	No
humTD03pl	Hummingbird	Yes	No	No
humTD03uhpl	Hummingbird	Yes	No	No
humTD03up	Hummingbird	Yes	No	No
humTD03upl	Hummingbird	Yes	No	No
JuruFull	IBM Research Haifa	Yes	Yes	Yes
JuruNoAnchor	IBM Research Haifa	Yes	No	Yes
JuruNoCohes	IBM Research Haifa	Yes	Yes	Yes
JuruNoQDiff	IBM Research Haifa	Yes	Yes	Yes
JuruNoSS	IBM Research Haifa	Yes	Yes	No
widittdb1	Indiana Univ. Bloomington	No	No	No
widittdb1r1	Indiana Univ. Bloomington	No	No	Yes
widittdf1r1	Indiana Univ. Bloomington	Yes	Yes	Yes
widittdf1r2	Indiana Univ. Bloomington	Yes	Yes	Yes
Merc1td	IRIT/SIG	No	No	No
Merc1ti	IRIT/SIG	No	No	No
Merc2tm	IRIT/SIG	No	No	No
Merc2tp	IRIT/SIG	No	No	No
KUCONTENT	Kasetsart Univ.	No	No	No
03wume206	Lehigh Univ.	No	Yes	Yes
03wume359	Lehigh Univ.	No	Yes	Yes
meijihilw1	Meiji Univ.	Yes	No	No
meijihilw2	Meiji Univ.	Yes	No	No
meijihilw3	Meiji Univ.	Yes	No	Yes
meijihilw4	Meiji Univ.	Yes	No	Yes
meijihilw5	Meiji Univ.	Yes	No	Yes
MSRA1001	Microsoft Research Asia	Yes	Yes	No
MSRA1002	Microsoft Research Asia	Yes	Yes	No
MSRA3	Microsoft Research Asia	Yes	Yes	Yes

Web Track, Topic Distillation Task (continued)

MSRA4002	Microsoft Research Asia	Yes	Yes	Yes
MSRA4003	Microsoft Research Asia	Yes	Yes	Yes
UniNEtd1	Universite de Neuchatel	Yes	Yes	Yes
UniNEtd2	Universite de Neuchatel	Yes	Yes	No
UniNEtd3	Universite de Neuchatel	Yes	Yes	Yes
UniNEtd4	Universite de Neuchatel	Yes	Yes	Yes
UniNEtd5	Universite de Neuchatel	Yes	No	Yes
topics0	Saarland Univ.	No	No	Yes
THUIRtd0301	Tsinghua Univ. (Ma)	Yes	Yes	Yes
THUIRtd0302	Tsinghua Univ. (Ma)	Yes	Yes	Yes
THUIRtd0303	Tsinghua Univ. (Ma)	Yes	Yes	Yes
THUIRtd0304	Tsinghua Univ. (Ma)	Yes	Yes	Yes
THUIRtd0305	Tsinghua Univ. (Ma)	Yes	Yes	Yes
UAmsT03WtLM3	Univ. of Amsterdam	Yes	Yes	No
UAmsT03WtLMI	Univ. of Amsterdam	No	No	Yes
UAmsT03WtOk3	Univ. of Amsterdam	Yes	Yes	No
UAmsT03WtOkC	Univ. of Amsterdam	No	No	No
UAmsT03WtOkI	Univ. of Amsterdam	No	No	Yes
uogtd1c	Univ. of Glasgow	No	No	No
uogtd2ca	Univ. of Glasgow	No	Yes	No
uogtd3cas	Univ. of Glasgow	No	Yes	No
uogtd4cahs	Univ. of Glasgow	No	Yes	No
uogtd5cass	Univ. of Glasgow	No	Yes	Yes
UIUC03W2s	Univ. of Illinois at Urbana-Champaign	No	No	No
UIUC03Wb	Univ. of Illinois at Urbana-Champaign	No	No	No
UIUC03Wp	Univ. of Illinois at Urbana-Champaign	No	No	Yes
UIUC03Wu1	Univ. of Illinois at Urbana-Champaign	No	No	Yes
UIUC03Wu2	Univ. of Illinois at Urbana-Champaign	No	No	Yes
C2A	Univ. of Maryland Baltimore County	No	No	Yes
C2B	Univ. of Maryland Baltimore County	No	No	Yes
MU03td01	Univ. of Melbourne	Yes	Yes	No
MU03td03	Univ. of Melbourne	Yes	Yes	No
MU03td04	Univ. of Melbourne	Yes	Yes	No
MU03td05	Univ. of Melbourne	Yes	No	No
SBBASE	Univ. of Sunderland	Yes	No	Yes
SBUNIQUE	Univ. of Sunderland	Yes	No	Yes
TBBASE	Univ. of Sunderland	Yes	No	Yes
TBUNIQUE	Univ. of Sunderland	Yes	No	Yes
VTtdgp33	Virginia Tech	No	Yes	No
VTtdgp41	Virginia Tech	No	Yes	No
VTtdgp5055	Virginia Tech	No	Yes	No
VTtdgp52	Virginia Tech	No	Yes	No
VTtdok4	Virginia Tech	No	Yes	No

1 Common Evaluation Measures

- Recall
  A measure of the ability of a system to present all relevant items.

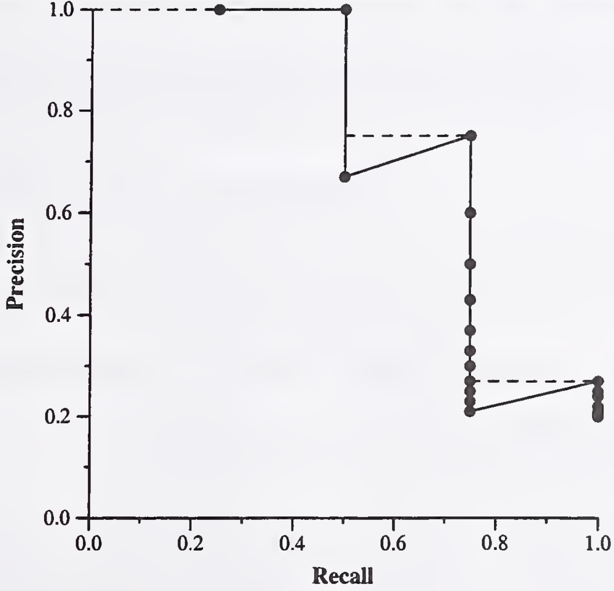
$$\text{recall} = \frac{\text{number of relevant items retrieved}}{\text{number of relevant items in collection}}$$

- Precision.
  A measure of the ability of a system to present only relevant items.

$$\text{precision} = \frac{\text{number of relevant items retrieved}}{\text{total number of items retrieved}}$$

Precision and recall are set-based measures. That is, they evaluate the quality of an unordered set of retrieved documents. To evaluate ranked lists, precision can be plotted against recall after each retrieved document as shown in the example below. To facilitate computing average performance over a set of topics— each with a different number of relevant documents— individual topic precision values are interpolated to a set of standard recall levels (0 to 1 in increments of .1). The particular rule used to interpolate precision at standard recall level i is to use the maximum precision obtained for the topic for any actual recall level greater than or equal to i. Note that while precision is not defined at a recall of 0.0, this interpolation rule does define an interpolated value for recall level 0.0. In the example, the actual precision values are plotted with circles (and connected by a solid line) and the interpolated precision is shown with the dashed line.

> Example: Assume a document collection has 20 documents, four of which are relevant to topic t. Further assume a retrieval system ranks the relevant documents first, second, fourth, and fifteenth. The exact recall points are 0.25, 0.5, 0.75, and 1.0. Using the interpolation rule, the interpolated precision for all standard recall levels up to .5 is 1, the interpolated precision for recall levels .6 and .7 is .75, and the interpolated precision for recall levels .8 or greater is .27.



2 trec_eval Evaluation Report

Retrieval tasks whose results are a ranked list of documents can be evaluated by the trec_eval program. Examples of such tasks are the task in the robust track, the document-level evaluation within the HARD track, and the primary task within the genome track. trec_eval was written by Chris Buckley when he was at Cornell University; it can be obtained by anonymous ftp from Cornell in the directory pub/smart at ftp.cs.cornell.edu. An evaluation report for a run evaluated by trec_eval is comprised of a header (containing the task and organization name), 3 tables, and 2 graphs as described below.

2.1 Tables

I. "Summary Statistics" Table
Table 1 is a sample "Summary Statistics" Table

Table 1: Sample "Summary Statistics" Table.

Summary Statistics	
Run	Cor7A1clt–automatic, title
Number of Topics	50
Total number of documents over all topics	
Retrieved:	50000
Relevant:	4674
Rel_ret:	2621

A. Run
A description of the run. It contains the run tag provided by the participant, and various details about the runs such as whether queries were constructed manually or automatically.

B. Number of Topics
Number of topics searched in this run (generally 50 topics are run for each task).

C. Total number of documents over all topics (the number of topics given in B).

i. Retrieved
Number of documents submitted to NIST. This is usually 50,000 (50 topics × 1000 documents), but is less when fewer than 1000 documents are retrieved per topic.

ii. Relevant
Total possible relevant documents within a given task and category.

iii. Rel_ret
Total number of relevant documents returned by a run over all the topics.

II. "Recall Level Precision Averages" Table.
Table 2 is a sample "Recall Level Precision Averages" Table.

A. Precision at 11 standard recall levels
The precision averages at 11 standard recall levels are used to compare the performance of different systems and as the input for plotting the recall-precision graph (see below). Each recall-precision average is computed by summing the interpolated precisions at the specified recall cutoff value (denoted by $\sum P_\lambda$ where P_λ is the interpolated precision at recall level λ) and then dividing by the number of topics.

$$\frac{\sum_{i=1}^{NUM} P_\lambda}{NUM} \qquad \lambda = \{0.0, 0.1, 0.2, 0.3, \ldots, 1.0\}$$

Table 2: Sample "Recall Level Precision Averages" Table.

Recall Level Precision Averages	
Recall	Precision
0.00	0.6169
0.10	0.4517
0.20	0.3938
0.30	0.3243
0.40	0.2715
0.50	0.2224
0.60	0.1642
0.70	0.1342
0.80	0.0904
0.90	0.0472
1.00	0.0031
Average precision over all relevant docs	
non-interpolated	0.2329

- Interpolating recall-precision
  Standard recall levels facilitate averaging and plotting retrieval results.

B. Average precision over all relevant documents, non-interpolated
This is a single-valued measure that reflects the performance over all relevant documents. It rewards systems that retrieve relevant documents quickly (highly ranked).

The measure is not an average of the precision at standard recall levels. Rather, it is the average of the precision value obtained after each relevant document is retrieved. (When a relevant document is not retrieved at all, its precision is assumed to be 0.) As an example, consider a query that has four relevant documents which are retrieved at ranks 1, 2, 4, and 7. The actual precision obtained when each relevant document is retrieved is 1, 1, 0.75, and 0.57, respectively, the mean of which is 0.83. Thus, the average precision over all relevant documents for this query is 0.83.

III. "Document Level Averages" Table
Table 3 is a sample "Document Level Averages" Table.

Table 3: Sample "Document Level Averages" Table.

Document Level Averages	
	Precision
At 5 docs	0.4280
At 10 docs	0.3960
At 15 docs	0.3493
At 20 docs	0.3370
At 30 docs	0.3100
At 100 docs	0.2106
At 200 docs	0.1544
At 500 docs	0.0875
At 1000 docs	0.0524
R−Precision (precision after R docs retrieved (where R is the number of relevant documents))	
Exact	0.2564

A. Precision at 9 document cutoff values

The precision computed after a given number of documents have been retrieved reflects the actual measured system performance as a user might see it. Each document precision average is computed by summing the precisions at the specified document cutoff value and dividing by the number of topics (50).

B. R-Precision

R-Precision is the precision after R documents have been retrieved, where R is the number of relevant documents for the topic. It de-emphasizes the exact ranking of the retrieved relevant documents, which can be particularly useful in TREC where there are large numbers of relevant documents.

The average R-Precision for a run is computed by taking the mean of the R-Precisions of the individual topics in the run. For example, assume a run consists of two topics, one with 50 relevant documents and another with 10 relevant documents. If the retrieval system returns 17 relevant documents in the top 50 documents for the first topic, and 7 relevant documents in the top 10 for the second topic, then the run's R-Precision would be $\frac{\frac{17}{50}+\frac{7}{10}}{2}$ or 0.52.

2.2 Graphs

I. Recall-Precision Graph

Figure 1 is a sample Recall-Precision Graph.

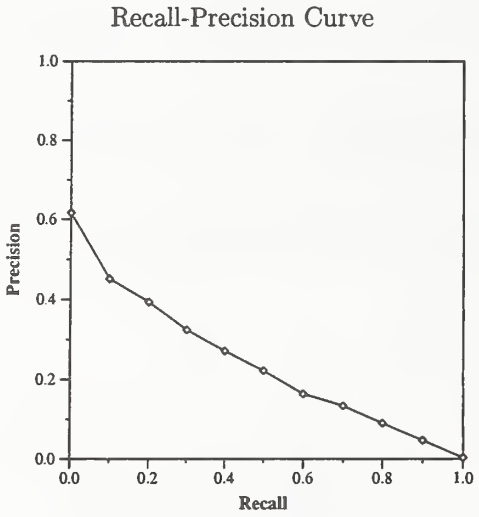


Figure 1: Sample Recall-Precision Graph.

The Recall-Precision Graph is created using the 11 cutoff values from the Recall Level Precision Averages. Typically these graphs slope downward from left to right, enforcing the notion that as more relevant documents are retrieved (recall increases), the more nonrelevant documents are retrieved (precision decreases).

This graph is the most commonly used method for comparing systems. The plots of different runs can be superimposed on the same graph to determine which run is superior. Curves closest to the upper right-hand corner of the graph (where recall and precision are maximized) indicate the best performance. Comparisons are best made in three different recall ranges: 0 to 0.2, 0.2 to 0.8, and 0.8 to 1. These ranges characterize high precision, middle recall, and high recall performance, respectively.

II. Average Precision Histogram.
Figure 2 is a sample Average Precision Histogram.

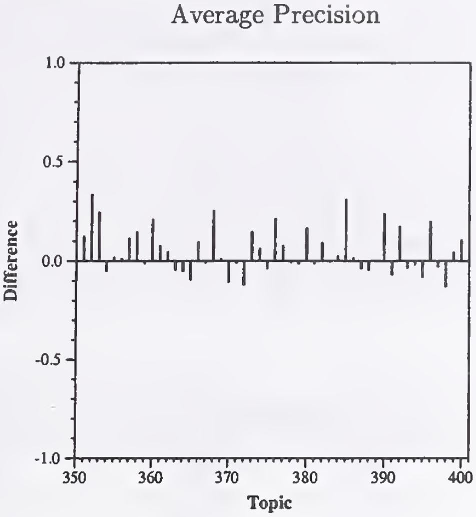


Figure 2: Sample Average Precision Histogram.

The Average Precision Histogram measures the average precision of a run on each topic against the median average precision of all corresponding runs on that topic. This graph is intended to give insight into the performance of individual systems and the types of topics that they handle well.

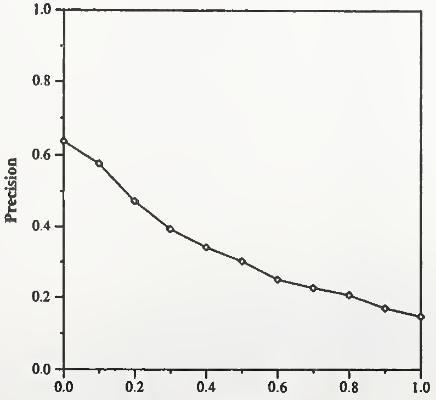
Genomics track, primary task results — Axontologic, Inc.

Summary Statistics	
Run ID:	axon1
Run Description	Primary task, automatic
Number of Topics:	50

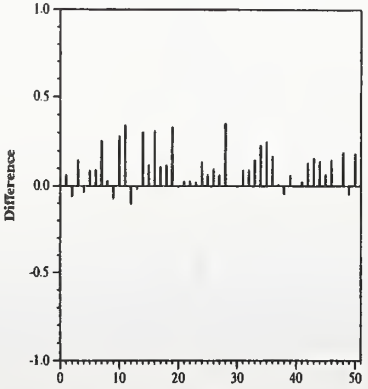
Total number of documents over all topics	
Retrieved:	39426
Relevant:	566
Rel-ret:	522

Recall Level Averages	
Recall	Precision
0.00	0.6372
0.10	0.5764
0.20	0.4725
0.30	0.3930
0.40	0.3420
0.50	0.3020
0.60	0.2504
0.70	0.2268
0.80	0.2070
0.90	0.1699
1.00	0.1468
Mean average precision	
non-interpolated	0.3118

Document Level Averages	
	Precision
At 5 docs	0.3200
At 10 docs	0.2400
At 15 docs	0.2120
At 20 docs	0.1890
At 30 docs	0.1493
At 100 docs	0.0742
At 200 docs	0.0437
At 500 docs	0.0203
At 1000 docs	0.0104
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.2935



Recall-Precision Curve



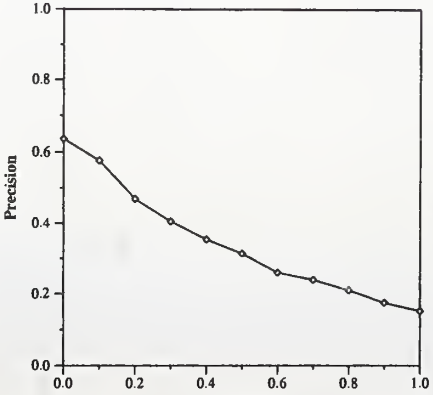
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	axon2
Run Description	Primary task, automatic
Number of Topics:	50

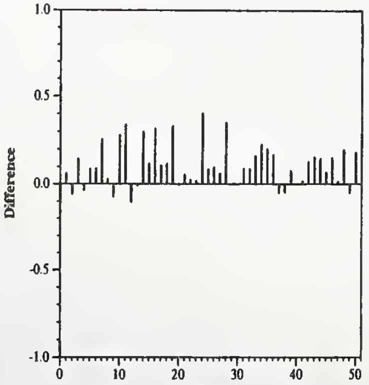
Total number of documents over all topics	
Retrieved:	39080
Relevant:	566
Rel-ret:	524

Recall Level Averages	
Recall	Precision
0.00	0.6361
0.10	0.5764
0.20	0.4697
0.30	0.4056
0.40	0.3543
0.50	0.3141
0.60	0.2598
0.70	0.2394
0.80	0.2100
0.90	0.1748
1.00	0.1518
Mean average precision	
non-interpolated	0.3173

Document Level Averages	
	Precision
At 5 docs	0.3200
At 10 docs	0.2500
At 15 docs	0.2187
At 20 docs	0.1930
At 30 docs	0.1520
At 100 docs	0.0748
At 200 docs	0.0433
At 500 docs	0.0201
At 1000 docs	0.0105
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.2959



Recall-Precision Curve



Difference from Median in Average Precision per Topic

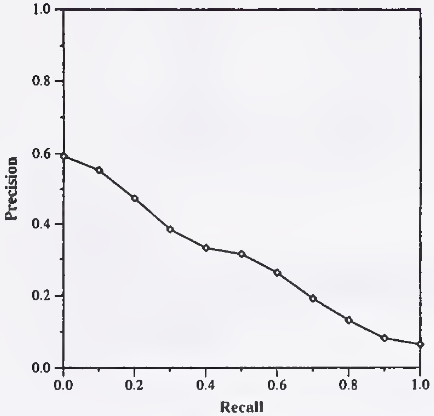
Genomics track, primary task results.— California State University San Marcos

Summary Statistics	
Run ID:	CSUSM1
Run Description	Primary task, automatic
Number of Topics:	50

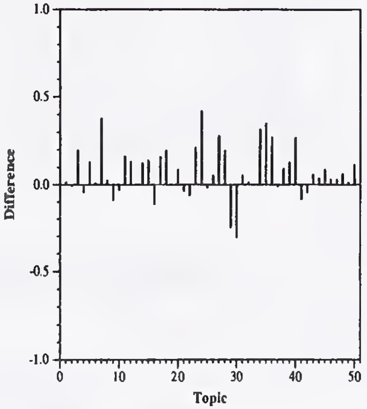
Total number of documents over all topics	
Retrieved:	2586
Relevant:	566
Rel-ret:	359

Recall Level Averages	
Recall	Precision
0.00	0.5929
0.10	0.5540
0.20	0.4737
0.30	0.3854
0.40	0.3324
0.50	0.3152
0.60	0.2629
0.70	0.1920
0.80	0.1321
0.90	0.0825
1.00	0.0652
Mean average precision	
non-interpolated	0.2859

Document Level Averages	
	Precision
At 5 docs	0.3240
At 10 docs	0.2560
At 15 docs	0.2053
At 20 docs	0.1760
At 30 docs	0.1407
At 100 docs	0.0718
At 200 docs	0.0359
At 500 docs	0.0144
At 1000 docs	0.0072
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.3032



Recall-Precision Curve



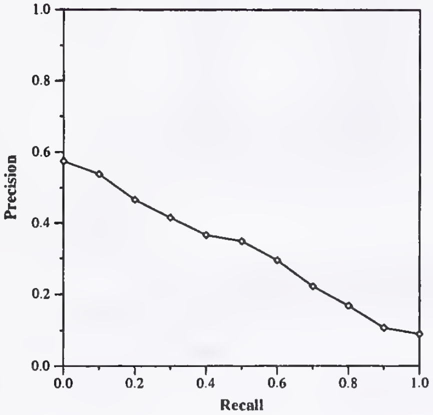
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	CSUSM2
Run Description	Primary task, automatic
Number of Topics:	50

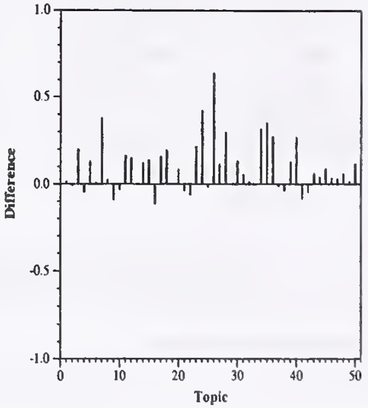
Total number of documents over all topics	
Retrieved:	2629
Relevant:	566
Rel-ret:	371

Recall Level Averages	
Recall	Precision
0.00	0.5749
0.10	0.5385
0.20	0.4657
0.30	0.4152
0.40	0.3656
0.50	0.3478
0.60	0.2941
0.70	0.2221
0.80	0.1680
0.90	0.1075
1.00	0.0902
Mean average precision	
non-interpolated	0.3079

Document Level Averages	
	Precision
At 5 docs	0.3320
At 10 docs	0.2680
At 15 docs	0.2160
At 20 docs	0.1880
At 30 docs	0.1487
At 100 docs	0.0742
At 200 docs	0.0371
At 500 docs	0.0148
At 1000 docs	0.0074
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.3099



Recall-Precision Curve



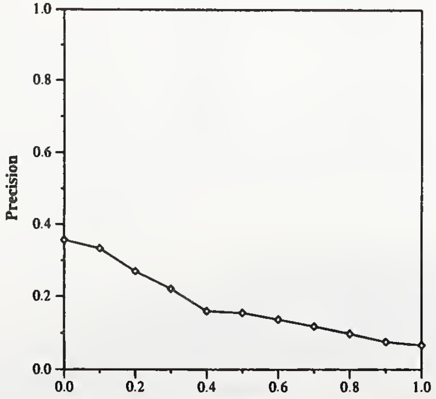
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	DcuMesh1
Run Description	Primary task, automatic
Number of Topics:	50

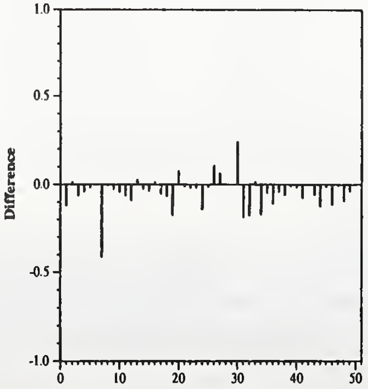
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	440

Recall Level Averages	
Recall	Precision
0.00	0.3562
0.10	0.3332
0.20	0.2696
0.30	0.2209
0.40	0.1602
0.50	0.1556
0.60	0.1372
0.70	0.1187
0.80	0.0984
0.90	0.0763
1.00	0.0669
Mean average precision	
non-interpolated	0.1669

Document Level Averages	
	Precision
At 5 docs	0.1560
At 10 docs	0.1360
At 15 docs	0.1120
At 20 docs	0.1040
At 30 docs	0.0827
At 100 docs	0.0522
At 200 docs	0.0338
At 500 docs	0.0160
At 1000 docs	0.0088
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1640



Recall-Precision Curve



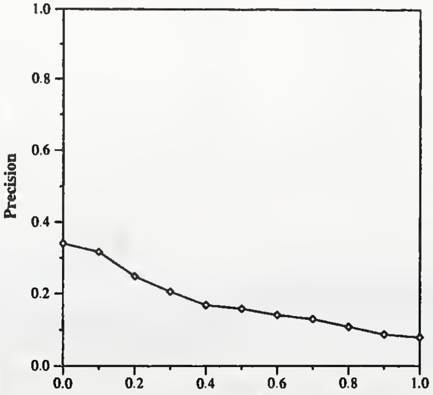
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	DcuMesh2
Run Description	Primary task, automatic
Number of Topics:	50

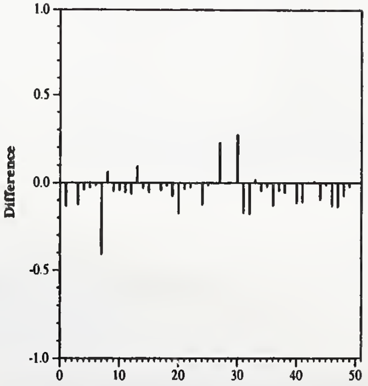
Total number of documents over all topics	
Retrieved:	49417
Relevant:	566
Rel-ret:	440

Recall Level Averages	
Recall	Precision
0.00	0.3399
0.10	0.3164
0.20	0.2484
0.30	0.2054
0.40	0.1684
0.50	0.1586
0.60	0.1412
0.70	0.1299
0.80	0.1086
0.90	0.0872
1.00	0.0790
Mean average precision	
non-interpolated	0.1667

Document Level Averages	
	Precision
At 5 docs	0.1680
At 10 docs	0.1360
At 15 docs	0.1187
At 20 docs	0.0980
At 30 docs	0.0840
At 100 docs	0.0558
At 200 docs	0.0357
At 500 docs	0.0164
At 1000 docs	0.0088
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1533



Recall-Precision Curve



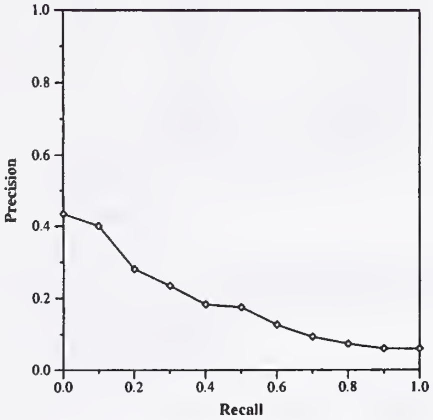
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	ErasmusMC2
Run Description	Primary task, automatic
Number of Topics:	50

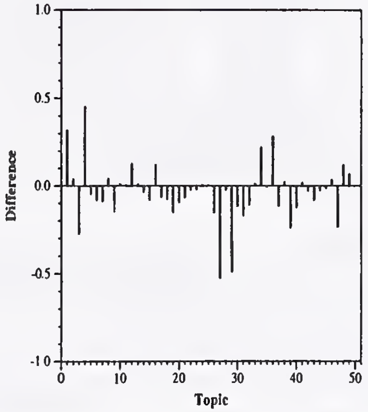
Total number of documents over all topics	
Retrieved:	35477
Relevant:	566
Rel-ret:	333

Recall Level Averages	
Recall	Precision
0.00	0.4341
0.10	0.4003
0.20	0.2807
0.30	0.2351
0.40	0.1837
0.50	0.1759
0.60	0.1272
0.70	0.0936
0.80	0.0737
0.90	0.0605
1.00	0.0599
Mean average precision	
non-interpolated	0.1754

Document Level Averages	
	Precision
At 5 docs	0.1800
At 10 docs	0.1380
At 15 docs	0.1253
At 20 docs	0.1160
At 30 docs	0.1020
At 100 docs	0.0496
At 200 docs	0.0285
At 500 docs	0.0126
At 1000 docs	0.0067
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1767



Recall-Precision Curve



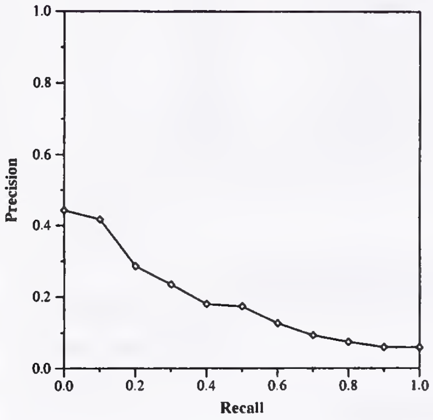
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	ErasmusMC3
Run Description	Primary task, automatic
Number of Topics:	50

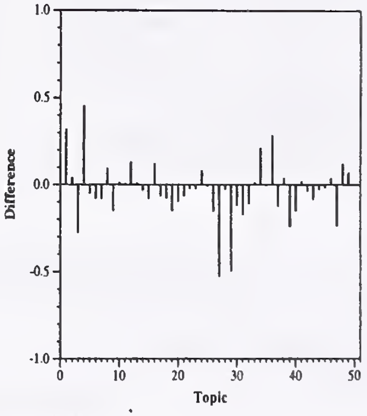
Total number of documents over all topics	
Retrieved:	45623
Relevant:	566
Rel-ret:	312

Recall Level Averages	
Recall	Precision
0.00	0.4429
0.10	0.4170
0.20	0.2863
0.30	0.2352
0.40	0.1806
0.50	0.1740
0.60	0.1268
0.70	0.0930
0.80	0.0739
0.90	0.0595
1.00	0.0588
Mean average precision	
non-interpolated	0.1770

Document Level Averages	
	Precision
At 5 docs	0.1800
At 10 docs	0.1360
At 15 docs	0.1227
At 20 docs	0.1140
At 30 docs	0.1007
At 100 docs	0.0500
At 200 docs	0.0289
At 500 docs	0.0123
At 1000 docs	0.0062
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1720



Recall-Precision Curve



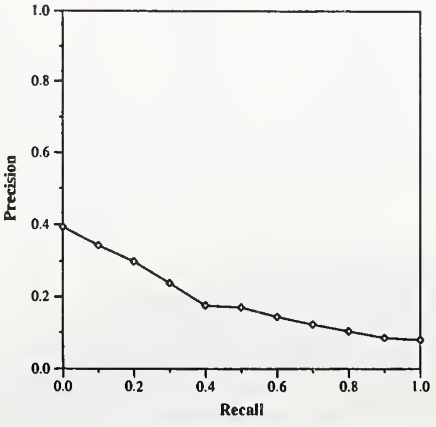
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	humG03ns
Run Description	Primary task, automatic
Number of Topics:	50

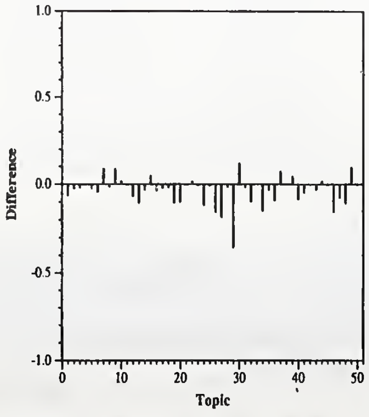
Total number of documents over all topics	
Retrieved:	49059
Relevant:	566
Rel-ret:	505

Recall Level Averages	
Recall	Precision
0.00	0.3935
0.10	0.3438
0.20	0.2989
0.30	0.2387
0.40	0.1762
0.50	0.1711
0.60	0.1446
0.70	0.1231
0.80	0.1039
0.90	0.0853
1.00	0.0795
Mean average precision	
non-interpolated	0.1753

Document Level Averages	
	Precision
At 5 docs	0.1640
At 10 docs	0.1480
At 15 docs	0.1253
At 20 docs	0.1170
At 30 docs	0.0987
At 100 docs	0.0566
At 200 docs	0.0392
At 500 docs	0.0185
At 1000 docs	0.0101
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1530



Recall-Precision Curve



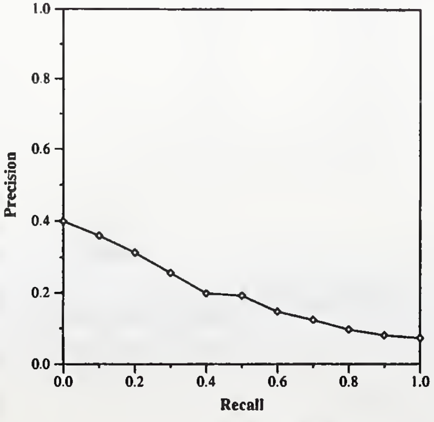
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	humG03ns5
Run Description	Primary task, automatic
Number of Topics:	50

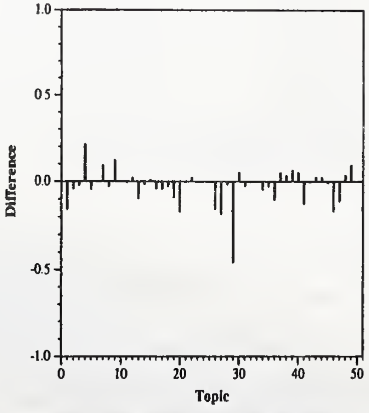
Total number of documents over all topics	
Retrieved:	49059
Relevant:	566
Rel-ret:	511

Recall Level Averages	
Recall	Precision
0.00	0.3992
0.10	0.3596
0.20	0.3128
0.30	0.2566
0.40	0.1987
0.50	0.1922
0.60	0.1470
0.70	0.1235
0.80	0.0964
0.90	0.0802
1.00	0.0722
Mean average precision	
non-interpolated	0.1847

Document Level Averages	
	Precision
At 5 docs	0.1800
At 10 docs	0.1580
At 15 docs	0.1413
At 20 docs	0.1230
At 30 docs	0.1080
At 100 docs	0.0622
At 200 docs	0.0403
At 500 docs	0.0190
At 1000 docs	0.0102
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1672



Recall-Precision Curve



Difference from Median in Average Precision per Topic

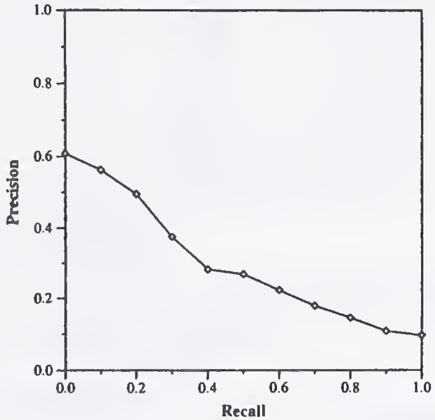
Genomics track, primary task results — IBM TJ Watson Research Center

Summary Statistics	
Run ID:	IBMbt1
Run Description	Primary task, automatic
Number of Topics:	50

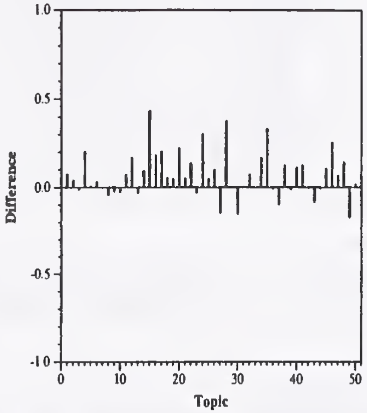
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	456

Recall Level Averages	
Recall	Precision
0.00	0.6080
0.10	0.5626
0.20	0.4952
0.30	0.3755
0.40	0.2827
0.50	0.2691
0.60	0.2234
0.70	0.1789
0.80	0.1459
0.90	0.1091
1.00	0.0971
Mean average precision	
non-interpolated	0.2823

Document Level Averages	
	Precision
At 5 docs	0.2840
At 10 docs	0.2260
At 15 docs	0.1880
At 20 docs	0.1660
At 30 docs	0.1347
At 100 docs	0.0686
At 200 docs	0.0403
At 500 docs	0.0176
At 1000 docs	0.0091
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.2780



Recall-Precision Curve



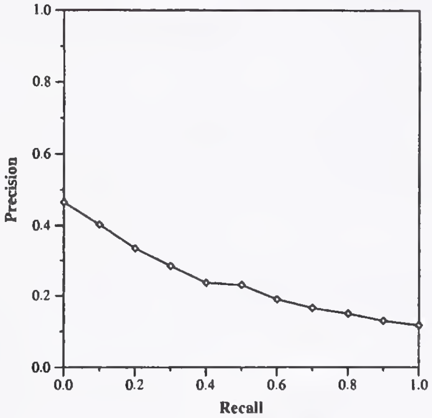
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	IBMbt2
Run Description	Primary task, automatic
Number of Topics:	50

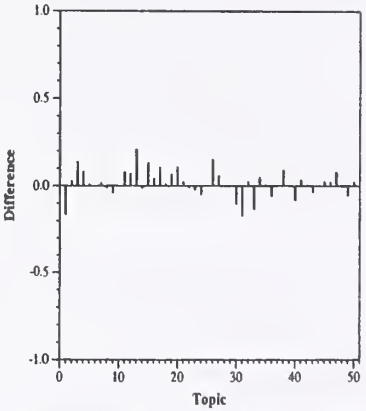
Total number of documents over all topics	
Retrieved:	48327
Relevant:	566
Rel-ret:	534

Recall Level Averages	
Recall	Precision
0.00	0.4649
0.10	0.4026
0.20	0.3346
0.30	0.2843
0.40	0.2367
0.50	0.2306
0.60	0.1894
0.70	0.1651
0.80	0.1495
0.90	0.1292
1.00	0.1169
Mean average precision	
non-interpolated	0.2259

Document Level Averages	
	Precision
At 5 docs	0.2080
At 10 docs	0.1800
At 15 docs	0.1547
At 20 docs	0.1420
At 30 docs	0.1260
At 100 docs	0.0694
At 200 docs	0.0438
At 500 docs	0.0203
At 1000 docs	0.0107
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.2057



Recall-Precision Curve



Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	vvP05mil3
Run Description	Primary task, automatic
Number of Topics:	50

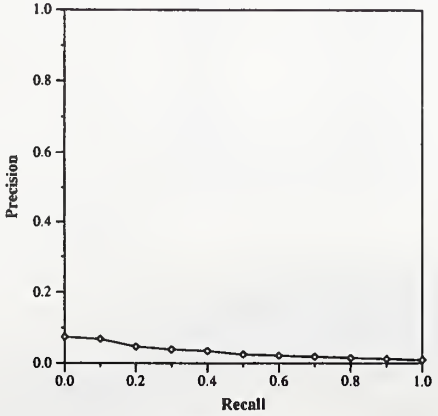
Total number of documents over all topics	
Retrieved:	26217
Relevant:	566
Rel-ret:	274

Recall Level Averages	
Recall	Precision
0.00	0.0743
0.10	0.0686
0.20	0.0469
0.30	0.0389
0.40	0.0348
0.50	0.0250
0.60	0.0217
0.70	0.0185
0.80	0.0151
0.90	0.0125
1.00	0.0092

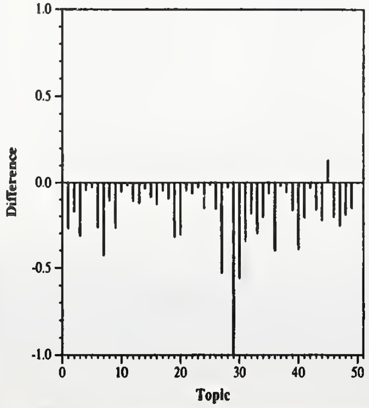
Mean average precision	
non-interpolated	0.0271

Document Level Averages	
	Precision
At 5 docs	0.0200
At 10 docs	0.0220
At 15 docs	0.0267
At 20 docs	0.0300
At 30 docs	0.0287
At 100 docs	0.0230
At 200 docs	0.0173
At 500 docs	0.0098
At 1000 docs	0.0055

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0249



Recall-Precision Curve



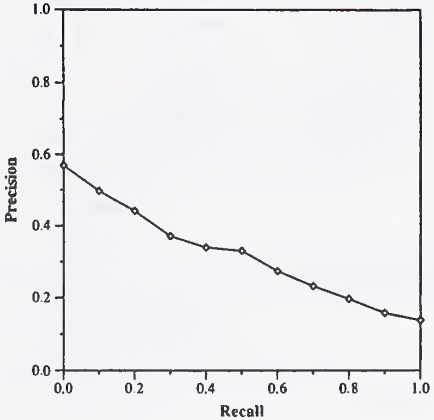
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	KUBIOIRNE
Run Description	Primary task, automatic
Number of Topics:	50

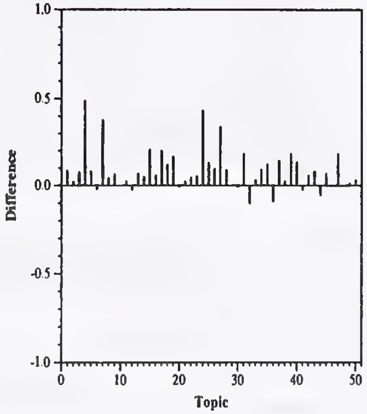
Total number of documents over all topics	
Retrieved:	48597
Relevant:	566
Rel-ret:	532

Recall Level Averages	
Recall	Precision
0.00	0.5685
0.10	0.4969
0.20	0.4417
0.30	0.3717
0.40	0.3402
0.50	0.3305
0.60	0.2741
0.70	0.2331
0.80	0.1972
0.90	0.1581
1.00	0.1380
Mean average precision	
non-interpolated	0.2980

Document Level Averages	
	Precision
At 5 docs	0.2960
At 10 docs	0.2320
At 15 docs	0.1960
At 20 docs	0.1710
At 30 docs	0.1507
At 100 docs	0.0750
At 200 docs	0.0455
At 500 docs	0.0202
At 1000 docs	0.0106
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.2837



Recall-Precision Curve



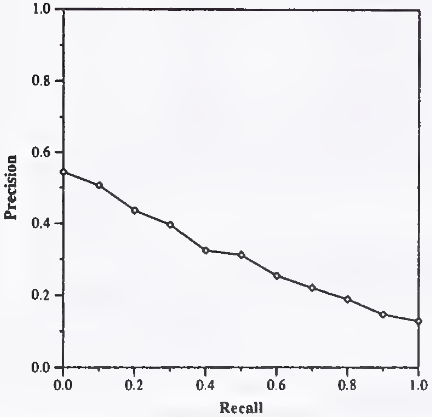
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	KUBIOIRRAW
Run Description	Primary task, automatic
Number of Topics:	50

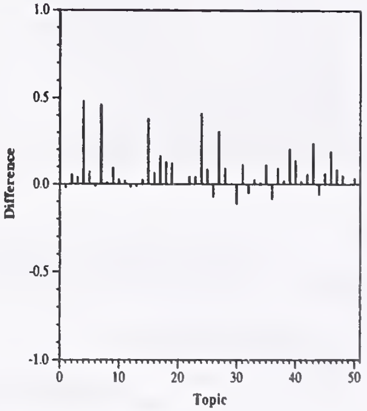
Total number of documents over all topics	
Retrieved:	48598
Relevant:	566
Rel-ret:	541

Recall Level Averages	
Recall	Precision
0.00	0.5442
0.10	0.5062
0.20	0.4362
0.30	0.3963
0.40	0.3242
0.50	0.3117
0.60	0.2538
0.70	0.2209
0.80	0.1890
0.90	0.1469
1.00	0.1283
Mean average precision	
non-interpolated	0.2937

Document Level Averages	
	Precision
At 5 docs	0.2840
At 10 docs	0.2240
At 15 docs	0.1920
At 20 docs	0.1690
At 30 docs	0.1473
At 100 docs	0.0762
At 200 docs	0.0467
At 500 docs	0.0208
At 1000 docs	0.0108
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.2696



Recall-Precision Curve



Difference from Median in Average Precision per Topic

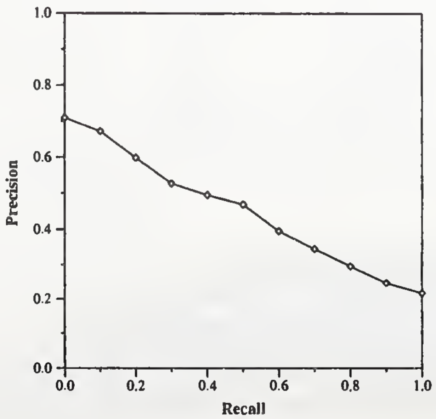
Genomics track, primary task results — National Library of Medicine and U. Maryland

Summary Statistics	
Run ID:	NLMUMDSE
Run Description	Primary task, automatic
Number of Topics:	50

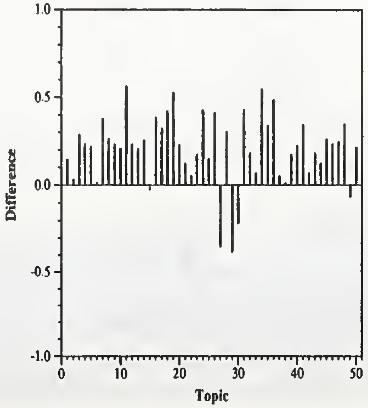
Total number of documents over all topics	
Retrieved:	25416
Relevant:	566
Rel-ret:	542

Recall Level Averages	
Recall	Precision
0.00	0.7099
0.10	0.6723
0.20	0.5984
0.30	0.5269
0.40	0.4950
0.50	0.4690
0.60	0.3953
0.70	0.3454
0.80	0.2956
0.90	0.2475
1.00	0.2170
Mean average precision	
non-interpolated	0.4165

Document Level Averages	
	Precision
At 5 docs	0.4000
At 10 docs	0.3160
At 15 docs	0.2733
At 20 docs	0.2420
At 30 docs	0.1967
At 100 docs	0.0898
At 200 docs	0.0504
At 500 docs	0.0215
At 1000 docs	0.0108
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.3926



Recall-Precision Curve



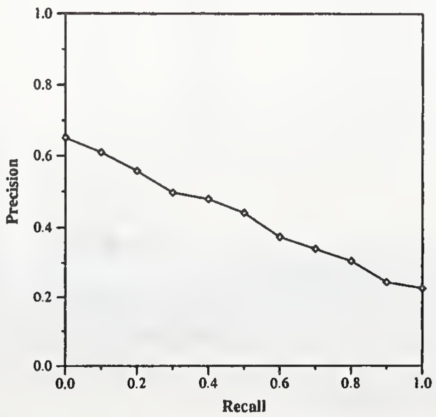
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	NLMUMDSRB
Run Description	Primary task, automatic
Number of Topics:	50

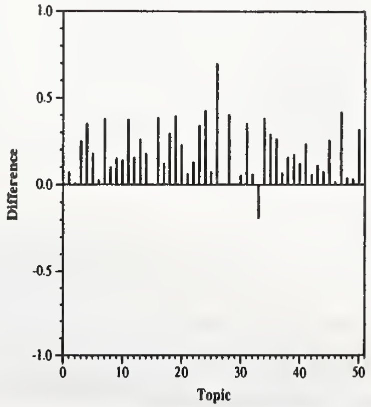
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	560

Recall Level Averages	
Recall	Precision
0.00	0.6510
0.10	0.6094
0.20	0.5576
0.30	0.4970
0.40	0.4790
0.50	0.4406
0.60	0.3733
0.70	0.3398
0.80	0.3052
0.90	0.2442
1.00	0.2260
Mean average precision	
non-interpolated	0.3994

Document Level Averages	
	Precision
At 5 docs	0.3800
At 10 docs	0.3200
At 15 docs	0.2667
At 20 docs	0.2280
At 30 docs	0.1827
At 100 docs	0.0870
At 200 docs	0.0505
At 500 docs	0.0220
At 1000 docs	0.0112
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.3892



Recall-Precision Curve



Difference from Median in Average Precision per Topic

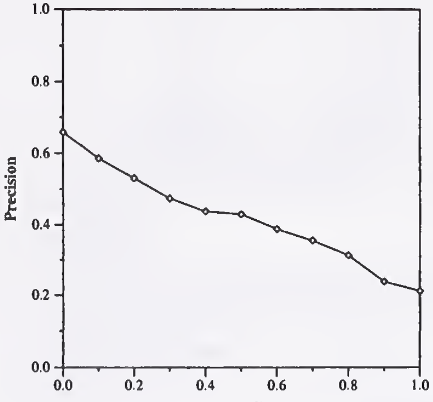
Genomics track, primary task results — National Research Council Canada

Summary Statistics	
Run ID:	nrc1
Run Description	Primary task, automatic
Number of Topics:	50

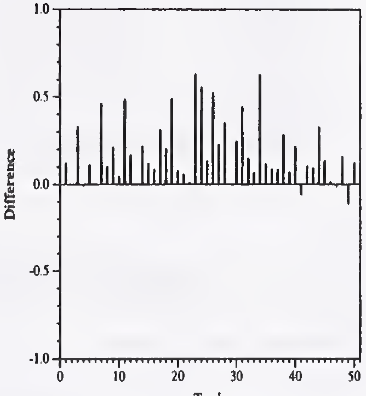
Total number of documents over all topics	
Retrieved:	9824
Relevant:	566
Rel-ret:	543

Recall Level Averages	
Recall	Precision
0.00	0.6587
0.10	0.5859
0.20	0.5303
0.30	0.4743
0.40	0.4376
0.50	0.4292
0.60	0.3873
0.70	0.3548
0.80	0.3122
0.90	0.2382
1.00	0.2110
Mean average precision	
non-interpolated	0.3941

Document Level Averages	
	Precision
At 5 docs	0.3840
At 10 docs	0.2940
At 15 docs	0.2453
At 20 docs	0.2190
At 30 docs	0.1873
At 100 docs	0.0854
At 200 docs	0.0497
At 500 docs	0.0216
At 1000 docs	0.0109
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.3429



Recall-Precision Curve



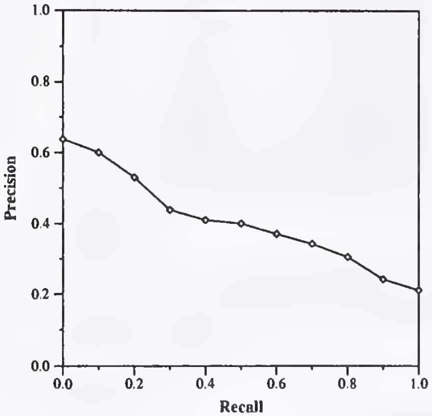
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	nrc2
Run Description	Primary task, automatic
Number of Topics:	50

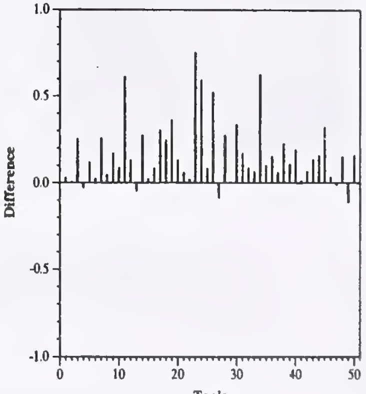
Total number of documents over all topics	
Retrieved:	9824
Relevant:	566
Rel-ret:	543

Recall Level Averages	
Recall	Precision
0.00	0.6380
0.10	0.6003
0.20	0.5308
0.30	0.4390
0.40	0.4098
0.50	0.4000
0.60	0.3705
0.70	0.3425
0.80	0.3046
0.90	0.2417
1.00	0.2105
Mean average precision	
non-interpolated	0.3771

Document Level Averages	
	Precision
At 5 docs	0.3520
At 10 docs	0.2760
At 15 docs	0.2440
At 20 docs	0.2180
At 30 docs	0.1820
At 100 docs	0.0856
At 200 docs	0.0497
At 500 docs	0.0216
At 1000 docs	0.0109
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.3462



Recall-Precision Curve



Difference from Median in Average Precision per Topic

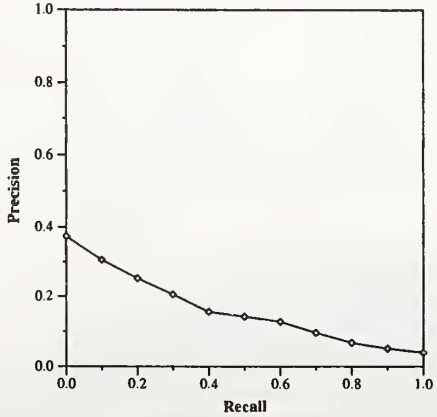
Genomics track, primary task results — NTT Communication Science Laboratories

Summary Statistics	
Run ID:	balsc2
Run Description	Primary task, automatic
Number of Topics:	50

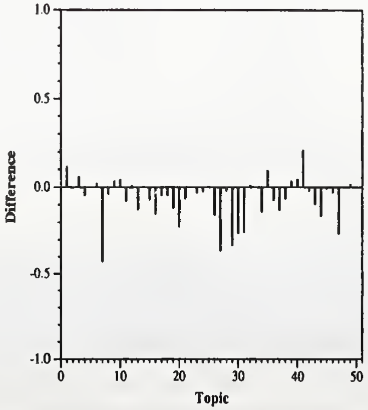
Total number of documents over all topics	
Retrieved:	9768
Relevant:	566
Rel-ret:	375

Recall Level Averages	
Recall	Precision
0.00	0.3734
0.10	0.3062
0.20	0.2526
0.30	0.2056
0.40	0.1562
0.50	0.1425
0.60	0.1278
0.70	0.0962
0.80	0.0678
0.90	0.0513
1.00	0.0397
Mean average precision	
non-interpolated	0.1481

Document Level Averages	
	Precision
At 5 docs	0.1600
At 10 docs	0.1360
At 15 docs	0.1293
At 20 docs	0.1170
At 30 docs	0.0953
At 100 docs	0.0518
At 200 docs	0.0315
At 500 docs	0.0146
At 1000 docs	0.0075
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1386



Recall-Precision Curve



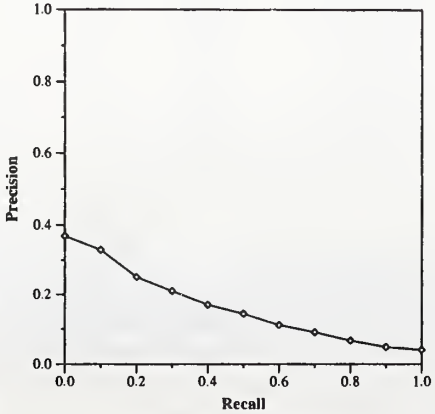
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	balsc3
Run Description	Primary task, automatic
Number of Topics:	50

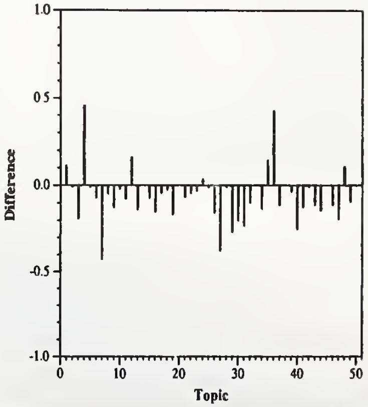
Total number of documents over all topics	
Retrieved:	12383
Relevant:	566
Rel-ret:	374

Recall Level Averages	
Recall	Precision
0.00	0.3688
0.10	0.3294
0.20	0.2511
0.30	0.2099
0.40	0.1702
0.50	0.1445
0.60	0.1118
0.70	0.0910
0.80	0.0672
0.90	0.0489
1.00	0.0407
Mean average precision	
non-interpolated	0.1528

Document Level Averages	
	Precision
At 5 docs	0.1680
At 10 docs	0.1440
At 15 docs	0.1200
At 20 docs	0.1050
At 30 docs	0.0860
At 100 docs	0.0498
At 200 docs	0.0300
At 500 docs	0.0138
At 1000 docs	0.0075
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1585



Recall-Precision Curve



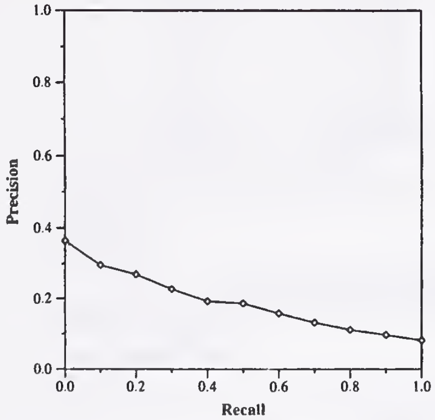
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	ohsuboost
Run Description	Primary task, automatic
Number of Topics:	50

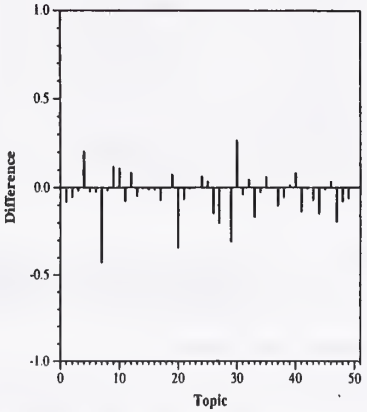
Total number of documents over all topics	
Retrieved:	14820
Relevant:	566
Rel-ret:	419

Recall Level Averages	
Recall	Precision
0.00	0.3645
0.10	0.2962
0.20	0.2697
0.30	0.2273
0.40	0.1926
0.50	0.1866
0.60	0.1583
0.70	0.1322
0.80	0.1116
0.90	0.0971
1.00	0.0814
Mean average precision	
non-interpolated	0.1747

Document Level Averages	
	Precision
At 5 docs	0.2040
At 10 docs	0.1580
At 15 docs	0.1320
At 20 docs	0.1180
At 30 docs	0.1020
At 100 docs	0.0592
At 200 docs	0.0359
At 500 docs	0.0162
At 1000 docs	0.0084
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1529



Recall-Precision Curve



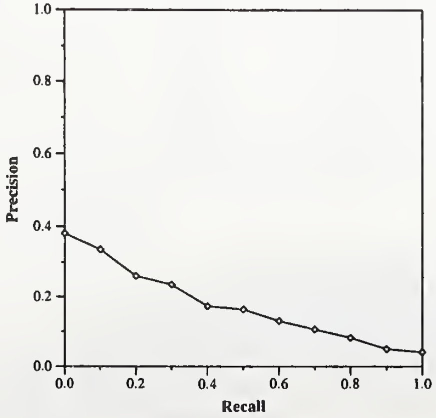
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	dayrutgers1
Run Description	Primary task, automatic
Number of Topics:	50

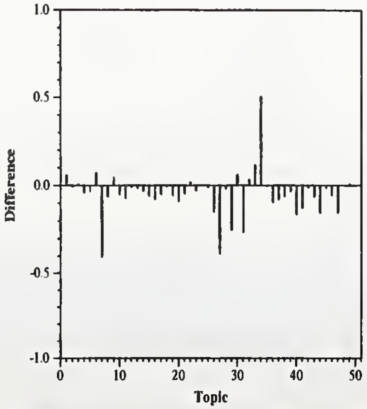
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	452

Recall Level Averages	
Recall	Precision
0.00	0.3807
0.10	0.3357
0.20	0.2605
0.30	0.2355
0.40	0.1734
0.50	0.1641
0.60	0.1308
0.70	0.1066
0.80	0.0826
0.90	0.0500
1.00	0.0408
Mean average precision	
non-interpolated	0.1652

Document Level Averages	
	Precision
At 5 docs	0.1600
At 10 docs	0.1340
At 15 docs	0.1187
At 20 docs	0.1200
At 30 docs	0.1060
At 100 docs	0.0548
At 200 docs	0.0344
At 500 docs	0.0164
At 1000 docs	0.0090
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1597



Recall-Precision Curve



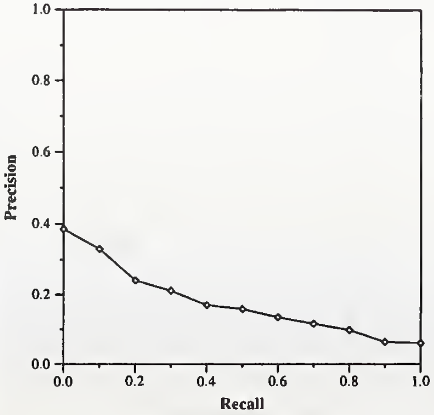
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	dayrutgers2
Run Description	Primary task, automatic
Number of Topics:	50

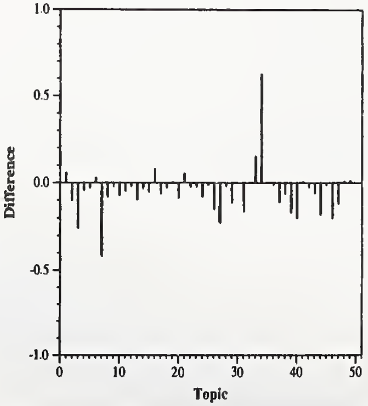
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	452

Recall Level Averages	
Recall	Precision
0.00	0.3859
0.10	0.3299
0.20	0.2411
0.30	0.2114
0.40	0.1695
0.50	0.1584
0.60	0.1344
0.70	0.1160
0.80	0.0978
0.90	0.0637
1.00	0.0602
Mean average precision	
non-interpolated	0.1636

Document Level Averages	
	Precision
At 5 docs	0.1720
At 10 docs	0.1320
At 15 docs	0.1173
At 20 docs	0.1030
At 30 docs	0.0813
At 100 docs	0.0478
At 200 docs	0.0310
At 500 docs	0.0163
At 1000 docs	0.0090
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1432



Recall-Precision Curve



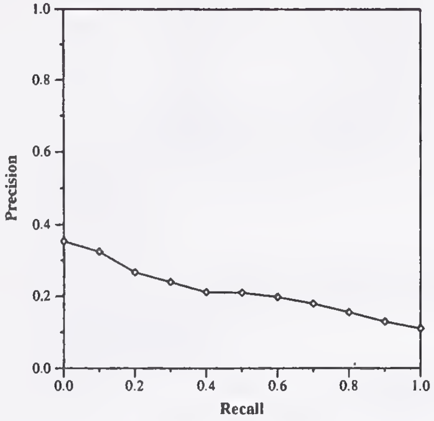
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	SCAI
Run Description	Primary task, automatic
Number of Topics:	50

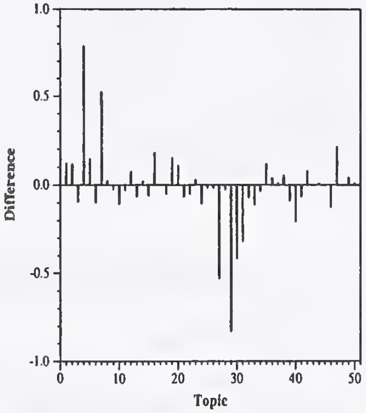
Total number of documents over all topics	
Retrieved:	12968
Relevant:	566
Rel-ret:	490

Recall Level Averages	
Recall	Precision
0.00	0.3527
0.10	0.3241
0.20	0.2664
0.30	0.2393
0.40	0.2115
0.50	0.2104
0.60	0.1982
0.70	0.1798
0.80	0.1556
0.90	0.1303
1.00	0.1111
Mean average precision	
non-interpolated	0.1960

Document Level Averages	
	Precision
At 5 docs	0.1480
At 10 docs	0.1420
At 15 docs	0.1387
At 20 docs	0.1300
At 30 docs	0.1113
At 100 docs	0.0600
At 200 docs	0.0404
At 500 docs	0.0188
At 1000 docs	0.0098
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1523



Recall-Precision Curve



Difference from Median in Average Precision per Topic

Genomics track, primary task results — State University of New York at Buffalo-CEDAR

Summary Statistics	
Run ID:	UBgenomRFB1
Run Description	Primary task, automatic
Number of Topics:	50

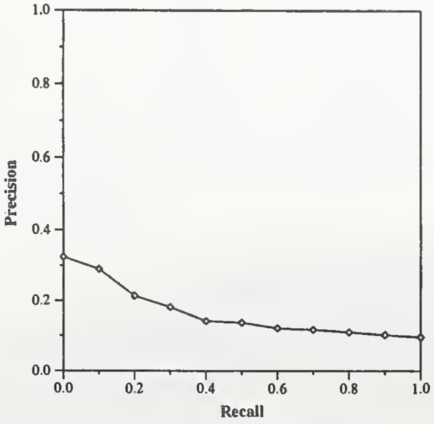
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	480

Recall Level Averages	
Recall	Precision
0.00	0.3239
0.10	0.2898
0.20	0.2138
0.30	0.1816
0.40	0.1417
0.50	0.1373
0.60	0.1212
0.70	0.1170
0.80	0.1099
0.90	0.1022
1.00	0.0957

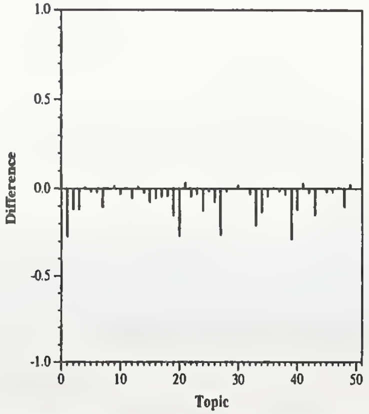
Mean average precision	
non-interpolated	0.1511

Document Level Averages	
	Precision
At 5 docs	0.1280
At 10 docs	0.1160
At 15 docs	0.1027
At 20 docs	0.0920
At 30 docs	0.0747
At 100 docs	0.0476
At 200 docs	0.0329
At 500 docs	0.0173
At 1000 docs	0.0096

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1232



Recall-Precision Curve



Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UBgenomRFB2
Run Description	Primary task, automatic
Number of Topics:	50

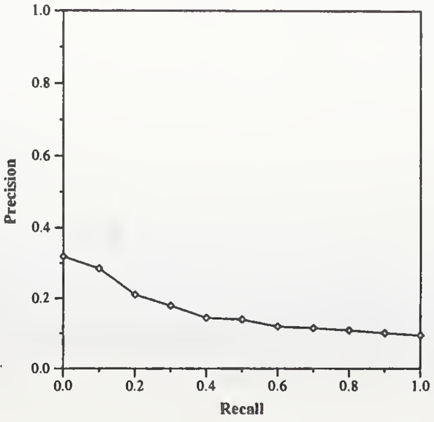
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	464

Recall Level Averages	
Recall	Precision
0.00	0.3193
0.10	0.2849
0.20	0.2111
0.30	0.1794
0.40	0.1446
0.50	0.1398
0.60	0.1194
0.70	0.1151
0.80	0.1088
0.90	0.1009
1.00	0.0945

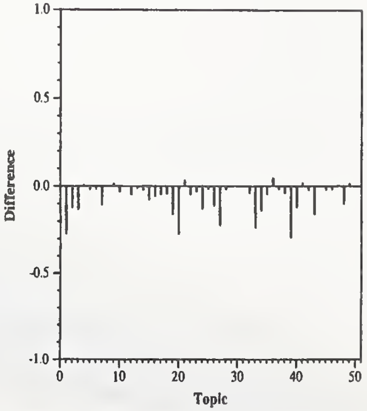
Mean average precision	
non-interpolated	0.1493

Document Level Averages	
	Precision
At 5 docs	0.1240
At 10 docs	0.1120
At 15 docs	0.1027
At 20 docs	0.0900
At 30 docs	0.0753
At 100 docs	0.0474
At 200 docs	0.0322
At 500 docs	0.0170
At 1000 docs	0.0093

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1141



Recall-Precision Curve



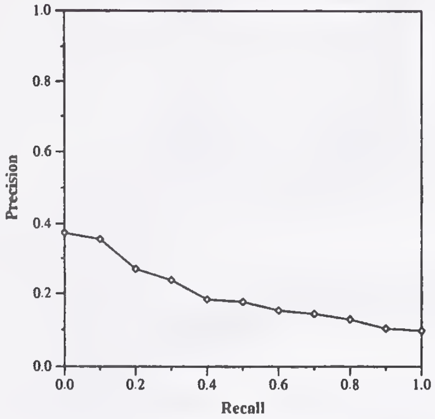
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UBgenomeBGNE
Run Description	Primary task, automatic
Number of Topics:	50

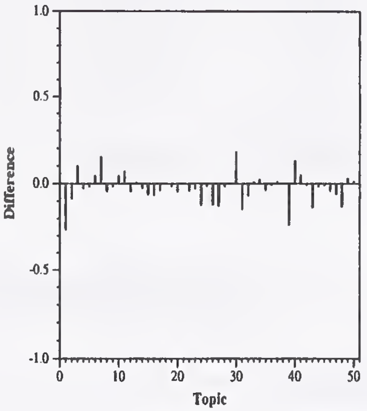
Total number of documents over all topics	
Retrieved:	49620
Relevant:	566
Rel-ret:	518

Recall Level Averages	
Recall	Precision
0.00	0.3737
0.10	0.3559
0.20	0.2698
0.30	0.2378
0.40	0.1834
0.50	0.1767
0.60	0.1531
0.70	0.1434
0.80	0.1281
0.90	0.1034
1.00	0.0974
Mean average precision	
non-interpolated	0.1867

Document Level Averages	
	Precision
At 5 docs	0.1560
At 10 docs	0.1440
At 15 docs	0.1213
At 20 docs	0.1070
At 30 docs	0.0913
At 100 docs	0.0524
At 200 docs	0.0371
At 500 docs	0.0190
At 1000 docs	0.0104
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1603



Recall-Precision Curve



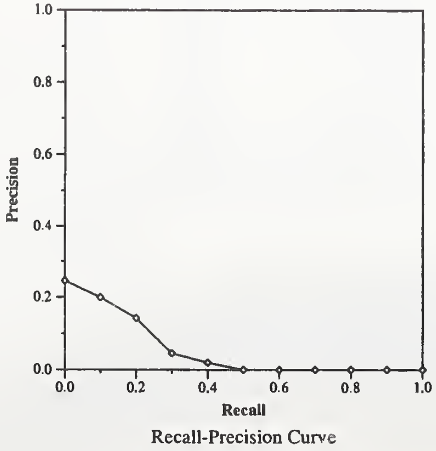
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	StreamSage3
Run Description	Primary task, automatic
Number of Topics:	50

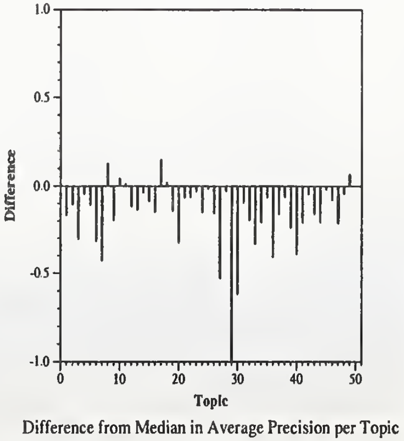
Total number of documents over all topics	
Retrieved:	470
Relevant:	566
Rel-ret:	40

Recall Level Averages	
Recall	Precision
0.00	0.2455
0.10	0.1997
0.20	0.1431
0.30	0.0455
0.40	0.0200
0.50	0.0000
0.60	0.0000
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000
Mean average precision	
non-interpolated	0.0508

Document Level Averages	
	Precision
At 5 docs	0.0960
At 10 docs	0.0700
At 15 docs	0.0533
At 20 docs	0.0400
At 30 docs	0.0267
At 100 docs	0.0080
At 200 docs	0.0040
At 500 docs	0.0016
At 1000 docs	0.0008
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0828



Recall-Precision Curve



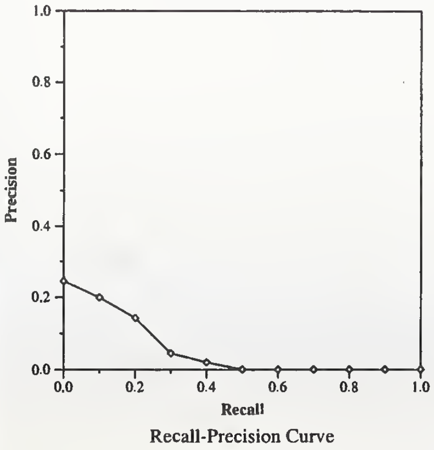
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	StreamSage4
Run Description	Primary task, automatic
Number of Topics:	50

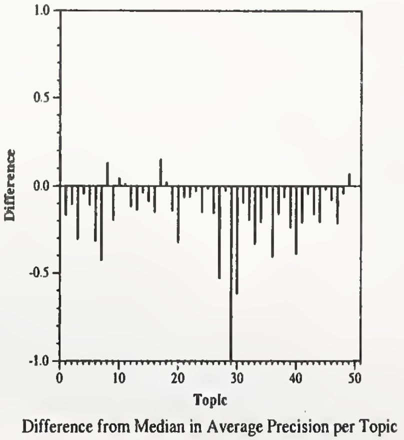
Total number of documents over all topics	
Retrieved:	335
Relevant:	566
Rel-ret:	40

Recall Level Averages	
Recall	Precision
0.00	0.2455
0.10	0.1997
0.20	0.1431
0.30	0.0455
0.40	0.0200
0.50	0.0000
0.60	0.0000
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000
Mean average precision	
non-interpolated	0.0508

Document Level Averages	
	Precision
At 5 docs	0.0960
At 10 docs	0.0700
At 15 docs	0.0533
At 20 docs	0.0400
At 30 docs	0.0267
At 100 docs	0.0080
At 200 docs	0.0040
At 500 docs	0.0016
At 1000 docs	0.0008
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0828



Recall-Precision Curve



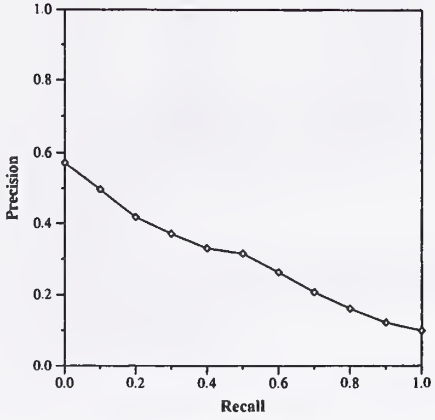
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	tgnBaseline
Run Description	Primary task, manual
Number of Topics:	50

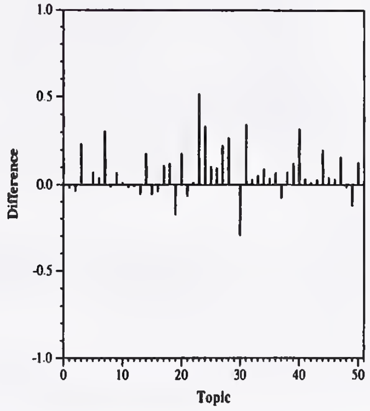
Total number of documents over all topics	
Retrieved:	7758
Relevant:	566
Rel-ret:	463

Recall Level Averages	
Recall	Precision
0.00	0.5721
0.10	0.4975
0.20	0.4179
0.30	0.3707
0.40	0.3298
0.50	0.3150
0.60	0.2622
0.70	0.2067
0.80	0.1614
0.90	0.1220
1.00	0.0992
Mean average precision	
non-interpolated	0.2837

Document Level Averages	
	Precision
At 5 docs	0.2640
At 10 docs	0.2180
At 15 docs	0.2000
At 20 docs	0.1760
At 30 docs	0.1473
At 100 docs	0.0752
At 200 docs	0.0426
At 500 docs	0.0184
At 1000 docs	0.0093
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.2850



Recall-Precision Curve



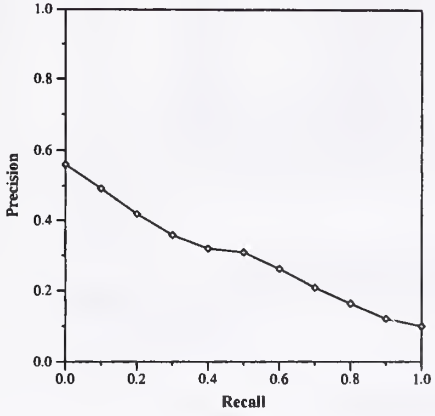
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	tgnVariant1
Run Description	Primary task, manual
Number of Topics:	50

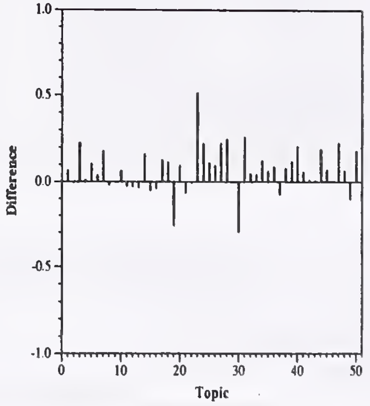
Total number of documents over all topics	
Retrieved:	7758
Relevant:	566
Rel-ret:	463

Recall Level Averages	
Recall	Precision
0.00	0.5606
0.10	0.4922
0.20	0.4192
0.30	0.3577
0.40	0.3193
0.50	0.3086
0.60	0.2616
0.70	0.2083
0.80	0.1637
0.90	0.1214
1.00	0.0998
Mean average precision	
non-interpolated	0.2791

Document Level Averages	
	Precision
At 5 docs	0.2720
At 10 docs	0.2220
At 15 docs	0.1973
At 20 docs	0.1780
At 30 docs	0.1480
At 100 docs	0.0758
At 200 docs	0.0422
At 500 docs	0.0184
At 1000 docs	0.0093
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.2852



Recall-Precision Curve



Difference from Median in Average Precision per Topic

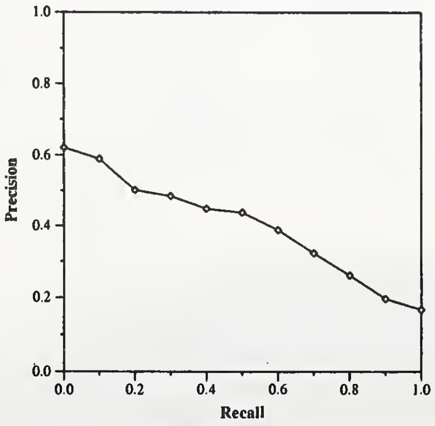
Genomics track, primary task results — University of California, Berkeley

Summary Statistics	
Run ID:	biotext0
Run Description	Primary task, automatic
Number of Topics:	50

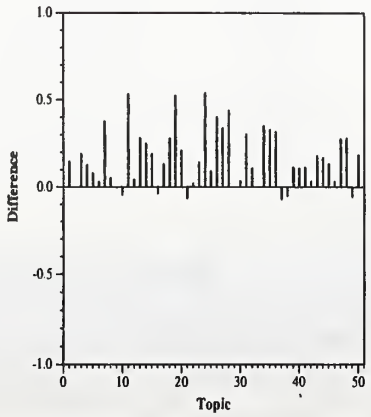
Total number of documents over all topics	
Retrieved:	13209
Relevant:	566
Rel-ret:	490

Recall Level Averages	
Recall	Precision
0.00	0.6206
0.10	0.5898
0.20	0.5012
0.30	0.4841
0.40	0.4489
0.50	0.4382
0.60	0.3881
0.70	0.3231
0.80	0.2617
0.90	0.1971
1.00	0.1682
Mean average precision	
non-interpolated	0.3753

Document Level Averages	
	Precision
At 5 docs	0.3760
At 10 docs	0.2920
At 15 docs	0.2480
At 20 docs	0.2150
At 30 docs	0.1700
At 100 docs	0.0784
At 200 docs	0.0442
At 500 docs	0.0188
At 1000 docs	0.0098
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.3701



Recall-Precision Curve



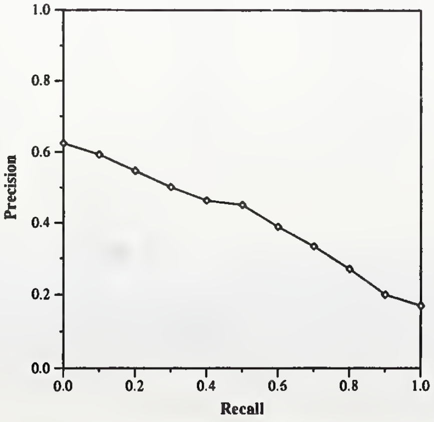
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	biotext1
Run Description	Primary task, automatic
Number of Topics:	50

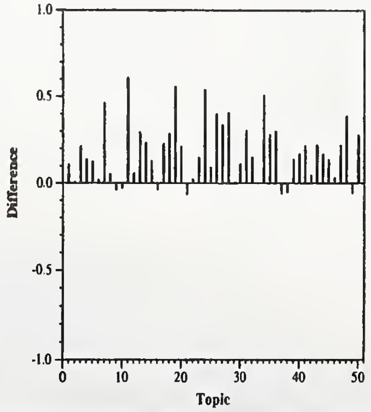
Total number of documents over all topics	
Retrieved:	13209
Relevant:	566
Rel-ret:	490

Recall Level Averages	
Recall	Precision
0.00	0.6246
0.10	0.5927
0.20	0.5475
0.30	0.5012
0.40	0.4630
0.50	0.4505
0.60	0.3890
0.70	0.3338
0.80	0.2710
0.90	0.2003
1.00	0.1704
Mean average precision	
non-interpolated	0.3912

Document Level Averages	
	Precision
At 5 docs	0.3960
At 10 docs	0.3060
At 15 docs	0.2507
At 20 docs	0.2230
At 30 docs	0.1760
At 100 docs	0.0812
At 200 docs	0.0447
At 500 docs	0.0191
At 1000 docs	0.0098
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.3770



Recall-Precision Curve



Difference from Median in Average Precision per Topic

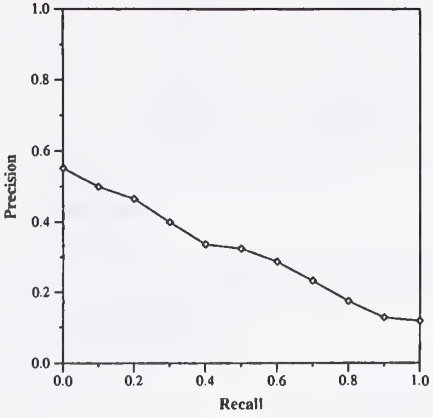
Genomics track, primary task results — U. of Edinburgh & Stanford U.

Summary Statistics	
Run ID:	edstanprec
Run Description	Primary task, automatic
Number of Topics:	50

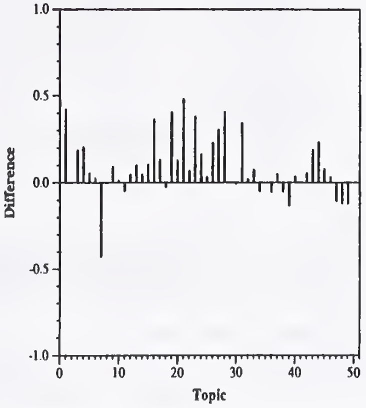
Total number of documents over all topics	
Retrieved:	10216
Relevant:	566
Rel-ret:	458

Recall Level Averages	
Recall	Precision
0.00	0.5520
0.10	0.4997
0.20	0.4655
0.30	0.3998
0.40	0.3371
0.50	0.3248
0.60	0.2874
0.70	0.2330
0.80	0.1745
0.90	0.1277
1.00	0.1173
Mean average precision	
non-interpolated	0.2984

Document Level Averages	
	Precision
At 5 docs	0.2800
At 10 docs	0.2600
At 15 docs	0.2200
At 20 docs	0.1870
At 30 docs	0.1420
At 100 docs	0.0692
At 200 docs	0.0408
At 500 docs	0.0181
At 1000 docs	0.0092
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.2757



Recall-Precision Curve



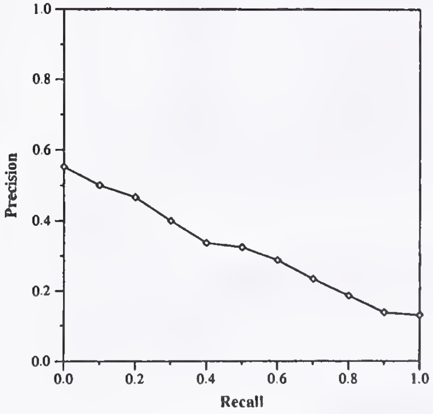
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	edstanrecall
Run Description	Primary task, automatic
Number of Topics:	50

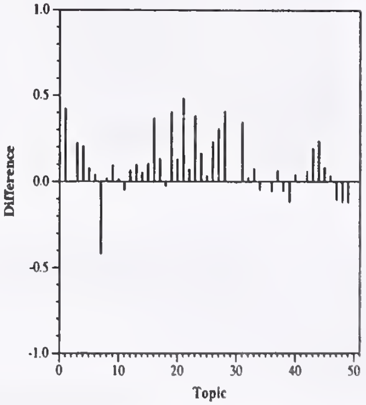
Total number of documents over all topics	
Retrieved:	25431
Relevant:	566
Rel-ret:	497

Recall Level Averages	
Recall	Precision
0.00	0.5525
0.10	0.5001
0.20	0.4660
0.30	0.3998
0.40	0.3371
0.50	0.3248
0.60	0.2877
0.70	0.2343
0.80	0.1858
0.90	0.1380
1.00	0.1298
Mean average precision	
non-interpolated	0.3015

Document Level Averages	
	Precision
At 5 docs	0.2800
At 10 docs	0.2600
At 15 docs	0.2200
At 20 docs	0.1870
At 30 docs	0.1420
At 100 docs	0.0700
At 200 docs	0.0419
At 500 docs	0.0191
At 1000 docs	0.0099
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.2757



Recall-Precision Curve



Difference from Median in Average Precision per Topic

Genomics track, primary task results — University of Illinois at Urbana-Champaign

Summary Statistics	
Run ID:	UIUC03Ga
Run Description	Primary task, automatic
Number of Topics:	50

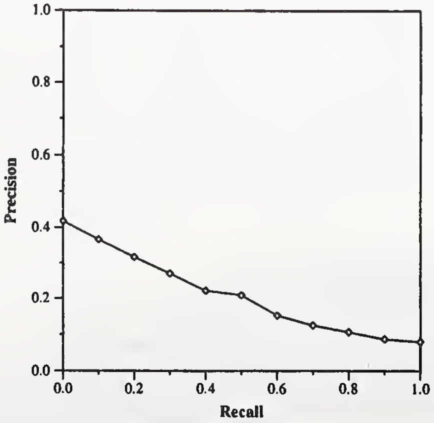
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	524

Recall Level Averages	
Recall	Precision
0.00	0.4173
0.10	0.3663
0.20	0.3164
0.30	0.2704
0.40	0.2221
0.50	0.2095
0.60	0.1523
0.70	0.1251
0.80	0.1068
0.90	0.0872
1.00	0.0800

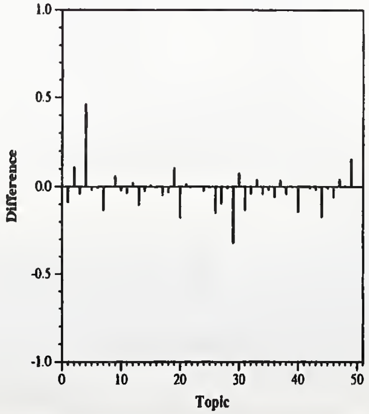
Mean average precision	
non-interpolated	0.1925

Document Level Averages	
	Precision
At 5 docs	0.1800
At 10 docs	0.1580
At 15 docs	0.1307
At 20 docs	0.1160
At 30 docs	0.0947
At 100 docs	0.0630
At 200 docs	0.0401
At 500 docs	0.0196
At 1000 docs	0.0105

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1839



Recall-Precision Curve



Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UIUC03Gb
Run Description	Primary task, automatic
Number of Topics:	50

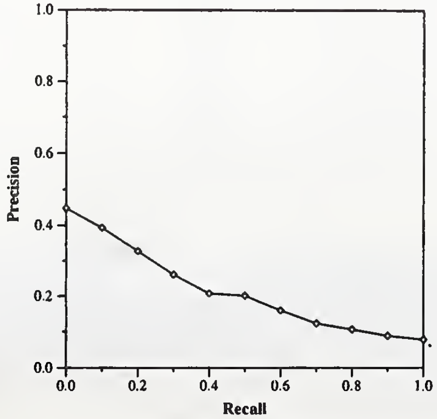
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	511

Recall Level Averages	
Recall	Precision
0.00	0.4484
0.10	0.3935
0.20	0.3279
0.30	0.2612
0.40	0.2079
0.50	0.2018
0.60	0.1597
0.70	0.1232
0.80	0.1067
0.90	0.0893
1.00	0.0791

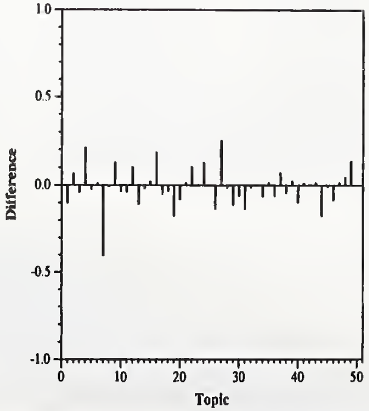
Mean average precision	
non-interpolated	0.2001

Document Level Averages	
	Precision
At 5 docs	0.1800
At 10 docs	0.1500
At 15 docs	0.1347
At 20 docs	0.1220
At 30 docs	0.1053
At 100 docs	0.0652
At 200 docs	0.0408
At 500 docs	0.0191
At 1000 docs	0.0102

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1730



Recall-Precision Curve



Difference from Median in Average Precision per Topic

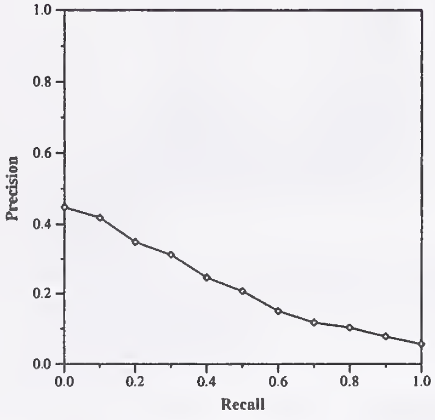
Genomics track, primary task results — University of Iowa

Summary Statistics	
Run ID:	UIowaGN1
Run Description	Primary task, automatic
Number of Topics:	50

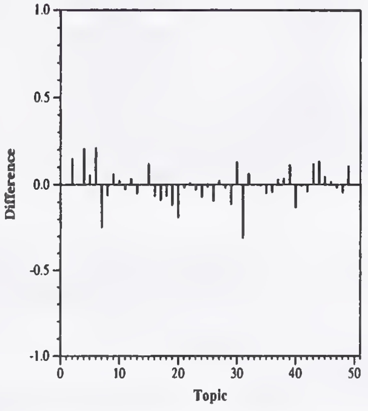
Total number of documents over all topics	
Retrieved:	12273
Relevant:	566
Rel-ret:	344

Recall Level Averages	
Recall	Precision
0.00	0.4486
0.10	0.4190
0.20	0.3495
0.30	0.3135
0.40	0.2473
0.50	0.2082
0.60	0.1504
0.70	0.1177
0.80	0.1034
0.90	0.0781
1.00	0.0562
Mean average precision	
non-interpolated	0.2064

Document Level Averages	
	Precision
At 5 docs	0.2320
At 10 docs	0.2020
At 15 docs	0.1853
At 20 docs	0.1700
At 30 docs	0.1393
At 100 docs	0.0624
At 200 docs	0.0338
At 500 docs	0.0138
At 1000 docs	0.0069
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.2281



Recall-Precision Curve



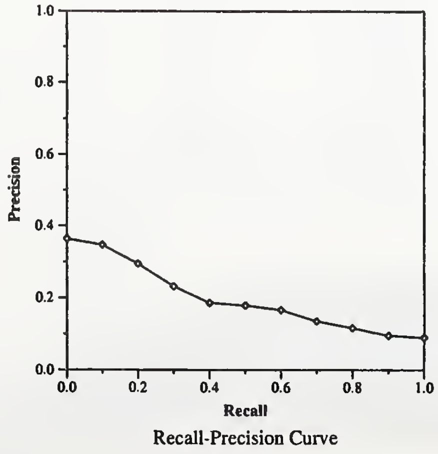
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UniNEg1
Run Description	Primary task, automatic
Number of Topics:	50

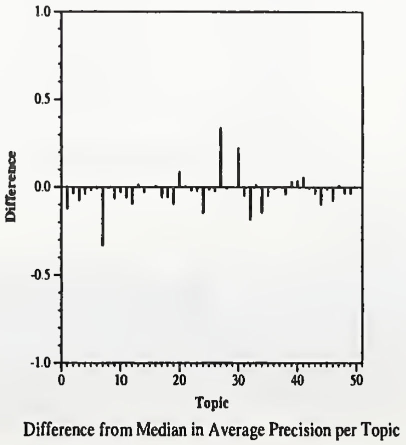
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	466

Recall Level Averages	
Recall	Precision
0.00	0.3637
0.10	0.3471
0.20	0.2948
0.30	0.2323
0.40	0.1863
0.50	0.1795
0.60	0.1669
0.70	0.1357
0.80	0.1170
0.90	0.0959
1.00	0.0899
Mean average precision	
non-interpolated	0.1852

Document Level Averages	
	Precision
At 5 docs	0.1760
At 10 docs	0.1280
At 15 docs	0.1173
At 20 docs	0.1060
At 30 docs	0.0860
At 100 docs	0.0568
At 200 docs	0.0344
At 500 docs	0.0171
At 1000 docs	0.0093
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1577



Recall-Precision Curve



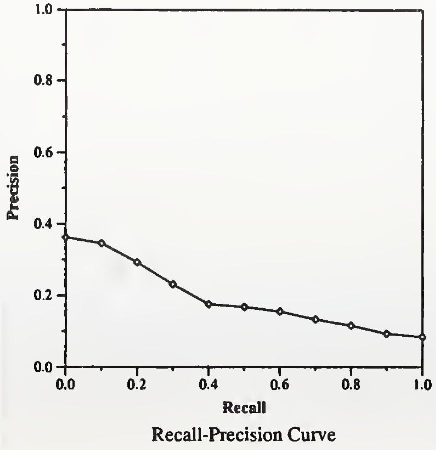
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UniNEg2
Run Description	Primary task, automatic
Number of Topics:	50

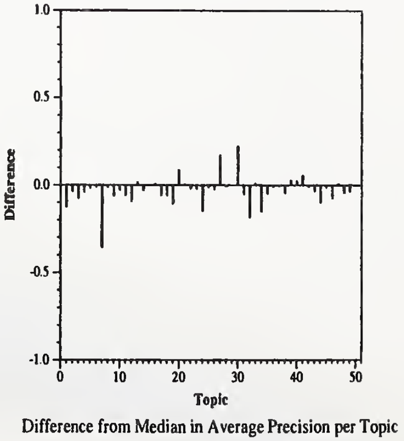
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	463

Recall Level Averages	
Recall	Precision
0.00	0.3620
0.10	0.3453
0.20	0.2928
0.30	0.2312
0.40	0.1759
0.50	0.1680
0.60	0.1554
0.70	0.1333
0.80	0.1158
0.90	0.0932
1.00	0.0840
Mean average precision	
non-interpolated	0.1802

Document Level Averages	
	Precision
At 5 docs	0.1720
At 10 docs	0.1300
At 15 docs	0.1187
At 20 docs	0.1050
At 30 docs	0.0873
At 100 docs	0.0562
At 200 docs	0.0341
At 500 docs	0.0166
At 1000 docs	0.0093
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1516



Recall-Precision Curve



Difference from Median in Average Precision per Topic

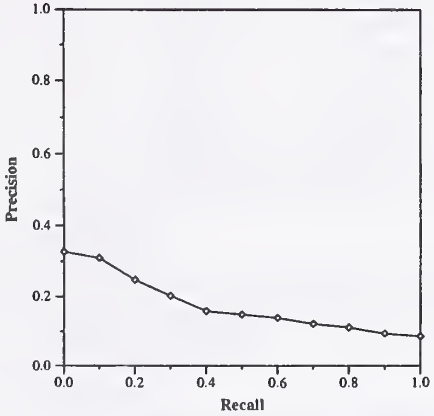
Genomics track, primary task results — Universite de Neuchatel

Summary Statistics	
Run ID:	UniNEg4
Run Description	Primary task, automatic
Number of Topics:	50

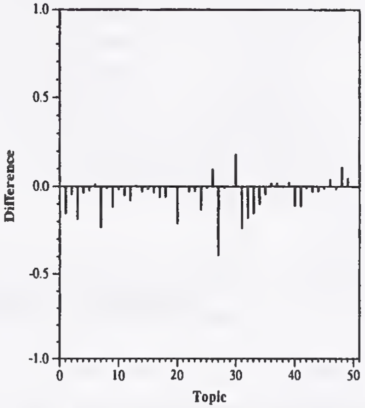
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	472

Recall Level Averages	
Recall	Precision
0.00	0.3266
0.10	0.3096
0.20	0.2480
0.30	0.2031
0.40	0.1592
0.50	0.1494
0.60	0.1396
0.70	0.1222
0.80	0.1120
0.90	0.0944
1.00	0.0862
Mean average precision	
non-interpolated	0.1623

Document Level Averages	
	Precision
At 5 docs	0.1680
At 10 docs	0.1300
At 15 docs	0.1093
At 20 docs	0.1050
At 30 docs	0.0873
At 100 docs	0.0572
At 200 docs	0.0360
At 500 docs	0.0177
At 1000 docs	0.0094
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1675



Recall-Precision Curve



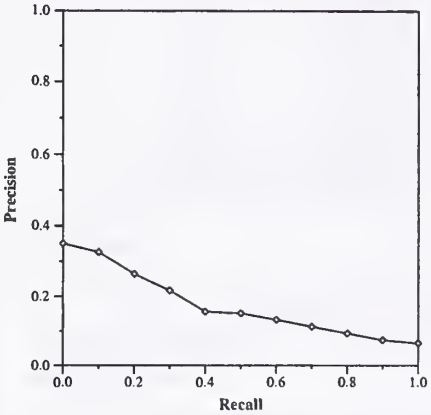
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UniNEg5
Run Description	Primary task, automatic
Number of Topics:	50

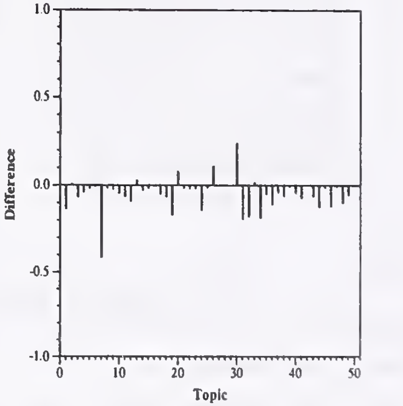
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	440

Recall Level Averages	
Recall	Precision
0.00	0.3506
0.10	0.3262
0.20	0.2648
0.30	0.2178
0.40	0.1566
0.50	0.1518
0.60	0.1331
0.70	0.1133
0.80	0.0937
0.90	0.0743
1.00	0.0649
Mean average precision	
non-interpolated	0.1635

Document Level Averages	
	Precision
At 5 docs	0.1480
At 10 docs	0.1280
At 15 docs	0.1133
At 20 docs	0.1000
At 30 docs	0.0827
At 100 docs	0.0522
At 200 docs	0.0338
At 500 docs	0.0160
At 1000 docs	0.0088
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1540



Recall-Precision Curve



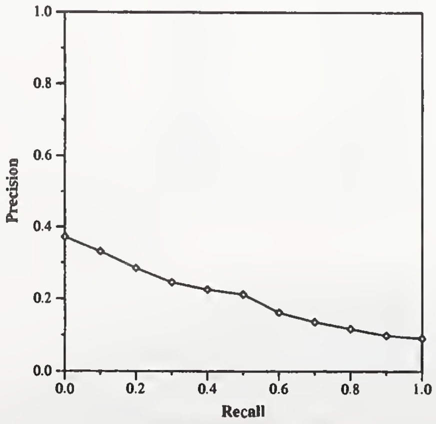
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	utaband
Run Description	Primary task, manual
Number of Topics:	50

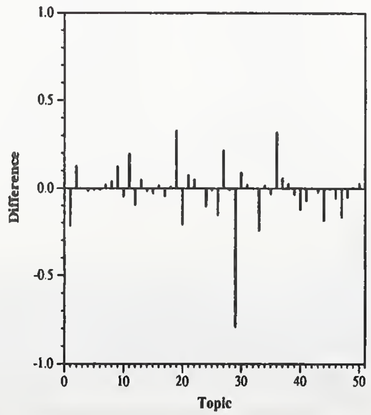
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	518

Recall Level Averages	
Recall	Precision
0.00	0.3723
0.10	0.3314
0.20	0.2848
0.30	0.2457
0.40	0.2258
0.50	0.2124
0.60	0.1627
0.70	0.1366
0.80	0.1177
0.90	0.0991
1.00	0.0912
Mean average precision	
non-interpolated	0.1927

Document Level Averages	
	Precision
At 5 docs	0.1800
At 10 docs	0.1540
At 15 docs	0.1400
At 20 docs	0.1310
At 30 docs	0.1140
At 100 docs	0.0626
At 200 docs	0.0405
At 500 docs	0.0197
At 1000 docs	0.0104
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1620



Recall-Precision Curve



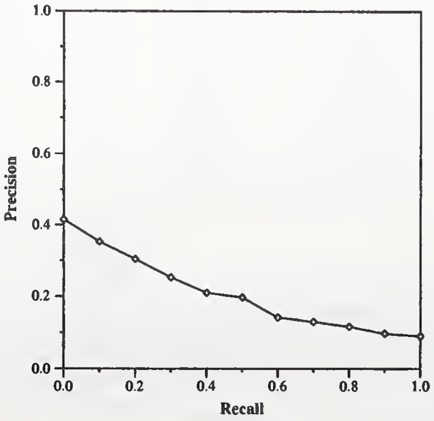
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	utafil
Run Description	Primary task, manual
Number of Topics:	50

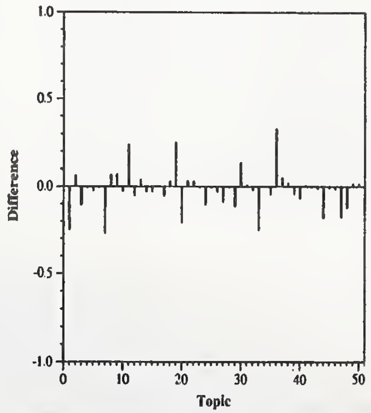
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	536

Recall Level Averages	
Recall	Precision
0.00	0.4153
0.10	0.3528
0.20	0.3041
0.30	0.2527
0.40	0.2099
0.50	0.1974
0.60	0.1419
0.70	0.1297
0.80	0.1162
0.90	0.0972
1.00	0.0901
Mean average precision	
non-interpolated	0.1931

Document Level Averages	
	Precision
At 5 docs	0.1800
At 10 docs	0.1480
At 15 docs	0.1307
At 20 docs	0.1200
At 30 docs	0.1040
At 100 docs	0.0630
At 200 docs	0.0397
At 500 docs	0.0198
At 1000 docs	0.0107
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1566



Recall-Precision Curve



Difference from Median in Average Precision per Topic

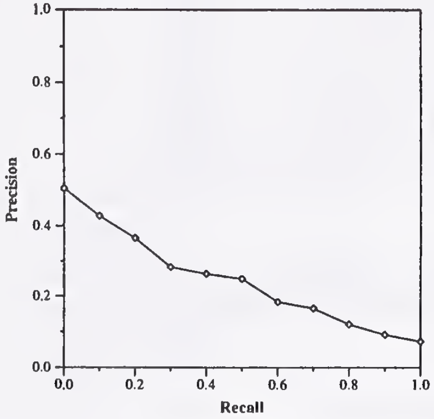
Genomics track, primary task results — University of Tokyo

Summary Statistics	
Run ID:	aoyama
Run Description	Primary task, automatic
Number of Topics:	50

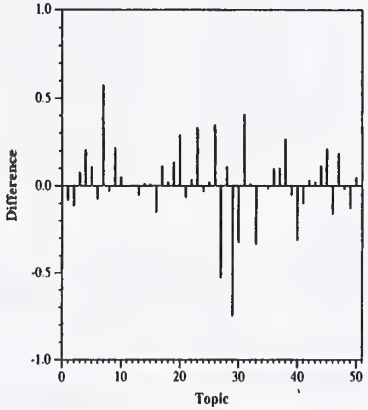
Total number of documents over all topics	
Retrieved:	4814
Relevant:	566
Rel-ret:	453

Recall Level Averages	
Recall	Precision
0.00	0.5051
0.10	0.4281
0.20	0.3649
0.30	0.2834
0.40	0.2644
0.50	0.2495
0.60	0.1840
0.70	0.1657
0.80	0.1209
0.90	0.0918
1.00	0.0725
Mean average precision	
non-interpolated	0.2276

Document Level Averages	
	Precision
At 5 docs	0.2480
At 10 docs	0.1920
At 15 docs	0.1693
At 20 docs	0.1460
At 30 docs	0.1227
At 100 docs	0.0666
At 200 docs	0.0395
At 500 docs	0.0177
At 1000 docs	0.0091
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.2174



Recall-Precision Curve



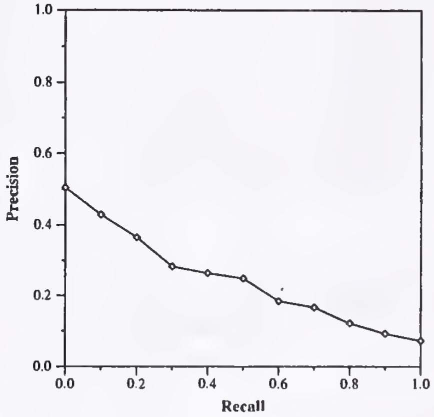
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	aoyama2
Run Description	Primary task, automatic
Number of Topics:	50

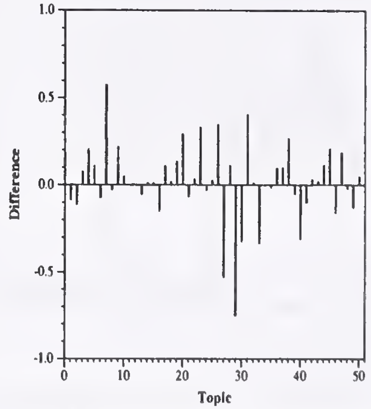
Total number of documents over all topics	
Retrieved:	4978
Relevant:	566
Rel-ret:	460

Recall Level Averages	
Recall	Precision
0.00	0.5044
0.10	0.4284
0.20	0.3648
0.30	0.2833
0.40	0.2643
0.50	0.2493
0.60	0.1839
0.70	0.1665
0.80	0.1217
0.90	0.0925
1.00	0.0725
Mean average precision	
non-interpolated	0.2277

Document Level Averages	
	Precision
At 5 docs	0.2480
At 10 docs	0.1900
At 15 docs	0.1693
At 20 docs	0.1460
At 30 docs	0.1227
At 100 docs	0.0666
At 200 docs	0.0397
At 500 docs	0.0177
At 1000 docs	0.0092
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.2174



Recall-Precision Curve



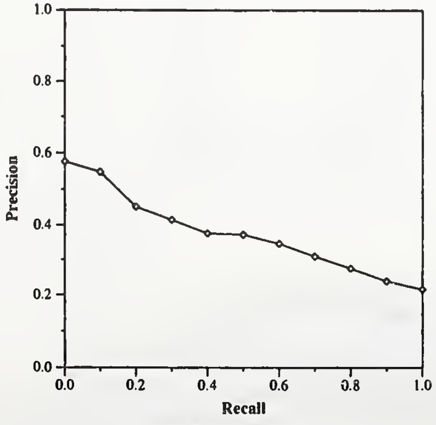
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	uwmtg03atrf
Run Description	Primary task, automatic
Number of Topics:	50

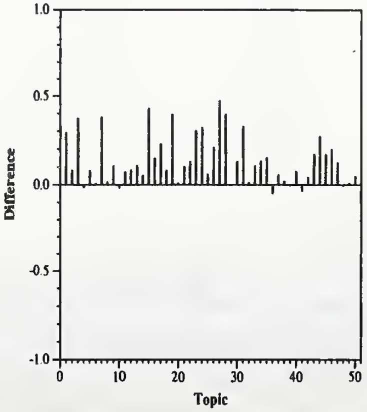
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	562

Recall Level Averages	
Recall	Precision
0.00	0.5766
0.10	0.5482
0.20	0.4514
0.30	0.4146
0.40	0.3758
0.50	0.3720
0.60	0.3459
0.70	0.3094
0.80	0.2750
0.90	0.2391
1.00	0.2159
Mean average precision	
non-interpolated	0.3479

Document Level Averages	
	Precision
At 5 docs	0.2760
At 10 docs	0.2480
At 15 docs	0.2227
At 20 docs	0.2000
At 30 docs	0.1640
At 100 docs	0.0822
At 200 docs	0.0485
At 500 docs	0.0219
At 1000 docs	0.0112
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.3013



Recall-Precision Curve



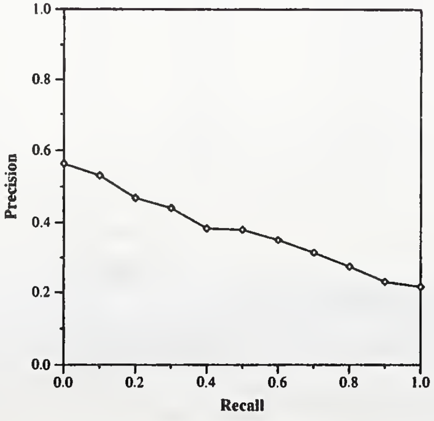
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	uwmtg03btrf
Run Description	Primary task, automatic
Number of Topics:	50

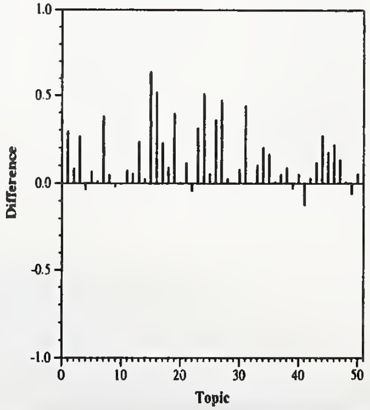
Total number of documents over all topics	
Retrieved:	50000
Relevant:	566
Rel-ret:	556

Recall Level Averages	
Recall	Precision
0.00	0.5635
0.10	0.5309
0.20	0.4689
0.30	0.4408
0.40	0.3834
0.50	0.3797
0.60	0.3503
0.70	0.3138
0.80	0.2744
0.90	0.2309
1.00	0.2168
Mean average precision	
non-interpolated	0.3534

Document Level Averages	
	Precision
At 5 docs	0.2840
At 10 docs	0.2280
At 15 docs	0.2027
At 20 docs	0.1840
At 30 docs	0.1533
At 100 docs	0.0782
At 200 docs	0.0484
At 500 docs	0.0217
At 1000 docs	0.0111
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.3113



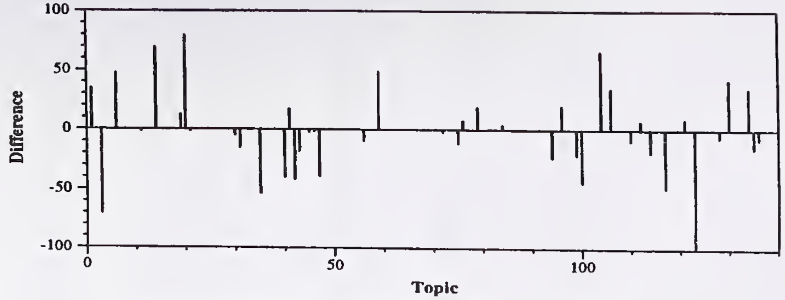
Recall-Precision Curve



Difference from Median in Average Precision per Topic

Run ID	CSUSMcand
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	49.3	51.3	35.0



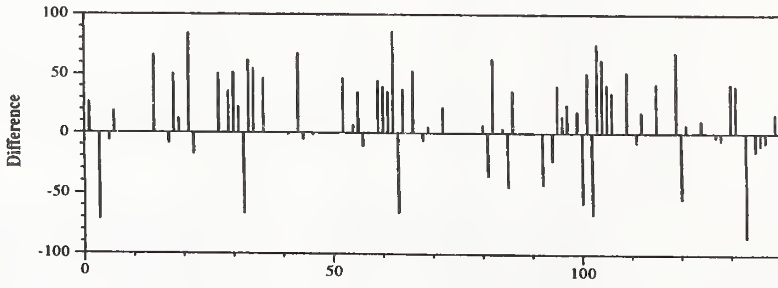
Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores

Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	82.8	82.3	80.0	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	54.5	58.8	53.3
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	40.8	39.0	20.5	107	61.5	73.3	71.4
3	10.0	20.7	7.4	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	50.0	52.2	19.1	40	23.5	0.0	0.0	75	9.1	28.6	0.0	110	26.7	26.1	0.0
6	91.9	89.7	81.5	41	30.8	34.5	14.8	76	42.1	29.6	24.0	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	61.1	53.9	41.7	77	24.2	13.8	7.4	112	6.1	28.6	0.0
8	90.5	89.7	81.5	43	19.4	9.5	0.0	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	13.3	40.0	25.0	114	22.2	20.0	0.0
10	100.0	100.0	100.0	45	30.8	29.6	8.0	80	19.1	14.3	0.0	115	34.0	36.8	22.2
11	0.0	8.7	0.0	46	22.2	7.1	0.0	81	62.5	71.4	50.0	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	17.6	45.5	30.0	82	21.6	14.8	0.0	117	37.2	38.7	20.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	90.9	90.3	82.8	49	44.4	50.0	44.4	84	26.7	30.8	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	20.5	19.4	13.8	86	58.5	48.0	0.0	121	74.2	68.3	61.5
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	9.5	0.0	0.0
19	59.3	52.6	35.3	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	98.3	100.0	100.0	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	25.8	9.5	0.0	56	38.7	45.5	10.0	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	28.6	23.1	8.3
23	79.0	86.7	78.6	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	34.3	41.7	0.0
24	45.7	40.0	8.7	59	94.1	91.9	85.7	94	27.6	19.6	8.2	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	72.0	73.7	70.6
26	91.9	92.9	84.6	61	51.3	50.0	26.7	96	97.3	100.0	100.0	131	39.0	33.3	7.1
27	37.8	41.4	22.2	62	25.8	9.5	0.0	97	52.9	69.0	37.0	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	16.0	21.1	0.0	134	87.8	93.8	93.3
30	21.3	16.7	5.9	65	61.1	51.9	32.0	100	22.2	23.1	8.3	135	85.1	85.0	84.2
31	19.5	7.1	0.0	66	41.7	35.9	16.2	101	40.7	46.5	19.5	136	31.2	34.8	9.5
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	89.2	88.9	84.6	139	33.3	40.0	11.1
35	20.0	21.4	7.7	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Run ID	emc4
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	57.8	59.6	46.8

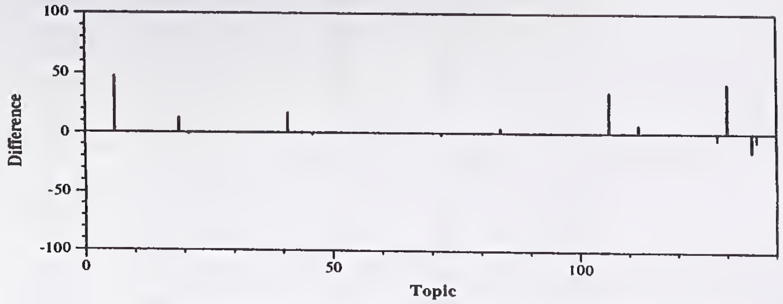


Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	75.0	73.7	70.6	36	81.2	80.0	77.8	71	78.8	88.9	88.0	106	54.5	58.8	53.3
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	55.8	62.9	48.5	107	61.5	73.3	71.4
3	9.8	20.0	7.1	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	93.5	97.8	97.7
5	35.9	45.7	24.2	40	43.8	40.0	26.1	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	46.8	60.6	19.4	41	11.8	16.0	0.0	76	30.8	21.4	7.7	111	61.2	54.0	51.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	33.3	38.9	29.4
8	90.5	89.7	81.5	43	96.8	95.2	94.7	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	37.0	36.4	20.0	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	20.7	21.1	0.0	115	84.2	79.1	73.2
11	8.0	10.5	0.0	46	22.2	7.1	0.0	81	24.2	35.7	7.7	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	70.6	84.6	66.7	82	81.5	78.0	76.9	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	88.9	87.5	80.0	49	44.4	50.0	44.4	84	26.7	30.8	0.0	119	96.5	100.0	100.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	32.6	40.0	6.1	120	31.6	41.7	27.3
16	100.0	100.0	100.0	51	20.5	19.4	13.8	86	81.6	83.9	75.9	121	71.9	66.7	60.0
17	40.0	28.6	16.7	52	77.5	76.5	75.0	87	48.6	58.1	27.6	122	26.7	28.6	10.5
18	89.8	91.9	91.4	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	59.3	52.6	35.3	54	16.0	22.2	0.0	89	75.0	66.7	60.0	124	82.0	80.0	71.4
20	24.6	20.5	5.4	55	87.7	89.4	88.9	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	86.5	95.2	94.7	56	37.5	43.5	9.5	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	26.7	35.7	15.4	57	80.0	81.5	80.0	92	42.5	42.4	6.5	127	22.2	19.4	6.9
23	79.0	86.7	78.6	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	34.3	41.7	0.0
24	45.7	40.0	8.7	59	88.9	87.2	81.1	94	20.6	19.6	8.2	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	64.7	69.6	38.1	95	83.7	90.3	89.7	130	72.0	73.7	70.6
26	91.9	92.9	84.6	61	81.1	85.7	76.9	96	90.0	94.7	94.4	131	76.4	73.7	72.2
27	89.8	91.9	91.4	62	86.5	95.2	94.7	97	77.8	93.3	71.4	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	21.1	28.6	7.7	98	71.4	75.0	72.7	133	15.4	14.6	0.0
29	92.3	87.0	81.8	64	94.7	95.5	95.2	99	56.2	60.9	47.6	134	57.1	58.8	31.2
30	76.9	73.2	71.8	65	61.1	51.9	32.0	100	6.7	9.5	0.0	135	85.1	85.0	84.2
31	43.6	44.4	11.8	66	88.5	88.9	82.3	101	94.7	97.7	97.6	136	41.4	31.6	0.0
32	5.0	0.0	0.0	67	41.4	58.3	36.4	102	5.0	0.0	0.0	137	7.7	9.5	0.0
33	92.3	94.1	93.8	68	23.3	25.0	13.3	103	97.8	100.0	100.0	138	66.7	75.9	59.3
34	76.2	66.7	64.0	69	58.8	51.4	42.4	104	85.3	85.7	81.5	139	55.0	56.2	40.0
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	90.9	93.3	92.9				

Run ID	IBMbtT2
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	50.5	52.6	34.8

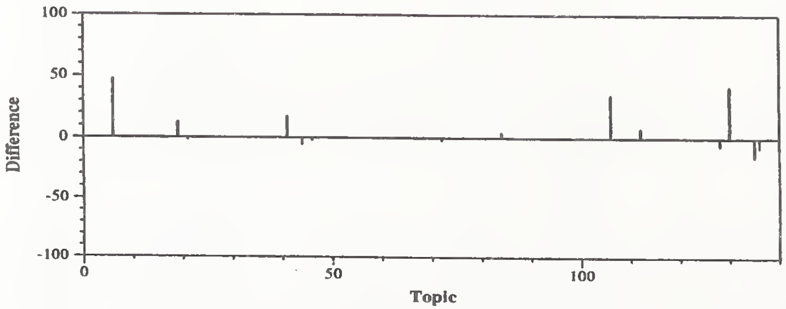


Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	54.5	58.8	53.3
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	40.8	39.0	20.5	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	50.0	52.2	19.1	40	43.8	40.0	26.1	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	91.9	89.7	81.5	41	30.8	34.5	14.8	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	6.1	28.6	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	19.1	14.3	0.0	115	34.0	36.8	22.2
11	8.0	10.5	0.0	46	22.2	7.1	0.0	81	62.5	71.4	50.0	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	70.6	84.6	66.7	82	21.6	14.8	0.0	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	18.2	21.1	0.0	49	44.4	50.0	44.4	84	26.7	30.8	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	20.5	19.4	13.8	86	58.5	48.0	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	59.3	52.6	35.3	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	25.8	9.5	0.0	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	28.6	23.1	8.3
23	79.0	86.7	78.6	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	34.3	41.7	0.0
24	45.7	40.0	8.7	59	37.2	42.4	12.9	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	72.0	73.7	70.6
26	91.9	92.9	84.6	61	51.3	50.0	26.7	96	48.3	80.0	57.1	131	39.0	33.3	7.1
27	37.8	41.4	22.2	62	25.8	9.5	0.0	97	52.9	69.0	37.0	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	57.1	58.8	31.2
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	85.1	85.0	84.2
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	40.7	46.5	19.5	136	31.2	34.8	9.5
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Run ID	IUB2003
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	50.4	52.6	34.8



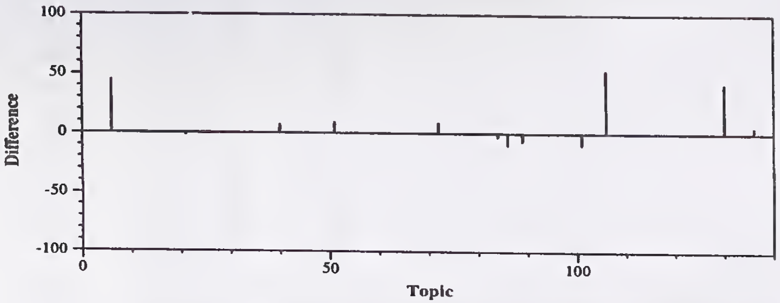
Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	54.5	58.8	53.3
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	40.8	39.0	20.5	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	50.0	52.2	19.1	40	43.8	40.0	26.1	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	91.9	89.7	81.5	41	30.8	34.5	14.8	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	6.1	28.6	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	37.0	36.4	20.0	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	19.1	14.3	0.0	115	34.0	36.8	22.2
11	8.0	10.5	0.0	46	22.2	7.1	0.0	81	62.5	71.4	50.0	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	70.6	84.6	66.7	82	21.6	14.8	0.0	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	18.2	21.1	0.0	49	44.4	50.0	44.4	84	26.7	30.8	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	20.5	19.4	13.8	86	58.5	48.0	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	59.3	52.6	35.3	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	25.8	9.5	0.0	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	28.6	23.1	8.3
23	79.0	86.7	78.6	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	34.3	41.7	0.0
24	45.7	40.0	8.7	59	37.2	42.4	12.9	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	72.0	73.7	70.6
26	91.9	92.9	84.6	61	51.3	50.0	26.7	96	48.3	80.0	57.1	131	39.0	33.3	7.1
27	37.8	41.4	22.2	62	25.8	9.5	0.0	97	52.9	69.0	37.0	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	57.1	58.8	31.2
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	85.1	85.0	84.2
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	40.7	46.5	19.5	136	31.2	34.8	9.5
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Genomics track, secondary task results — National Library of Medicine and U. Maryland

Run ID	NLMUMDLIN
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	50.4	52.6	35.0



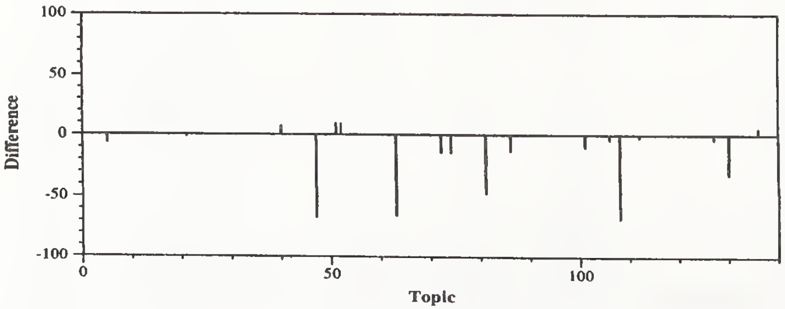
Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores

Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	66.7	76.9	72.7
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	47.6	50.0	26.7	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	50.0	52.2	19.1	40	48.0	47.1	40.0	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	83.9	87.0	85.7	41	25.0	17.4	0.0	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	0.0	21.1	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	19.1	14.3	0.0	115	34.0	36.8	22.2
11	8.0	10.5	0.0	46	26.7	9.1	0.0	81	62.5	71.4	50.0	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	70.6	84.6	66.7	82	21.6	14.8	0.0	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	18.2	21.1	0.0	49	44.4	50.0	44.4	84	20.0	23.5	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	32.0	28.6	21.1	86	40.0	38.1	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	52.2	40.0	15.4	54	11.8	15.4	0.0	89	61.5	60.0	50.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	63.2	87.5	57.1
21	25.8	9.5	0.0	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	28.6	23.1	8.3
23	79.0	86.7	78.6	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	38.7	47.6	0.0
24	47.1	40.0	8.7	59	37.2	42.4	12.9	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	20.7	30.0	0.0	95	50.0	50.0	30.8	130	72.0	73.7	70.6
26	91.9	92.9	84.6	61	51.3	50.0	26.7	96	48.3	80.0	57.1	131	39.0	33.3	7.1
27	37.8	41.4	22.2	62	25.8	9.5	0.0	97	52.9	69.0	37.0	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	57.1	58.8	31.2
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	100.0	100.0	100.0
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	37.5	36.8	22.2	136	33.3	47.1	13.3
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Run ID	nwe
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	47.6	49.4	31.6

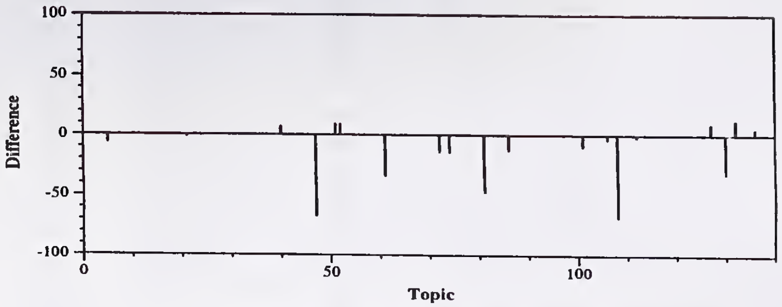


Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	14.3	20.0	0.0
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	25.4	27.4	16.3	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	13.6	12.1	0.0
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	90.9	85.7	78.8	109	38.1	45.7	18.2
5	35.9	45.7	24.2	40	48.0	47.1	40.0	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	46.1	42.1	23.5	41	25.0	17.4	0.0	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	8.7	19.1	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	19.1	14.3	0.0	115	34.0	36.8	22.2
11	8.0	10.5	0.0	46	26.7	9.1	0.0	81	26.3	24.4	15.4	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	13.3	17.4	9.5	82	21.6	14.8	0.0	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	18.2	21.1	0.0	49	44.4	50.0	44.4	84	21.1	26.7	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	32.0	28.6	21.1	86	46.7	35.3	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	50.0	38.7	13.8	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	52.2	40.0	15.4	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	25.8	9.5	0.0	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	24.4	19.4	6.9
23	81.1	86.7	78.6	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	38.7	47.6	0.0
24	45.7	40.0	8.7	59	37.2	42.4	12.9	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	0.0	0.0	0.0
26	91.9	92.9	84.6	61	51.3	50.0	26.7	96	48.3	80.0	57.1	131	39.0	33.3	7.1
27	37.8	41.4	22.2	62	25.8	9.5	0.0	97	52.9	69.0	37.0	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	21.1	28.6	7.7	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	57.1	58.8	31.2
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	100.0	100.0	100.0
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	37.5	36.8	22.2	136	33.3	47.1	13.3
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Run ID	we
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	48.1	49.8	32.3



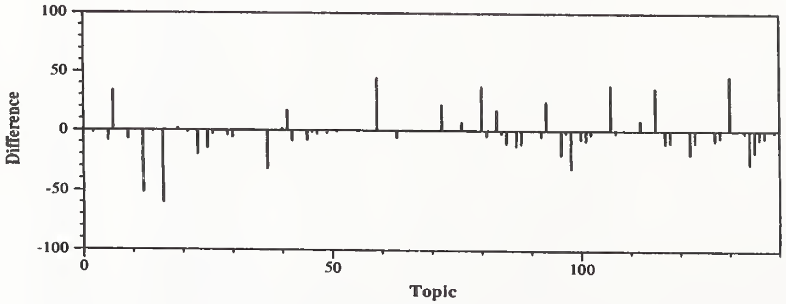
Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores

Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	14.3	20.0	0.0
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	25.4	27.4	16.3	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	13.6	12.1	0.0
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	90.9	85.7	78.8	109	38.1	45.7	18.2
5	35.9	45.7	24.2	40	48.0	47.1	40.0	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	46.1	42.1	23.5	41	25.0	17.4	0.0	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	8.7	19.1	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	19.1	14.3	0.0	115	34.0	36.8	22.2
11	8.0	10.5	0.0	46	26.7	9.1	0.0	81	26.3	24.4	15.4	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	13.3	17.4	9.5	82	21.6	14.8	0.0	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	18.2	21.1	0.0	49	44.4	50.0	44.4	84	21.1	26.7	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	32.0	28.6	21.1	86	46.7	35.3	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	50.0	38.7	13.8	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	52.2	40.0	15.4	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	25.8	9.5	0.0	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	31.1	32.3	20.7
23	81.1	86.7	78.6	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	38.7	47.6	0.0
24	45.7	40.0	8.7	59	37.2	42.4	12.9	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	0.0	0.0	0.0
26	91.9	92.9	84.6	61	28.6	16.0	8.3	96	48.3	80.0	57.1	131	39.0	33.3	7.1
27	37.8	41.4	22.2	62	25.8	9.5	0.0	97	52.9	69.0	37.0	132	44.7	38.8	21.5
28	31.6	66.7	30.8	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	57.1	58.8	31.2
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	100.0	100.0	100.0
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	37.5	36.8	22.2	136	33.3	47.1	13.3
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Run ID	balscsec1
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	48.9	50.5	32.4



Per-topic difference from median for modified unigram Dice

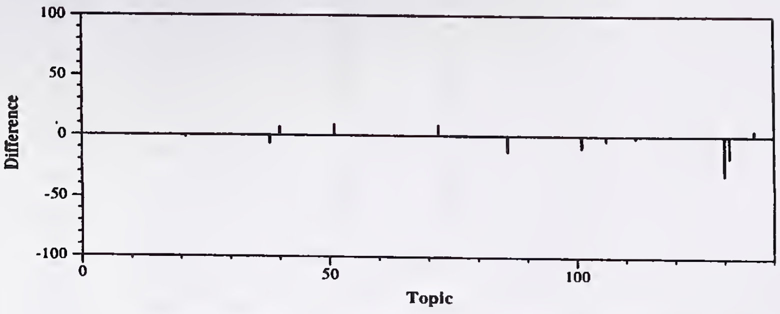
Per-topic Dice Scores

Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	54.5	62.5	57.1
2	46.1	41.0	5.4	37	12.5	4.9	0.0	72	55.8	62.9	48.5	107	60.0	71.0	69.0
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	42.9	43.5	9.5	40	43.8	41.7	27.3	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	81.1	75.9	59.3	41	31.6	34.5	14.8	76	35.9	28.6	15.4	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	90.3	87.0	76.2	77	24.2	13.8	7.4	112	6.2	29.6	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	94.4	92.9	92.3	44	46.1	41.7	18.2	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	27.6	24.0	0.0	80	62.9	51.9	48.0	115	79.3	72.7	66.7
11	8.0	10.5	0.0	46	22.9	7.4	0.0	81	58.8	66.7	46.1	116	35.7	40.0	26.1
12	17.1	22.2	8.0	47	68.6	81.5	64.0	82	21.6	14.8	0.0	117	77.5	75.7	62.9
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	52.0	58.5	35.9	118	12.8	13.9	4.9
14	17.8	20.5	0.0	49	42.9	47.6	42.1	84	20.7	24.0	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	52.2	73.7	35.3	120	83.9	95.2	94.7
16	30.8	40.0	7.1	51	20.0	18.2	12.9	86	55.0	48.0	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	37.8	45.2	6.9	122	6.2	9.1	0.0
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	92.9	88.9	75.0	123	93.3	90.0	77.8
19	53.9	42.1	23.5	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	25.8	9.5	0.0	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	80.0	80.0	68.4	127	22.2	14.8	0.0
23	64.9	66.7	42.9	58	58.8	76.9	72.7	93	41.0	60.0	28.6	128	29.4	41.7	0.0
24	45.7	40.0	8.7	59	88.9	87.2	81.1	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	19.5	6.5	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	75.0	77.8	75.0
26	89.5	89.7	74.1	61	51.3	50.0	26.7	96	41.4	60.0	14.3	131	39.0	33.3	7.1
27	37.8	41.4	22.2	62	25.8	9.5	0.0	97	51.4	66.7	28.6	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	87.8	88.2	87.5	98	44.4	43.5	19.1	133	98.2	97.8	93.0
29	41.7	46.8	22.2	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	39.0	31.2	6.7
30	33.3	15.8	0.0	65	61.1	51.9	32.0	100	50.0	59.3	32.0	135	80.8	82.0	81.1
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	34.0	38.1	15.0	136	32.3	34.8	9.5
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	63,2	58.8	137	15.4	12.1	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	32.0	38.1	10.5
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Genomics track, secondary task results — State University of New York at Buffalo-CEDAR

Run ID	UBGenT2BL1
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	49.3	51.2	33.6



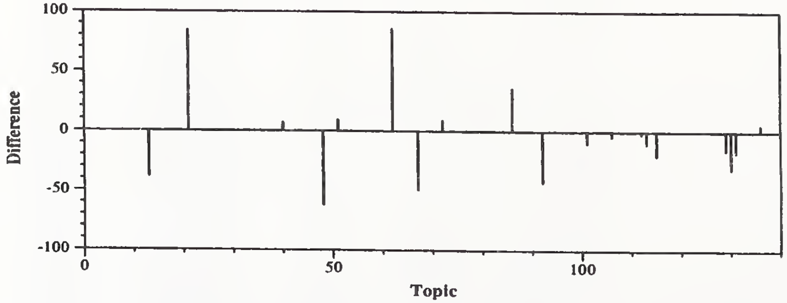
Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	14.3	20.0	0.0
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	47.6	50.0	26.7	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	0.0	0.0	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	50.0	52.2	19.1	40	48.0	47.1	40.0	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	46.1	42.1	23.5	41	25.0	17.4	0.0	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	8.7	19.1	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	19.1	14.3	0.0	115	34.0	36.8	22.2
11	8.0	10.5	0.0	46	26.7	9.1	0.0	81	62.5	71.4	50.0	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	70.6	84.6	66.7	82	21.6	14.8	0.0	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	18.2	21.1	0.0	49	44.4	50.0	44.4	84	21.1	26.7	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	32.0	28.6	21.1	86	46.7	35.3	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	52.2	40.0	15.4	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	25.8	9.5	0.0	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	28.6	23.1	8.3
23	79.0	86.7	78.6	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	38.7	47.6	0.0
24	45.7	40.0	8.7	59	37.2	42.4	12.9	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	0.0	0.0	0.0
26	91.9	92.9	84.6	61	51.3	50.0	26.7	96	48.3	80.0	57.1	131	30.8	16.1	3.3
27	37.8	41.4	22.2	62	25.8	9.5	0.0	97	52.9	69.0	37.0	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	57.1	58.8	31.2
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	100.0	100.0	100.0
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	37.5	36.8	22.2	136	33.3	47.1	13.3
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Genomics track, secondary task results — State University of New York at Buffalo-CEDAR

Run ID	UBGenT2R1
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	49.0	51.2	33.9

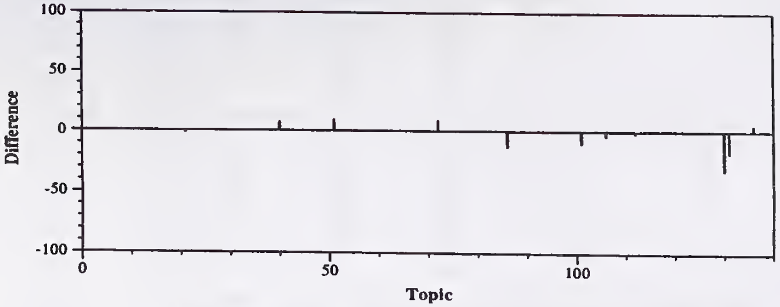


Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	14.3	20.0	0.0
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	47.6	50.0	26.7	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	50.0	52.2	19.1	40	48.0	47.1	40.0	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	46.1	42.1	23.5	41	25.0	17.4	0.0	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	8.7	19.1	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	69.0	55.6	25.0	113	32.3	27.3	10.0
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	19.1	14.3	0.0	115	26.1	16.7	0.0
11	8.0	10.5	0.0	46	26.7	9.1	0.0	81	62.5	71.4	50.0	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	70.6	84.6	66.7	82	21.6	14.8	0.0	117	85.7	86.5	85.7
13	17.4	16.2	0.0	48	26.3	22.2	16.0	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	18.2	21.1	0.0	49	44.4	50.0	44.4	84	21.1	26.7	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	32.0	28.6	21.1	86	81.6	83.9	75.9	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	52.2	40.0	15.4	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	86.5	95.2	94.7	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	42.5	42.4	6.5	127	28.6	23.1	8.3
23	79.0	86.7	78.6	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	38.7	47.6	0.0
24	45.7	40.0	8.7	59	37.2	42.4	12.9	94	40.7	42.5	13.3	129	13.6	12.9	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	0.0	0.0	0.0
26	91.9	92.9	84.6	61	51.3	50.0	26.7	96	48.3	80.0	57.1	131	30.8	16.1	3.3
27	37.8	41.4	22.2	62	86.5	95.2	94.7	97	52.9	69.0	37.0	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	57.1	58.8	31.2
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	100.0	100.0	100.0
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	37.5	36.8	22.2	136	33.3	47.1	13.3
32	28.6	66.7	60.0	67	6.9	9.1	0.0	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Run ID	UBGenT2R2
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	49.4	51.3	33.6

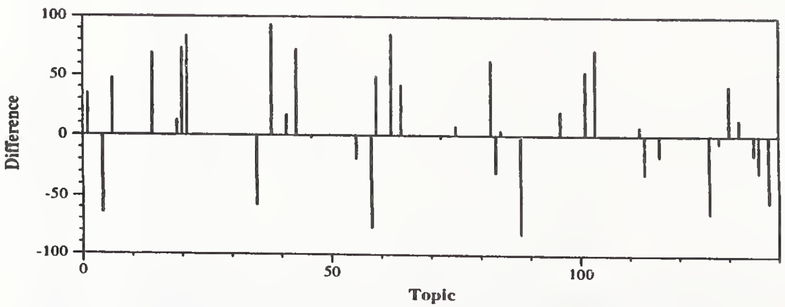


Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	14.3	20.0	0.0
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	47.6	50.0	26.7	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	50.0	52.2	19.1	40	48.0	47.1	40.0	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	46.1	42.1	23.5	41	25.0	17.4	0.0	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	8.7	19.1	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	19.1	14.3	0.0	115	34.0	36.8	22.2
11	8.0	10.5	0.0	46	26.7	9.1	0.0	81	62.5	71.4	50.0	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	70.6	84.6	66.7	82	21.6	14.8	0.0	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	18.2	21.1	0.0	49	44.4	50.0	44.4	84	21.1	26.7	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	32.0	28.6	21.1	86	46.7	35.3	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	52.2	40.0	15.4	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	25.8	9.5	0.0	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	28.6	23.1	8.3
23	79.0	86.7	78.6	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	38.7	47.6	0.0
24	45.7	40.0	8.7	59	37.2	42.4	12.9	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	0.0	0.0	0.0
26	91.9	92.9	84.6	61	51.3	50.0	26.7	96	48.3	80.0	57.1	131	30.8	16.1	3.3
27	37.8	41.4	22.2	62	25.8	9.5	0.0	97	52.9	69.0	37.0	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	57.1	58.8	31.2
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	100.0	100.0	100.0
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	37.5	36.8	22.2	136	33.3	47.1	13.3
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Run ID	biotextTask2
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	53.0	54.6	38.6



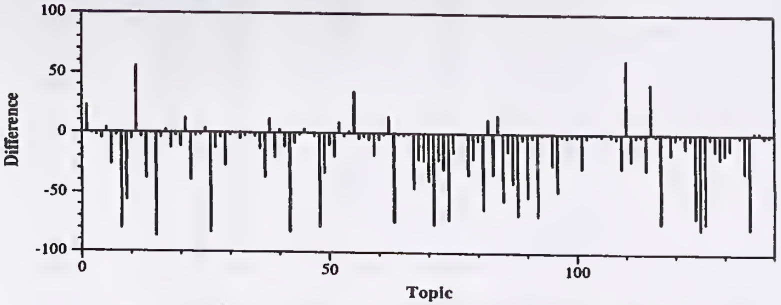
Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores

Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	85.7	82.3	80.0	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	9.5	23.5	0.0
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	40.8	39.0	20.5	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	100.0	100.0	100.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	12.5	15.4	0.0	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	50.0	52.2	19.1	40	43.8	40.0	26.1	75	53.9	47.6	10.5	110	41.4	34.8	9.5
6	91.9	89.7	81.5	41	30.8	34.5	14.8	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	6.1	28.6	0.0
8	90.5	89.7	81.5	43	100.0	100.0	100.0	78	69.0	55.6	25.0	113	11.8	6.9	0.0
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	19.1	14.3	0.0	115	34.0	36.8	22.2
11	8.0	10.5	0.0	46	22.2	7.1	0.0	81	62.5	71.4	50.0	116	13.3	23.8	5.0
12	58.3	73.7	47.1	47	70.6	84.6	66.7	82	81.5	78.0	76.9	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	8.9	11.4	0.0	118	35.9	24.2	6.5
14	90.9	90.3	82.8	49	44.4	50.0	44.4	84	26.7	30.8	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	20.5	19.4	13.8	86	58.5	48.0	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	28.6	17.9	7.4	123	100.0	100.0	100.0
19	59.3	52.6	35.3	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	94.5	94.1	93.8	55	36.7	35.9	16.2	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	93.3	94.7	94.1	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	30.0	36.4	0.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	28.6	23.1	8.3
23	79.0	86.7	78.6	58	0.0	0.0	0.0	93	31.6	35.7	7.7	128	34.3	41.7	0.0
24	45.7	40.0	8.7	59	94.1	91.9	85.7	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	72.0	73.7	70.6
26	91.9	92.9	84.6	61	51.3	50.0	26.7	96	100.0	100.0	100.0	131	39.0	33.3	7.1
27	37.8	41.4	22.2	62	93.3	94.7	94.1	97	52.9	69.0	37.0	132	46.6	40.0	22.2
28	31.6	66.7	30.8	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	100.0	100.0	100.0	99	40.0	42.1	35.3	134	57.1	58.8	31.2
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	85.1	85.0	84.2
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	100.0	100.0	100.0	136	26.3	13.3	0.0
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	95.2	97.1	97.0	138	21.3	22.2	11.8
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	13.3	17.4	0.0	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Run ID	EDISTFruns2
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	35.8	35.9	20.1



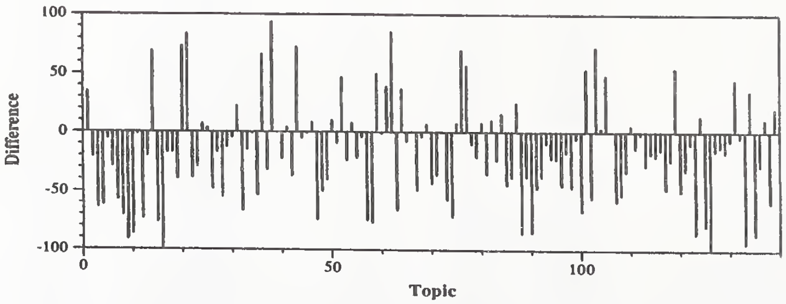
Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores

Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	72.7	70.0	66.7	36	33.3	20.0	0.0	71	14.6	12.5	0.0	106	17.1	24.0	8.7
2	46.1	42.1	16.7	37	11.4	0.0	0.0	72	21.4	19.1	0.0	107	60.0	71.0	69.0
3	86.4	88.9	82.3	38	21.1	18.6	4.9	73	44.4	37.0	16.0	108	69.0	76.2	73.7
4	73.7	75.0	42.9	39	5.1	12.9	0.0	74	24.4	27.8	5.9	109	22.2	17.6	6.2
5	46.1	56.0	43.5	40	40.0	42.9	0.0	75	18.8	24.0	0.0	110	97.3	96.8	96.5
6	11.1	15.4	0.0	41	16.3	5.9	0.0	76	20.5	21.4	7.7	111	38.1	40.0	28.6
7	90.5	88.2	87.5	42	19.1	12.9	0.0	77	23.5	13.3	7.1	112	8.3	18.2	0.0
8	17.5	9.1	0.0	43	25.8	19.1	0.0	78	23.5	21.4	7.7	113	34.8	35.3	13.3
9	29.3	43.8	20.0	44	44.4	40.0	17.4	79	7.4	0.0	0.0	114	14.3	9.1	0.0
10	95.2	94.7	94.1	45	17.4	35.3	13.3	80	20.0	8.3	0.0	115	84.2	79.1	73.2
11	50.0	66.7	40.0	46	25.8	8.7	0.0	81	0.0	8.3	0.0	116	34.5	38.5	25.0
12	56.0	70.0	44.4	47	68.6	81.5	64.0	82	35.9	26.7	0.0	117	27.3	12.1	0.0
13	17.4	16.2	0.0	48	15.0	6.1	0.0	83	11.4	7.4	0.0	118	35.0	23.5	6.2
14	17.8	20.5	0.0	49	27.6	16.7	9.1	84	23.1	42.1	0.0	119	20.5	14.8	0.0
15	21.4	9.5	0.0	50	0.0	0.0	0.0	85	24.2	28.6	7.7	120	81.2	90.9	90.0
16	96.5	95.7	95.2	51	0.0	0.0	0.0	86	45.2	33.3	0.0	121	61.9	57.1	7.7
17	47.1	40.0	25.0	52	50.0	38.7	13.8	87	21.6	17.1	12.1	122	13.3	17.4	0.0
18	20.8	28.6	6.1	53	46.1	50.0	33.3	88	38.1	32.3	13.8	123	96.8	95.2	94.7
19	35.7	37.5	0.0	54	5.6	17.4	0.0	89	70.6	61.5	54.5	124	15.8	0.0	0.0
20	20.3	9.3	0.0	55	87.7	89.4	88.9	90	54.0	42.9	23.1	125	13.8	8.3	0.0
21	24.0	23.5	0.0	56	47.1	50.0	20.0	91	56.0	70.0	55.6	126	31.2	26.1	9.5
22	25.6	13.8	7.4	57	77.4	78.6	76.9	92	20.3	16.3	4.3	127	22.2	19.4	6.9
23	76.9	83.9	75.9	58	55.6	71.4	66.7	93	30.8	34.5	7.4	128	40.0	34.3	18.2
24	44.4	38.5	8.3	59	27.0	24.1	7.1	94	40.0	41.7	13.0	129	12.5	8.7	0.0
25	22.2	25.0	0.0	60	26.7	25.0	13.3	95	22.6	23.8	5.0	130	25.0	14.8	0.0
26	21.4	9.5	0.0	61	50.0	48.5	25.8	96	25.8	32.3	0.0	131	29.2	20.7	0.0
27	20.8	28.6	6.1	62	24.0	23.5	0.0	97	51.4	66.7	35.7	132	26.1	25.6	0.0
28	30.0	62.5	28.6	63	20.5	20.0	0.0	98	69.0	72.0	69.6	133	98.2	97.8	97.7
29	22.2	24.0	4.2	64	45.8	56.4	27.0	99	38.5	40.0	33.3	134	20.7	27.3	0.0
30	16.3	19.5	0.0	65	59.5	50.0	30.8	100	54.5	64.3	46.1	135	26.1	21.6	11.4
31	23.8	21.4	7.7	66	40.8	35.0	15.8	101	22.2	18.2	4.8	136	32.0	44.4	12.5
32	26.7	61.5	54.5	67	10.8	12.9	0.0	102	66.7	63.2	58.8	137	21.3	20.5	10.8
33	19.5	29.6	8.0	68	12.1	9.5	0.0	103	21.1	24.2	0.0	138	64.7	73.3	57.1
34	14.3	11.1	0.0	69	22.2	23.5	0.0	104	26.7	22.2	0.0	139	32.0	38.1	10.5
35	77.4	72.0	60.9	70	11.4	14.8	0.0	105	47.4	50.0	15.4				

Run ID	tg2hug
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	35.2	34.6	20.0



Per-topic difference from median for modified unigram Dice

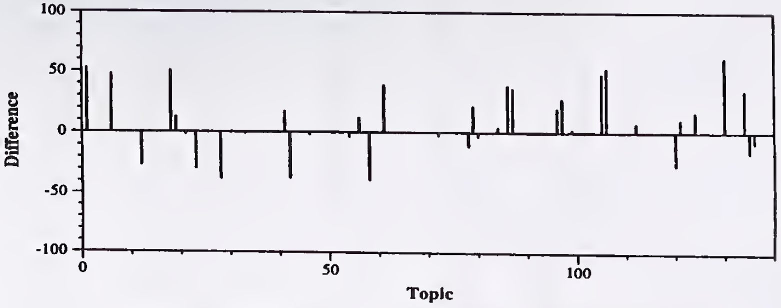
Per-topic Dice Scores

Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	85.7	82.3	80.0	36	100.0	100.0	100.0	71	44.4	52.2	38.1	106	9.1	23.5	0.0
2	26.9	22.2	9.3	37	12.5	4.9	0.0	72	34.8	41.2	25.0	107	21.1	15.4	8.3
3	21.1	27.6	14.8	38	100.0	100.0	100.0	73	20.0	9.5	0.0	108	27.9	27.0	0.0
4	18.2	18.2	0.0	39	21.6	33.3	7.1	74	25.0	28.6	6.1	109	21.3	12.1	0.0
5	35.3	46.1	25.0	40	18.2	17.4	0.0	75	53.9	47.6	10.5	110	28.6	40.0	8.7
6	14.3	12.9	0.0	41	27.0	21.4	7.7	76	91.7	91.4	90.9	111	55.8	48.5	45.2
7	31.6	33.3	7.1	42	68.8	58.3	45.5	77	77.3	70.6	68.8	112	22.9	18.8	13.3
8	20.0	18.8	0.0	43	100.0	100.0	100.0	78	51.4	45.5	10.0	113	11.8	9.1	0.0
9	16.7	8.7	0.0	44	37.0	36.4	20.0	79	0.0	0.0	0.0	114	23.1	20.0	0.0
10	16.2	13.3	0.0	45	28.6	30.8	8.3	80	21.4	22.2	0.0	115	26.1	16.7	0.0
11	0.0	8.7	0.0	46	27.6	17.4	9.5	81	24.2	35.7	7.7	116	13.8	25.0	0.0
12	8.3	0.0	0.0	47	20.0	10.5	0.0	82	31.8	25.0	6.7	117	38.1	38.7	20.7
13	30.8	34.3	6.1	48	30.3	34.8	28.6	83	18.2	17.1	0.0	118	15.0	0.0	0.0
14	90.9	90.3	82.8	49	7.7	9.5	0.0	84	23.1	42.1	0.0	119	88.0	85.7	83.3
15	25.6	20.0	0.0	50	24.0	20.0	0.0	85	34.1	40.0	6.1	120	33.3	45.5	30.0
16	13.8	0.0	0.0	51	7.4	10.0	0.0	86	11.1	9.5	0.0	121	26.7	26.7	0.0
17	29.6	20.0	0.0	52	77.5	76.5	75.0	87	78.4	82.9	82.0	122	13.8	18.2	0.0
18	22.7	24.2	6.5	53	6.9	28.6	0.0	88	15.0	14.3	7.7	123	31.6	14.3	0.0
19	25.0	0.0	0.0	54	16.7	23.5	0.0	89	25.0	28.6	0.0	124	86.5	82.8	74.1
20	94.5	94.1	93.8	55	35.1	32.6	4.9	90	21.4	10.5	0.0	125	12.5	8.0	0.0
21	93.3	94.7	94.1	56	44.4	50.0	11.1	91	25.0	26.7	15.4	126	14.3	0.0	0.0
22	26.3	14.3	7.7	57	0.0	6.7	0.0	92	47.6	46.7	7.1	127	14.6	6.9	0.0
23	56.6	57.1	30.0	58	0.0	0.0	0.0	93	13.3	26.1	0.0	128	33.3	34.8	9.5
24	33.3	47.1	26.7	59	94.1	91.9	85.7	94	21.5	20.4	8.5	129	21.4	11.1	0.0
25	22.2	25.0	0.0	60	30.3	28.6	0.0	95	32.6	27.0	0.0	130	23.5	24.0	0.0
26	46.5	45.2	13.8	61	85.7	88.9	80.0	96	32.4	35.3	25.0	131	80.8	77.8	76.5
27	22.7	24.2	6.5	62	93.3	94.7	94.1	97	34.1	51.6	27.6	132	30.0	21.4	0.0
28	10.0	11.8	6.2	63	27.8	28.6	7.7	98	16.7	28.6	0.0	133	8.7	6.1	0.0
29	41.7	38.3	13.3	64	94.1	95.0	94.7	99	41.2	36.4	10.0	134	92.7	93.8	93.3
30	21.7	16.7	5.9	65	42.9	43.8	20.0	100	7.1	0.0	0.0	135	9.5	13.3	0.0
31	43.6	44.4	11.8	66	33.3	36.4	6.5	101	98.2	100.0	100.0	136	26.3	13.3	0.0
32	5.9	0.0	0.0	67	7.4	9.5	0.0	102	8.7	11.1	0.0	137	25.0	28.6	7.7
33	28.6	17.4	9.5	68	11.1	27.3	0.0	103	95.2	97.1	97.0	138	13.8	16.7	9.1
34	19.4	10.5	0.0	69	61.2	52.9	43.8	104	35.1	26.1	4.5	139	59.5	60.0	42.9
35	20.5	21.4	7.7	70	9.5	11.1	0.0	105	100.0	100.0	100.0				

Genomics track, secondary task results — University Hospital of Geneva

Run ID	tgIIhugLASt
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	52.8	54.3	37.7



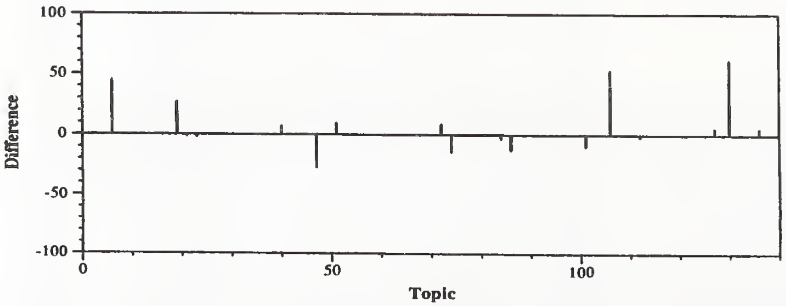
Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	100.0	100.0	100.0	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	66.7	76.9	72.7
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	40.8	39.0	20.5	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	50.0	52.2	19.1	40	43.8	40.0	26.1	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	91.9	89.7	81.5	41	30.8	34.5	14.8	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	68.8	58.3	45.5	77	24.2	13.8	7.4	112	6.1	28.6	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	51.4	45.5	10.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	31.6	42.9	16.7	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	23.1	11.1	0.0	115	34.0	36.8	22.2
11	8.0	10.5	0.0	46	22.2	7.1	0.0	81	62.5	71.4	50.0	116	35.7	40.0	26.1
12	43.5	47.1	26.7	47	70.6	84.6	66.7	82	21.6	14.8	0.0	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	18.2	21.1	0.0	49	44.4	50.0	44.4	84	26.7	30.8	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	62.9	69.6	47.6
16	100.0	100.0	100.0	51	20.5	19.4	13.8	86	88.9	86.7	78.6	121	75.4	70.0	63.2
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	95.2	94.4	94.1	122	22.2	28.6	16.7
18	95.7	91.9	91.4	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	59.3	52.6	35.3	54	9.1	11.8	0.0	89	75.0	66.7	60.0	124	88.9	85.7	76.9
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	25.8	9.5	0.0	56	58.8	66.7	20.0	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	28.6	23.1	8.3
23	56.6	57.1	30.0	58	22.2	38.1	0.0	93	31.6	35.7	7.7	128	38.7	47.6	0.0
24	45.7	40.0	8.7	59	37.2	42.4	12.9	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	94.7	93.3	92.3
26	91.9	92.9	84.6	61	85.7	88.9	80.0	96	100.0	100.0	100.0	131	39.0	33.3	7.1
27	37.8	41.4	22.2	62	25.8	9.5	0.0	97	80.0	96.5	74.1	132	26.7	26.3	0.0
28	21.1	28.6	16.7	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	41.7	44.4	0.0	134	92.7	93.8	93.3
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	85.1	85.0	84.2
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	40.7	46.5	19.5	136	31.2	34.8	9.5
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	26.7	31.2	13.3	68	15.4	31.6	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	100.0	100.0	100.0				

Genomics track, secondary task results — University of Iowa

Run ID	UIowaSecCan
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	50.7	52.7	35.3

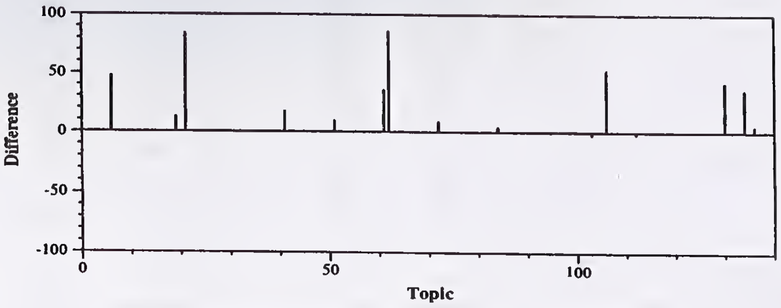


Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	66.7	76.9	72.7
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	47.6	50.0	26.7	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	87.8	85.7	84.8	109	38.1	45.7	18.2
5	50.0	52.2	19.1	40	48.0	47.1	40.0	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	83.9	87.0	85.7	41	25.0	17.4	0.0	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	8.7	19.1	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	19.1	14.3	0.0	115	34.0	36.8	22.2
11	8.0	10.5	0.0	46	26.7	9.1	0.0	81	62.5	71.4	50.0	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	42.9	57.1	42.1	82	21.6	14.8	0.0	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	18.2	21.1	0.0	49	44.4	50.0	44.4	84	20.0	23.5	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	32.0	28.6	21.1	86	46.7	35.3	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	66.7	66.7	57.1	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	25.8	9.5	0.0	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	35.7	28.6	10.5
23	81.2	84.6	83.3	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	38.7	47.6	0.0
24	45.7	40.0	8.7	59	37.2	42.4	12.9	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	94.7	93.3	92.3
26	91.9	92.9	84.6	61	51.3	50.0	26.7	96	48.3	80.0	57.1	131	39.0	33.3	7.1
27	37.8	41.4	22.2	62	25.8	9.5	0.0	97	52.9	69.0	37.0	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	57.1	58.8	31.2
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	100.0	100.0	100.0
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	37.5	36.8	22.2	136	33.3	47.1	13.3
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Run ID	UniNEie1
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	52.3	54.8	37.4

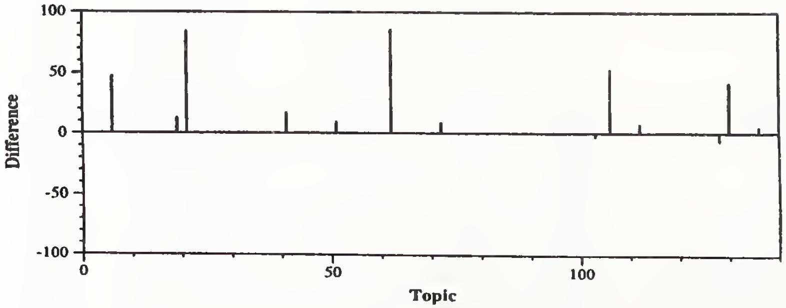


Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	66.7	76.9	72.7
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	47.6	50.0	26.7	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	50.0	52.2	19.1	40	43.8	40.0	26.1	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	91.9	89.7	81.5	41	30.8	34.5	14.8	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	8.7	19.1	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	19.1	14.3	0.0	115	34.0	36.8	22.2
11	8.0	10.5	0.0	46	26.7	9.1	0.0	81	62.5	71.4	50.0	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	70.6	84.6	66.7	82	21.6	14.8	0.0	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	18.2	21.1	0.0	49	44.4	50.0	44.4	84	26.7	30.8	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	32.0	28.6	21.1	86	58.5	48.0	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	59.3	52.6	35.3	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	86.5	95.2	94.7	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	28.6	23.1	8.3
23	79.0	86.7	78.6	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	38.7	47.6	0.0
24	45.7	40.0	8.7	59	37.2	42.4	12.9	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	72.0	73.7	70.6
26	91.9	92.9	84.6	61	81.1	85.7	76.9	96	48.3	80.0	57.1	131	39.0	33.3	7.1
27	37.8	41.4	22.2	62	86.5	95.2	94.7	97	52.9	69.0	37.0	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	93.6	94.4	94.1
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	100.0	100.0	100.0
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	40.7	46.5	19.5	136	33.3	47.1	13.3
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	27.8	22.2	8.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Run ID	UniNEie2
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	51.7	54.3	36.6

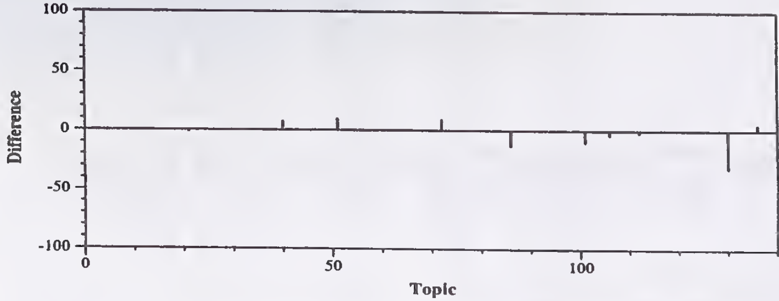


Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	66.7	76.9	72.7
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	47.6	50.0	26.7	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	50.0	52.2	19.1	40	43.8	40.0	26.1	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	91.9	89.7	81.5	41	30.8	34.5	14.8	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	6.1	28.6	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	19.1	14.3	0.0	115	34.0	36.8	22.2
11	8.0	10.5	0.0	46	26.7	9.1	0.0	81	62.5	71.4	50.0	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	70.6	84.6	66.7	82	21.6	14.8	0.0	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	18.2	21.1	0.0	49	44.4	50.0	44.4	84	21.1	26.7	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	32.0	28.6	21.1	86	58.5	48.0	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	59.3	52.6	35.3	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	86.5	95.2	94.7	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	28.6	23.1	8.3
23	79.0	86.7	78.6	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	34.3	41.7	0.0
24	45.7	40.0	8.7	59	37.2	42.4	12.9	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	72.0	73.7	70.6
26	91.9	92.9	84.6	61	51.3	50.0	26.7	96	48.3	80.0	57.1	131	39.0	33.3	7.1
27	37.8	41.4	22.2	62	86.5	95.2	94.7	97	52.9	69.0	37.0	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	57.1	58.8	31.2
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	100.0	100.0	100.0
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	40.7	46.5	19.5	136	33.3	47.1	13.3
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	27.8	22.2	8.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Run ID	UniNEie3
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	49.5	51.4	33.6



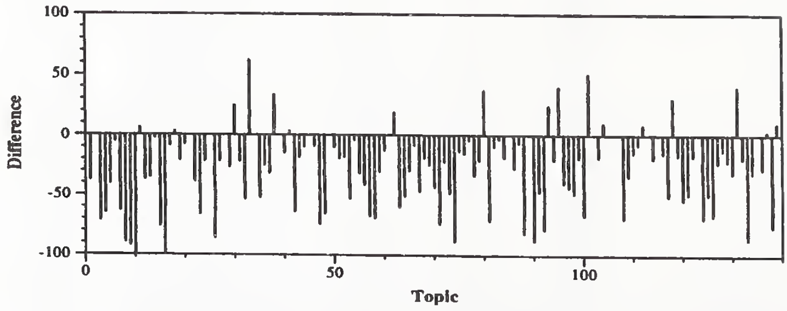
Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores

Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	78.8	88.9	88.0	106	14.3	20.0	0.0
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	47.6	50.0	26.7	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	71.4	80.0	77.8
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	50.0	52.2	19.1	40	48.0	47.1	40.0	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	46.1	42.1	23.5	41	25.0	17.4	0.0	76	30.8	21.4	7.7	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	96.8	95.7	95.2	77	24.2	13.8	7.4	112	8.7	19.1	0.0
8	90.5	89.7	81.5	43	25.0	27.6	7.4	78	69.0	55.6	25.0	113	36.4	37.5	14.3
9	100.0	100.0	100.0	44	46.1	41.7	18.2	79	23.1	21.1	11.8	114	30.3	38.5	25.0
10	100.0	100.0	100.0	45	34.5	32.0	0.0	80	19.1	14.3	0.0	115	34.0	36.8	22.2
11	8.0	10.5	0.0	46	26.7	9.1	0.0	81	62.5	71.4	50.0	116	35.7	40.0	26.1
12	58.3	73.7	47.1	47	70.6	84.6	66.7	82	21.6	14.8	0.0	117	85.7	86.5	85.7
13	50.0	54.5	20.0	48	88.2	84.6	83.3	83	23.8	41.2	12.5	118	35.9	24.2	6.5
14	18.2	21.1	0.0	49	44.4	50.0	44.4	84	21.1	26.7	0.0	119	40.0	31.6	0.0
15	97.6	96.3	88.0	50	8.3	10.0	0.0	85	60.9	84.2	58.8	120	83.9	95.2	94.7
16	100.0	100.0	100.0	51	32.0	28.6	21.1	86	46.7	35.3	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	48.6	58.1	27.6	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	100.0	100.0	100.0	123	100.0	100.0	100.0
19	52.2	40.0	15.4	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	25.8	9.5	0.0	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	84.0	85.0	73.7	127	28.6	23.1	8.3
23	79.0	86.7	78.6	58	58.8	76.9	72.7	93	31.6	35.7	7.7	128	38.7	47.6	0.0
24	45.7	40.0	8.7	59	37.2	42.4	12.9	94	40.7	42.5	13.3	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	0.0	0.0	0.0
26	91.9	92.9	84.6	61	51.3	50.0	26.7	96	48.3	80.0	57.1	131	39.0	33.3	7.1
27	37.8	41.4	22.2	62	25.8	9.5	0.0	97	52.9	69.0	37.0	132	26.7	26.3	0.0
28	31.6	66.7	30.8	63	92.7	94.1	87.5	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	45.8	51.1	31.1	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	57.1	58.8	31.2
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	56.2	66.7	48.0	135	100.0	100.0	100.0
31	24.4	22.2	8.0	66	41.7	35.9	16.2	101	37.5	36.8	22.2	136	33.3	47.1	13.3
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	70.0	66.7	62.5	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	15.4	31.6	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	50.0	46.1	16.7	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Run ID	UniNEie4
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	25.9	25.3	12.0

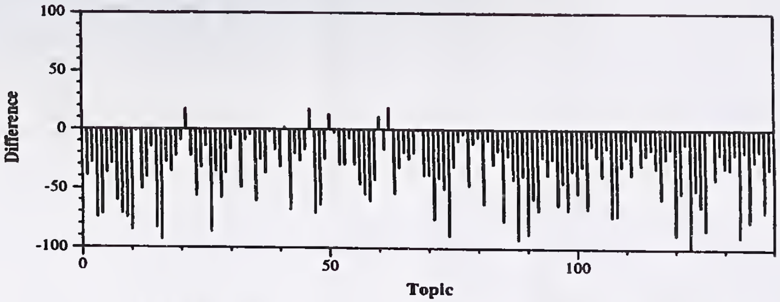


Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	20.0	9.5	0.0	36	17.6	8.0	0.0	71	18.2	14.8	0.0	106	17.1	24.0	8.7
2	47.1	43.2	17.1	37	12.5	5.0	0.0	72	21.4	19.1	0.0	107	61.5	73.3	71.4
3	26.3	20.0	7.1	38	30.0	40.0	18.6	73	23.3	18.8	0.0	108	16.7	10.8	0.0
4	12.1	14.8	0.0	39	28.6	33.3	0.0	74	15.4	11.8	0.0	109	20.8	11.8	0.0
5	9.5	11.1	0.0	40	32.4	25.0	0.0	75	17.4	26.7	0.0	110	22.9	20.0	7.1
6	32.6	36.4	25.8	41	26.3	20.7	7.4	76	17.0	6.2	0.0	111	61.2	54.0	51.4
7	19.5	27.6	7.4	42	38.5	31.6	23.5	77	16.7	9.5	5.0	112	28.1	29.8	17.8
8	4.2	0.0	0.0	43	25.8	9.1	0.0	78	0.0	22.2	0.0	113	36.4	37.5	14.3
9	5.7	7.4	0.0	44	20.7	32.0	8.7	79	0.0	0.0	0.0	114	10.3	19.4	0.0
10	8.7	0.0	0.0	45	34.5	32.0	0.0	80	62.9	51.9	48.0	115	34.1	36.4	25.8
11	12.5	16.7	0.0	46	16.2	0.0	0.0	81	0.0	0.0	0.0	116	13.8	25.0	0.0
12	34.8	36.4	20.0	47	16.7	10.3	0.0	82	13.3	5.3	0.0	117	37.2	36.4	19.4
13	13.8	19.1	0.0	48	19.1	18.8	6.7	83	13.9	37.5	6.7	118	60.0	54.5	38.7
14	26.7	18.2	0.0	49	44.4	50.0	44.4	84	5.9	8.0	0.0	119	20.5	14.8	0.0
15	25.6	20.0	0.0	50	0.0	0.0	0.0	85	60.9	84.2	58.8	120	31.6	41.7	27.3
16	13.8	0.0	0.0	51	0.0	0.0	0.0	86	23.3	20.7	0.0	121	15.7	10.5	0.0
17	40.0	28.6	16.7	52	23.3	11.4	0.0	87	39.0	51.4	24.2	122	8.0	11.1	0.0
18	22.9	44.4	0.0	53	0.0	0.0	0.0	88	18.8	18.2	10.0	123	100.0	100.0	100.0
19	12.9	19.1	0.0	54	7.4	11.1	0.0	89	75.0	66.7	60.0	124	12.9	0.0	0.0
20	17.8	12.5	0.0	55	27.4	22.9	12.1	90	19.1	7.4	0.0	125	30.8	38.1	21.1
21	22.2	11.1	0.0	56	18.2	13.3	0.0	91	25.0	26.7	15.4	126	35.3	33.3	18.2
22	26.3	14.3	7.7	57	18.2	14.0	3.6	92	13.3	6.5	0.0	127	16.0	0.0	0.0
23	30.4	20.5	5.4	58	5.7	8.0	0.0	93	41.0	60.0	28.6	128	29.4	34.8	0.0
24	13.8	18.2	0.0	59	17.4	12.1	0.0	94	23.7	22.2	9.3	129	14.3	6.9	0.0
25	29.6	21.1	0.0	60	25.0	17.4	9.5	95	83.7	90.3	89.7	130	0.0	0.0	0.0
26	19.5	6.9	0.0	61	51.3	50.0	26.7	96	21.1	40.0	24.2	131	76.4	73.7	72.2
27	19.1	19.4	6.9	62	37.2	28.6	15.4	97	19.1	25.5	8.9	132	10.8	6.9	0.0
28	31.6	66.7	30.8	63	34.8	34.1	5.1	98	15.4	26.7	0.0	133	13.3	13.3	0.0
29	22.6	24.5	4.3	64	16.2	7.4	0.0	99	25.0	23.1	0.0	134	20.7	27.3	0.0
30	41.3	46.1	36.0	65	27.0	22.2	0.0	100	9.5	0.0	0.0	135	100.0	100.0	100.0
31	0.0	0.0	0.0	66	26.2	27.3	9.5	101	94.7	97.7	97.6	136	36.4	14.3	0.0
32	10.0	13.3	0.0	67	9.1	11.8	0.0	102	70.0	66.7	62.5	137	21.7	21.1	11.1
33	92.3	94.1	93.8	68	17.4	12.5	0.0	103	10.3	6.7	0.0	138	0.0	0.0	0.0
34	14.8	11.8	0.0	69	23.3	21.4	0.0	104	30.2	32.6	4.9	139	33.3	50.0	22.2
35	21.3	22.9	6.1	70	8.3	11.1	0.0	105	48.6	51.9	16.0				

Run ID	UniNEie5
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	9.4	14.2	0.1

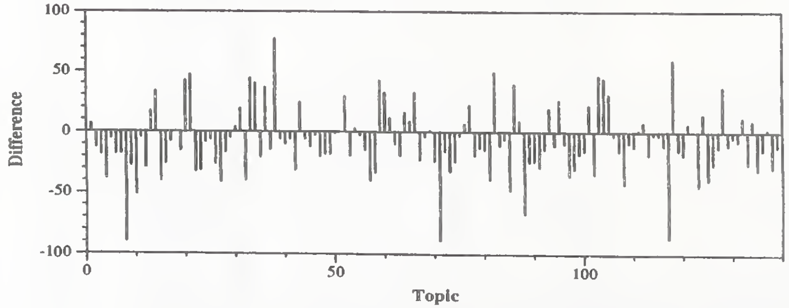


Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	6.7	8.0	0.0	36	6.5	7.7	0.0	71	12.5	13.3	0.0	106	0.0	8.3	0.0
2	4.1	14.6	0.0	37	0.0	0.0	0.0	72	0.0	0.0	0.0	107	0.0	0.0	0.0
3	10.0	16.7	0.0	38	4.3	4.9	0.0	73	7.4	16.7	0.0	108	6.5	7.4	0.0
4	7.1	7.7	0.0	39	8.0	16.0	0.0	74	9.8	10.5	0.0	109	12.8	15.0	0.0
5	14.3	14.8	0.0	40	7.4	8.3	0.0	75	0.0	8.7	0.0	110	11.1	12.1	0.0
6	11.1	12.9	0.0	41	10.8	19.4	0.0	76	10.0	11.8	0.0	111	15.8	24.2	0.0
7	10.8	30.3	0.0	42	24.2	27.6	0.0	77	9.5	10.0	0.0	112	13.3	13.3	0.0
8	10.3	18.2	0.0	43	6.1	7.1	0.0	78	7.1	8.7	0.0	113	6.9	7.7	0.0
9	16.7	25.0	0.0	44	7.4	15.4	0.0	79	0.0	9.5	0.0	114	16.0	16.7	0.0
10	14.3	14.8	0.0	45	13.8	14.8	0.0	80	6.7	7.7	0.0	115	13.0	20.0	0.0
11	8.7	9.1	0.0	46	23.5	26.7	0.0	81	0.0	8.7	0.0	116	7.7	7.7	0.0
12	21.4	23.1	0.0	47	12.5	13.8	0.0	82	5.0	5.9	0.0	117	14.6	28.6	0.0
13	13.3	13.8	0.0	48	12.1	20.7	0.0	83	5.4	11.8	0.0	118	0.0	0.0	0.0
14	0.0	6.1	0.0	49	15.4	25.0	0.0	84	7.7	8.3	0.0	119	12.5	15.4	0.0
15	5.3	12.9	0.0	50	14.8	23.1	0.0	85	0.0	7.1	0.0	120	6.1	7.1	0.0
16	6.2	6.9	0.0	51	8.0	16.7	0.0	86	20.0	25.8	6.9	121	4.9	6.2	0.0
17	8.7	9.5	0.0	52	0.0	0.0	0.0	87	15.0	16.2	0.0	122	7.7	16.7	0.0
18	5.0	5.7	0.0	53	13.3	23.1	0.0	88	6.2	7.4	0.0	123	0.0	0.0	0.0
19	15.4	17.4	0.0	54	7.7	8.0	0.0	89	25.0	27.3	0.0	124	11.1	18.8	0.0
20	8.5	11.1	0.0	55	22.2	25.0	0.0	90	0.0	7.1	0.0	125	14.8	23.1	0.0
21	17.6	28.6	0.0	56	0.0	8.3	0.0	91	14.3	15.4	0.0	126	13.8	16.0	0.0
22	14.3	30.8	0.0	57	12.5	26.7	0.0	92	9.5	17.1	0.0	127	18.2	20.7	0.0
23	16.2	30.3	0.0	58	8.0	16.7	0.0	93	5.1	12.1	0.0	128	0.0	6.2	0.0
24	6.9	7.7	0.0	59	0.0	0.0	0.0	94	3.9	4.4	0.0	129	6.9	8.0	0.0
25	0.0	6.7	0.0	60	23.5	41.4	0.0	95	11.1	25.0	0.0	130	0.0	0.0	0.0
26	5.6	6.2	0.0	61	29.4	33.3	0.0	96	16.7	16.7	0.0	131	0.0	0.0	0.0
27	5.0	5.7	0.0	62	17.6	28.6	0.0	97	17.1	24.2	0.0	132	4.8	5.4	0.0
28	0.0	8.7	0.0	63	31.6	40.0	0.0	98	6.9	7.4	0.0	133	4.3	10.0	0.0
29	4.7	13.9	0.0	64	17.8	25.6	0.0	99	0.0	7.1	0.0	134	25.0	40.0	6.1
30	0.0	4.5	0.0	65	22.9	32.3	0.0	100	6.2	13.3	0.0	135	10.5	22.9	0.0
31	9.1	17.1	0.0	66	9.3	11.1	0.0	101	8.9	15.4	0.0	136	6.9	16.0	0.0
32	16.0	17.4	0.0	67	30.8	38.5	8.3	102	0.0	0.0	0.0	137	0.0	0.0	0.0
33	14.3	23.5	0.0	68	14.3	30.8	0.0	103	10.0	11.1	0.0	138	0.0	6.9	0.0
34	5.9	7.4	0.0	69	5.7	7.4	0.0	104	0.0	0.0	0.0	139	6.7	7.4	0.0
35	12.9	14.3	0.0	70	7.7	16.0	0.0	105	5.3	12.5	0.0				

Run ID	uwb2
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	44.4	44.1	2.3

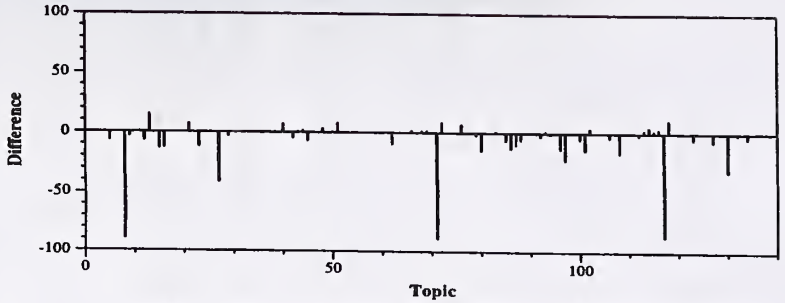


Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	61.5	53.9	0.0	36	70.3	69.6	0.0	71	0.0	0.0	0.0	106	8.3	21.1	0.0
2	34.7	30.8	0.0	37	32.7	22.7	0.0	72	28.2	25.4	0.0	107	49.0	57.9	0.0
3	69.1	72.7	0.0	38	90.6	84.0	0.0	73	33.3	34.3	6.1	108	36.7	37.8	5.7
4	37.8	41.4	0.0	39	18.6	27.8	0.0	74	78.0	76.0	4.2	109	35.1	37.2	0.0
5	39.0	46.7	0.0	40	29.2	30.3	0.0	75	38.9	37.0	0.0	110	27.9	23.5	6.2
6	32.6	24.2	6.5	41	20.4	11.8	0.0	76	35.0	27.6	0.0	111	60.7	63.4	0.0
7	71.7	73.2	0.0	42	71.8	64.3	0.0	77	42.9	35.6	0.0	112	25.4	28.0	0.0
8	0.0	0.0	0.0	43	57.1	51.4	6.1	78	46.5	37.0	0.0	113	20.0	19.4	0.0
9	61.8	72.2	0.0	44	36.4	35.7	0.0	79	10.0	8.3	0.0	114	28.6	36.4	6.5
10	45.5	48.6	11.4	45	25.0	20.5	0.0	80	0.0	0.0	0.0	115	41.4	34.0	4.4
11	4.5	6.1	0.0	46	24.4	6.5	0.0	81	27.0	32.3	0.0	116	24.5	29.3	0.0
12	35.3	44.4	0.0	47	48.0	64.7	0.0	82	67.7	64.0	0.0	117	0.0	0.0	0.0
13	71.0	71.4	0.0	48	66.7	66.7	6.5	83	24.6	30.4	0.0	118	83.6	83.3	0.0
14	62.3	54.2	8.7	49	30.8	32.3	0.0	84	19.6	21.1	0.0	119	25.9	17.1	0.0
15	62.3	55.8	4.9	50	6.9	8.7	0.0	85	34.8	36.8	0.0	120	61.9	76.9	8.3
16	70.3	74.1	0.0	51	20.0	19.4	0.0	86	89.4	86.7	0.0	121	71.9	65.1	0.0
17	35.7	30.0	0.0	52	66.7	58.5	5.1	87	66.7	66.7	5.0	122	22.9	28.6	7.7
18	37.9	41.9	4.9	53	30.0	33.3	0.0	88	40.6	33.3	4.3	123	63.6	56.2	6.7
19	37.5	25.0	9.1	54	13.8	18.2	0.0	89	48.0	42.1	0.0	124	86.5	82.8	0.0
20	66.7	62.8	4.1	55	50.8	52.0	4.2	90	71.8	71.4	0.0	125	40.0	48.5	6.5
21	62.2	58.1	0.0	56	34.3	40.0	0.0	91	36.8	45.2	0.0	126	73.3	73.7	11.8
22	30.8	20.7	0.0	57	39.3	41.5	7.8	92	71.4	71.1	4.7	127	16.3	10.8	0.0
23	57.7	55.0	5.3	58	33.3	43.5	9.5	93	40.0	54.5	6.5	128	87.0	83.9	0.0
24	33.3	31.8	0.0	59	87.1	84.4	0.0	94	31.7	32.3	0.0	129	21.7	18.2	0.0
25	22.2	14.8	0.0	60	52.4	62.1	0.0	95	78.3	75.7	5.7	130	24.2	27.3	0.0
26	68.0	66.7	0.0	61	66.7	61.1	11.8	96	87.2	71.0	0.0	131	36.4	26.7	4.7
27	0.0	0.0	0.0	62	0.0	0.0	0.0	97	22.2	33.3	5.0	132	38.6	37.1	0.0
28	28.6	50.0	0.0	63	72.0	75.0	5.3	98	40.8	45.0	0.0	133	76.7	74.6	10.5
29	43.1	46.4	3.7	64	79.4	73.5	8.5	99	20.8	24.2	0.0	134	72.0	66.7	5.4
30	35.5	25.5	0.0	65	56.6	60.0	0.0	100	44.4	51.4	0.0	135	69.0	69.6	4.5
31	43.1	40.9	0.0	66	77.4	68.1	17.8	101	78.1	68.0	4.2	136	29.2	27.8	0.0
32	14.3	26.7	0.0	67	26.4	35.6	0.0	102	29.2	32.4	5.7	137	22.9	20.0	0.0
33	78.7	76.2	10.0	68	20.0	27.8	0.0	103	75.5	71.1	0.0	138	38.6	46.8	0.0
34	65.3	51.6	6.9	69	55.6	47.4	0.0	104	79.5	66.7	0.0	139	22.2	28.6	0.0
35	57.1	54.5	6.5	70	23.5	30.8	0.0	105	81.6	82.3	18.8				

Run ID	uwb3
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	46.5	48.2	29.5



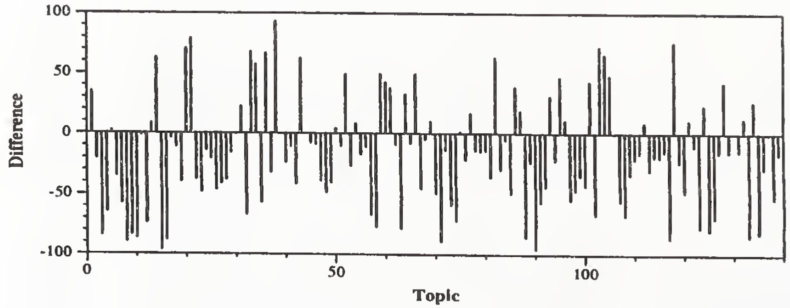
Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores

Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	41.4	47.6	21.1	36	34.5	33.3	12.5	71	0.0	0.0	0.0	106	14.3	20.0	0.0
2	47.1	43.2	17.1	37	43.8	37.0	0.0	72	47.6	50.0	26.7	107	61.5	73.3	71.4
3	88.4	91.4	84.8	38	15.8	6.7	0.0	73	59.3	66.7	37.5	108	60.0	63.6	60.0
4	77.8	80.0	46.1	39	28.6	33.3	0.0	74	97.9	100.0	100.0	109	38.1	45.7	18.2
5	44.4	45.5	10.0	40	48.0	47.1	40.0	75	38.1	40.0	15.4	110	41.4	34.8	9.5
6	46.1	42.1	23.5	41	25.0	17.4	0.0	76	35.9	28.6	15.4	111	50.0	62.1	44.4
7	92.7	90.9	90.3	42	93.3	90.9	80.0	77	24.2	13.8	7.4	112	8.7	19.1	0.0
8	0.0	0.0	0.0	43	25.6	28.6	7.7	78	69.0	55.6	25.0	113	38.1	40.0	15.4
9	97.1	96.3	96.0	44	46.1	43.5	19.1	79	21.4	19.1	10.5	114	33.3	43.5	28.6
10	100.0	100.0	100.0	45	28.6	25.0	0.0	80	0.0	0.0	0.0	115	39.1	38.9	23.5
11	8.3	11.1	0.0	46	26.7	9.1	0.0	81	62.5	71.4	50.0	116	38.5	43.5	28.6
12	52.2	66.7	25.0	47	70.6	84.6	66.7	82	21.6	14.8	0.0	117	0.0	0.0	0.0
13	64.0	69.6	19.1	48	90.9	88.0	87.0	83	24.4	42.4	12.9	118	45.0	35.3	12.5
14	18.2	20.5	0.0	49	44.4	50.0	44.4	84	21.1	26.7	0.0	119	40.0	31.6	0.0
15	88.4	82.8	66.7	50	8.7	10.5	0.0	85	54.5	77.8	37.5	120	83.9	95.2	94.7
16	89.7	87.0	76.2	51	30.8	27.3	20.0	86	46.7	35.3	0.0	121	63.4	59.3	8.0
17	43.5	37.5	14.3	52	36.8	29.6	16.0	87	40.0	48.3	7.4	122	22.2	28.6	16.7
18	37.8	41.4	22.2	53	48.0	52.6	35.3	88	96.3	94.1	80.0	123	96.5	94.7	82.3
19	52.2	40.0	15.4	54	11.8	15.4	0.0	89	75.0	66.7	60.0	124	76.5	69.2	41.7
20	24.6	20.5	5.4	55	60.5	54.5	25.8	90	96.5	95.2	94.7	125	70.0	87.5	57.1
21	31.2	18.2	0.0	56	50.0	54.5	22.2	91	58.3	73.7	58.8	126	100.0	100.0	100.0
22	42.1	53.3	30.8	57	80.0	81.5	80.0	92	81.6	82.0	70.3	127	23.5	16.0	0.0
23	73.3	75.0	54.5	58	58.8	76.9	72.7	93	32.4	37.0	8.0	128	38.7	47.6	0.0
24	45.7	40.0	8.7	59	37.2	42.4	12.9	94	40.0	41.7	13.0	129	38.1	28.6	0.0
25	29.6	21.1	0.0	60	21.4	30.0	0.0	95	50.0	50.0	30.8	130	0.0	0.0	0.0
26	91.9	92.9	84.6	61	51.3	50.0	26.7	96	42.9	66.7	16.0	131	39.0	33.3	7.1
27	0.0	0.0	0.0	62	0.0	0.0	0.0	97	35.3	46.7	21.4	132	25.5	25.0	0.0
28	31.6	66.7	30.8	63	92.3	93.8	93.3	98	71.4	75.0	72.7	133	100.0	100.0	100.0
29	42.5	47.8	22.7	64	46.8	57.9	27.8	99	40.0	42.1	35.3	134	53.7	54.5	19.4
30	38.3	21.6	0.0	65	61.1	51.9	32.0	100	51.6	61.5	33.3	135	100.0	100.0	100.0
31	24.4	22.2	8.0	66	42.5	37.8	17.1	101	34.0	32.4	17.1	136	32.0	42.1	11.8
32	28.6	66.7	60.0	67	41.4	58.3	36.4	102	73.7	70.6	66.7	137	23.1	18.2	0.0
33	31.8	32.3	13.8	68	16.0	33.3	0.0	103	21.6	25.0	0.0	138	66.7	75.9	59.3
34	14.8	11.8	0.0	69	51.3	48.0	17.4	104	27.3	22.9	0.0	139	33.3	40.0	11.1
35	80.0	75.0	63.6	70	37.0	54.5	40.0	105	48.6	51.9	16.0				

Run ID	uwb4
Run Description	Secondary task, automatic
Number of Topics	139

	Average score over all topics		
	Classic	Unigram	Bigram
Average	36.3	35.2 .	22.7



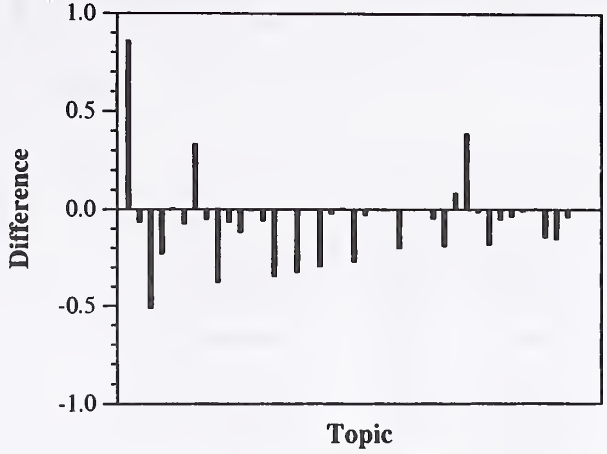
Per-topic difference from median for modified unigram Dice

Per-topic Dice Scores															
Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram	Topic	Classic	Unigram	Bigram
1	85.7	82.3	80.0	36	100.0	100.0	100.0	71	0.0	0.0	0.0	106	9.1	23.5	0.0
2	26.9	22.2	9.3	37	12.8	5.0	0.0	72	25.4	27.4	16.3	107	25.8	18.2	10.0
3	11.1	7.1	0.0	38	100.0	100.0	100.0	73	17.6	8.3	0.0	108	14.6	12.9	0.0
4	12.5	15.4	0.0	39	21.6	33.3	7.1	74	25.0	28.6	6.1	109	21.3	12.1	0.0
5	48.3	54.5	20.0	40	17.1	16.0	0.0	75	46.7	41.7	9.1	110	22.2	13.8	7.4
6	5.7	7.1	0.0	41	18.2	6.5	0.0	76	0.0	0.0	0.0	111	52.2	45.7	42.4
7	30.8	33.3	7.1	42	61.1	53.9	41.7	77	36.7	30.8	16.2	112	28.1	29.8	17.8
8	0.0	0.0	0.0	43	93.3	90.0	77.8	78	48.6	41.7	9.1	113	11.8	7.4	0.0
9	19.1	16.0	0.0	44	41.7	42.1	23.5	79	5.1	6.2	0.0	114	22.2	19.1	0.0
10	16.2	13.3	0.0	45	23.5	24.0	0.0	80	0.0	0.0	0.0	115	27.9	17.1	0.0
11	0.0	8.7	0.0	46	17.1	0.0	0.0	81	24.2	35.7	7.7	116	13.9	25.0	5.3
12	0.0	0.0	0.0	47	17.6	45.5	30.0	82	81.5	78.0	76.9	117	0.0	0.0	0.0
13	60.9	63.2	11.8	48	31.2	36.4	30.0	83	12.5	11.8	0.0	118	97.9	100.0	100.0
14	86.4	83.9	69.0	49	7.7	9.5	0.0	84	22.2	21.1	5.6	119	14.3	7.7	0.0
15	9.1	0.0	0.0	50	11.1	14.3	0.0	85	28.6	35.3	0.0	120	35.3	47.6	31.6
16	23.1	11.8	0.0	51	7.1	9.1	0.0	86	88.9	86.7	78.6	121	75.4	70.0	63.2
17	47.1	33.3	20.0	52	80.8	78.8	77.4	87	73.5	76.9	70.3	122	13.8	18.2	0.0
18	21.7	30.3	6.5	53	24.2	26.1	9.5	88	27.1	14.8	3.9	123	33.3	23.1	0.0
19	25.0	0.0	0.0	54	16.7	23.5	0.0	89	42.1	42.9	33.3	124	94.1	92.3	83.3
20	91.2	91.4	90.9	55	36.4	37.2	14.6	90	22.2	0.0	0.0	125	12.5	8.0	0.0
21	82.3	90.0	88.9	56	37.5	43.5	9.5	91	13.3	17.4	9.5	126	34.8	30.8	0.0
22	28.6	15.4	8.3	57	16.0	14.8	3.9	92	43.9	41.4	7.4	127	15.0	7.1	0.0
23	42.5	38.9	0.0	58	0.0	0.0	0.0	93	47.1	66.7	32.0	128	95.2	89.7	81.5
24	18.6	25.8	6.9	59	94.1	91.9	85.7	94	21.5	20.4	8.5	129	14.3	13.3	0.0
25	14.3	0.0	0.0	60	66.7	72.7	40.0	95	94.7	96.5	96.3	130	27.6	31.6	23.5
26	47.6	46.7	14.3	61	90.9	88.0	87.0	96	97.1	90.9	71.0	131	27.4	18.8	0.0
27	0.0	0.0	0.0	62	0.0	0.0	0.0	97	13.0	14.6	10.3	132	44.7	38.8	21.5
28	30.0	28.6	16.7	63	11.8	16.0	0.0	98	16.7	28.6	0.0	133	16.3	15.4	0.0
29	29.8	35.0	15.8	64	92.6	90.5	90.0	99	5.1	7.4	0.0	134	85.7	84.8	77.4
30	24.0	21.1	5.6	65	41.9	43.8	20.0	100	23.5	24.0	0.0	135	19.1	18.2	6.5
31	43.6	44.4	11.8	66	89.8	85.7	72.7	101	96.2	89.5	77.8	136	26.3	13.3	0.0
32	5.3	0.0	0.0	67	11.1	13.3	0.0	102	5.1	0.0	0.0	137	14.8	17.4	9.5
33	100.0	100.0	100.0	68	25.6	26.7	14.3	103	95.2	97.1	97.0	138	22.7	22.9	12.1
34	78.0	69.2	66.7	69	65.2	56.2	46.7	104	90.6	88.9	84.6	139	17.1	23.1	8.3
35	14.3	18.2	0.0	70	5.1	5.6	0.0	105	97.6	100.0	100.0				

Run ID	OPEN
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.1203	0.0098
At 10 passages	0.1217	0.0148
At 15 passages	0.1175	0.0179
At 20 passages	0.1151	0.0207
At 30 passages	0.1152	0.0269
At 50 passages	0.1074	0.0343
At 100 passages	0.0975	0.0369
R-Precision	0.0954	

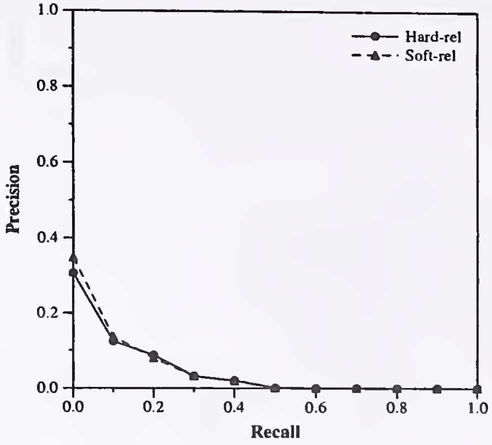


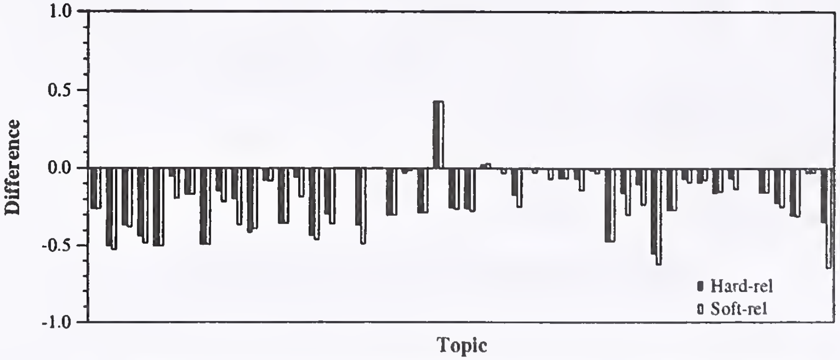
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	10953	10953
Total relevant	5123	7576
Total rel. ret.	763	1036
MAP	0.0324	0.0368

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.1583	0.2125
At 10 docs	0.1500	0.1937
At 15 docs	0.1458	0.1861
At 20 docs	0.1427	0.1854
At 30 docs	0.1285	0.1639
At 100 docs	0.0765	0.0996
At 200 docs	0.0525	0.0700
At 500 docs	0.0295	0.0400
At 1000 docs	0.0159	0.0216
R-Precision	0.0679	0.0792



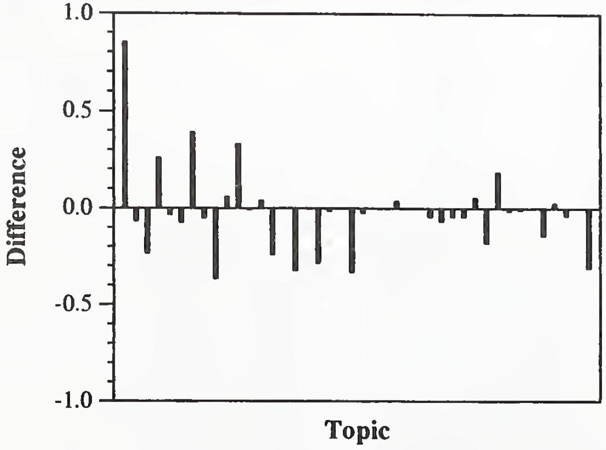


Difference from median in Document based R-precision per topic

Run ID	OPEN1
Run Description	not a baseline, used metadata; title+desc+narr

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.1944	0.0146
At 10 passages	0.1841	0.0260
At 15 passages	0.1625	0.0295
At 20 passages	0.1498	0.0343
At 30 passages	0.1482	0.0404
At 50 passages	0.1489	0.0531
At 100 passages	0.1314	0.0614
R-Precision	0.1381	

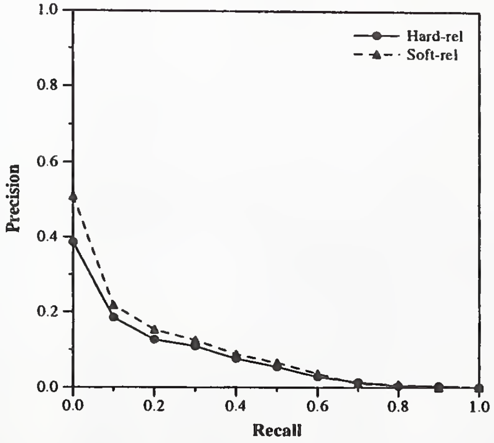


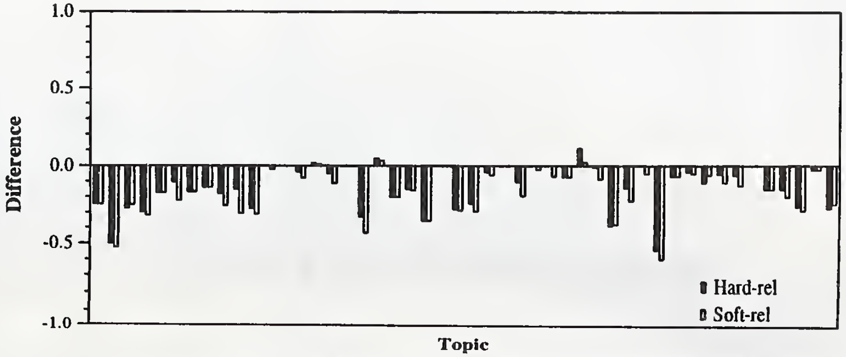
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	15134	15134
Total relevant	5123	7576
Total rel. ret.	1146	1652
MAP	0.0715	0.0858

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.2125	0.3042
At 10 docs	0.1937	0.2688
At 15 docs	0.1778	0.2431
At 20 docs	0.1656	0.2260
At 30 docs	0.1590	0.2132
At 100 docs	0.1235	0.1677
At 200 docs	0.0856	0.1189
At 500 docs	0.0450	0.0653
At 1000 docs	0.0239	0.0344
R-Precision	0.1210	0.1406



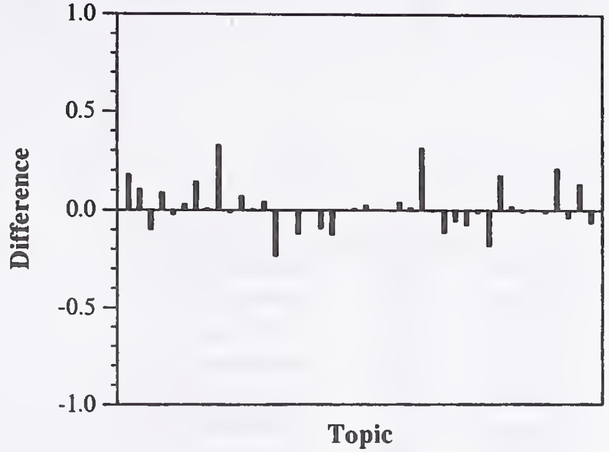


Difference from median in Document based R-precision per topic

Run ID	CLAISTDNG
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2274	0.0269
At 10 passages	0.2056	0.0373
At 15 passages	0.1901	0.0481
At 20 passages	0.1905	0.0537
At 30 passages	0.1919	0.0733
At 50 passages	0.1668	0.0703
At 100 passages	0.1546	0.0845
R-Precision	0.1737	

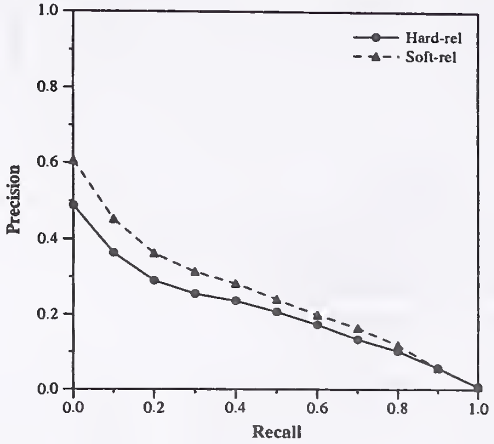


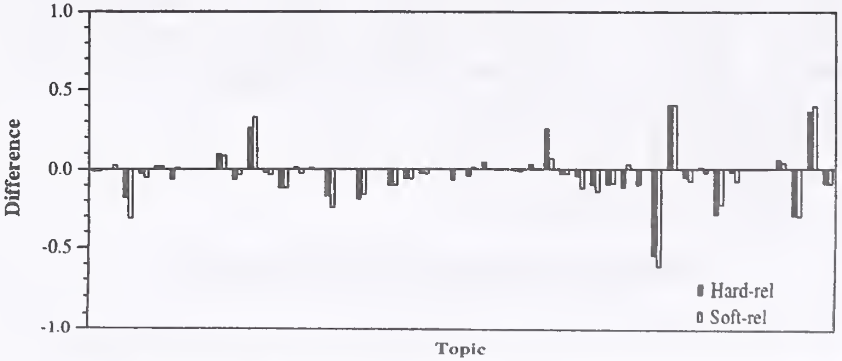
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3147	4384
MAP	0.1909	0.2345

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3000	0.4292
At 10 docs	0.2750	0.3917
At 15 docs	0.2667	0.3806
At 20 docs	0.2740	0.3750
At 30 docs	0.2722	0.3674
At 100 docs	0.2181	0.2985
At 200 docs	0.1736	0.2397
At 500 docs	0.1060	0.1456
At 1000 docs	0.0656	0.0913
R-Precision	0.2267	0.2712



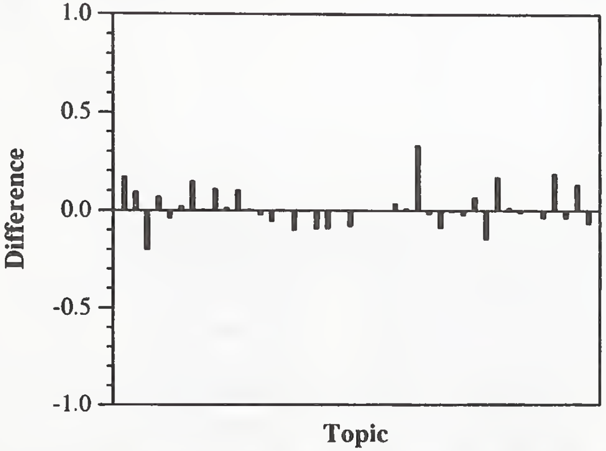


Difference from median in Document based R-precision per topic

Run ID	CLSTD630
Run Description	baseline run; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2308	0.0314
At 10 passages	0.2100	0.0439
At 15 passages	0.2038	0.0496
At 20 passages	0.1997	0.0543
At 30 passages	0.1938	0.0700
At 50 passages	0.1767	0.0749
At 100 passages	0.1461	0.0858
R-Precision	0.1726	

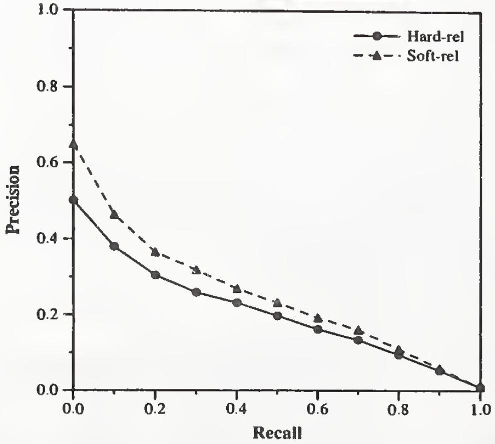


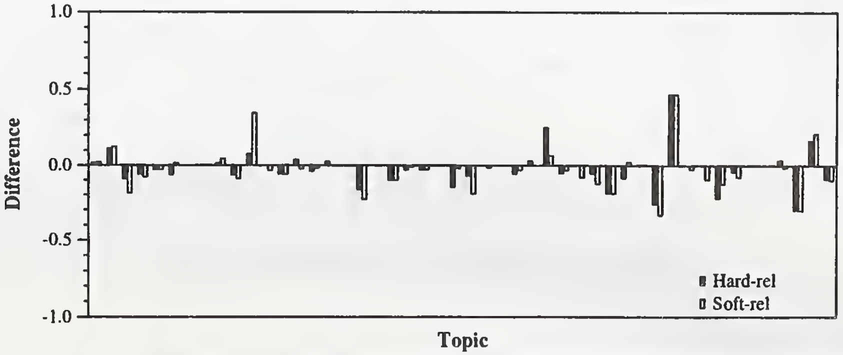
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	46450	46450
Total relevant	5123	7576
Total rel. ret.	3194	4386
MAP	0.1925	0.2341

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3042	0.4500
At 10 docs	0.2646	0.3938
At 15 docs	0.2708	0.3958
At 20 docs	0.2708	0.3844
At 30 docs	0.2597	0.3611
At 100 docs	0.2221	0.2981
At 200 docs	0.1720	0.2348
At 500 docs	0.1041	0.1414
At 1000 docs	0.0665	0.0914
R-Precision	0.2339	0.2772



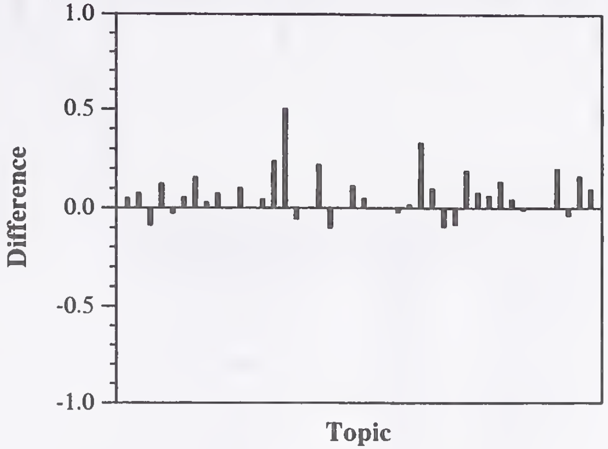


Difference from median in Document based R-precision per topic

Run ID	CLAI1G
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3350	0.0554
At 10 passages	0.3235	0.0654
At 15 passages	0.2781	0.0666
At 20 passages	0.2765	0.0750
At 30 passages	0.2554	0.0905
At 50 passages	0.2239	0.0956
At 100 passages	0.1789	0.1051
R-Precision	0.2426	

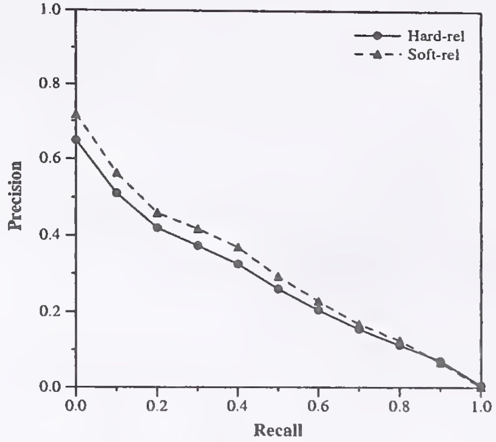


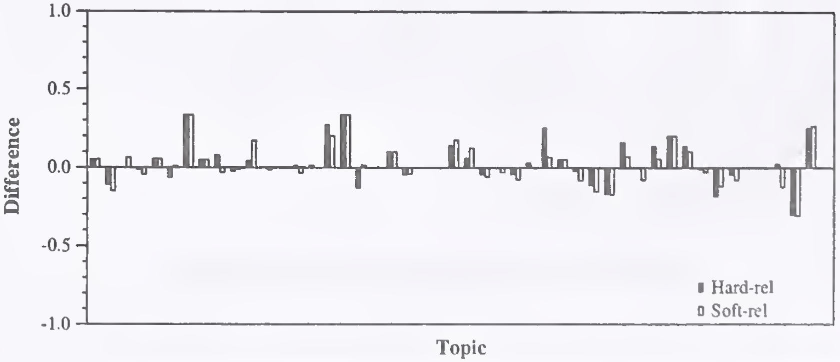
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3484	4783
MAP	0.2579	0.2884

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4208	0.5167
At 10 docs	0.3854	0.4729
At 15 docs	0.3611	0.4500
At 20 docs	0.3521	0.4396
At 30 docs	0.3319	0.4188
At 100 docs	0.2598	0.3358
At 200 docs	0.1978	0.2574
At 500 docs	0.1170	0.1547
At 1000 docs	0.0726	0.0996
R-Precision	0.2867	0.3229



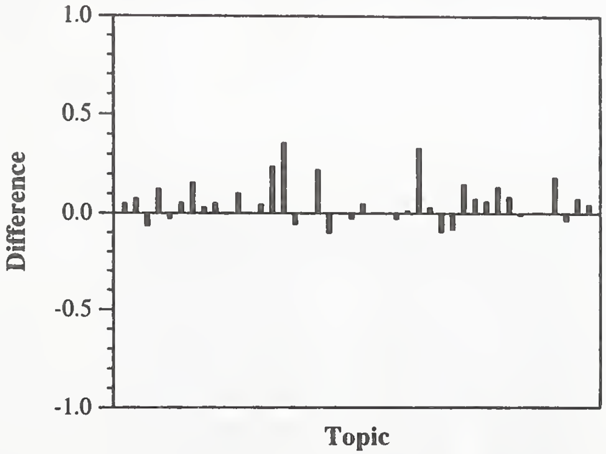


Difference from median in Document based R-precision per topic

Run ID	CLAI1NG
Run Description	not a baseline, used form; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3022	0.0537
At 10 passages	0.2886	0.0627
At 15 passages	0.2530	0.0654
At 20 passages	0.2538	0.0729
At 30 passages	0.2390	0.0851
At 50 passages	0.2077	0.0898
At 100 passages	0.1710	0.1038
R-Precision	0.2266	

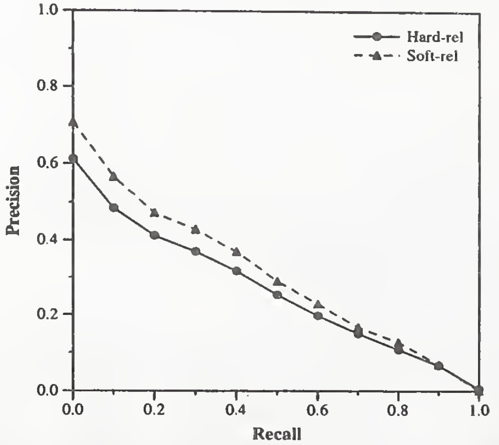


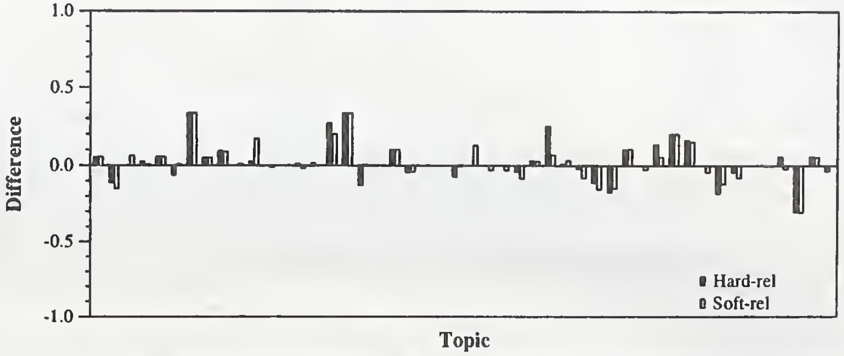
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3484	4783
MAP	0.2495	0.2917

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4042	0.5250
At 10 docs	0.3646	0.4729
At 15 docs	0.3431	0.4611
At 20 docs	0.3354	0.4490
At 30 docs	0.3160	0.4243
At 100 docs	0.2502	0.3390
At 200 docs	0.1945	0.2618
At 500 docs	0.1176	0.1590
At 1000 docs	0.0726	0.0996
R-Precision	0.2775	0.3246



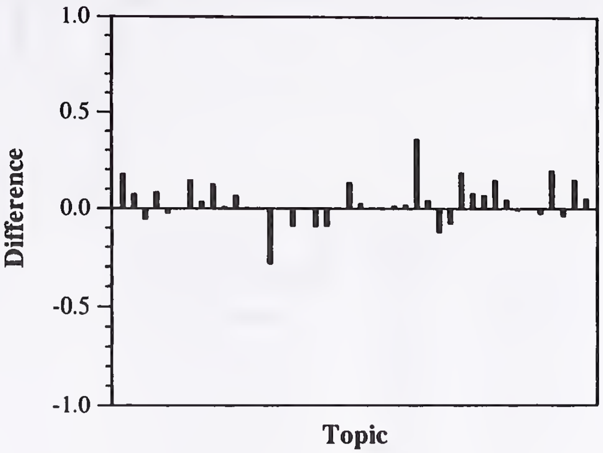


Difference from median in Document based R-precision per topic

Run ID	CLAI2G
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2757	0.0350
At 10 passages	0.2538	0.0485
At 15 passages	0.2351	0.0559
At 20 passages	0.2287	0.0631
At 30 passages	0.2379	0.0814
At 50 passages	0.2025	0.0862
At 100 passages	0.1615	0.0983
R-Precision	0.1900	

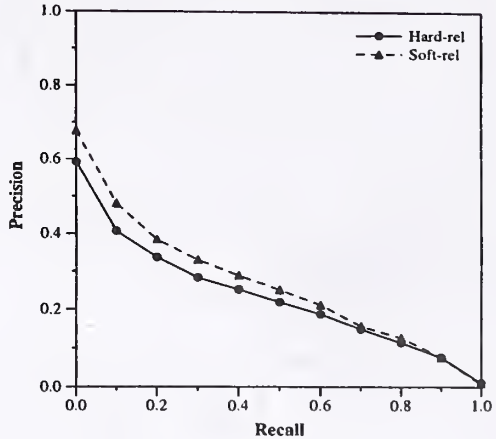


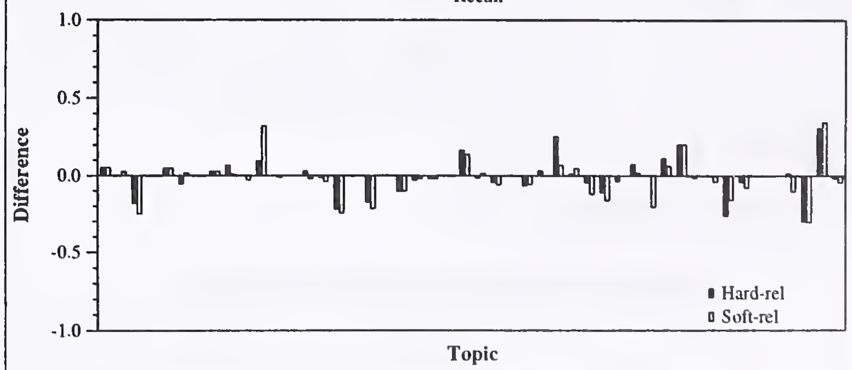
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3404	4652
MAP	0.2181	0.2499

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3375	0.4375
At 10 docs	0.3063	0.4188
At 15 docs	0.3111	0.4111
At 20 docs	0.3031	0.3896
At 30 docs	0.2986	0.3799
At 100 docs	0.2381	0.3108
At 200 docs	0.1914	0.2502
At 500 docs	0.1185	0.1562
At 1000 docs	0.0709	0.0969
R-Precision	0.2506	0.2861



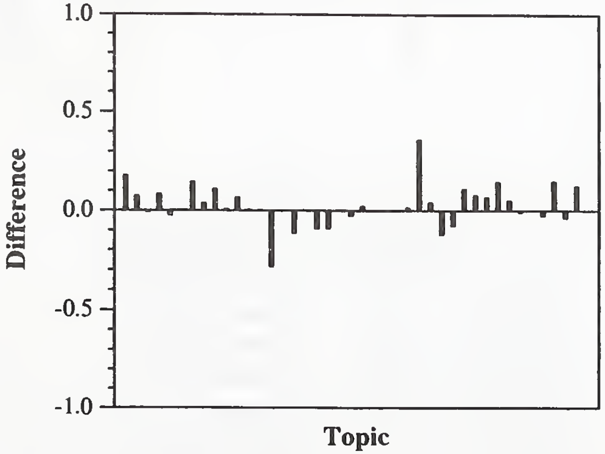


Difference from median in Document based R-precision per topic

Run ID	CLAI2NG
Run Description	not a baseline, used form; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2643	0.0348
At 10 passages	0.2201	0.0467
At 15 passages	0.2035	0.0534
At 20 passages	0.2040	0.0613
At 30 passages	0.2081	0.0781
At 50 passages	0.1879	0.0838
At 100 passages	0.1539	0.0940
R-Precision	0.1815	

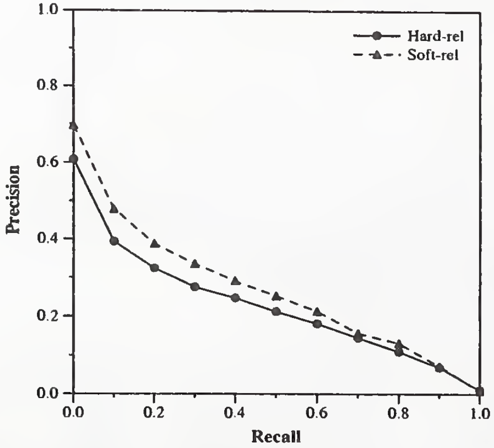


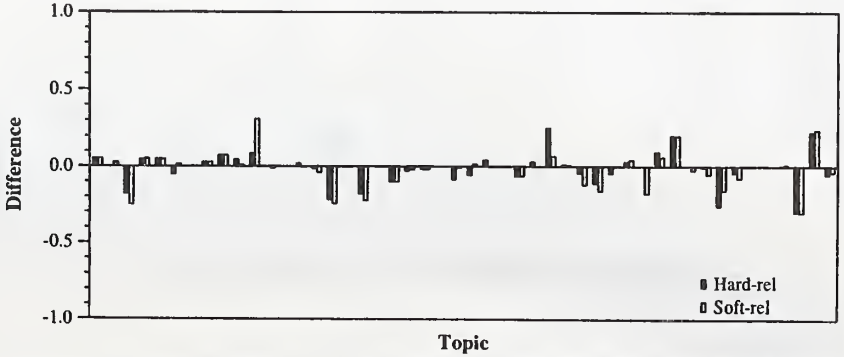
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3404	4652
MAP	0.2115	0.2514

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3458	0.4625
At 10 docs	0.2896	0.4146
At 15 docs	0.2944	0.4056
At 20 docs	0.2896	0.3875
At 30 docs	0.2813	0.3694
At 100 docs	0.2308	0.3142
At 200 docs	0.1843	0.2501
At 500 docs	0.1181	0.1588
At 1000 docs	0.0709	0.0969
R-Precision	0.2437	0.2870



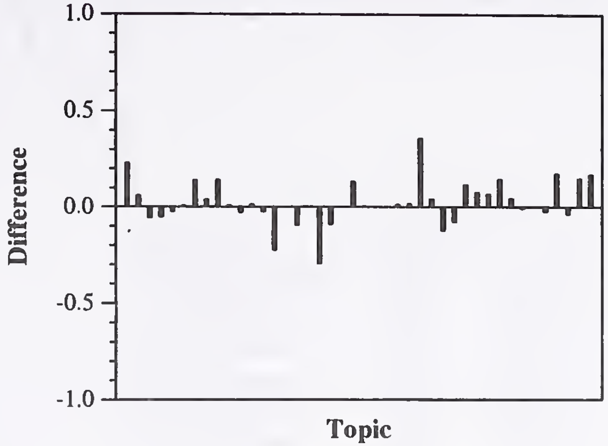


Difference from median in Document based R-precision per topic

Run ID	CLAI2RTG
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2487	0.0296
At 10 passages	0.2507	0.0412
At 15 passages	0.2227	0.0489
At 20 passages	0.2178	0.0585
At 30 passages	0.2273	0.0762
At 50 passages	0.2042	0.0859
At 100 passages	0.1601	0.0958
R-Precision	0.1773	

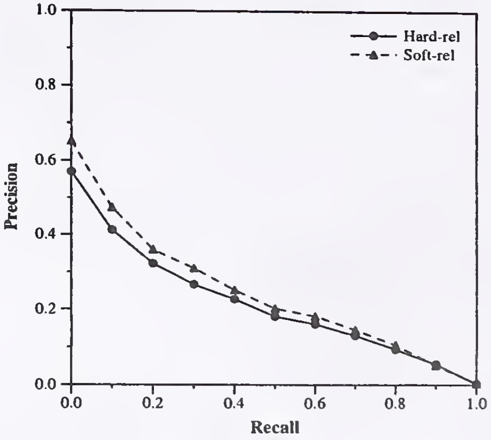


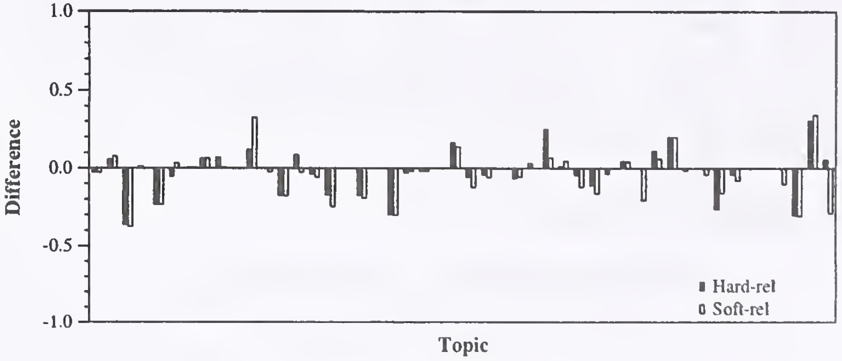
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3198	4372
MAP	0.1965	0.2218

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3167	0.4000
At 10 docs	0.3063	0.4125
At 15 docs	0.2903	0.3931
At 20 docs	0.2844	0.3792
At 30 docs	0.2847	0.3632
At 100 docs	0.2340	0.2971
At 200 docs	0.1841	0.2352
At 500 docs	0.1088	0.1424
At 1000 docs	0.0666	0.0911
R-Precision	0.2356	0.2634



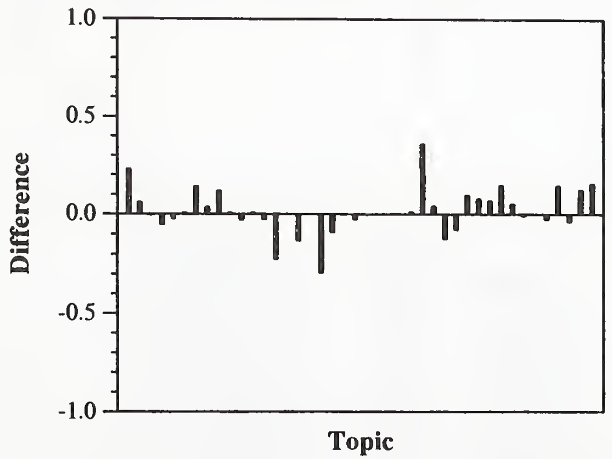


Difference from median in Document based R-precision per topic

Run ID	CLAI2RTNG
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2434	0.0296
At 10 passages	0.2176	0.0391
At 15 passages	0.1975	0.0471
At 20 passages	0.1936	0.0564
At 30 passages	0.2009	0.0728
At 50 passages	0.1847	0.0821
At 100 passages	0.1529	0.0913
R-Precision	0.1708	

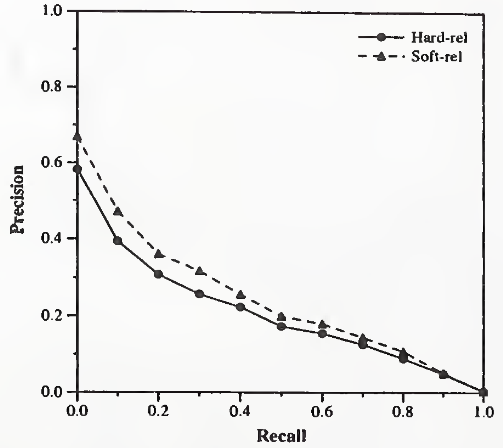


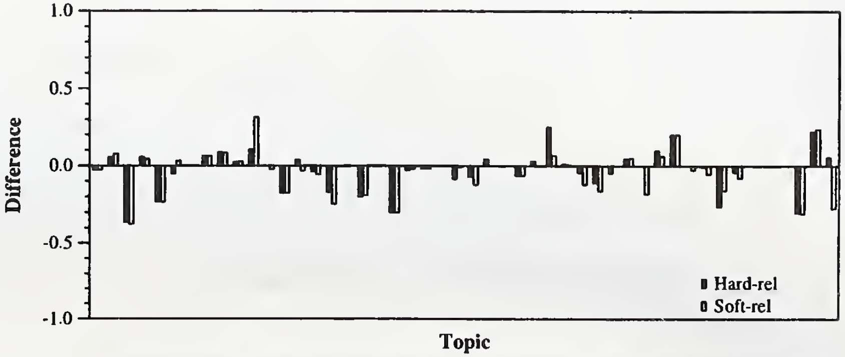
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3198	4372
MAP	0.1897	0.2225

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3292	0.4292
At 10 docs	0.2875	0.4083
At 15 docs	0.2792	0.3903
At 20 docs	0.2698	0.3740
At 30 docs	0.2694	0.3597
At 100 docs	0.2258	0.3000
At 200 docs	0.1764	0.2342
At 500 docs	0.1084	0.1449
At 1000 docs	0.0666	0.0911
R-Precision	0.2296	0.2633



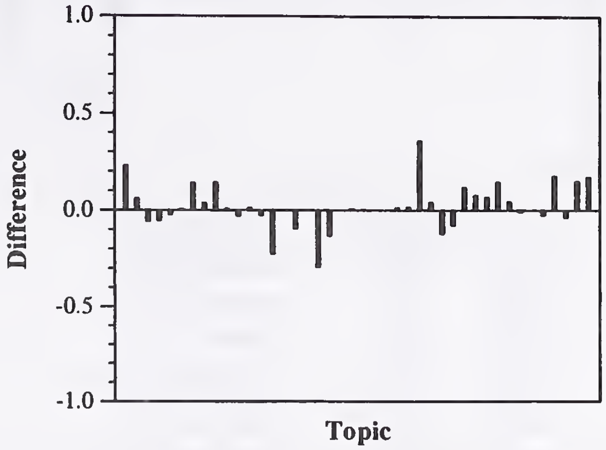


Difference from median in Document based R-precision per topic

Run ID	CLAI2WRTG
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2487	0.0296
At 10 passages	0.2483	0.0411
At 15 passages	0.2220	0.0489
At 20 passages	0.2212	0.0586
At 30 passages	0.2228	0.0761
At 50 passages	0.2022	0.0858
At 100 passages	0.1616	0.0960
R-Precision	0.1733	

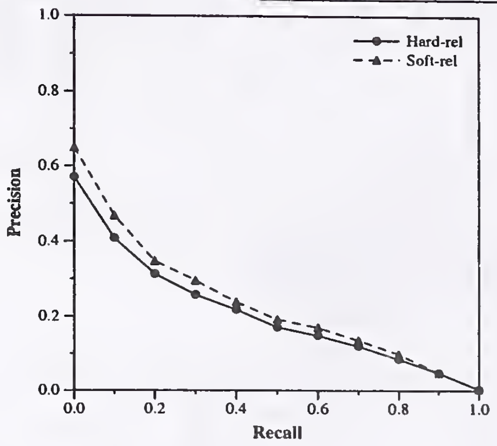


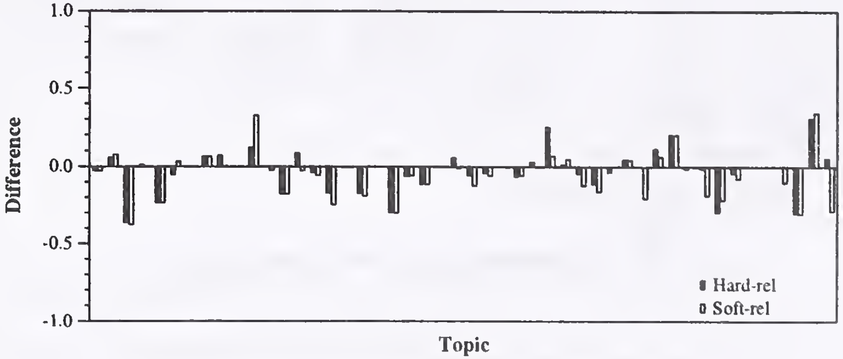
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3032	4082
MAP	0.1894	0.2138

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3250	0.4000
At 10 docs	0.3125	0.4104
At 15 docs	0.2931	0.3903
At 20 docs	0.2885	0.3813
At 30 docs	0.2840	0.3604
At 100 docs	0.2335	0.2965
At 200 docs	0.1796	0.2303
At 500 docs	0.1022	0.1328
At 1000 docs	0.0632	0.0850
R-Precision	0.2298	0.2533



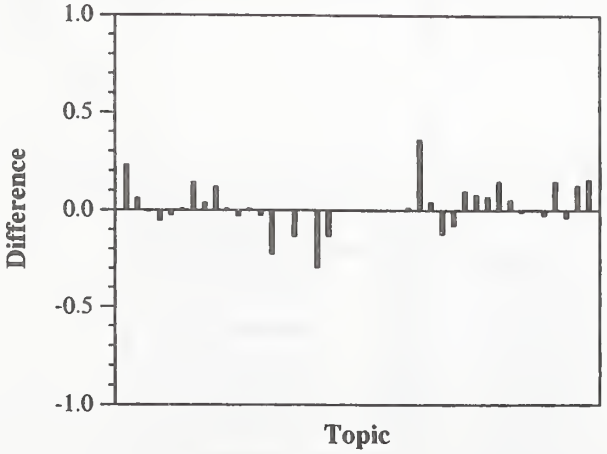


Difference from median in Document based R-precision per topic

Run ID	CLAI2WRTNG
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2434	0.0296
At 10 passages	0.2142	0.0391
At 15 passages	0.1952	0.0470
At 20 passages	0.1939	0.0564
At 30 passages	0.2009	0.0728
At 50 passages	0.1871	0.0821
At 100 passages	0.1537	0.0915
R-Precision	0.1704	

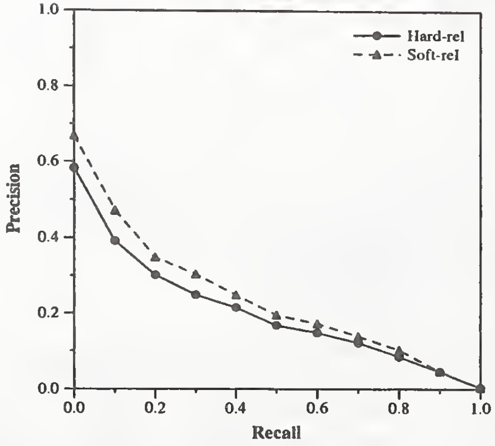


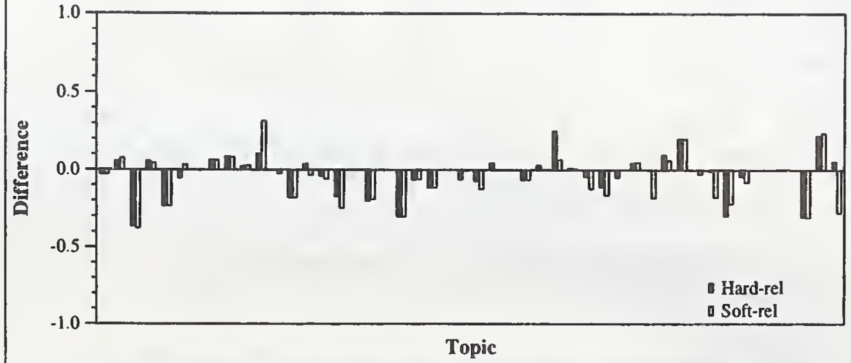
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3032	4082
MAP	0.1854	0.2170

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3375	0.4292
At 10 docs	0.2938	0.4063
At 15 docs	0.2819	0.3875
At 20 docs	0.2719	0.3740
At 30 docs	0.2715	0.3597
At 100 docs	0.2258	0.2996
At 200 docs	0.1751	0.2325
At 500 docs	0.1030	0.1372
At 1000 docs	0.0632	0.0850
R-Precision	0.2268	0.2569



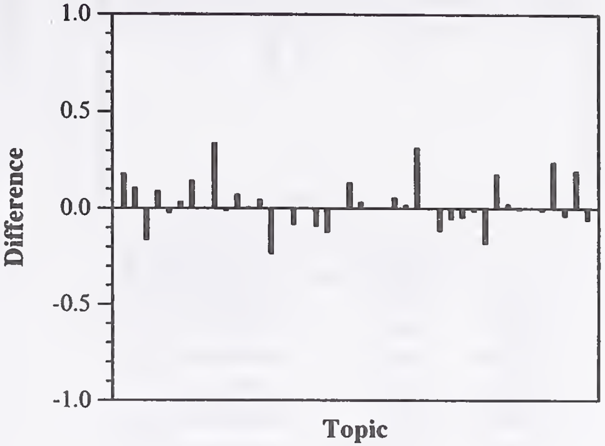


Difference from median in Document based R-precision per topic

Run ID	CLAISTDG
Run Description	not a baseline, used metadata; title only

Passages-based Evaluation (42 topics)

	Passage Level Averages	
	Precision	F Score
At 5 passages	0.2547	0.0285
At 10 passages	0.2230	0.0379
At 15 passages	0.2139	0.0501
At 20 passages	0.2129	0.0539
At 30 passages	0.2133	0.0738
At 50 passages	0.1838	0.0749
At 100 passages	0.1571	0.0877
R-Precision	0.1799	

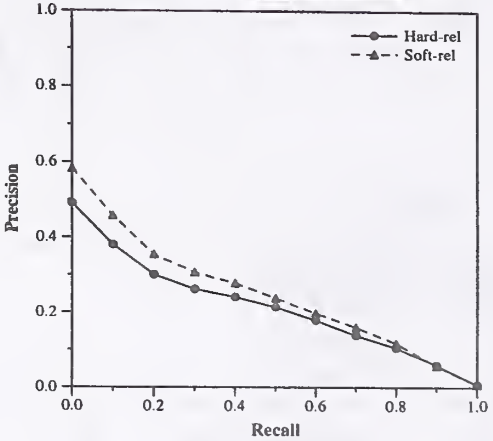


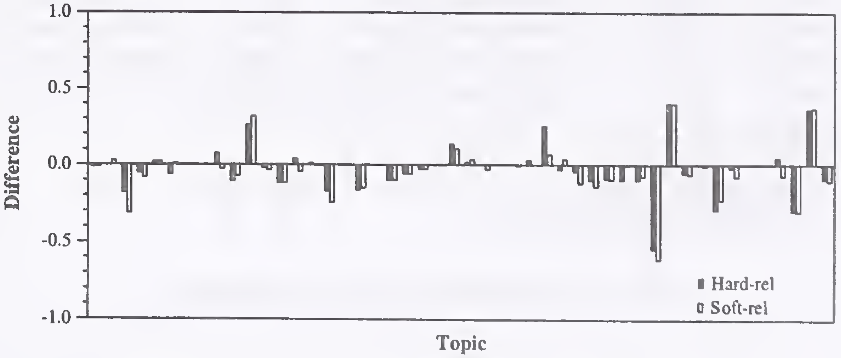
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3147	4384
MAP	0.1968	0.2309

	Document Level Averages	
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3167	0.4208
At 10 docs	0.2854	0.3750
At 15 docs	0.2847	0.3722
At 20 docs	0.2896	0.3667
At 30 docs	0.2840	0.3583
At 100 docs	0.2283	0.2973
At 200 docs	0.1764	0.2351
At 500 docs	0.1057	0.1410
At 1000 docs	0.0656	0.0913
R-Precision	0.2306	0.2658



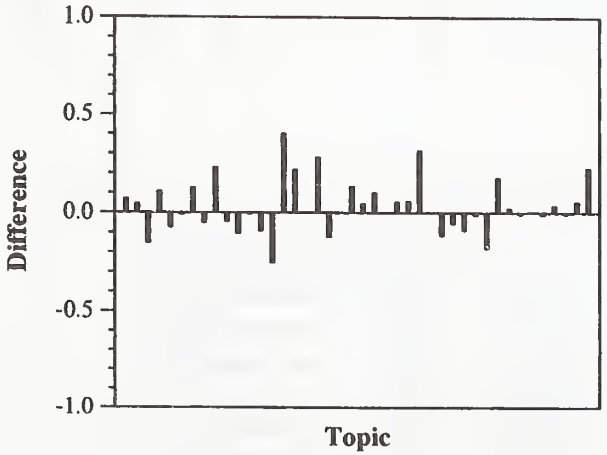


Difference from median in Document based R-precision per topic

Run ID	CLAISTDRTG
Run Description	not a baseline, used metadata; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2776	0.0495
At 10 passages	0.2610	0.0614
At 15 passages	0.2198	0.0614
At 20 passages	0.2271	0.0716
At 30 passages	0.2162	0.0826
At 50 passages	0.1903	0.0787
At 100 passages	0.1490	0.0858
R-Precision	0.2076	

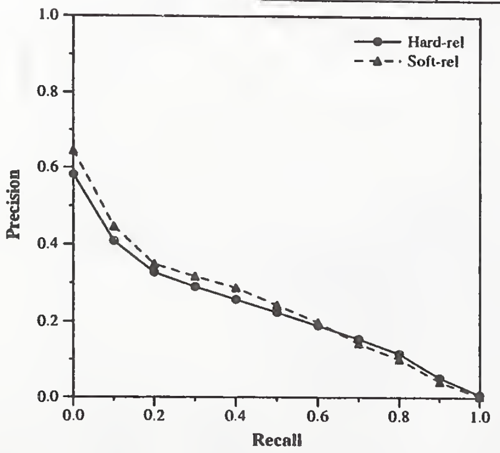


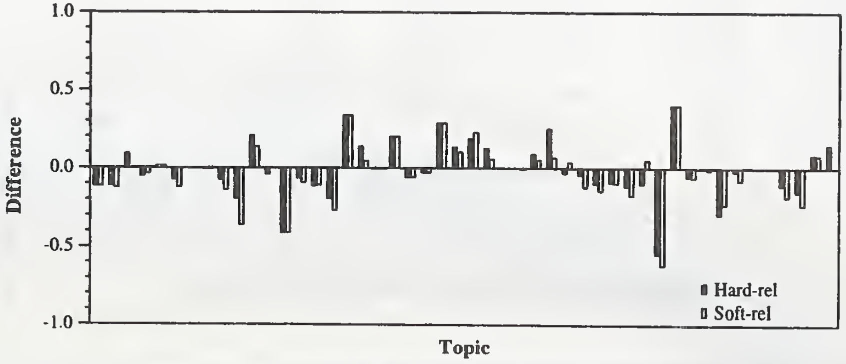
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	2981	4084
MAP	0.2200	0.2334

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3667	0.4250
At 10 docs	0.3396	0.3979
At 15 docs	0.3153	0.3722
At 20 docs	0.3198	0.3708
At 30 docs	0.2833	0.3319
At 100 docs	0.2037	0.2610
At 200 docs	0.1634	0.2114
At 500 docs	0.1011	0.1347
At 1000 docs	0.0621	0.0851
R-Precision	0.2454	0.2686



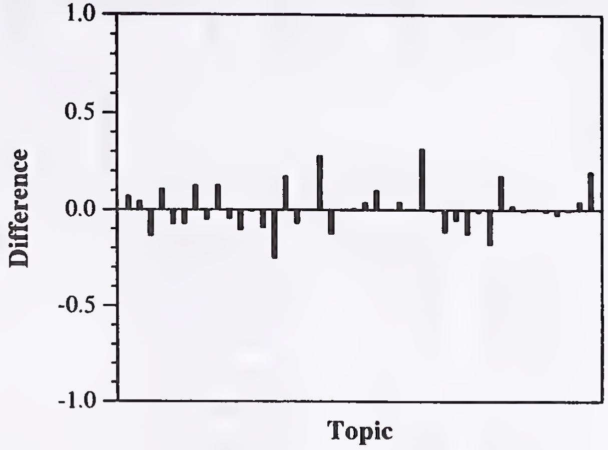


Difference from median in Document based R-precision per topic

Run ID	CLAISTDRTNG
Run Description	not a baseline, used metadata; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2326	0.0424
At 10 passages	0.2246	0.0560
At 15 passages	0.1863	0.0559
At 20 passages	0.1926	0.0646
At 30 passages	0.1833	0.0745
At 50 passages	0.1642	0.0718
At 100 passages	0.1372	0.0793
R-Precision	0.1782	

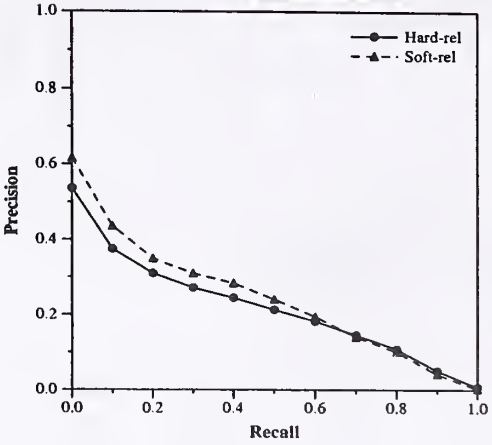


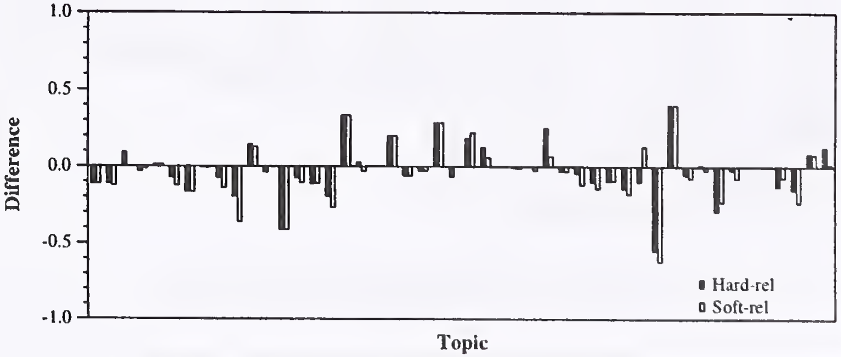
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	2981	4084
MAP	0.2055	0.2285

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3417	0.4375
At 10 docs	0.3146	0.4000
At 15 docs	0.2931	0.3667
At 20 docs	0.2969	0.3656
At 30 docs	0.2625	0.3250
At 100 docs	0.1965	0.2613
At 200 docs	0.1572	0.2077
At 500 docs	0.0997	0.1346
At 1000 docs	0.0621	0.0851
R-Precision	0.2310	0.2618



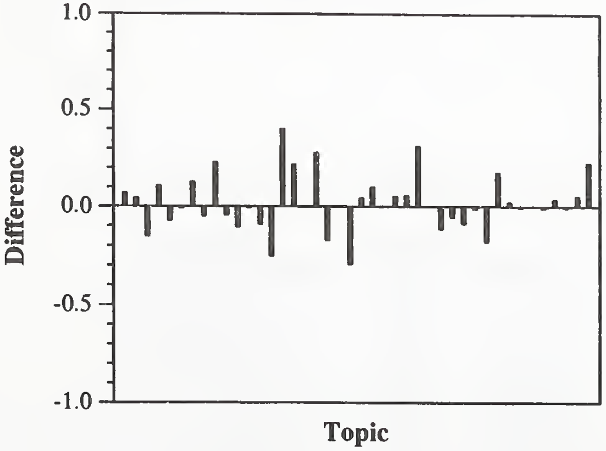


Difference from median in Document based R-precision per topic

Run ID	CLAISTDWRTG
Run Description	not a baseline, used metadata; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2847	0.0495
At 10 passages	0.2582	0.0613
At 15 passages	0.2131	0.0614
At 20 passages	0.2130	0.0714
At 30 passages	0.1973	0.0822
At 50 passages	0.1725	0.0780
At 100 passages	0.1355	0.0842
R-Precision	0.1963	

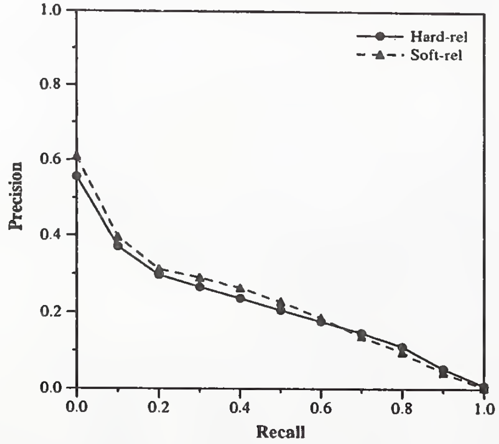


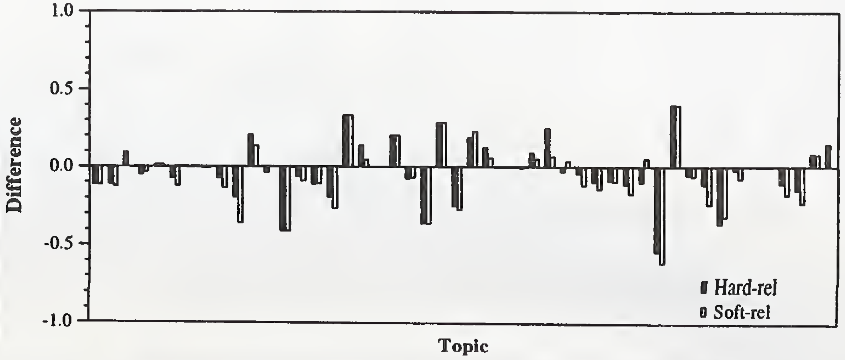
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	2640	3550
MAP	0.2025	0.2134

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3542	0.4042
At 10 docs	0.3208	0.3729
At 15 docs	0.2903	0.3361
At 20 docs	0.2854	0.3260
At 30 docs	0.2444	0.2826
At 100 docs	0.1733	0.2196
At 200 docs	0.1422	0.1829
At 500 docs	0.0880	0.1166
At 1000 docs	0.0550	0.0740
R-Precision	0.2260	0.2468



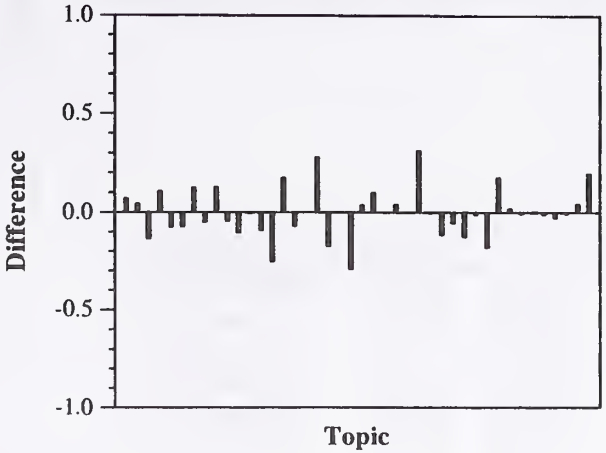


Difference from median in Document based R-precision per topic

Run ID	CLAISTDWRTNG
Run Description	not a baseline, used metadata; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2382	0.0424
At 10 passages	0.2243	0.0561
At 15 passages	0.1883	0.0561
At 20 passages	0.1902	0.0646
At 30 passages	0.1785	0.0744
At 50 passages	0.1558	0.0718
At 100 passages	0.1241	0.0782
R-Precision	0.1700	

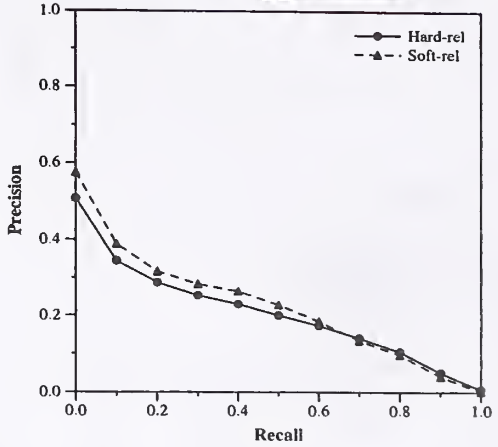


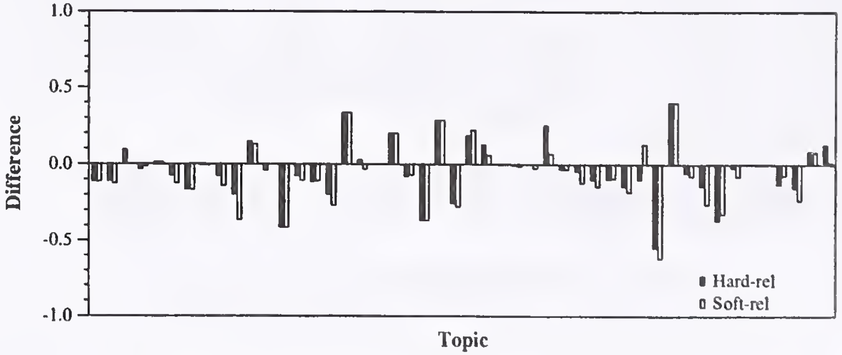
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	2640	3550
MAP	0.1925	0.2105

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3250	0.4083
At 10 docs	0.2917	0.3646
At 15 docs	0.2694	0.3278
At 20 docs	0.2677	0.3219
At 30 docs	0.2313	0.2799
At 100 docs	0.1675	0.2169
At 200 docs	0.1396	0.1810
At 500 docs	0.0873	0.1162
At 1000 docs	0.0550	0.0740
R-Precision	0.2150	0.2420



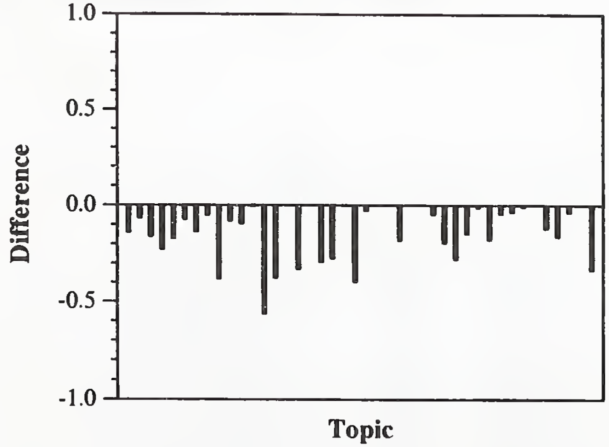


Difference from median in Document based R-precision per topic

Run ID	IITBHDBSLN1
Run Description	baseline run; description only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.0319	0.0064
At 10 passages	0.0305	0.0061
At 15 passages	0.0299	0.0068
At 20 passages	0.0293	0.0074
At 30 passages	0.0265	0.0084
At 50 passages	0.0228	0.0085
At 100 passages	0.0166	0.0082
R-Precision	0.0143	

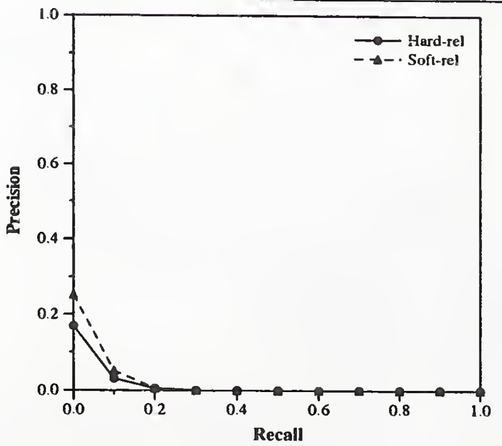


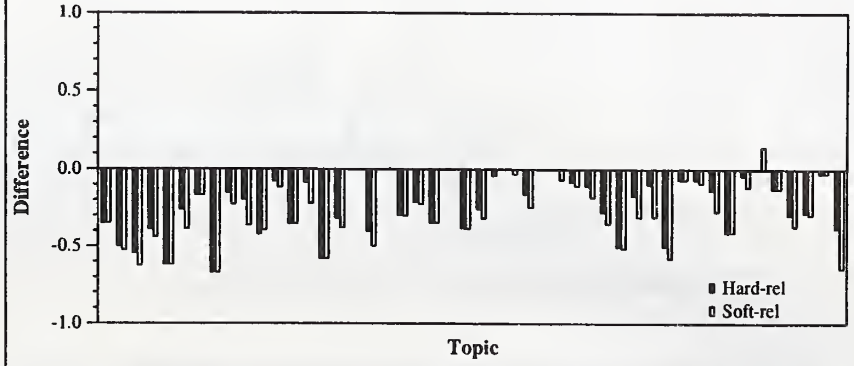
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	22892	22892
Total relevant	5123	7576
Total rel. ret.	145	205
MAP	0.0085	0.0122

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.0875	0.1208
At 10 docs	0.0750	0.0917
At 15 docs	0.0694	0.0847
At 20 docs	0.0615	0.0750
At 30 docs	0.0514	0.0646
At 100 docs	0.0244	0.0346
At 200 docs	0.0130	0.0184
At 500 docs	0.0060	0.0085
At 1000 docs	0.0030	0.0043
R-Precision	0.0194	0.0250



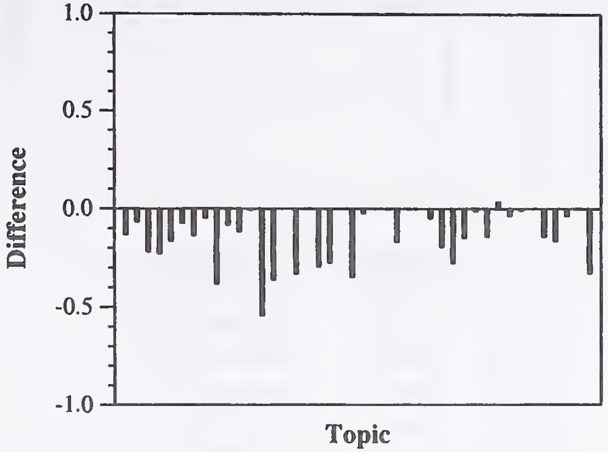


Difference from median in Document based R-precision per topic

Run ID	IITBHDBSLN2
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.0732	0.0120
At 10 passages	0.0679	0.0179
At 15 passages	0.0557	0.0194
At 20 passages	0.0503	0.0179
At 30 passages	0.0454	0.0194
At 50 passages	0.0359	0.0150
At 100 passages	0.0260	0.0125
R-Precision	0.0197	

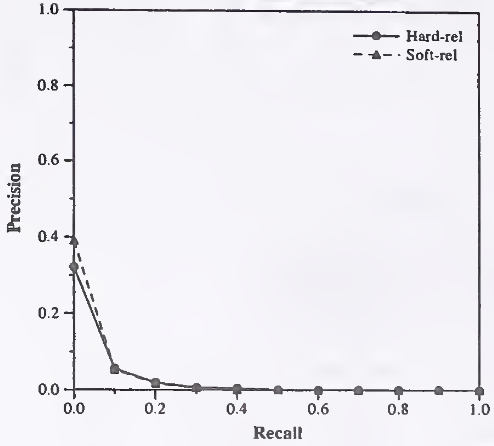


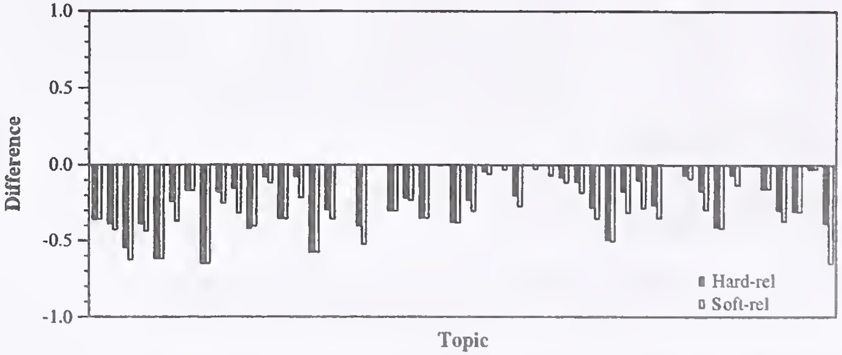
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	22862	22862
Total relevant	5123	7576
Total rel. ret.	186	224
MAP	0.0156	0.0160

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.1750	0.2250
At 10 docs	0.1333	0.1583
At 15 docs	0.1111	0.1347
At 20 docs	0.0969	0.1187
At 30 docs	0.0729	0.0931
At 100 docs	0.0277	0.0346
At 200 docs	0.0161	0.0197
At 500 docs	0.0078	0.0093
At 1000 docs	0.0039	0.0047
R-Precision	0.0281	0.0283



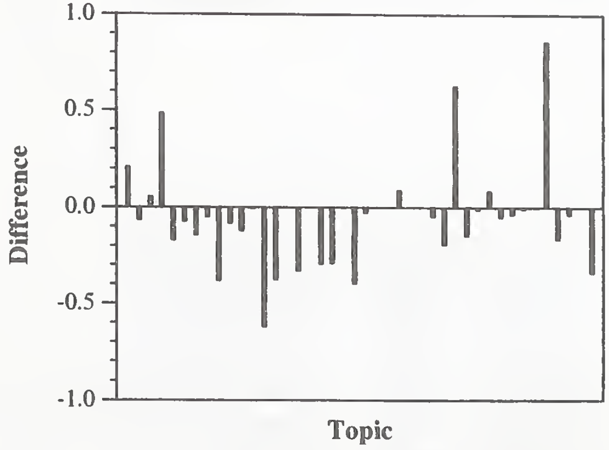


Difference from median in Document based R-precision per topic

Run ID	IITBHDF1
Run Description	not a baseline, used metadata; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.0974	0.0082
At 10 passages	0.0989	0.0097
At 15 passages	0.0989	0.0097
At 20 passages	0.0989	0.0097
At 30 passages	0.0989	0.0097
At 50 passages	0.0989	0.0097
At 100 passages	0.0989	0.0097
R-Precision	0.0989	

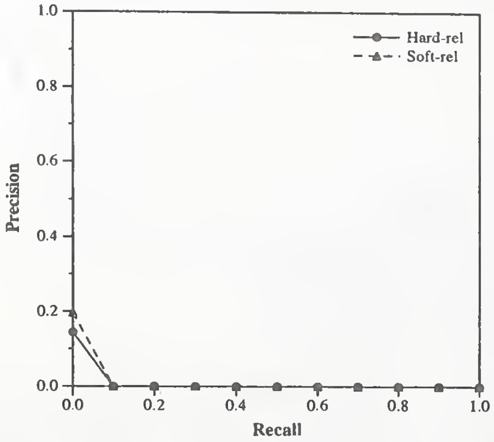


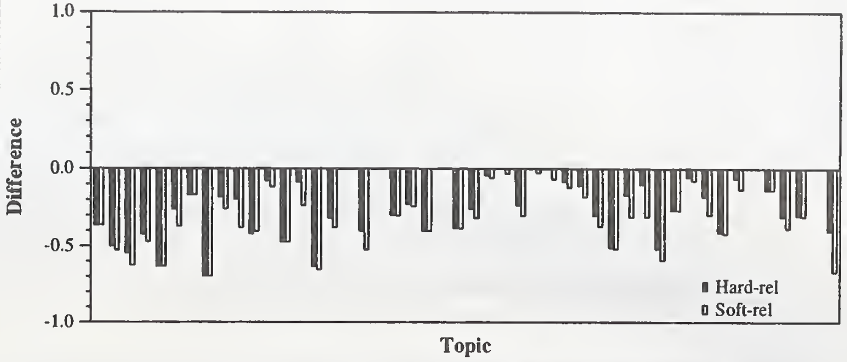
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	153	153
Total relevant	5123	7576
Total rel. ret.	22	29
MAP	0.0024	0.0026

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.0708	0.0958
At 10 docs	0.0458	0.0604
At 15 docs	0.0306	0.0403
At 20 docs	0.0229	0.0302
At 30 docs	0.0153	0.0201
At 100 docs	0.0046	0.0060
At 200 docs	0.0023	0.0030
At 500 docs	0.0009	0.0012
At 1000 docs	0.0005	0.0006
R-Precision	0.0038	0.0038



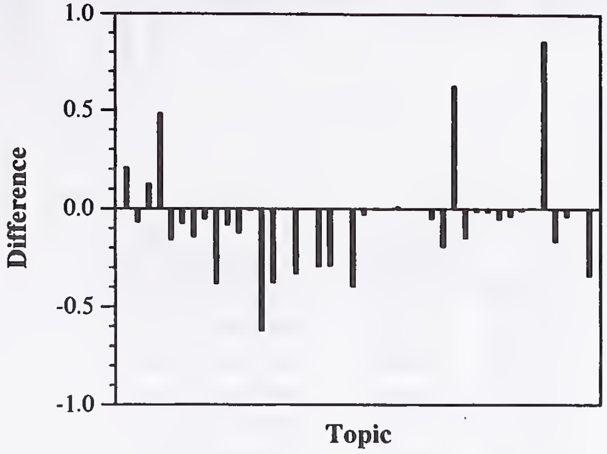


Difference from median in Document based R-precision per topic

Run ID	IITBHDF2
Run Description	not a baseline, used metadata; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.1036	0.0083
At 10 passages	0.0999	0.0089
At 15 passages	0.0966	0.0090
At 20 passages	0.0966	0.0090
At 30 passages	0.0966	0.0090
At 50 passages	0.0966	0.0090
At 100 passages	0.0966	0.0090
R-Precision	0.0966	

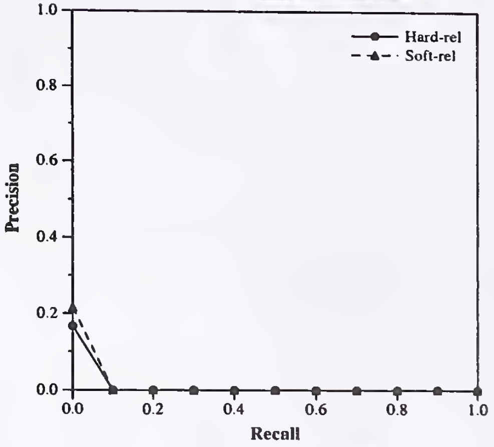


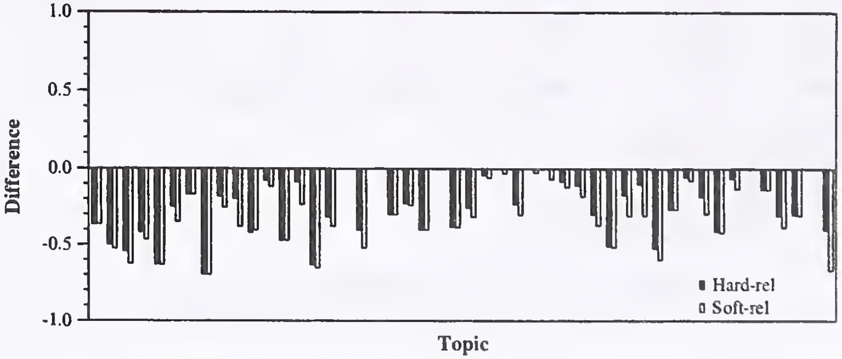
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	182	182
Total relevant	5123	7576
Total rel. ret.	25	34
MAP	0.0025	0.0032

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.0792	0.1083
At 10 docs	0.0500	0.0687
At 15 docs	0.0347	0.0472
At 20 docs	0.0260	0.0354
At 30 docs	0.0174	0.0236
At 100 docs	0.0052	0.0071
At 200 docs	0.0026	0.0035
At 500 docs	0.0010	0.0014
At 1000 docs	0.0005	0.0007
R-Precision	0.0040	0.0043



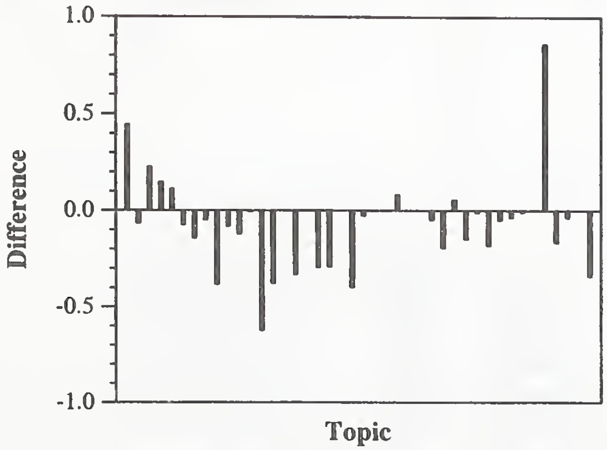


Difference from median in Document based R-precision per topic

Run ID	IITBHDF3
Run Description	not a baseline, used metadata; narrative only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.0966	0.0009
At 10 passages	0.0889	0.0010
At 15 passages	0.0874	0.0010
At 20 passages	0.0874	0.0010
At 30 passages	0.0874	0.0010
At 50 passages	0.0874	0.0010
At 100 passages	0.0874	0.0010
R-Precision	0.0874	

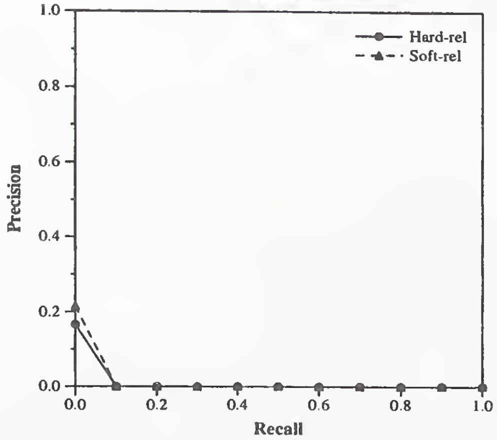


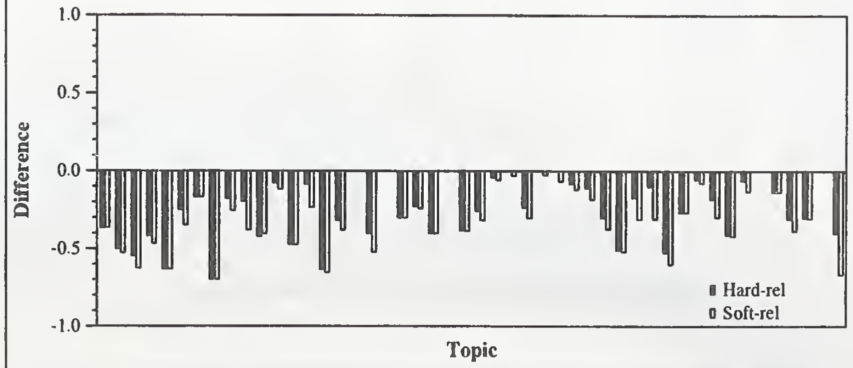
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	182	182
Total relevant	5123	7576
Total rel. ret.	25	34
MAP	0.0025	0.0032

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.0792	0.1083
At 10 docs	0.0500	0.0687
At 15 docs	0.0347	0.0472
At 20 docs	0.0260	0.0354
At 30 docs	0.0174	0.0236
At 100 docs	0.0052	0.0071
At 200 docs	0.0026	0.0035
At 500 docs	0.0010	0.0014
At 1000 docs	0.0005	0.0007
R-Precision	0.0040	0.0043



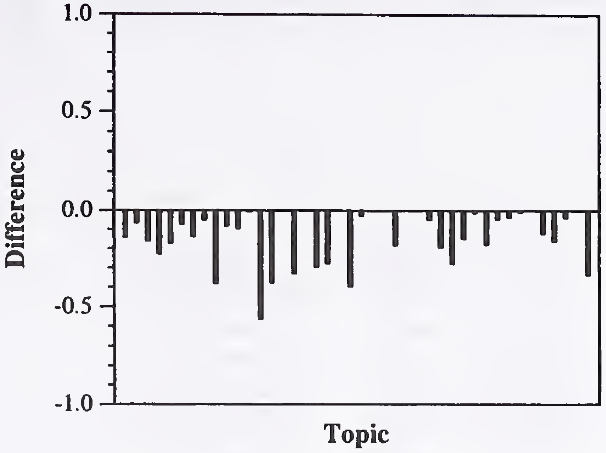


Difference from median in Document based R-precision per topic

Run ID	IITBHDFTEMP
Run Description	not a baseline, used metadata; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.0319	0.0064
At 10 passages	0.0305	0.0061
At 15 passages	0.0299	0.0068
At 20 passages	0.0293	0.0074
At 30 passages	0.0265	0.0084
At 50 passages	0.0228	0.0085
At 100 passages	0.0166	0.0082
R-Precision	0.0143	

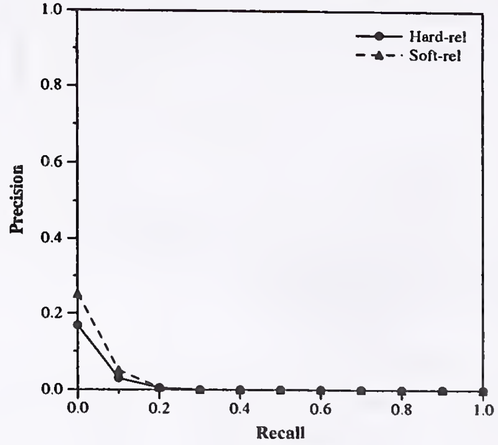


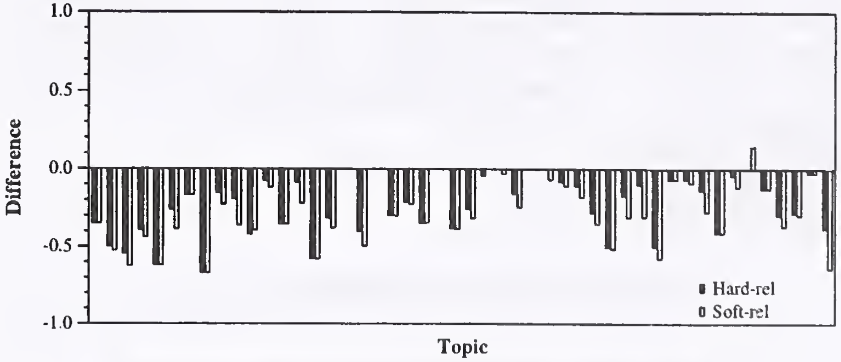
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	22892	22892
Total relevant	5123	7576
Total rel. ret.	145	205
MAP	0.0085	0.0122

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.0875	0.1208
At 10 docs	0.0750	0.0917
At 15 docs	0.0694	0.0847
At 20 docs	0.0615	0.0750
At 30 docs	0.0514	0.0646
At 100 docs	0.0244	0.0346
At 200 docs	0.0130	0.0184
At 500 docs	0.0060	0.0085
At 1000 docs	0.0030	0.0043
R-Precision	0.0194	0.0250



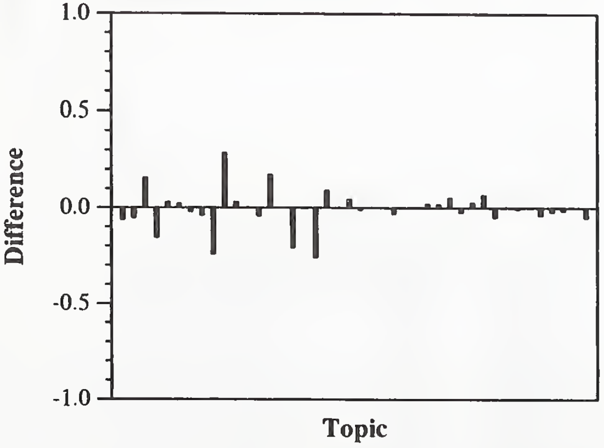


Difference from median in Document based R-precision per topic

Run ID	MSRCbase
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2706	0.0555
At 10 passages	0.2364	0.0700
At 15 passages	0.2104	0.0844
At 20 passages	0.1976	0.0914
At 30 passages	0.1775	0.0989
At 50 passages	0.1693	0.1167
At 100 passages	0.1505	0.1430
R-Precision	0.1424	

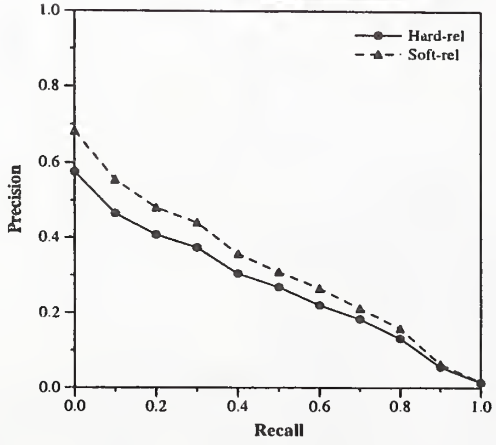


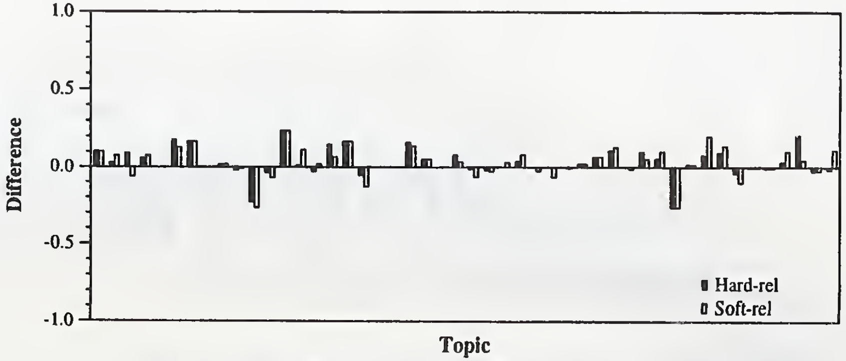
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3633	5118
MAP	0.2555	0.3051

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3667	0.5042
At 10 docs	0.3750	0.4813
At 15 docs	0.3736	0.4792
At 20 docs	0.3625	0.4667
At 30 docs	0.3382	0.4340
At 100 docs	0.2725	0.3577
At 200 docs	0.2101	0.2821
At 500 docs	0.1260	0.1729
At 1000 docs	0.0757	0.1066
R-Precision	0.2881	0.3326



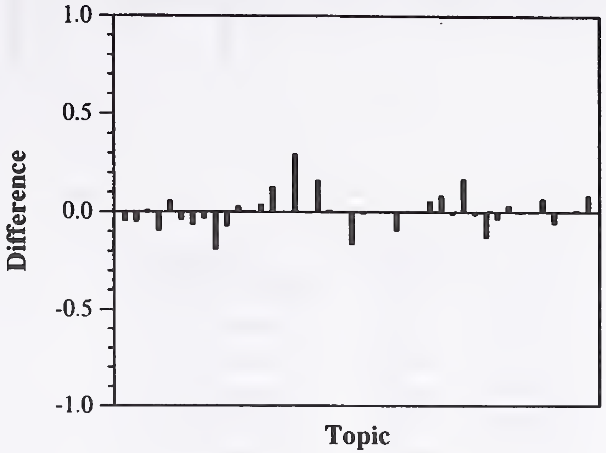


Difference from median in Document based R-precision per topic

Run ID	MSRCs1e0p0B
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2418	0.0425
At 10 passages	0.2479	0.0713
At 15 passages	0.2227	0.0719
At 20 passages	0.2203	0.0856
At 30 passages	0.2131	0.1029
At 50 passages	0.1853	0.1177
At 100 passages	0.1486	0.1365
R-Precision	0.1621	

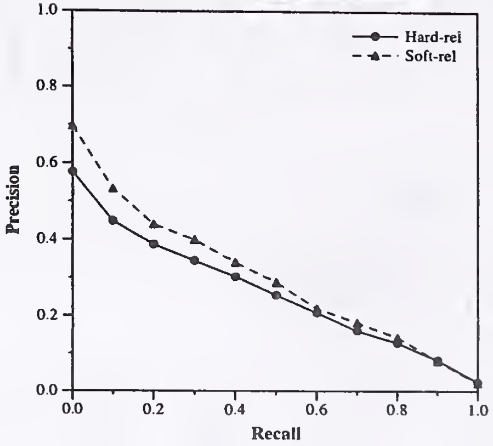


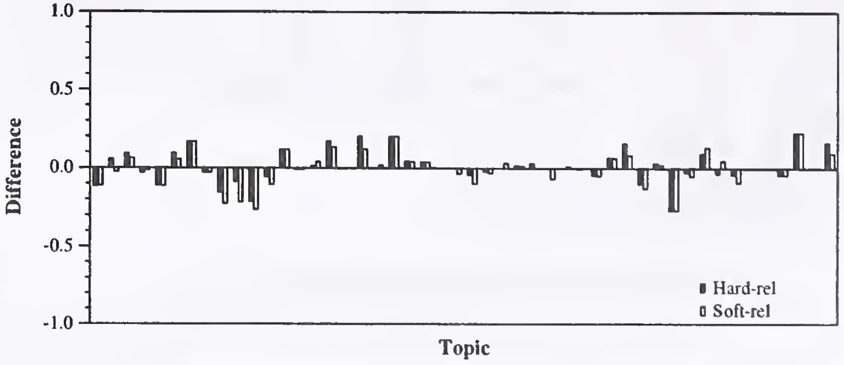
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	46818	46818
Total relevant	5123	7576
Total rel. ret.	3249	4484
MAP	0.2490	0.2851

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3875	0.5083
At 10 docs	0.3875	0.4896
At 15 docs	0.3625	0.4639
At 20 docs	0.3552	0.4531
At 30 docs	0.3347	0.4326
At 100 docs	0.2483	0.3248
At 200 docs	0.1890	0.2514
At 500 docs	0.1105	0.1507
At 1000 docs	0.0677	0.0934
R-Precision	0.2673	0.2994



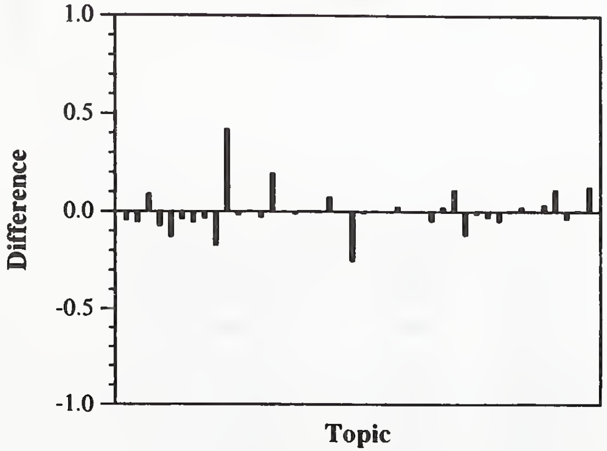


Difference from median in Document based R-precision per topic

Run ID	MSRCs1e0p0
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2423	0.0351
At 10 passages	0.2204	0.0609
At 15 passages	0.2202	0.0672
At 20 passages	0.2100	0.0745
At 30 passages	0.1925	0.0909
At 50 passages	0.1809	0.1103
At 100 passages	0.1485	0.1211
R-Precision	0.1553	

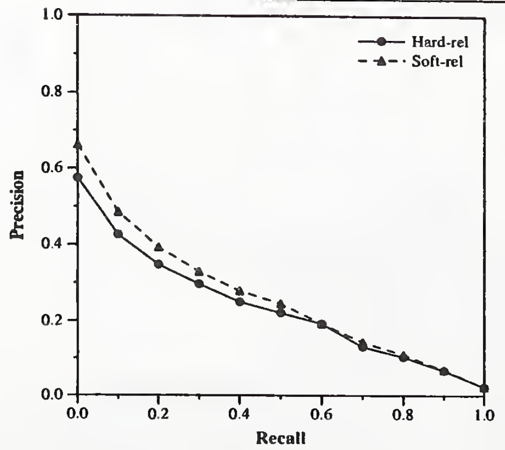


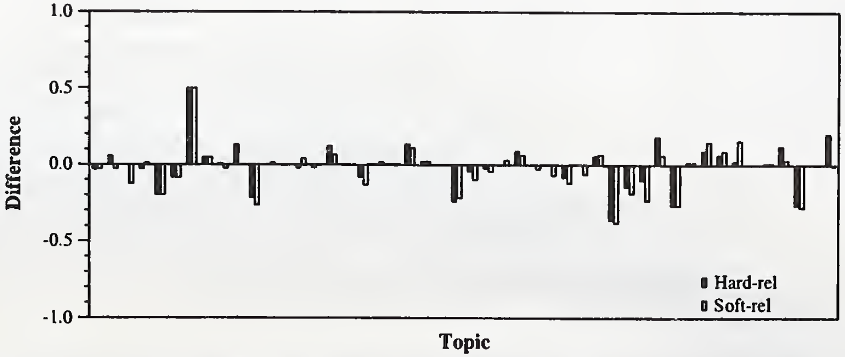
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	47599	47599
Total relevant	5123	7576
Total rel. ret.	3303	4530
MAP	0.2167	0.2422

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3750	0.4708
At 10 docs	0.3729	0.4521
At 15 docs	0.3611	0.4403
At 20 docs	0.3438	0.4219
At 30 docs	0.3326	0.4021
At 100 docs	0.2329	0.2992
At 200 docs	0.1740	0.2259
At 500 docs	0.1084	0.1445
At 1000 docs	0.0688	0.0944
R-Precision	0.2493	0.2766



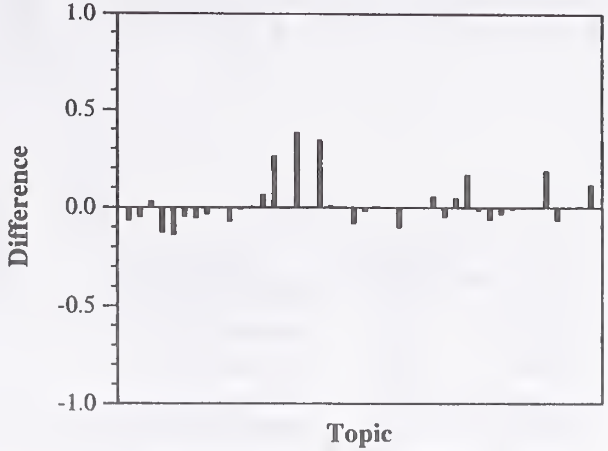


Difference from median in Document based R-precision per topic

Run ID	MSRCs1e0p1B
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

	Passage Level Averages	
	Precision	F Score
At 5 passages	0.2570	0.0431
At 10 passages	0.2544	0.0811
At 15 passages	0.2349	0.0747
At 20 passages	0.2324	0.0878
At 30 passages	0.2182	0.1055
At 50 passages	0.1967	0.1236
At 100 passages	0.1563	0.1394
R-Precision	0.1798	

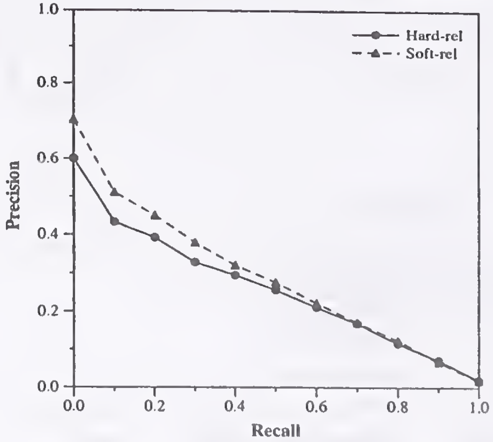


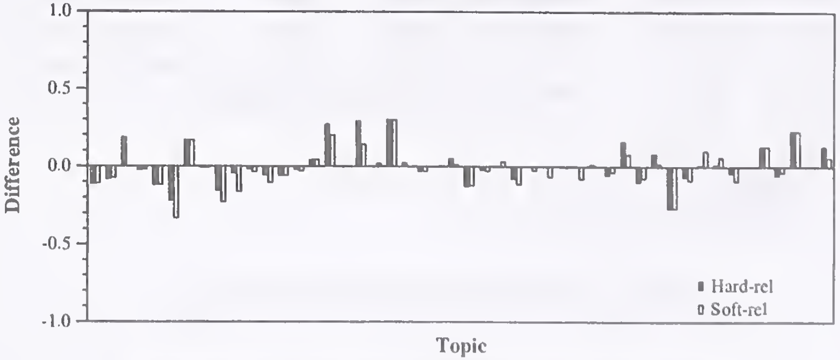
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	46816	46816
Total relevant	5123	7576
Total rel. ret.	3267	4508
MAP	0.2429	0.2706

	Document Level Averages	
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4083	0.4917
At 10 docs	0.3854	0.4625
At 15 docs	0.3708	0.4472
At 20 docs	0.3542	0.4281
At 30 docs	0.3354	0.4097
At 100 docs	0.2321	0.2973
At 200 docs	0.1825	0.2391
At 500 docs	0.1090	0.1477
At 1000 docs	0.0681	0.0939
R-Precision	0.2624	0.2885



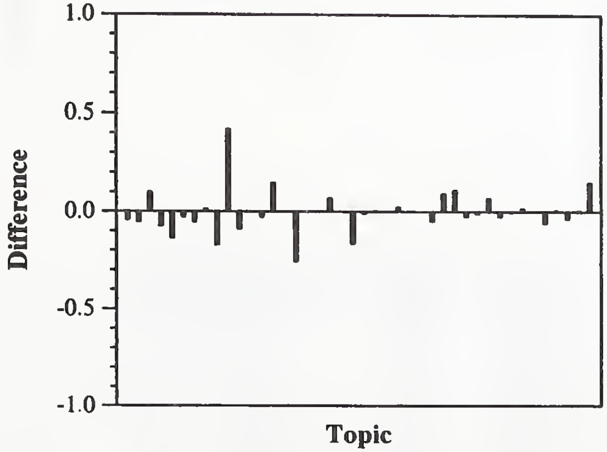


. Difference from median in Document based R-precision per topic

Run ID	MSRCs1e0p1
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2710	0.0387
At 10 passages	0.2609	0.0661
At 15 passages	0.2307	0.0672
At 20 passages	0.2264	0.0806
At 30 passages	0.2059	0.1000
At 50 passages	0.1917	0.1175
At 100 passages	0.1432	0.1196
R-Precision	0.1526	

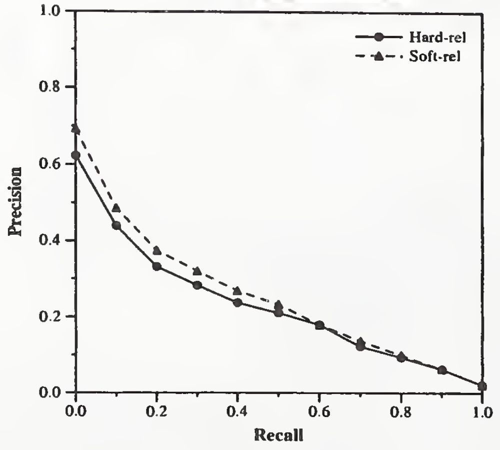


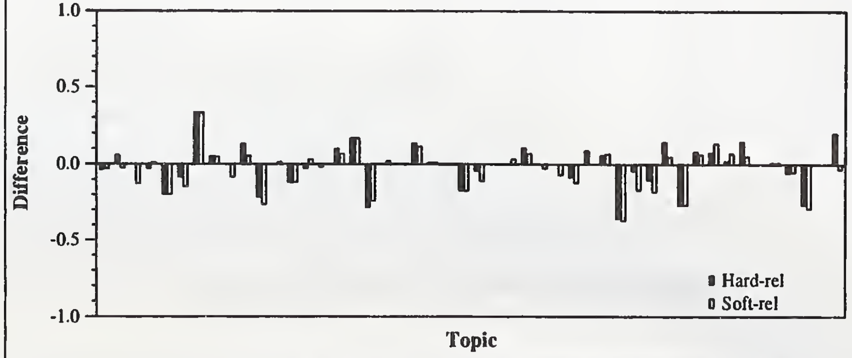
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	47683	47683
Total relevant	5123	7576
Total rel. ret.	3290	4508
MAP	0.2115	0.2337

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3708	0.4375
At 10 docs	0.3729	0.4333
At 15 docs	0.3528	0.4153
At 20 docs	0.3542	0.4177
At 30 docs	0.3403	0.4007
At 100 docs	0.2323	0.2933
At 200 docs	0.1741	0.2245
At 500 docs	0.1071	0.1416
At 1000 docs	0.0685	0.0939
R-Precision	0.2462	0.2703



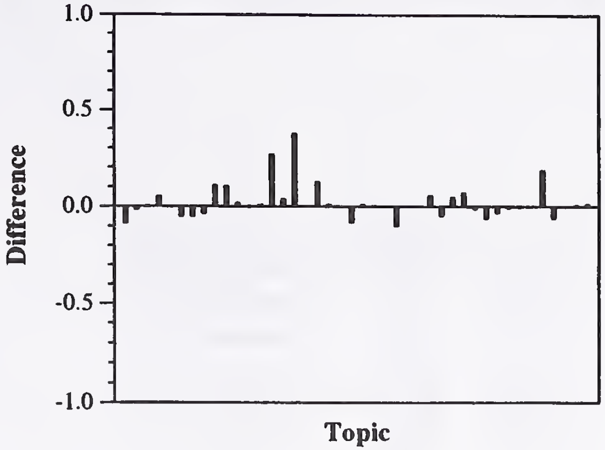


Difference from median in Document based R-precision per topic

Run ID	MSRCs1e1p1B
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3098	0.0568
At 10 passages	0.2843	0.0834
At 15 passages	0.2587	0.0830
At 20 passages	0.2467	0.0945
At 30 passages	0.2375	0.1161
At 50 passages	0.2109	0.1336
At 100 passages	0.1669	0.1499
R-Precision	0.1794	

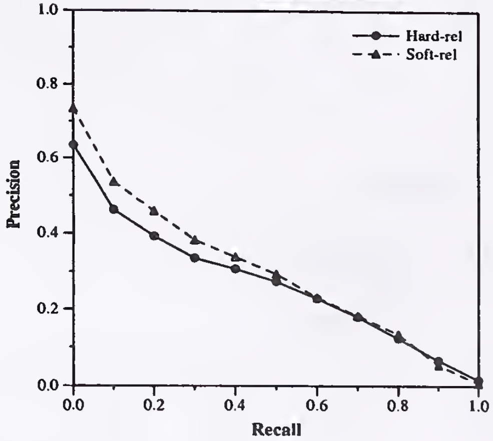


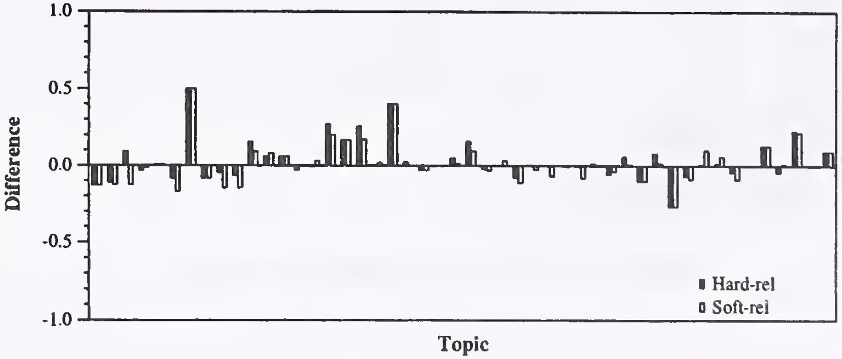
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	47141	47141
Total relevant	5123	7576
Total rel. ret.	3379	4654
MAP	0.2558	0.2841

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4542	0.5500
At 10 docs	0.4167	0.4958
At 15 docs	0.3903	0.4708
At 20 docs	0.3708	0.4500
At 30 docs	0.3576	0.4326
At 100 docs	0.2467	0.3167
At 200 docs	0.1910	0.2503
At 500 docs	0.1132	0.1531
At 1000 docs	0.0704	0.0970
R-Precision	0.2874	0.3177



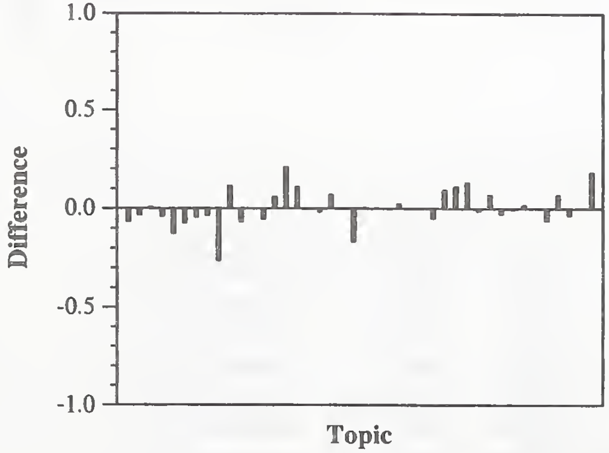


Difference from median in Document based R-precision per topic

Run ID	MSRCs1e1p1
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2821	0.0502
At 10 passages	0.2715	0.0733
At 15 passages	0.2619	0.0830
At 20 passages	0.2492	0.0971
At 30 passages	0.2285	0.1162
At 50 passages	0.1923	0.1286
At 100 passages	0.1414	0.1318
R-Precision	0.1620	

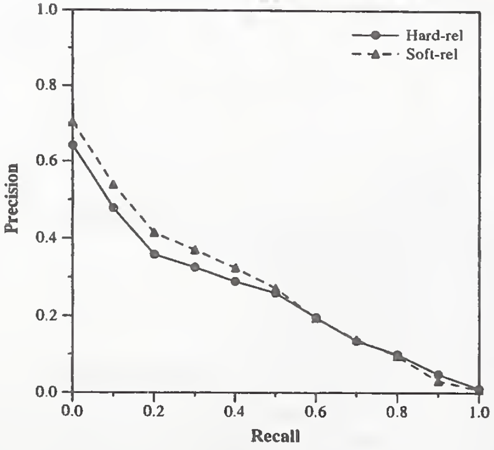


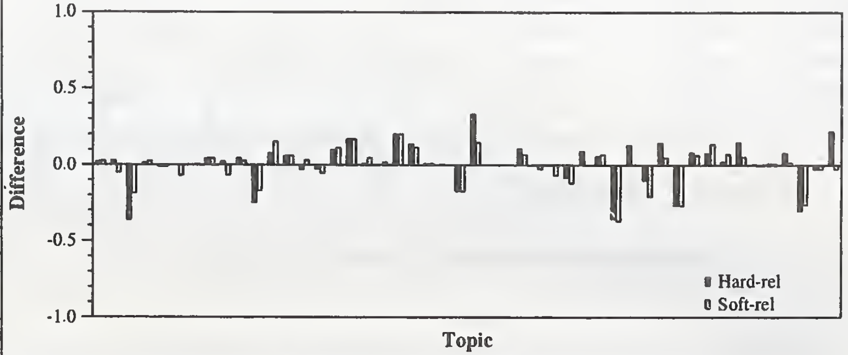
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	47767	47767
Total relevant	5123	7576
Total rel. ret.	3385	4605
MAP	0.2370	0.2610

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4000	0.4875
At 10 docs	0.4000	0.4833
At 15 docs	0.3958	0.4694
At 20 docs	0.3885	0.4688
At 30 docs	0.3694	0.4465
At 100 docs	0.2615	0.3360
At 200 docs	0.1882	0.2482
At 500 docs	0.1125	0.1498
At 1000 docs	0.0705	0.0959
R-Precision	0.2648	0.2956



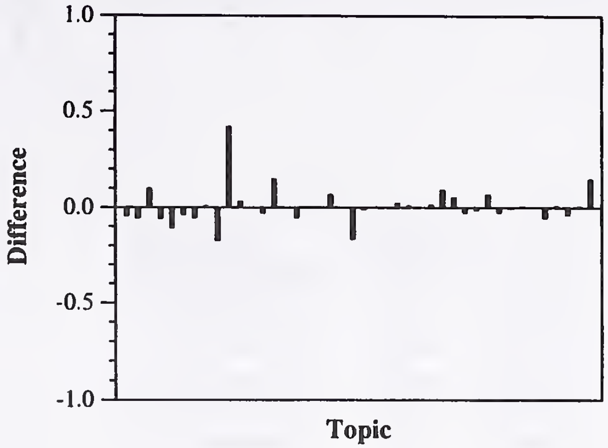


Difference from median in Document based R-precision per topic

Run ID	MSRCs2e0p1
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2888	0.0468
At 10 passages	0.2654	0.0719
At 15 passages	0.2363	0.0737
At 20 passages	0.2242	0.0829
At 30 passages	0.1961	0.0976
At 50 passages	0.1843	0.1114
At 100 passages	0.1501	0.1240
R-Precision	0.1611	

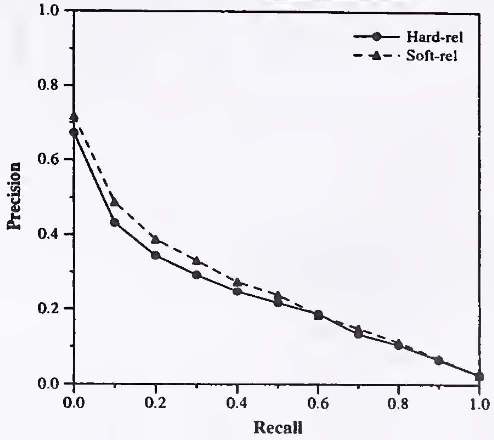


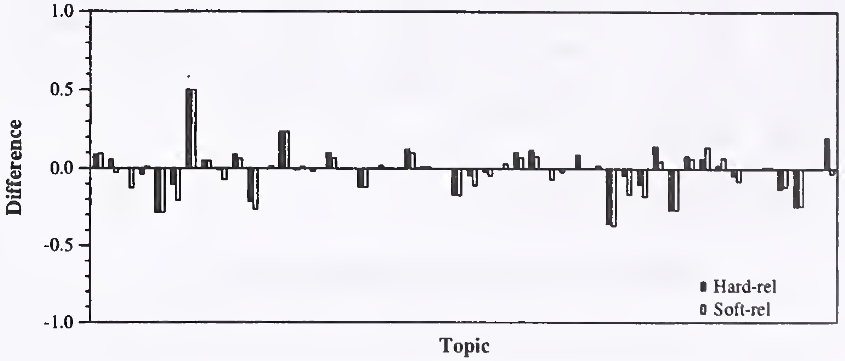
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	47659	47659
Total relevant	5123	7576
Total rel. ret.	3288	4502
MAP	0.2201	0.2423

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4042	0.4833
At 10 docs	0.3896	0.4521
At 15 docs	0.3597	0.4222
At 20 docs	0.3458	0.4135
At 30 docs	0.3285	0.3958
At 100 docs	0.2304	0.2923
At 200 docs	0.1724	0.2246
At 500 docs	0.1083	0.1431
At 1000 docs	0.0685	0.0938
R-Precision	0.2529	0.2789



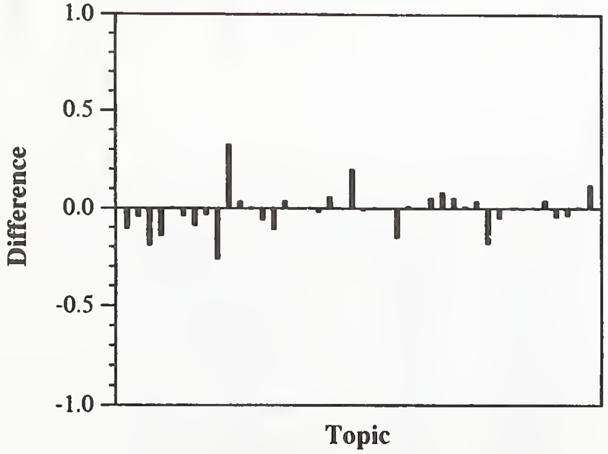


Difference from median in Document based R-precision per topic

Run ID	MSRCs9e1p0
Run Description	not a baseline, used metadata; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2751	0.0483
At 10 passages	0.2410	0.0669
At 15 passages	0.2302	0.0750
At 20 passages	0.2044	0.0797
At 30 passages	0.2004	0.0965
At 50 passages	0.1724	0.1092
At 100 passages	0.1407	0.1250
R-Precision	0.1445	

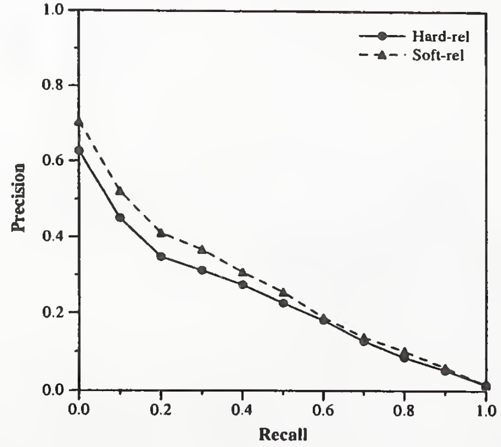


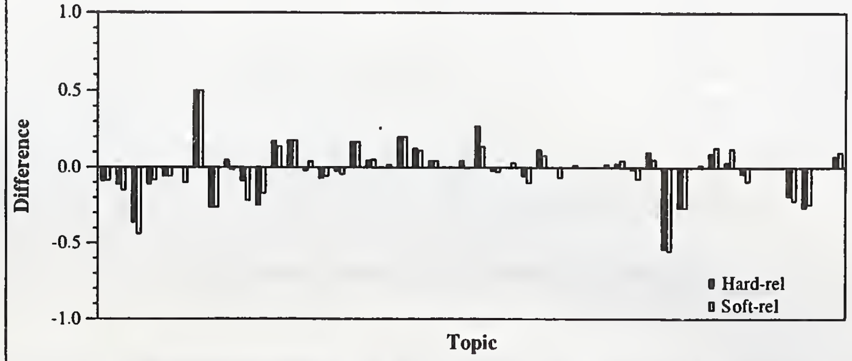
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	47398	47398
Total relevant	5123	7576
Total rel. ret.	3418	4742
MAP	0.2207	0.2533

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4042	0.4875
At 10 docs	0.3792	0.4646
At 15 docs	0.3583	0.4458
At 20 docs	0.3302	0.4188
At 30 docs	0.3118	0.3965
At 100 docs	0.2225	0.2969
At 200 docs	0.1763	0.2374
At 500 docs	0.1139	0.1548
At 1000 docs	0.0712	0.0988
R-Precision	0.2435	0.2812



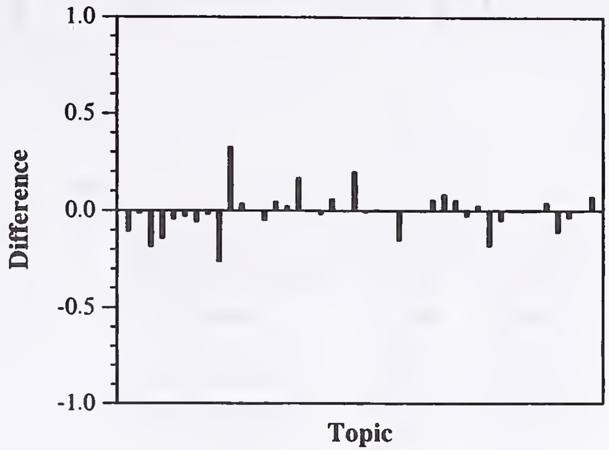


Difference from median in Document based R-precision per topic

Run ID	MSRCs9e1p1
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3126	0.0520
At 10 passages	0.2574	0.0702
At 15 passages	0.2464	0.0809
At 20 passages	0.2184	0.0868
At 30 passages	0.2154	0.1032
At 50 passages	0.1762	0.1117
At 100 passages	0.1442	0.1272
R-Precision	0.1484	

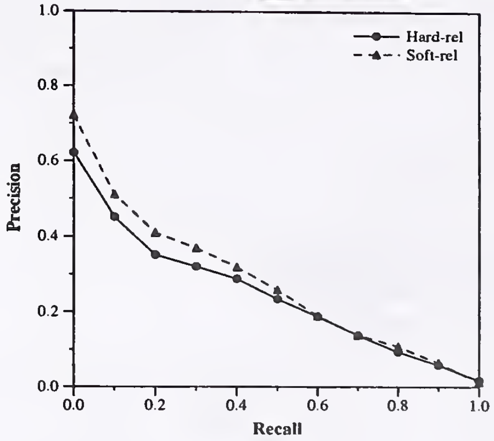


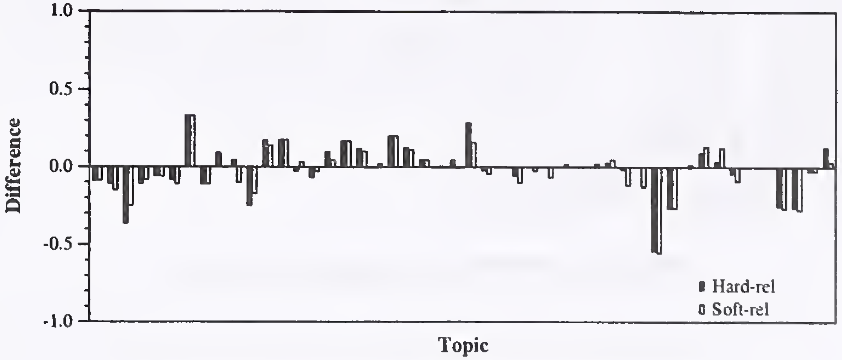
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	47502	47502
Total relevant	5123	7576
Total rel. ret.	3472	4758
MAP	0.2274	0.2559

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4042	0.4917
At 10 docs	0.3646	0.4396
At 15 docs	0.3583	0.4333
At 20 docs	0.3302	0.4083
At 30 docs	0.3069	0.3833
At 100 docs	0.2248	0.2983
At 200 docs	0.1804	0.2419
At 500 docs	0.1158	0.1565
At 1000 docs	0.0723	0.0991
R-Precision	0.2440	0.2796



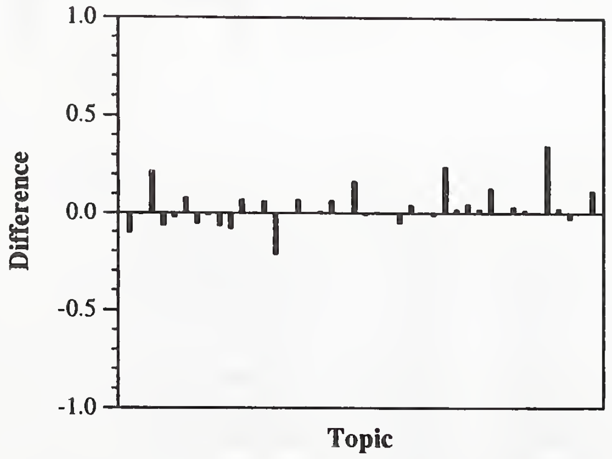


Difference from median in Document based R-precision per topic

Run ID	pircHDBt1
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2867	0.0577
At 10 passages	0.2809	0.0946
At 15 passages	0.2710	0.1074
At 20 passages	0.2628	0.1232
At 30 passages	0.2491	0.1491
At 50 passages	0.2158	0.1594
At 100 passages	0.1739	0.1741
R-Precision	0.1810	

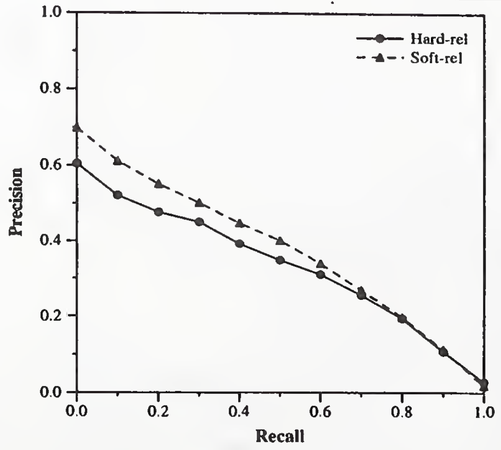


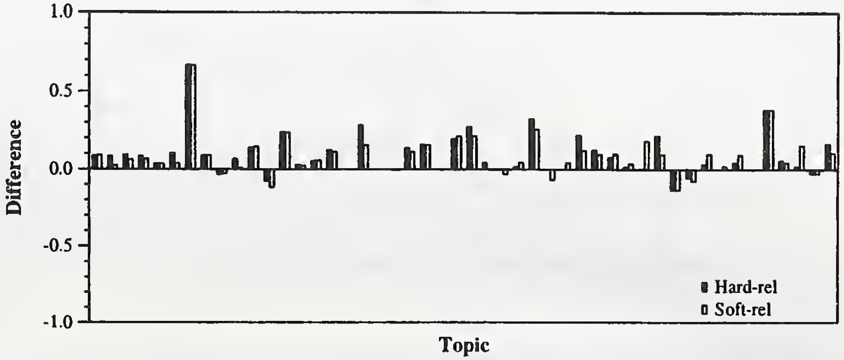
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3893	5372
MAP	0.3219	0.3650

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4333	0.5375
At 10 docs	0.4417	0.5396
At 15 docs	0.4306	0.5333
At 20 docs	0.4229	0.5260
At 30 docs	0.4132	0.5167
At 100 docs	0.3287	0.4179
At 200 docs	0.2428	0.3165
At 500 docs	0.1366	0.1827
At 1000 docs	0.0811	0.1119
R-Precision	0.3460	0.3857



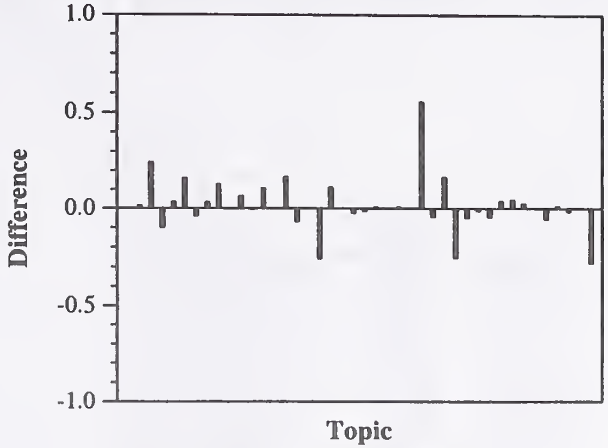


Difference from median in Document based R-precision per topic

Run ID	pircHDBtd1
Run Description	baseline run; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3602	0.0981
At 10 passages	0.3186	0.1125
At 15 passages	0.2931	0.1207
At 20 passages	0.2721	0.1300
At 30 passages	0.2316	0.1280
At 50 passages	0.1988	0.1391
At 100 passages	0.1561	0.1551
R-Precision	0.1699	

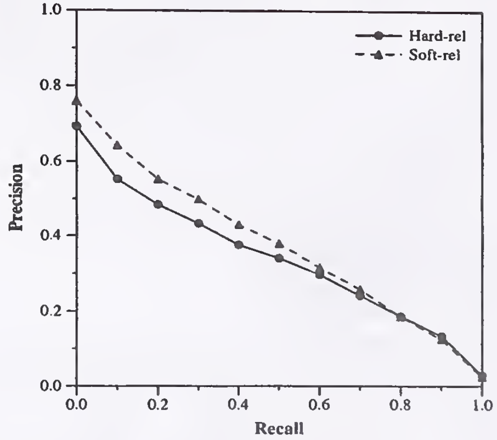


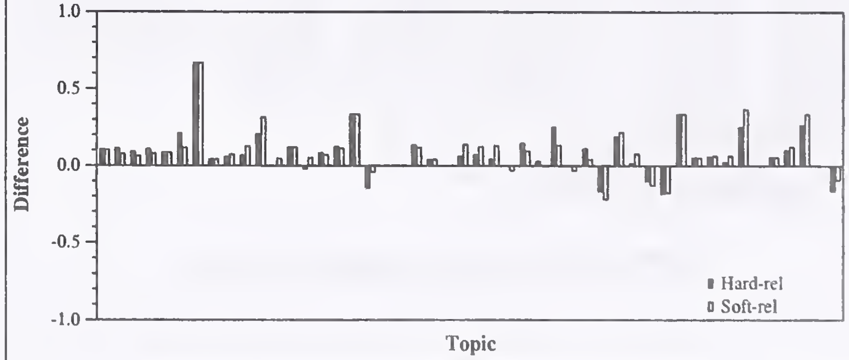
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3958	5430
MAP	0.3277	0.3656

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5292	0.6250
At 10 docs	0.4875	0.5875
At 15 docs	0.4375	0.5583
At 20 docs	0.4167	0.5365
At 30 docs	0.3743	0.4826
At 100 docs	0.3017	0.3867
At 200 docs	0.2357	0.3081
At 500 docs	0.1375	0.1843
At 1000 docs	0.0825	0.1131
R-Precision	0.3360	0.3932



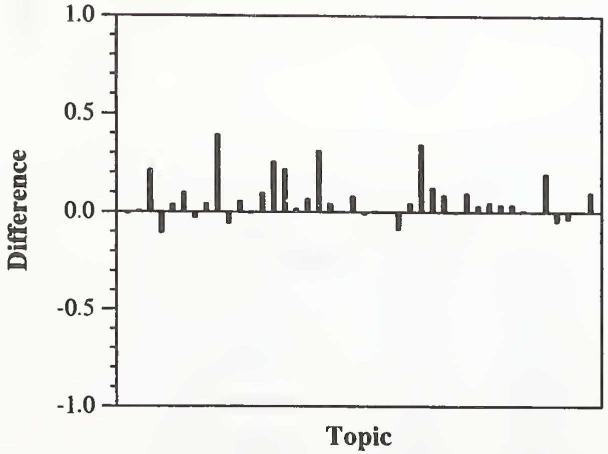


Difference from median in Document based R-precision per topic

Run ID	pircHDC1t1
Run Description	not a baseline, used form; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3751	0.1093
At 10 passages	0.3152	0.1350
At 15 passages	0.2694	0.1238
At 20 passages	0.2638	0.1362
At 30 passages	0.2369	0.1479
At 50 passages	0.2272	0.1654
At 100 passages	0.1812	0.1833
R-Precision	0.2335	

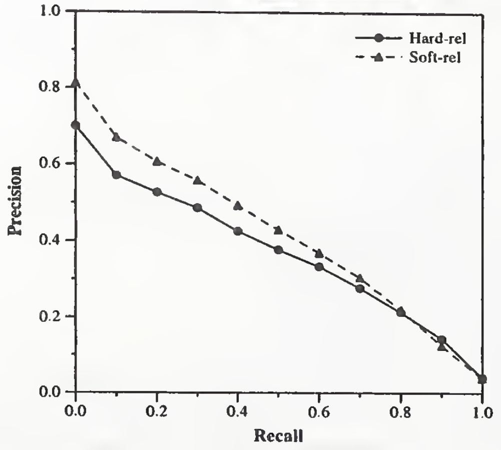


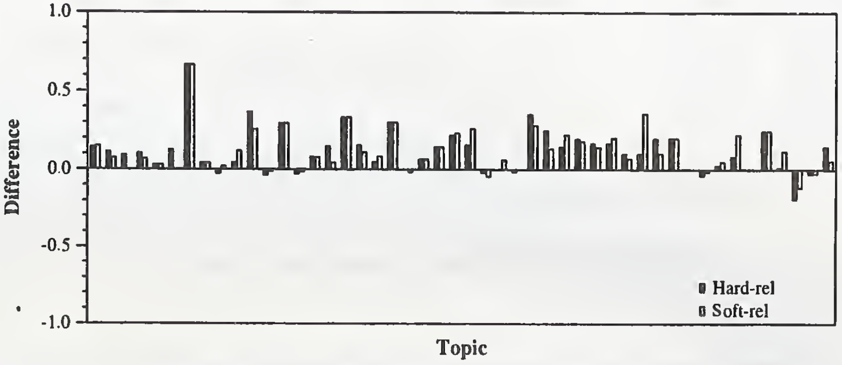
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	47911	47911
Total relevant	5123	7576
Total rel. ret.	3999	5533
MAP	0.3583	0.4069

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5292	0.6542
At 10 docs	0.4854	0.5979
At 15 docs	0.4486	0.5611
At 20 docs	0 4344	0.5469
At 30 docs	0.4111	0.5299
At 100 docs	0.3283	0.4185
At 200 docs	0.2447	0.3210
At 500 docs	0.1384	0.1873
At 1000 docs	0.0833	0.1153
R-Precision	0.3740	0.4250



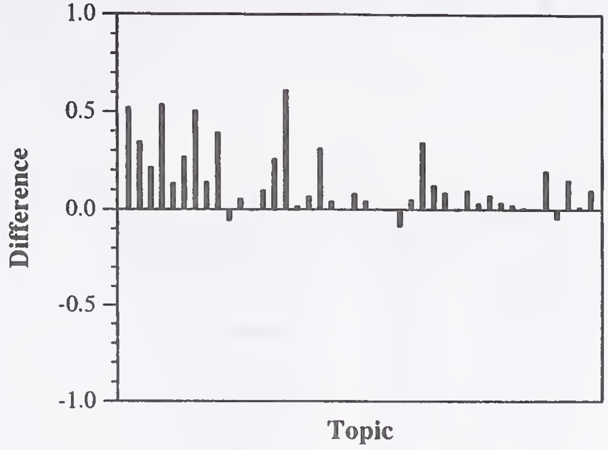


Difference from median in Document based R-precision per topic

Run ID	pircHDC1tp
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.4319	0.0772
At 10 passages	0.3770	0.1083
At 15 passages	0.3300	0.1009
At 20 passages	0.3171	0.1122
At 30 passages	0.3081	0.1403
At 50 passages	0.2891	0.1598
At 100 passages	0.2352	0.1884
R-Precision	0.3195	

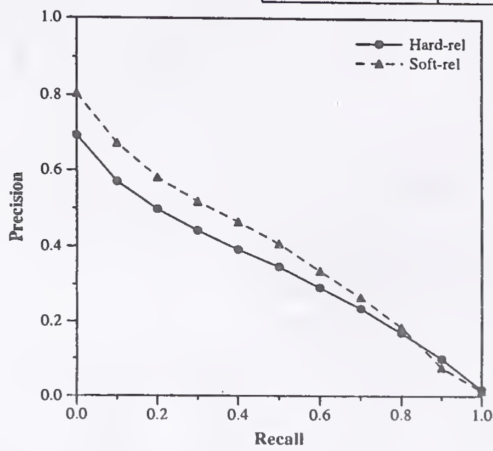


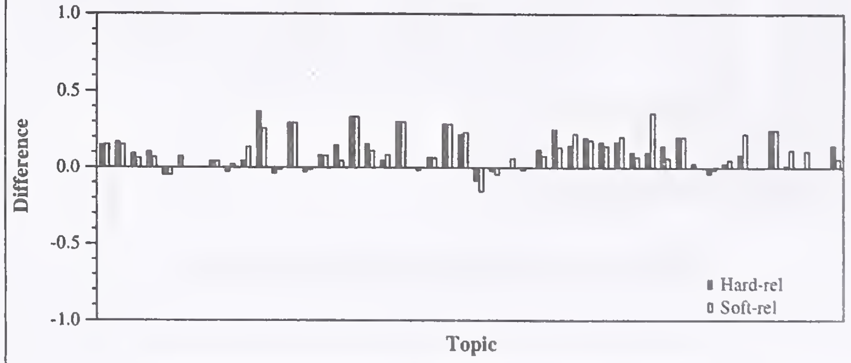
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	37386	37386
Total relevant	5123	7576
Total rel. ret.	3712	5084
MAP	0.3251	0.3790

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5125	0.6375
At 10 docs	0.4583	0.5854
At 15 docs	0.4319	0.5514
At 20 docs	0.4188	0.5333
At 30 docs	0.4028	0.5194
At 100 docs	0.3217	0.4112
At 200 docs	0.2377	0.3113
At 500 docs	0.1319	0.1769
At 1000 docs	0.0773	0.1059
R-Precision	0.3556	0.4072



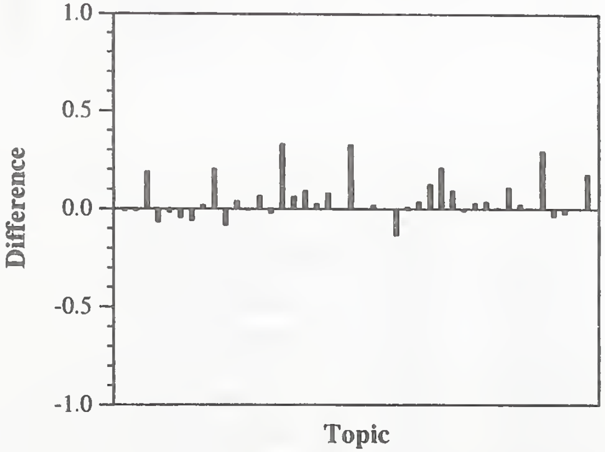


Difference from median in Document based R-precision per topic

Run ID	pircHDC2t1
Run Description	not a baseline, used form; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3768	0.0989
At 10 passages	0.3209	0.1199
At 15 passages	0.3121	0.1307
At 20 passages	0.2985	0.1434
At 30 passages	0.2766	0.1549
At 50 passages	0.2359	0.1712
At 100 passages	0.1881	0.1845
R-Precision	0.2145	

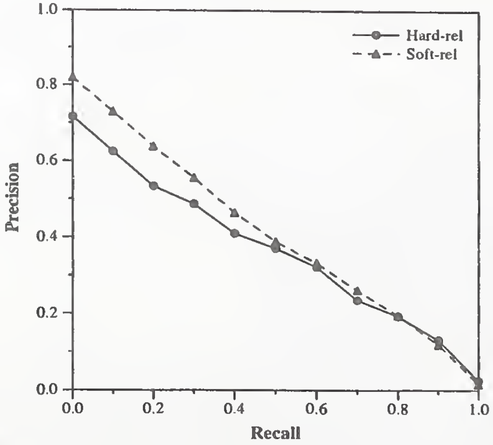


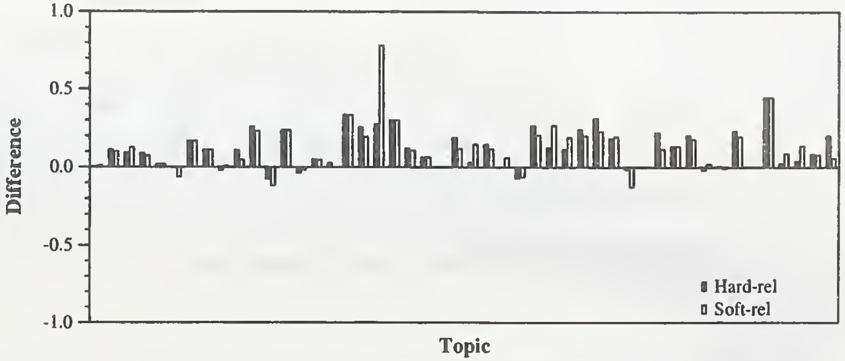
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3812	5215
MAP	0.3536	0.3986

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5583	0.6917
At 10 docs	0.5083	0.6500
At 15 docs	0.4875	0.6250
At 20 docs	0.4802	0.6104
At 30 docs	0.4535	0.5799
At 100 docs	0.3502	0.4490
At 200 docs	0.2466	0.3187
At 500 docs	0.1332	0.1774
At 1000 docs	0.0794	0.1086
R-Precision	0.3717	0.4242



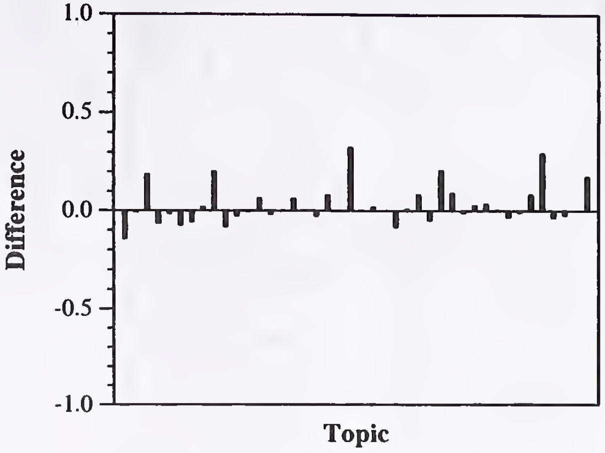


Difference from median in Document based R-precision per topic

Run ID	pircHDC2t2
Run Description	not a baseline, used form; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3383	0.0760
At 10 passages	0.2964	0.0945
At 15 passages	0.2817	0.1013
At 20 passages	0.2648	0.1110
At 30 passages	0.2553	0.1320
At 50 passages	0.2204	0.1506
At 100 passages	0.1791	0.1704
R-Precision	0.1872	

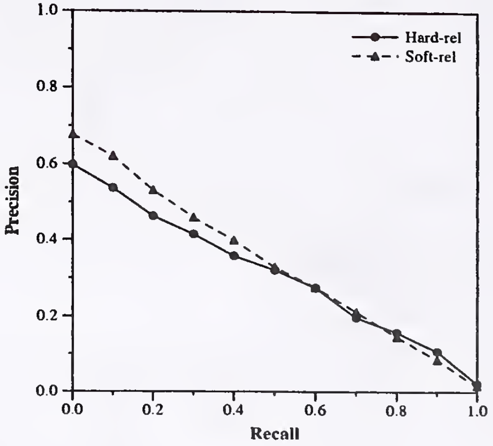


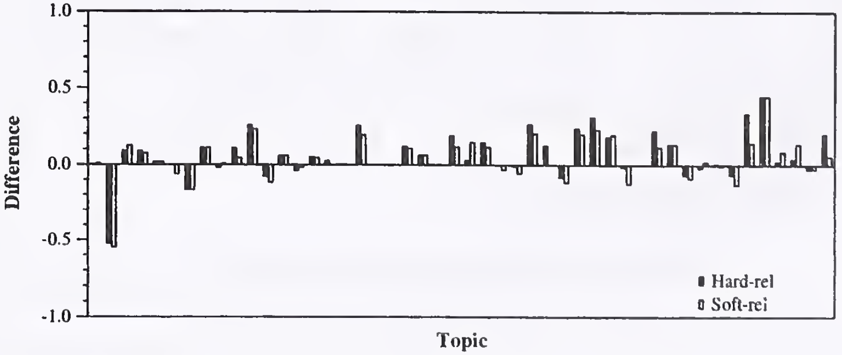
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3589	4891
MAP	0.3048	0.3314

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4833	0.5708
At 10 docs	0.4542	0.5583
At 15 docs	0.4403	0.5417
At 20 docs	0.4354	0.5323
At 30 docs	0.4125	0.5062
At 100 docs	0.3246	0.4044
At 200 docs	0.2321	0.2947
At 500 docs	0.1257	0.1661
At 1000 docs	0.0748	0.1019
R-Precision	0.3191	0.3454



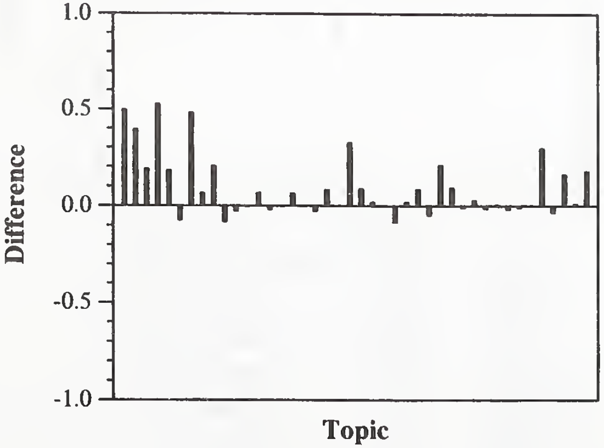


Difference from median in Document based R-precision per topic

Run ID	pircHDC2tp
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.4056	0.0589
At 10 passages	0.3754	0.0822
At 15 passages	0.3566	0.0932
At 20 passages	0.3426	0.1050
At 30 passages	0.3191	0.1269
At 50 passages	0.2779	0.1487
At 100 passages	0.2359	0.1801
R-Precision	0.2508	

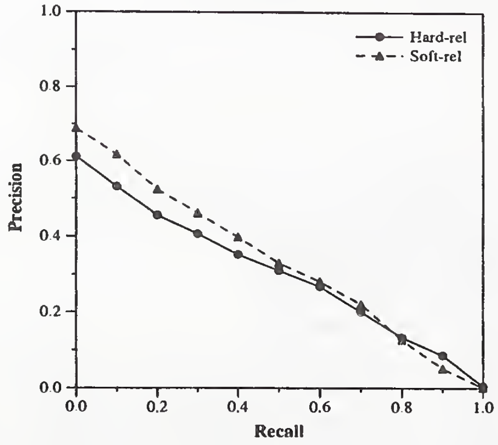


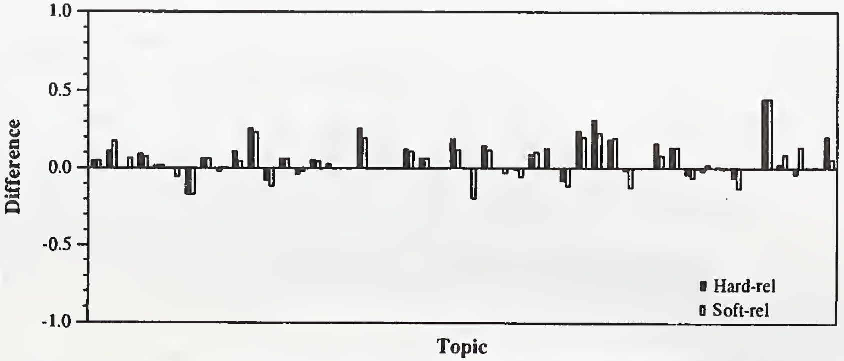
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	36647	36647
Total relevant	5123	7576
Total rel. ret.	3377	4561
MAP	0.2926	0.3248

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5042	0.6042
At 10 docs	0.4688	0.5750
At 15 docs	0.4458	0.5431
At 20 docs	0.4448	0.5354
At 30 docs	0.4181	0.5056
At 100 docs	0.3252	0.4008
At 200 docs	0.2315	0.2917
At 500 docs	0.1215	0.1586
At 1000 docs	0.0704	0.0950
R-Precision	0.3174	0.3473



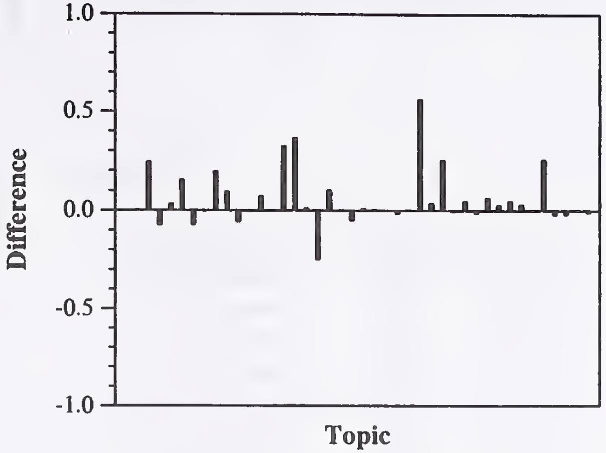


Difference from median in Document based R-precision per topic

Run ID	pircHDC3td1
Run Description	not a baseline, used form; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3601	0.1023
At 10 passages	0.3359	0.1271
At 15 passages	0.3065	0.1315
At 20 passages	0.2862	0.1415
At 30 passages	0.2374	0.1452
At 50 passages	0.2243	0.1680
At 100 passages	0.1910	0.1909
R-Precision	0.2141	

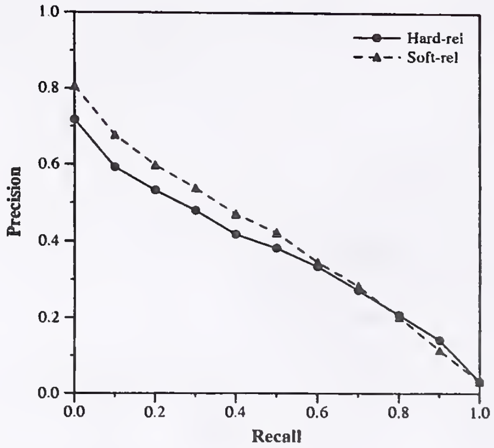


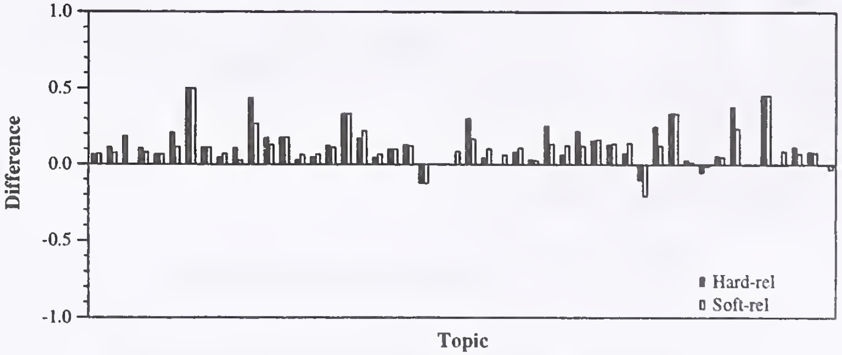
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3915	5470
MAP	0.3589	0.3934

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5583	0.6625
At 10 docs	0.5271	0.6271
At 15 docs	0.4917	0.5889
At 20 docs	0.4802	0.5896
At 30 docs	0.4306	0.5403
At 100 docs	0.3317	0.4223
At 200 docs	0.2435	0.3168
At 500 docs	0.1389	0.1875
At 1000 docs	0.0816	0.1140
R-Precision	0.3813	0.4131



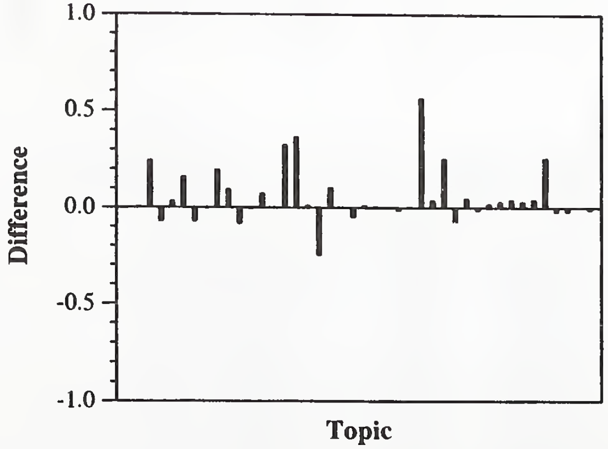


Difference from median in Document based R-precision per topic

Run ID	pircHDC3td2
Run Description	not a baseline, used form; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3607	0.1108
At 10 passages	0.3257	0.1198
At 15 passages	0.3022	0.1287
At 20 passages	0.2803	0.1397
At 30 passages	0.2291	0.1400
At 50 passages	0.2185	0.1615
At 100 passages	0.1888	0.1876
R-Precision	0.2127	

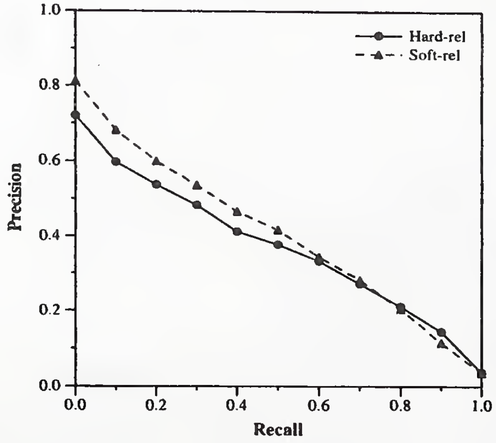


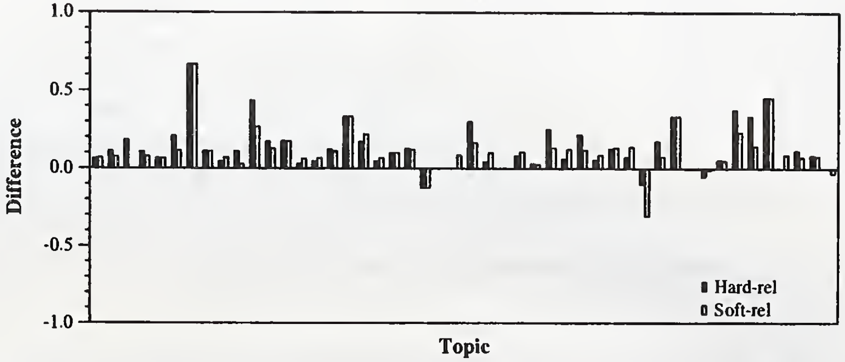
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3901	5445
MAP	0.3604	0.3937

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5542	0.6583
At 10 docs	0.5146	0.6167
At 15 docs	0.4833	0.5833
At 20 docs	0.4740	0.5865
At 30 docs	0.4236	0.5319
At 100 docs	0.3277	0.4181
At 200 docs	0.2419	0.3150
At 500 docs	0.1382	0.1867
At 1000 docs	0.0813	0.1134
R-Precision	0.3875	0.4133



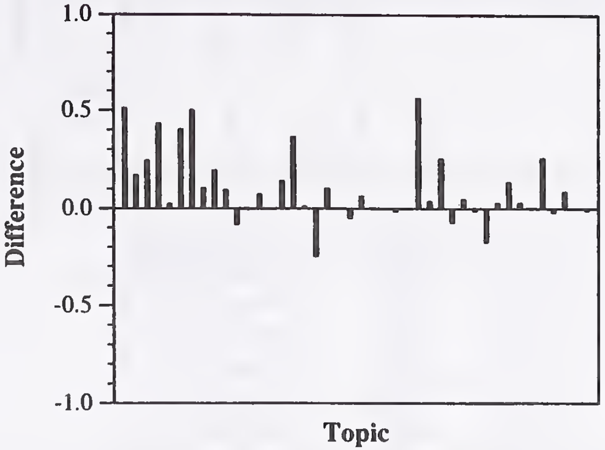


Difference from median in Document based R-precision per topic

Run ID	pircHDC3tdp
Run Description	not a baseline, used both; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3536	0.0663
At 10 passages	0.3353	0.0864
At 15 passages	0.3124	0.0971
At 20 passages	0.3019	0.1071
At 30 passages	0.2772	0.1283
At 50 passages	0.2652	0.1586
At 100 passages	0.2349	0.1913
R-Precision	0.2555	

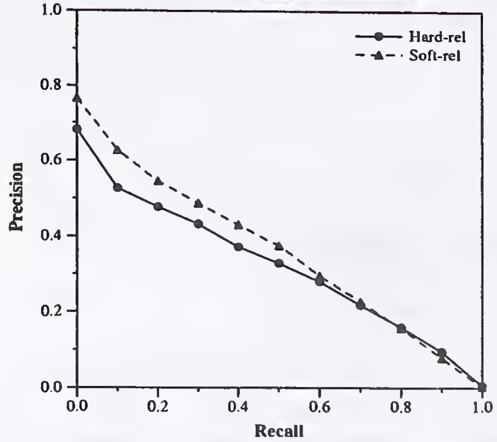


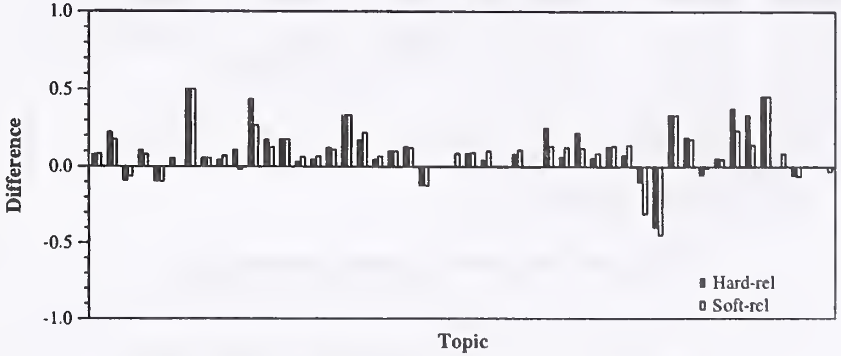
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	39228	39228
Total relevant	5123	7576
Total rel. ret.	3686	5149
MAP	0.3072	0.3449

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4708	0.5750
At 10 docs	0.4583	0.5542
At 15 docs	0.4333	0.5306
At 20 docs	0.4219	0.5344
At 30 docs	0.3819	0.4896
At 100 docs	0.2994	0.3867
At 200 docs	0.2280	0.2983
At 500 docs	0.1332	0.1801
At 1000 docs	0.0768	0.1073
R-Precision	0.3549	0.3893



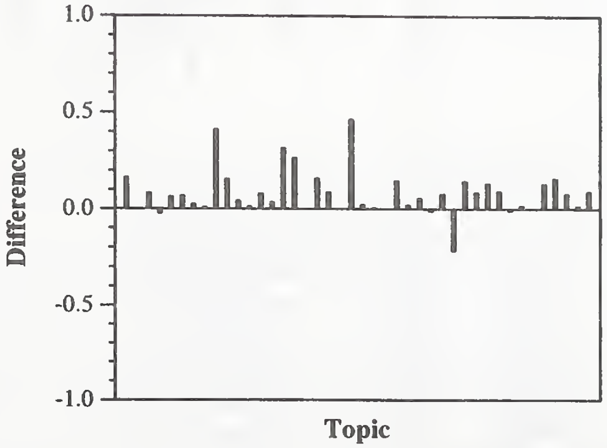


Difference from median in Document based R-precision per topic

Run ID	rutbase1
Run Description	baseline run; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3425	0.0923
At 10 passages	0.3396	0.1223
At 15 passages	0.3231	0.1326
At 20 passages	0.3072	0.1411
At 30 passages	0.2901	0.1618
At 50 passages	0.2619	0.1832
At 100 passages	0.2103	0.1994
R-Precision	0.2553	

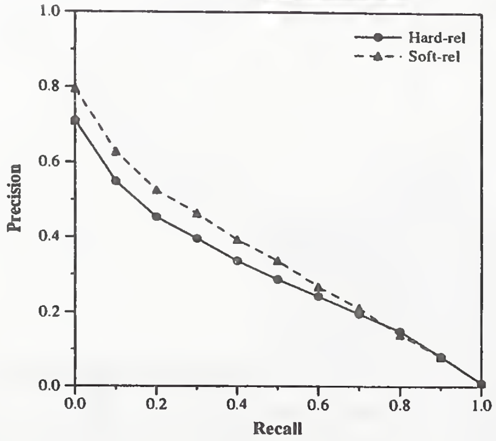


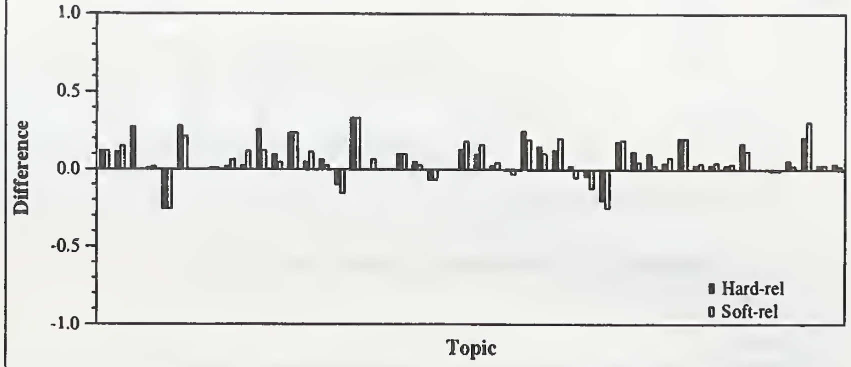
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3545	5010
MAP	0.2899	0.3310

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4625	0.5750
At 10 docs	0.4521	0.5500
At 15 docs	0.4333	0.5278
At 20 docs	0.4073	0.5000
At 30 docs	0.3813	0.4771
At 100 docs	0.2888	0.3819
At 200 docs	0.2178	0.2906
At 500 docs	0.1237	0.1710
At 1000 docs	0.0739	0.1044
R-Precision	0.3256	0.3642



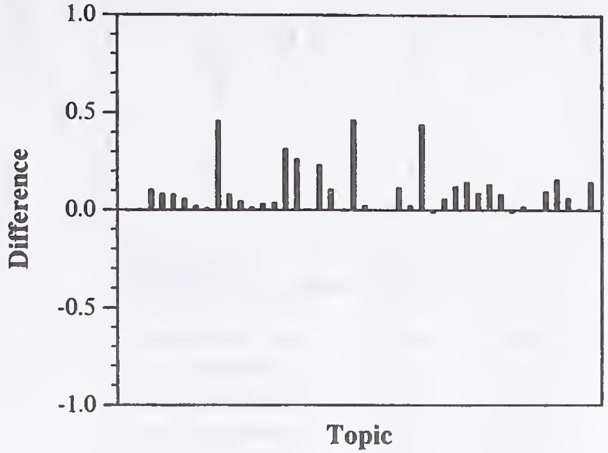


Difference from median in Document based R-precision per topic

Run ID	rutbase2
Run Description	baseline run; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3745	0.0991
At 10 passages	0.3769	0.1320
At 15 passages	0.3370	0.1315
At 20 passages	0.3214	0.1463
At 30 passages	0.3055	0.1738
At 50 passages	0.2580	0.1819
At 100 passages	0.2371	0.2205
R-Precision	0.2706	

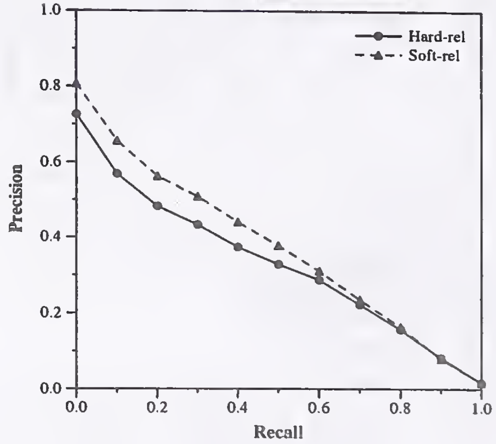


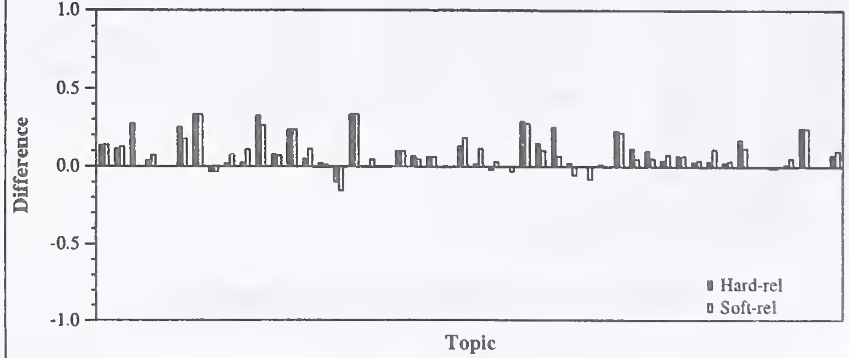
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3736	5265
MAP	0.3186	0.3638

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5083	0.6333
At 10 docs	0.4750	0.5875
At 15 docs	0.4444	0.5597
At 20 docs	0.4302	0.5396
At 30 docs	0.4104	0.5132
At 100 docs	0.3173	0.4142
At 200 docs	0.2344	0.3126
At 500 docs	0.1318	0.1809
At 1000 docs	0.0778	0.1097
R-Precision	0.3451	0.3854



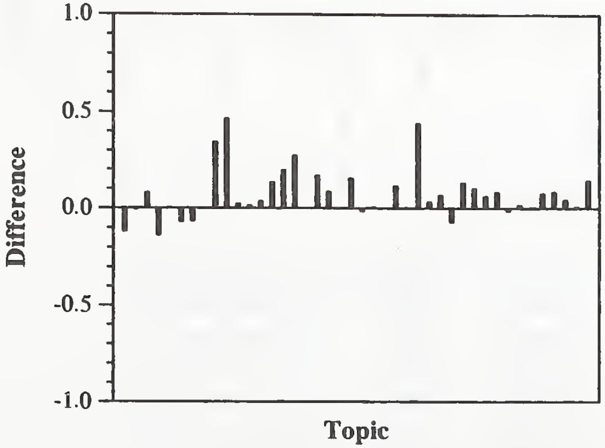


Difference from median in Document based R-precision per topic

Run ID	Rutmeta
Run Description	not a baseline, used metadata; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3581	0.0907
At 10 passages	0.3617	0.1279
At 15 passages	0.3239	0.1273
At 20 passages	0.3089	0.1415
At 30 passages	0.2737	0.1558
At 50 passages	0.2434	0.1698
At 100 passages	0.2035	0.1922
R-Precision	0.2383	

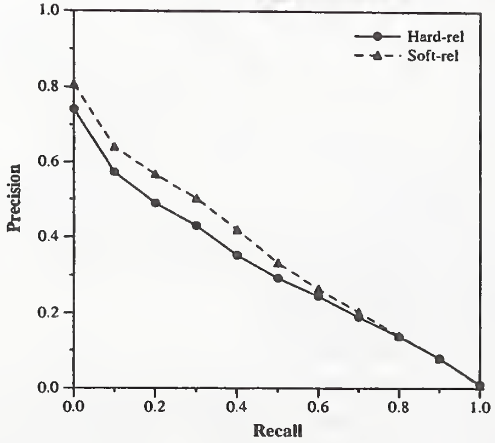


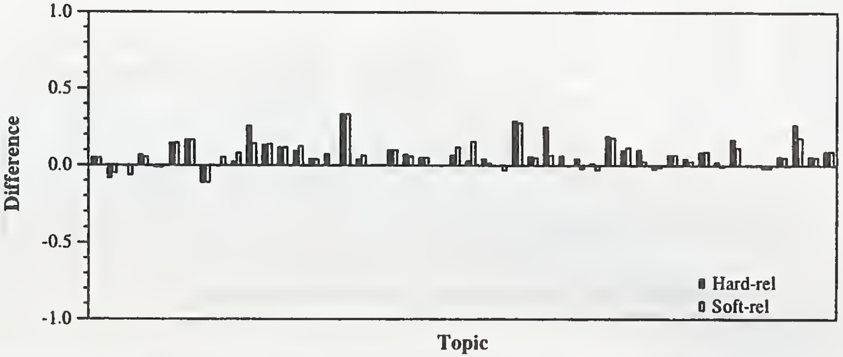
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3728	5218
MAP	0.3019	0.3436

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5125	0.6250
At 10 docs	0.4750	0.5917
At 15 docs	0.4458	0.5694
At 20 docs	0.4312	0.5479
At 30 docs	0.4125	0.5194
At 100 docs	0.3115	0.4090
At 200 docs	0.2328	0.3070
At 500 docs	0.1300	0.1780
At 1000 docs	0.0777	0.1087
R-Precision	0.3308	0.3685



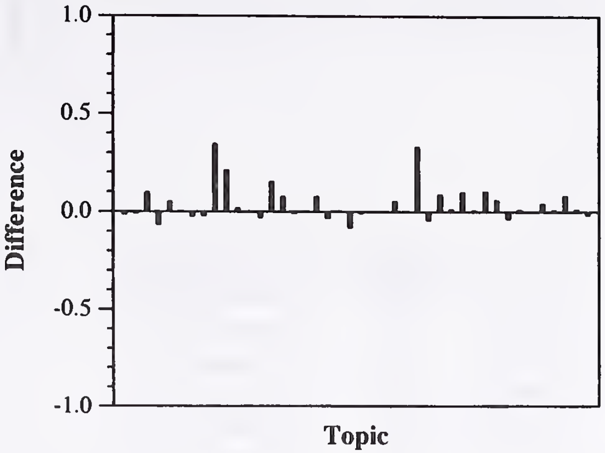


Difference from median in Document based R-precision per topic

Run ID	ub03cugTD
Run Description	baseline run; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3065	0.0862
At 10 passages	0.2987	0.1151
At 15 passages	0.2666	0.1161
At 20 passages	0.2667	0.1305
At 30 passages	0.2467	0.1528
At 50 passages	0.2126	0.1653
At 100 passages	0.1841	0.1858
R-Precision	0.1997	

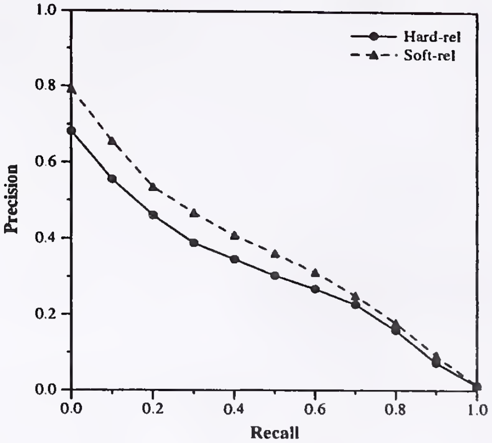


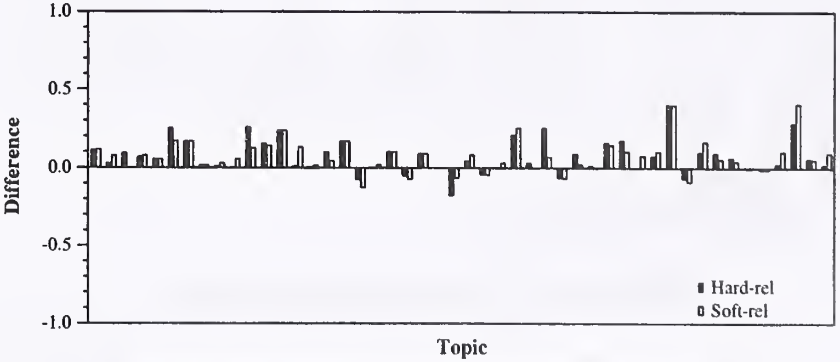
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3548	5069
MAP	0.2981	0.3540

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4708	0.6292
At 10 docs	0.4542	0.5854
At 15 docs	0.4278	0.5556
At 20 docs	0.4229	0.5375
At 30 docs	0.4042	0.5076
At 100 docs	0.3002	0.3940
At 200 docs	0.2253	0.3047
At 500 docs	0.1271	0.1766
At 1000 docs	0.0739	0.1056
R-Precision	0.3286	0.3784



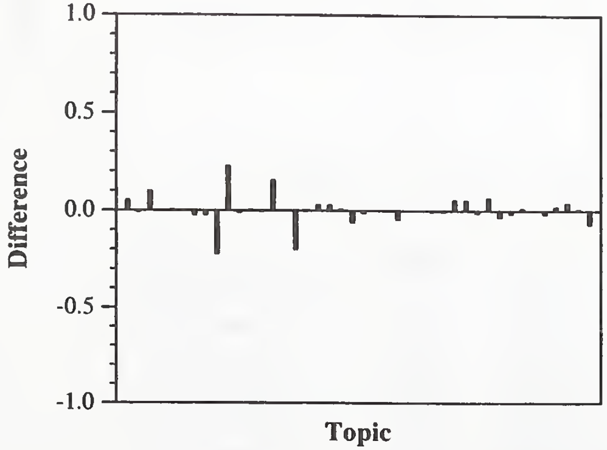


Difference from median in Document based R-precision per topic

Run ID	ub03sugT
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2261	0.0539
At 10 passages	0.2399	0.0829
At 15 passages	0.2278	0.0902
At 20 passages	0.2107	0.1004
At 30 passages	0.1993	0.1178
At 50 passages	0.1859	0.1354
At 100 passages	0.1572	0.1541
R-Precision	0.1594	

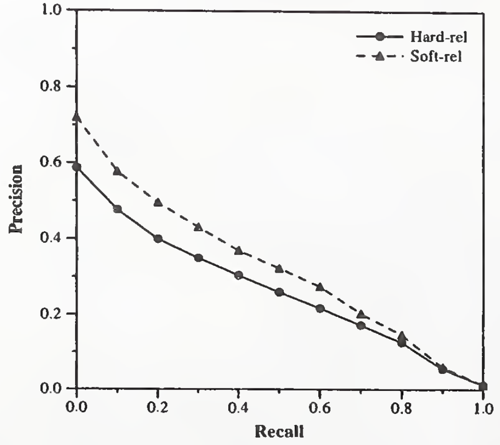


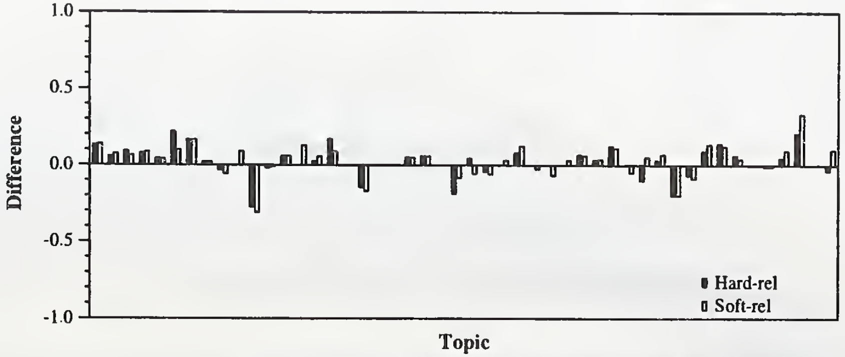
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3513	4990
MAP	0.2543	0.3126

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3875	0.5458
At 10 docs	0.4042	0.5375
At 15 docs	0.3875	0.5042
At 20 docs	0.3771	0.4958
At 30 docs	0.3597	0.4674
At 100 docs	0.2792	0.3729
At 200 docs	0.2098	0.2856
At 500 docs	0.1226	0.1700
At 1000 docs	0.0732	0.1040
R-Precision	0.2764	0.3335



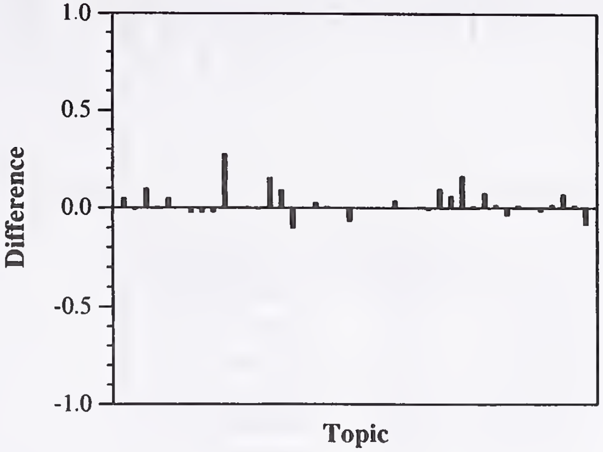


Difference from median in Document based R-precision per topic

Run ID	ub03ugTcugTD
Run Description	baseline run; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2828	0.0737
At 10 passages	0.2763	0.0964
At 15 passages	0.2501	0.1003
At 20 passages	0.2441	0.1169
At 30 passages	0.2325	0.1368
At 50 passages	0.2159	0.1585
At 100 passages	0.1813	0.1810
R-Precision	0.1825	

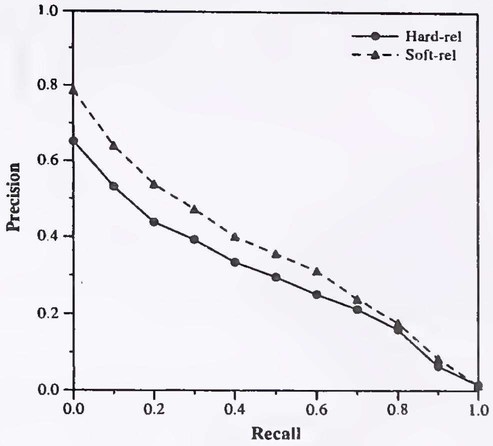


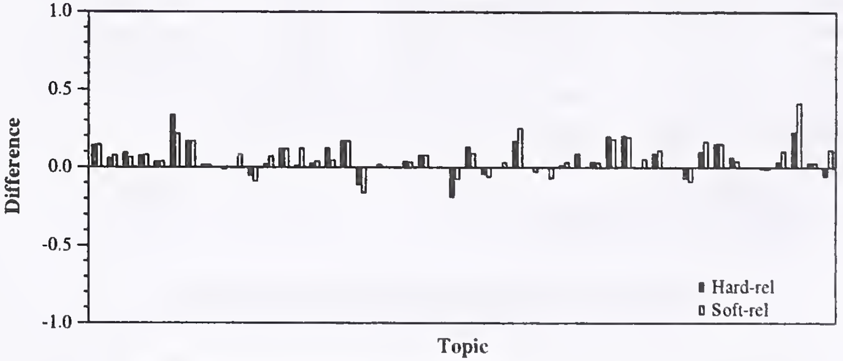
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3682	5265
MAP	0.2896	0.3495

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4542	0.6042
At 10 docs	0.4479	0.5729
At 15 docs	0.4194	0.5361
At 20 docs	0.4062	0.5219
At 30 docs	0.3840	0.4924
At 100 docs	0.3027	0.3977
At 200 docs	0.2247	0.3055
At 500 docs	0.1297	0.1797
At 1000 docs	0.0767	0.1097
R-Precision	0.3075	0.3663



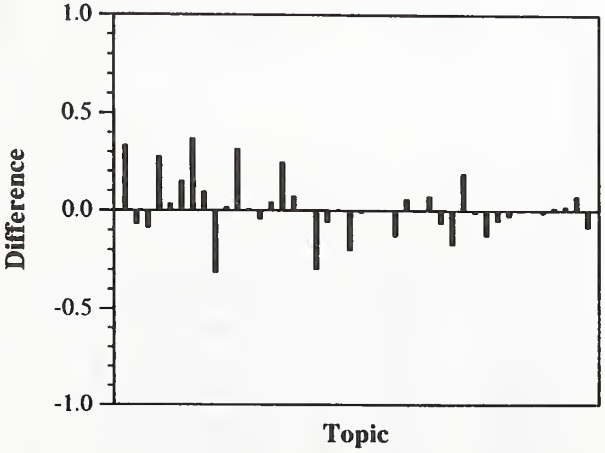


Difference from median in Document based R-precision per topic

Run ID	ub03smfugTD
Run Description	not a baseline, used metadata; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2395	0.0305
At 10 passages	0.2313	0.0495
At 15 passages	0.2198	0.0630
At 20 passages	0.2187	0.0726
At 30 passages	0.1929	0.0798
At 50 passages	0.1744	0.0989
At 100 passages	0.1518	0.1228
R-Precision	0.1699	

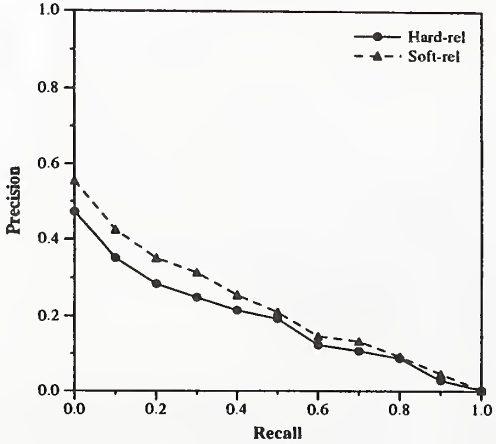


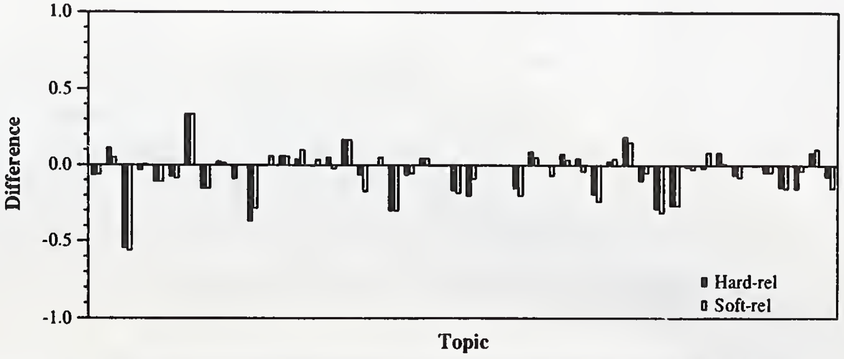
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	33675	33675
Total relevant	5123	7576
Total rel. ret.	3236	4543
MAP	0.1726	0.2073

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.2917	0.3667
At 10 docs	0.2771	0.3583
At 15 docs	0.2722	0.3569
At 20 docs	0.2656	0.3469
At 30 docs	0.2465	0.3278
At 100 docs	0.2150	0.2846
At 200 docs	0.1741	0.2322
At 500 docs	0.1155	0.1597
At 1000 docs	0.0674	0.0946
R-Precision	0.2078	0.2562



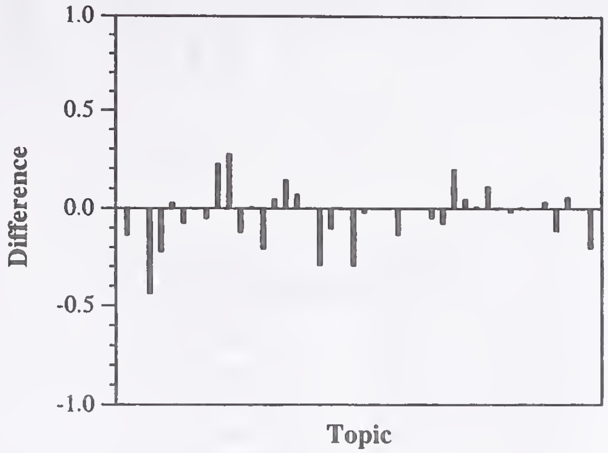


Difference from median in Document based R-precision per topic

Run ID	TUCSHardBR1
Run Description	baseline run; title+desc+narr

Passages-based Evaluation (42 topics)

	Passage Level Averages	
	Precision	F Score
At 5 passages	0.2419	0.0418
At 10 passages	0.2000	0.0560
At 15 passages	0.2076	0.0687
At 20 passages	0.1839	0.0762
At 30 passages	0.1667	0.0912
At 50 passages	0.1508	0.1036
At 100 passages	0.1297	0.1294
R-Precision	0.1235	

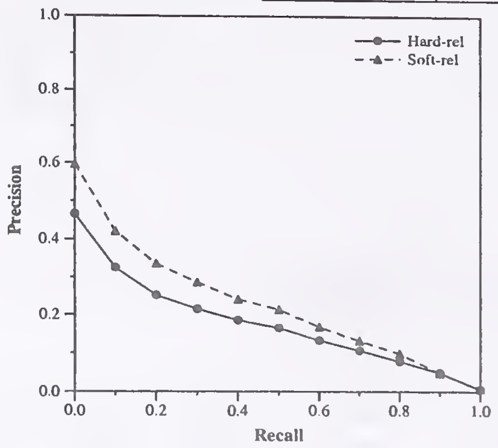


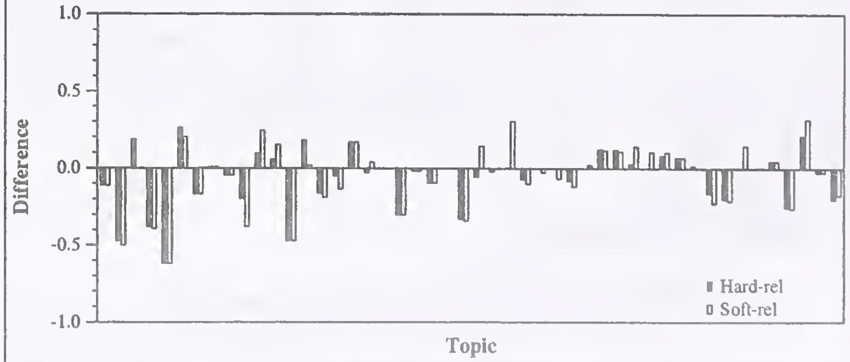
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	2437	3479
MAP	0.1632	0.2121

	Document Level Averages	
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.2833	0.3833
At 10 docs	0.2583	0.3667
At 15 docs	0.2556	0.3583
At 20 docs	0.2469	0.3500
At 30 docs	0.2361	0.3389
At 100 docs	0.1954	0.2821
At 200 docs	0.1494	0.2139
At 500 docs	0.0825	0.1175
At 1000 docs	0.0508	0.0725
R-Precision	0.1960	0.2520



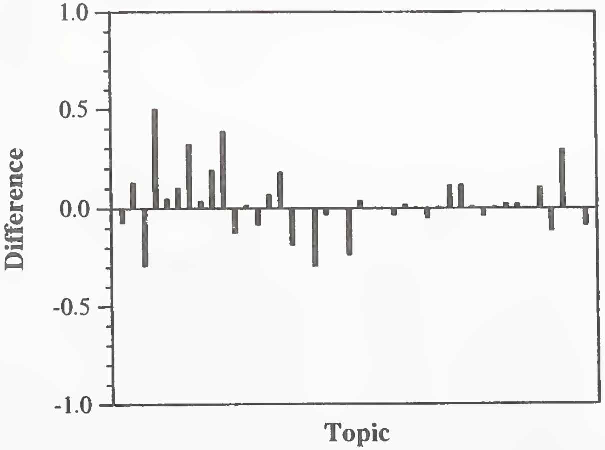


Difference from median in Document based R-precision per topic

Run ID	TUCSHARD1
Run Description	not a baseline, used both; title+desc+narr

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2509	0.0393
At 10 passages	0.2424	0.0608
At 15 passages	0.2452	0.0689
At 20 passages	0.2464	0.0797
At 30 passages	0.2305	0.0965
At 50 passages	0.2094	0.1132
At 100 passages	0.1828	0.1396
R-Precision	0.1868	

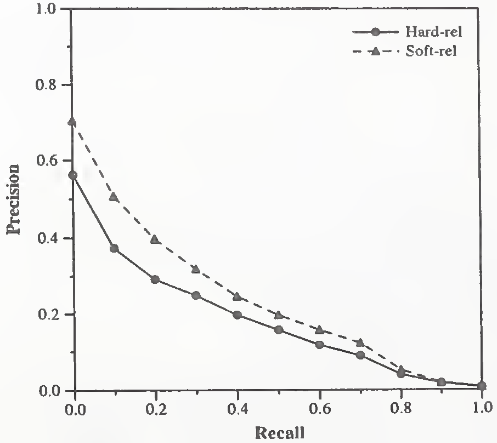


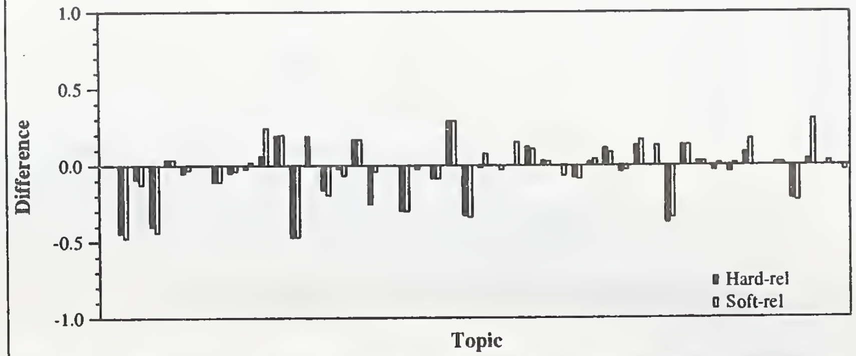
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	30714	30714
Total relevant	5123	7576
Total rel. ret.	2402	3453
MAP	0.1701	0.2262

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3125	0.4542
At 10 docs	0.3250	0.4562
At 15 docs	0.3014	0.4333
At 20 docs	0.2938	0.4156
At 30 docs	0.2729	0.3951
At 100 docs	0.2048	0.3058
At 200 docs	0.1609	0.2346
At 500 docs	0.0925	0.1325
At 1000 docs	0.0500	0.0719
R-Precision	0.2148	0.2818



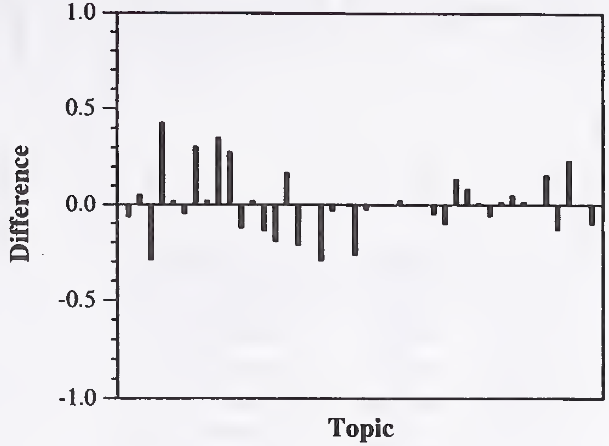


Difference from median in Document based R-precision per topic

Run ID	TUCSHARD2
Run Description	not a baseline, used both; title+desc+narr

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2519	0.0355
At 10 passages	0.2561	0.0506
At 15 passages	0.2421	0.0604
At 20 passages	0.2246	0.0648
At 30 passages	0.2045	0.0799
At 50 passages	0.1915	0.0950
At 100 passages	0.1734	0.1296
R-Precision	0.1655	

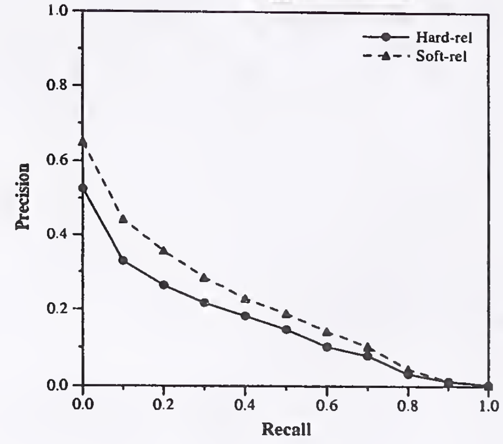


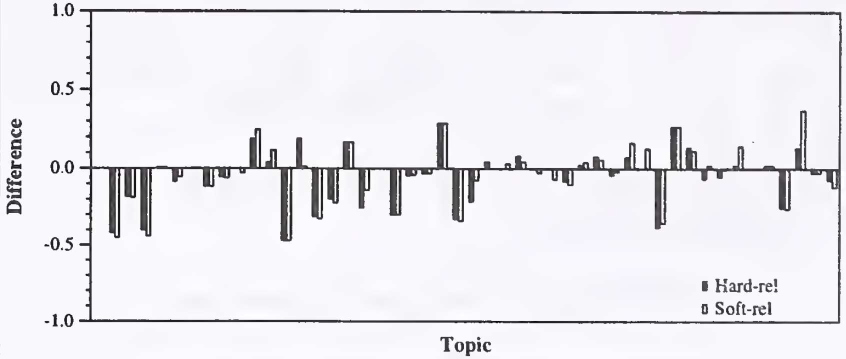
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	30714	30714
Total relevant	5123	7576
Total rel. ret.	2402	3453
MAP	0.1516	0.2034

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3167	0.4458
At 10 docs	0.3208	0.4417
At 15 docs	0.3097	0.4181
At 20 docs	0.2812	0.3865
At 30 docs	0.2514	0.3653
At 100 docs	0.1908	0.2790
At 200 docs	0.1507	0.2197
At 500 docs	0.0898	0.1290
At 1000 docs	0.0500	0.0719
R-Precision	0.2012	0.2627



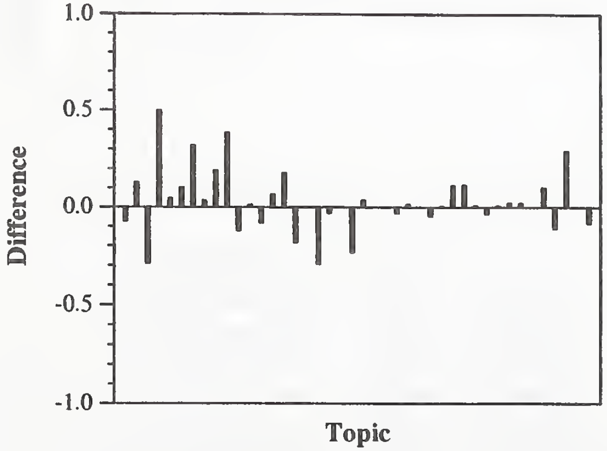


Difference from median in Document based R-precision per topic

Run ID	TUCSHARD3
Run Description	not a baseline, used both; title+desc+narr

Passages-based Evaluation (42 topics)

	Passage Level Averages	
	Precision	F Score
At 5 passages	0.2509	0.0393
At 10 passages	0.2424	0.0608
At 15 passages	0.2452	0.0689
At 20 passages	0.2464	0.0797
At 30 passages	0.2305	0.0965
At 50 passages	0.2094	0.1132
At 100 passages	0.1828	0.1396
R-Precision	0.1868	

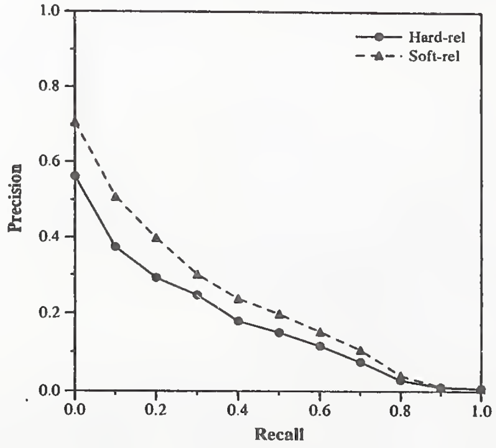


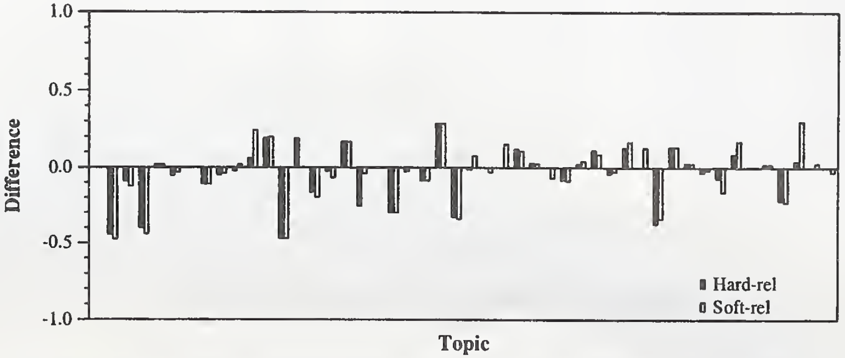
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	26908	26908
Total relevant	5123	7576
Total rel. ret.	2237	3198
MAP	0.1651	0.2200

	Document Level Averages	
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3125	0.4542
At 10 docs	0.3250	0.4562
At 15 docs	0.3014	0.4333
At 20 docs	0.2938	0.4156
At 30 docs	0.2729	0.3951
At 100 docs	0.2048	0.3058
At 200 docs	0.1598	0.2334
At 500 docs	0.0879	0.1258
At 1000 docs	0.0466	0.0666
R-Precision	0.2138	0.2771



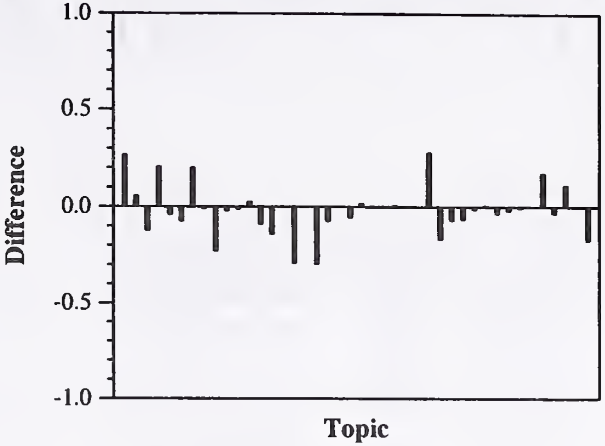


Difference from median in Document based R-precision per topic

Run ID	TUKE1
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2024	0.0077
At 10 passages	0.2053	0.0165
At 15 passages	0.2039	0.0222
At 20 passages	0.1998	0.0289
At 30 passages	0.1784	0.0341
At 50 passages	0.1605	0.0423
At 100 passages	0.1445	0.0592
R-Precision	0.1356	

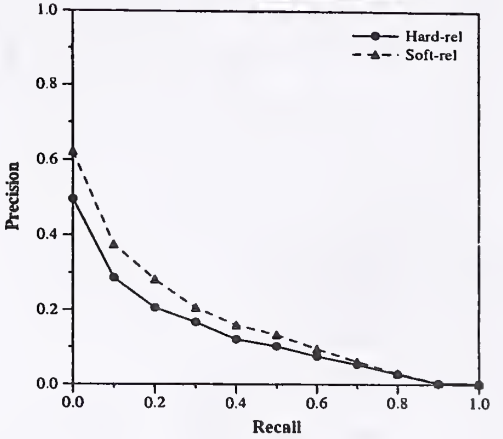


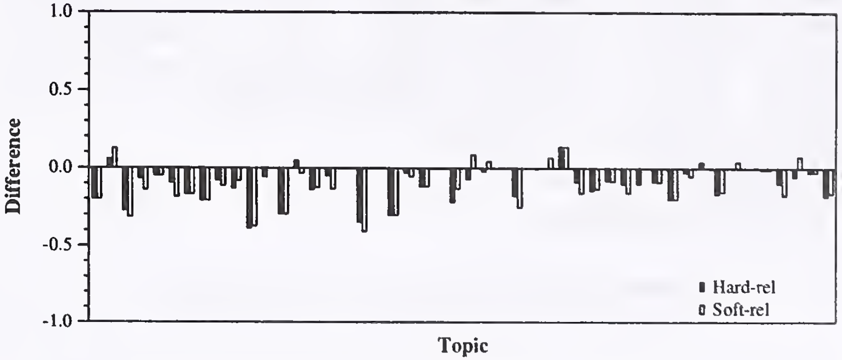
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	21712	21712
Total relevant	5123	7576
Total rel. ret.	1924	2781
MAP	0.1186	0.1535

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3000	0.4208
At 10 docs	0.2875	0.3729
At 15 docs	0.2667	0.3583
At 20 docs	0.2552	0.3458
At 30 docs	0.2313	0.3236
At 100 docs	0.1896	0.2692
At 200 docs	0.1420	0.2016
At 500 docs	0.0792	0.1145
At 1000 docs	0.0401	0.0579
R-Precision	0.1612	0.2112



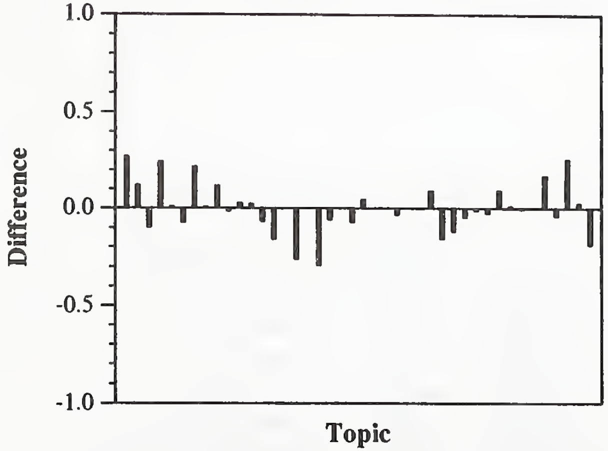


Difference from median in Document based R-precision per topic

Run ID	TUKE2
Run Description	baseline run; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.1950	0.0090
At 10 passages	0.1979	0.0157
At 15 passages	0.1941	0.0223
At 20 passages	0.1960	0.0273
At 30 passages	0.1886	0.0363
At 50 passages	0.1672	0.0441
At 100 passages	0.1628	0.0680
R-Precision	0.1579	

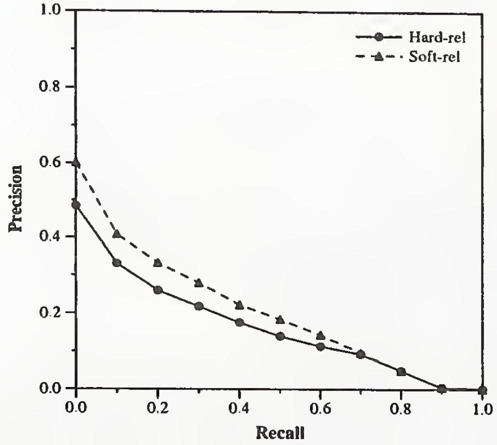


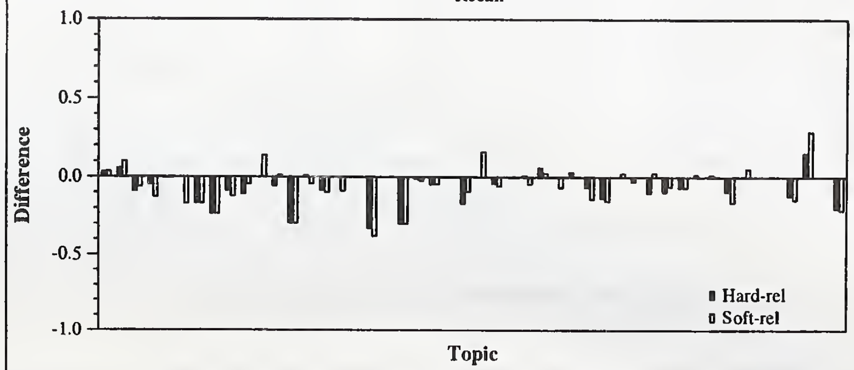
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	21091	21091
Total relevant	5123	7576
Total rel. ret.	2376	3298
MAP	0.1489	0.1881

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.2917	0.3792
At 10 docs	0.3021	0.3729
At 15 docs	0.2931	0.3750
At 20 docs	0.2823	0.3698
At 30 docs	0.2632	0.3569
At 100 docs	0.2142	0.2950
At 200 docs	0.1692	0.2328
At 500 docs	0.0981	0.1362
At 1000 docs	0.0495	0.0687
R-Precision	0.2019	0.2519



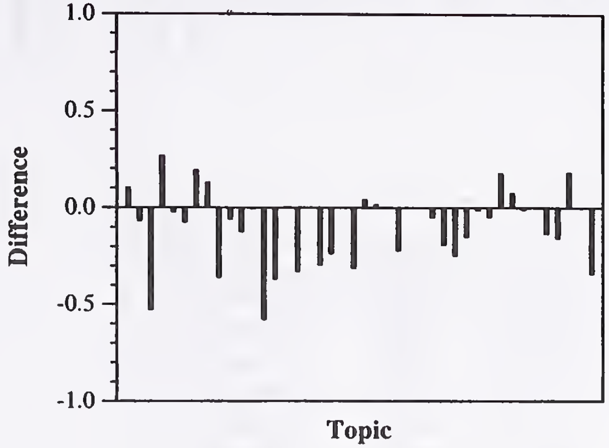


Difference from median in Document based R-precision per topic

Run ID	TUKE3
Run Description	not a baseline, used both; title+desc+narr

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.1429	0.0129
At 10 passages	0.1247	0.0188
At 15 passages	0.0901	0.0182
At 20 passages	0.0840	0.0203
At 30 passages	0.0810	0.0276
At 50 passages	0.0750	0.0292
At 100 passages	0.0655	0.0382
R-Precision	0.0694	

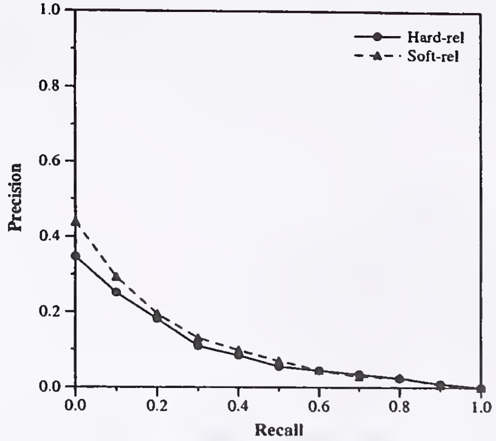


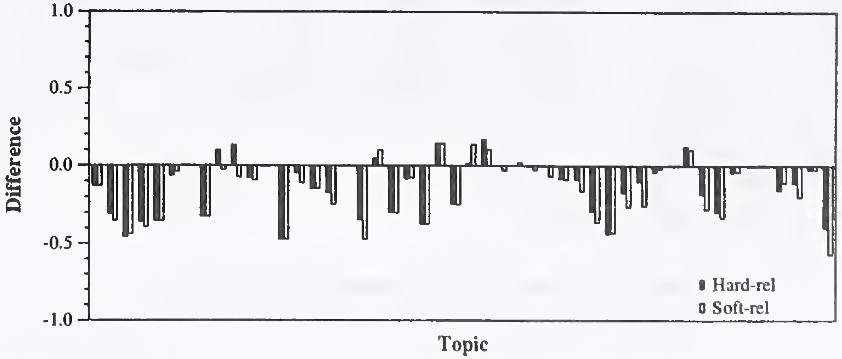
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	28007	28007
Total relevant	5123	7576
Total rel. ret.	1791	2316
MAP	0.0873	0.0989

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.1917	0.2542
At 10 docs	0.1896	0.2417
At 15 docs	0.1819	0.2347
At 20 docs	0.1750	0.2333
At 30 docs	0.1708	0.2257
At 100 docs	0.1348	0.1763
At 200 docs	0.1060	0.1374
At 500 docs	0.0618	0.0802
At 1000 docs	0.0373	0.0483
R-Precision	0.1325	0.1531



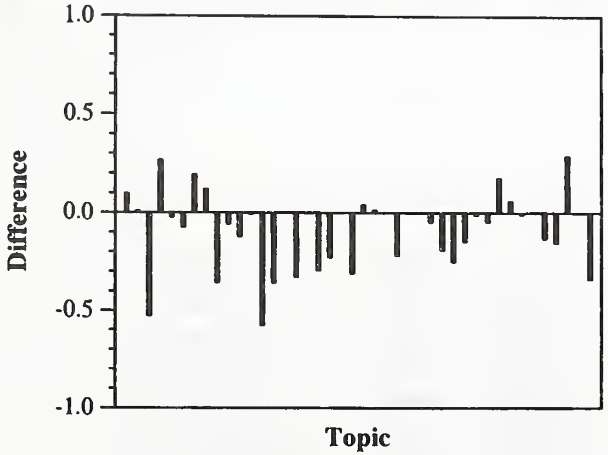


Difference from median in Document based R-precision per topic

Run ID	TUKE4
Run Description	not a baseline, used both; title+desc+narr

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.1462	0.0140
At 10 passages	0.1281	0.0193
At 15 passages	0.1059	0.0211
At 20 passages	0.0948	0.0226
At 30 passages	0.0931	0.0299
At 50 passages	0.0796	0.0318
At 100 passages	0.0706	0.0393
R-Precision	0.0764	

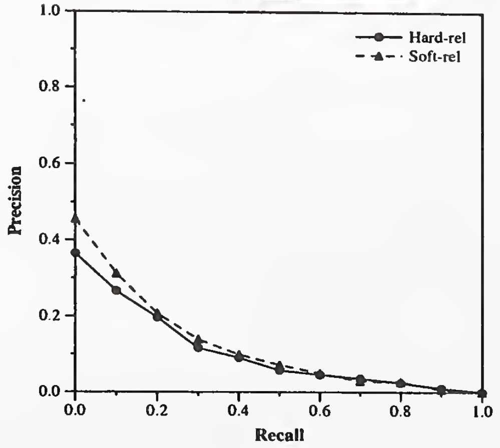


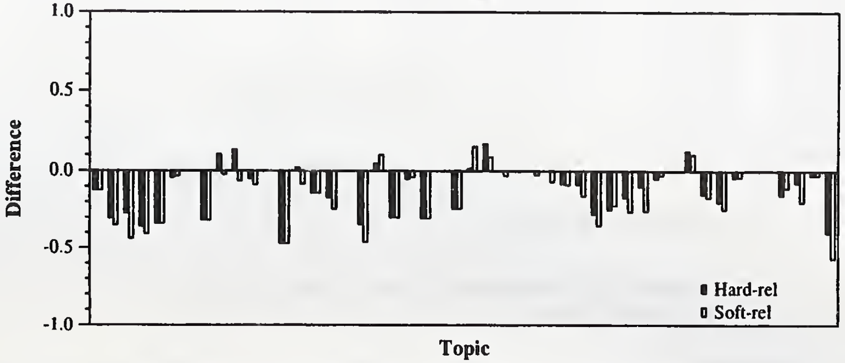
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	28156	28156
Total relevant	5123	7576
Total rel. ret.	1927	2541
MAP	0.0929	0.1049

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.1917	0.2583
At 10 docs	0.2000	0.2521
At 15 docs	0.1903	0.2417
At 20 docs	0.1792	0.2354
At 30 docs	0.1882	0.2437
At 100 docs	0.1502	0.1933
At 200 docs	0.1175	0.1527
At 500 docs	0.0683	0.0895
At 1000 docs	0.0401	0.0529
R-Precision	0.1457	0.1618



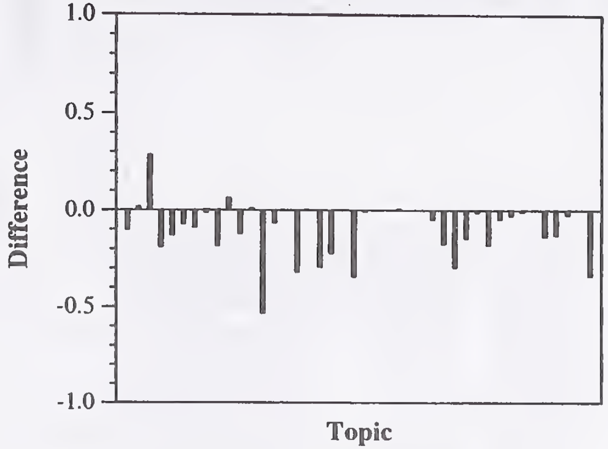


Difference from median in Document based R-precision per topic

Run ID	HKbas1tni1td
Run Description	baseline run; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.1258	0.0060
At 10 passages	0.1058	0.0138
At 15 passages	0.0926	0.0189
At 20 passages	0.0899	0.0200
At 30 passages	0.0812	0.0250
At 50 passages	0.0694	0.0349
At 100 passages	0.0555	0.0378
R-Precision	0.0562	

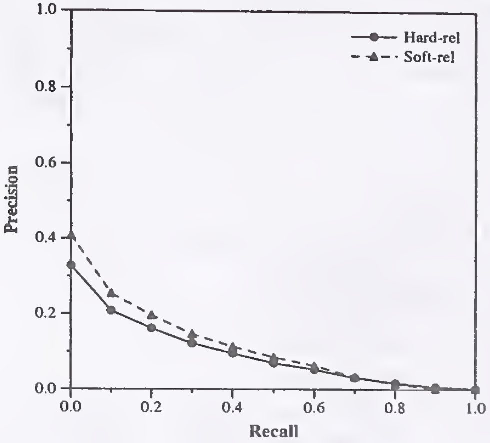


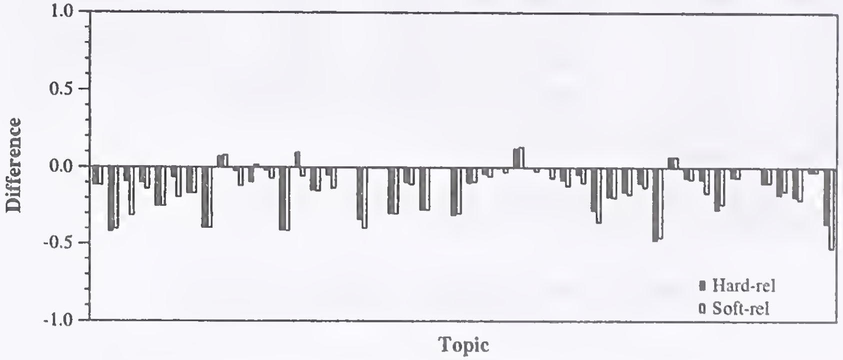
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	41098	41098
Total relevant	5123	7576
Total rel. ret.	2245	3089
MAP	0.0825	0.0986

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.1375	0.2000
At 10 docs	0.1417	0.2021
At 15 docs	0.1542	0.2153
At 20 docs	0.1552	0.2219
At 30 docs	0.1528	0.2083
At 100 docs	0.1369	0.1840
At 200 docs	0.1113	0.1485
At 500 docs	0.0721	0.0983
At 1000 docs	0.0468	0.0644
R-Precision	0.1290	0.1520



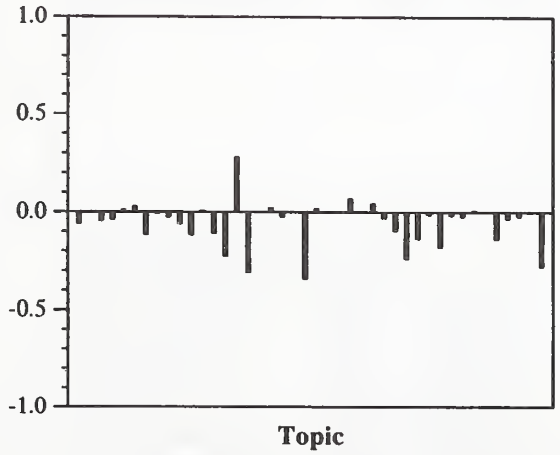


Difference from median in Document based R-precision per topic

Run ID	HKIdoc1t1td
Run Description	baseline run; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2497	0.0584
At 10 passages	0.2113	0.0696
At 15 passages	0.1768	0.0699
At 20 passages	0.1609	0.0743
At 30 passages	0.1440	0.0841
At 50 passages	0.1224	0.0881
At 100 passages	0.0980	0.0912
R-Precision	0.1103	

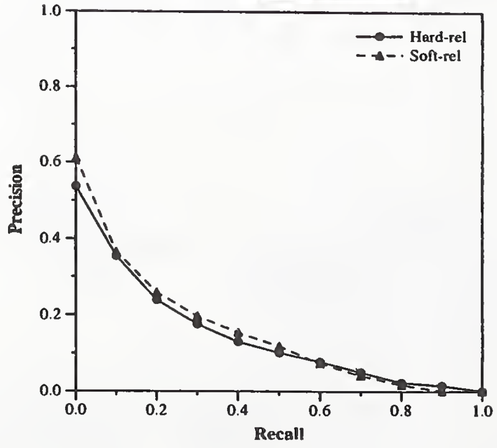


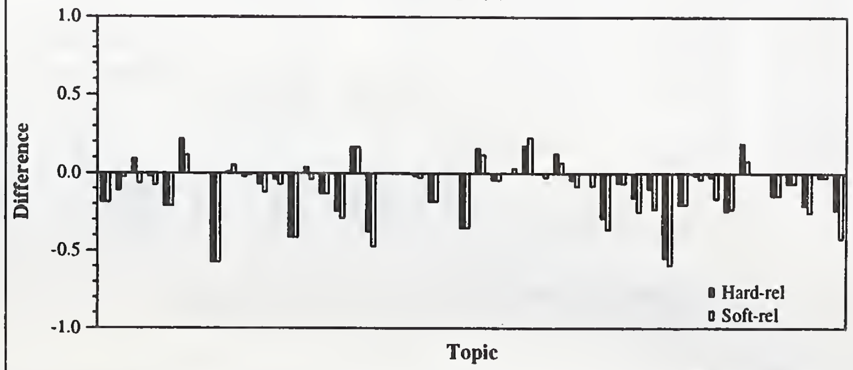
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	2275	3083
MAP	0.1354	0.1454

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3625	0.4333
At 10 docs	0.3021	0.3688
At 15 docs	0.2750	0.3472
At 20 docs	0.2594	0.3333
At 30 docs	0.2361	0.3042
At 100 docs	0.1600	0.2050
At 200 docs	0.1196	0.1550
At 500 docs	0.0723	0.0970
At 1000 docs	0.0474	0.0642
R-Precision	0.1694	0.1875



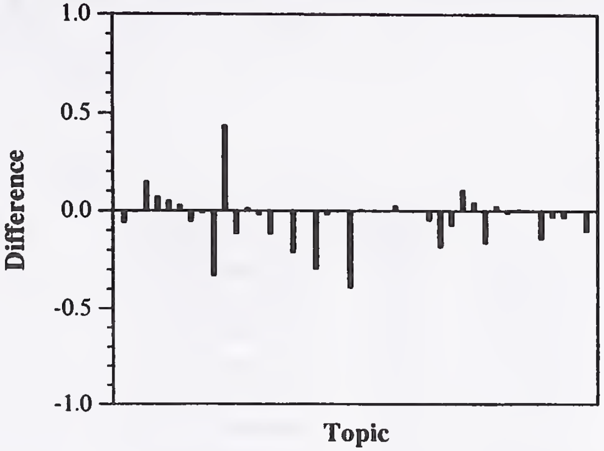


Difference from median in Document based R-precision per topic

Run ID	HKIdoc1t1t
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.1814	0.0315
At 10 passages	0.1725	0.0510
At 15 passages	0.1578	0.0589
At 20 passages	0.1479	0.0681
At 30 passages	0.1516	0.0787
At 50 passages	0.1294	0.0840
At 100 passages	0.1117	0.0985
R-Precision	0.1159	

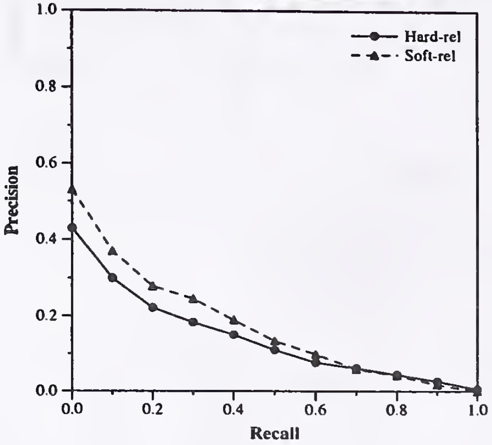


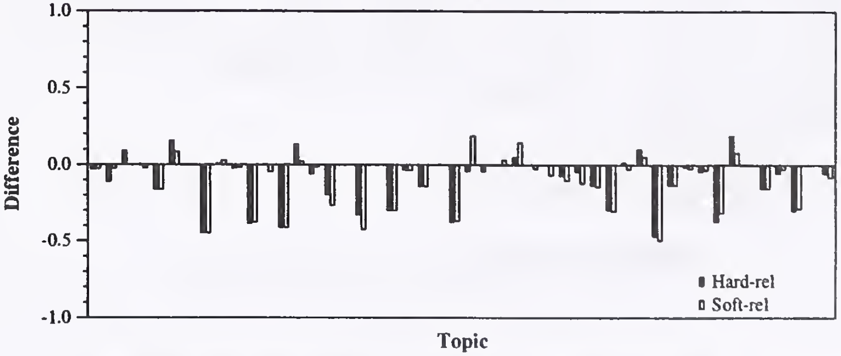
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	45902	45902
Total relevant	5123	7576
Total rel. ret.	2550	3560
MAP	0.1286	0.1615

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.2667	0.3792
At 10 docs	0.2521	0.3479
At 15 docs	0.2431	0.3417
At 20 docs	0.2375	0.3240
At 30 docs	0.2257	0.3049
At 100 docs	0.1571	0.2202
At 200 docs	0.1253	0.1756
At 500 docs	0.0804	0.1112
At 1000 docs	0.0531	0.0742
R-Precision	0.1627	0.2047



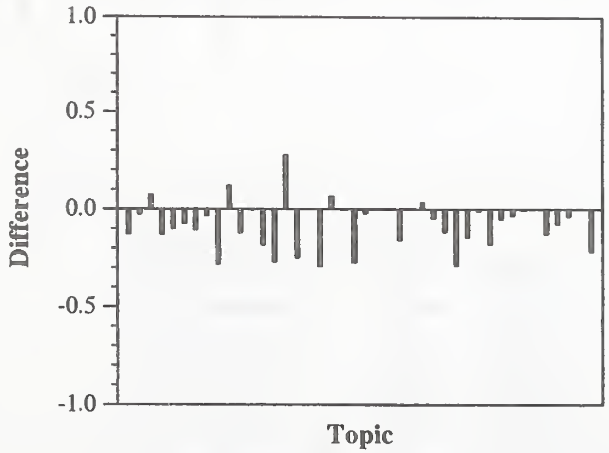


Difference from median in Document based R-precision per topic

Run ID	HKIdoc1ti1td
Run Description	baseline run; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2471	0.0486
At 10 passages	0.2139	0.0523
At 15 passages	0.1832	0.0521
At 20 passages	0.1602	0.0572
At 30 passages	0.1411	0.0625
At 50 passages	0.1151	0.0657
At 100 passages	0.0822	0.0634
R-Precision	0.0780	

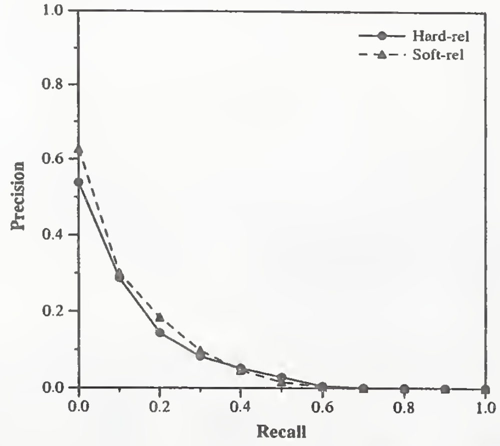


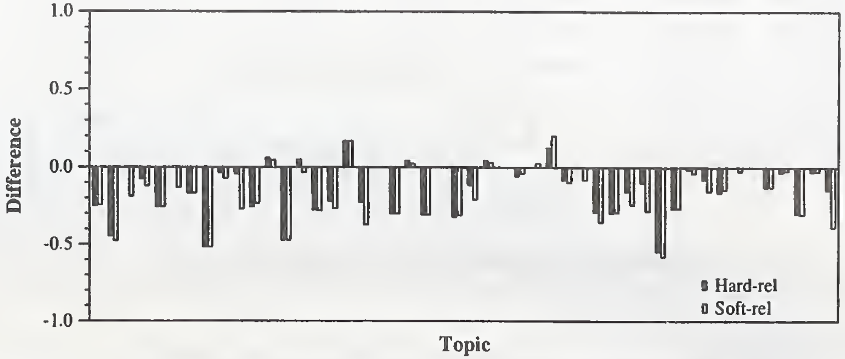
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	43549	43549
Total relevant	5123	7576
Total rel. ret.	1531	2138
MAP	0.0819	0.0898

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3708	0.4500
At 10 docs	0.3333	0.4062
At 15 docs	0.2847	0.3500
At 20 docs	0.2615	0.3333
At 30 docs	0.2292	0.3014
At 100 docs	0.1371	0.1881
At 200 docs	0.0896	0.1226
At 500 docs	0.0504	0.0702
At 1000 docs	0.0319	0.0445
R-Precision	0.1211	0.1345



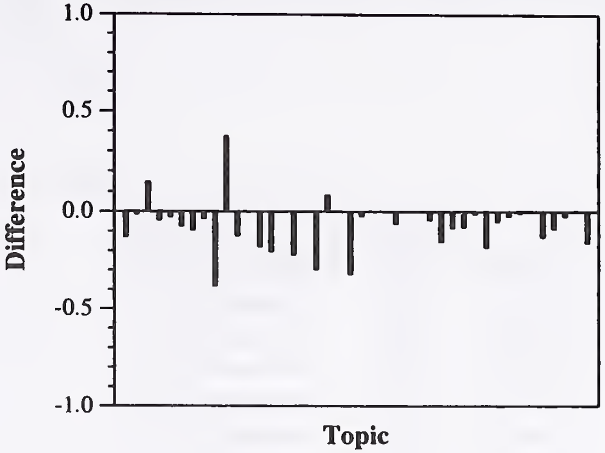


Difference from median in Document based R-precision per topic

Run ID	HKIdoc1ti1t
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2110	0.0299
At 10 passages	0.2026	0.0452
At 15 passages	0.1902	0.0538
At 20 passages	0.1747	0.0610
At 30 passages	0.1605	0.0720
At 50 passages	0.1303	0.0757
At 100 passages	0.0991	0.0785
R-Precision	0.0860	

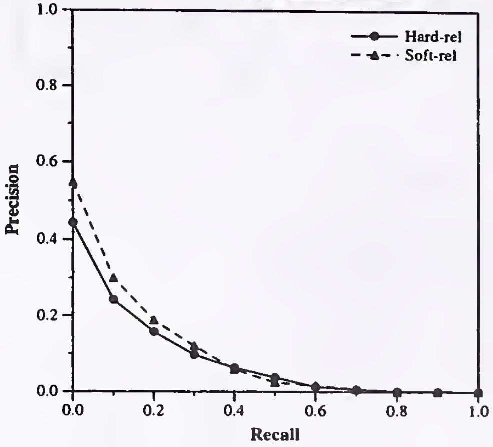


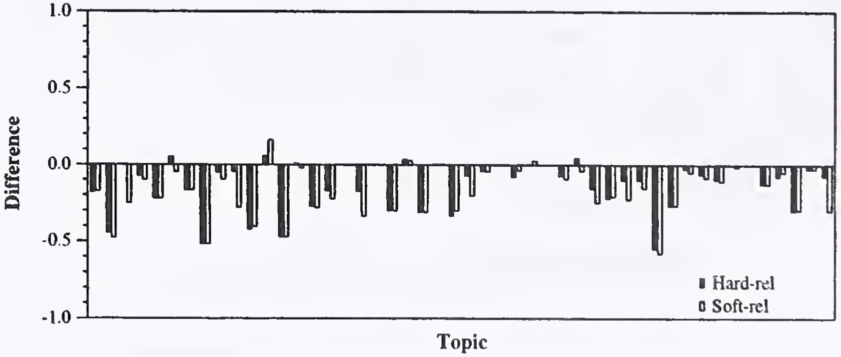
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	36472	36472
Total relevant	5123	7576
Total rel. ret.	1798	2477
MAP	0.0806	0.0937

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3167	0.4167
At 10 docs	0.3021	0.3771
At 15 docs	0.2819	0.3569
At 20 docs	0.2573	0.3375
At 30 docs	0.2396	0.3118
At 100 docs	0.1419	0.1925
At 200 docs	0.0997	0.1344
At 500 docs	0.0573	0.0796
At 1000 docs	0.0375	0.0516
R-Precision	0.1230	0.1402



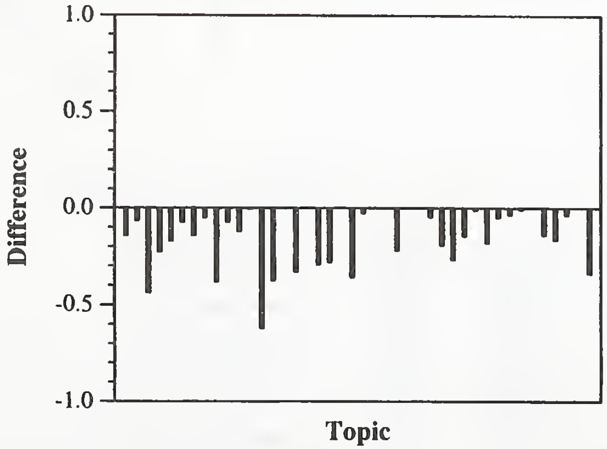


Difference from median in Document based R-precision per topic

Run ID	HKupw1tni1td
Run Description	baseline run; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.0240	0.0009
At 10 passages	0.0136	0.0011
At 15 passages	0.0117	0.0014
At 20 passages	0.0122	0.0023
At 30 passages	0.0090	0.0025
At 50 passages	0.0085	0.0039
At 100 passages	0.0078	0.0055
R-Precision	0.0046	

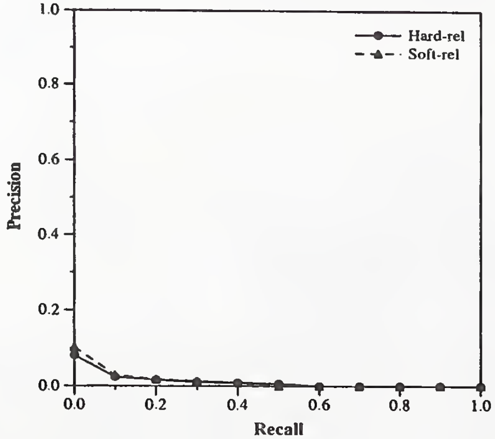


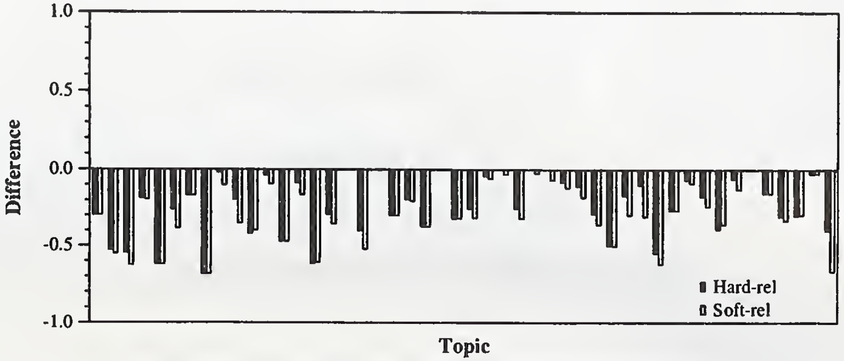
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	953	1248
MAP	0.0083	0.0088

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.0417	0.0542
At 10 docs	0.0312	0.0417
At 15 docs	0.0319	0.0417
At 20 docs	0.0312	0.0385
At 30 docs	0.0271	0.0340
At 100 docs	0.0233	0.0281
At 200 docs	0.0177	0.0228
At 500 docs	0.0179	0.0247
At 1000 docs	0.0199	0.0260
R-Precision	0.0162	0.0228



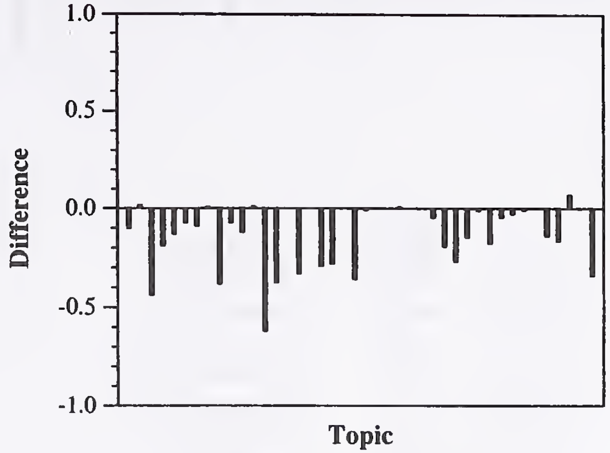


Difference from median in Document based R-precision per topic

Run ID	HKup1tni1td
Run Description	not a baseline, used metadata; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.0659	0.0026
At 10 passages	0.0615	0.0086
At 15 passages	0.0473	0.0120
At 20 passages	0.0461	0.0138
At 30 passages	0.0399	0.0163
At 50 passages	0.0397	0.0244
At 100 passages	0.0280	0.0257
R-Precision	0.0235	

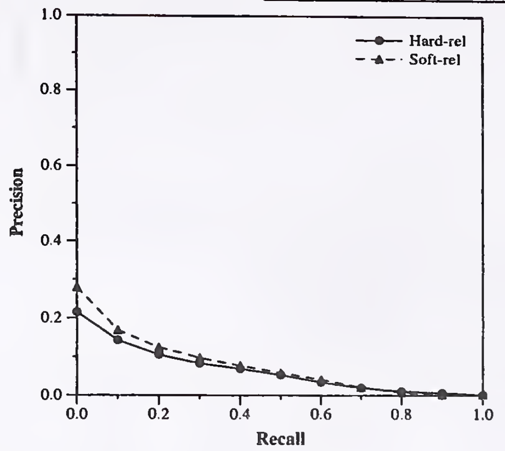


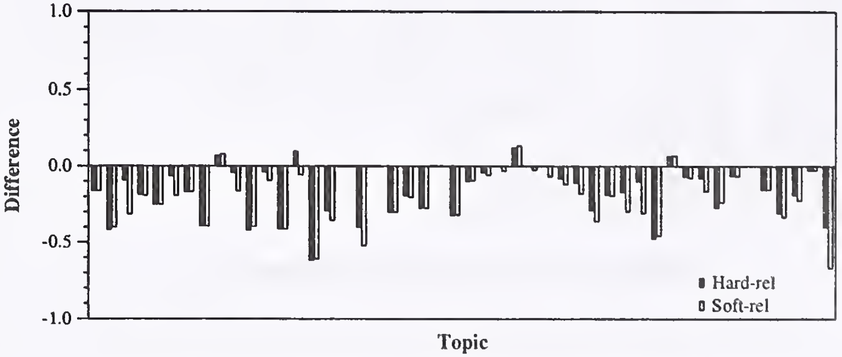
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	43695	43695
Total relevant	5123	7576
Total rel. ret.	1791	2426
MAP	0.0558	0.0656

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.0792	0.1417
At 10 docs	0.0875	0.1313
At 15 docs	0.0986	0.1361
At 20 docs	0.1000	0.1417
At 30 docs	0.1035	0.1437
At 100 docs	0.0973	0.1254
At 200 docs	0.0776	0.1018
At 500 docs	0.0515	0.0710
At 1000 docs	0.0373	0.0505
R-Precision	0.0935	0.1042



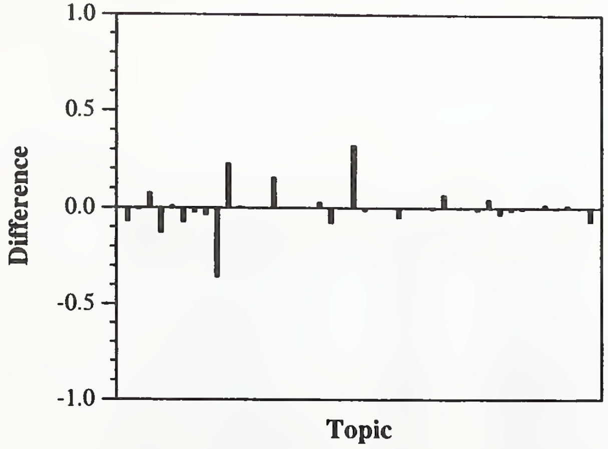


Difference from median in Document based R-precision per topic

Run ID	uiuchard
Run Description	baseline run; fields not specified

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2251	0.0453
At 10 passages	0.2409	0.0720
At 15 passages	0.2169	0.0835
At 20 passages	0.2102	0.0973
At 30 passages	0.1819	0.1084
At 50 passages	0.1798	0.1302
At 100 passages	0.1591	0.1530
R-Precision	0.1563	

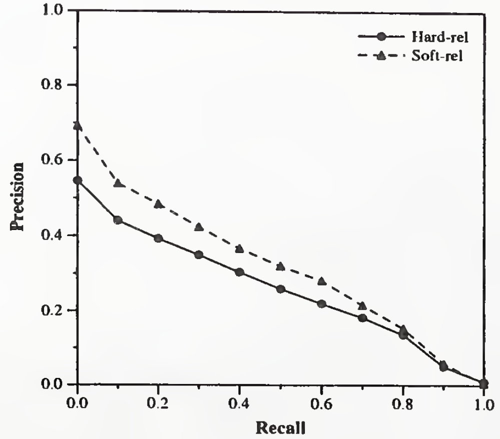


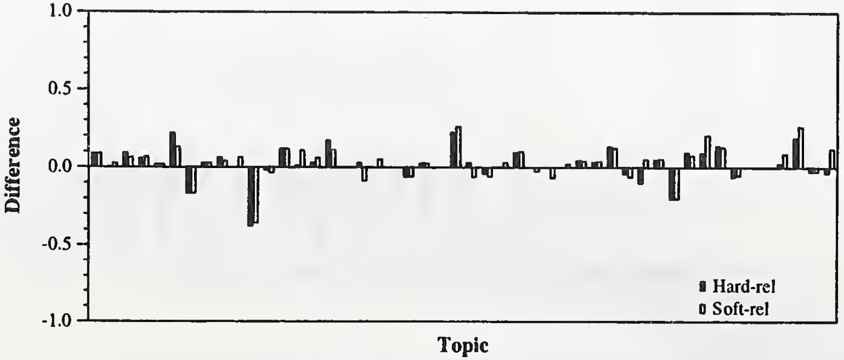
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3618	5105
MAP	0.2505	0.3077

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3667	0.5125
At 10 docs	0.3812	0.5125
At 15 docs	0.3806	0.4944
At 20 docs	0.3687	0.4854
At 30 docs	0.3396	0.4465
At 100 docs	0.2848	0.3750
At 200 docs	0.2136	0.2903
At 500 docs	0.1247	0.1731
At 1000 docs	0.0754	0.1064
R-Precision	0.2757	0.3327



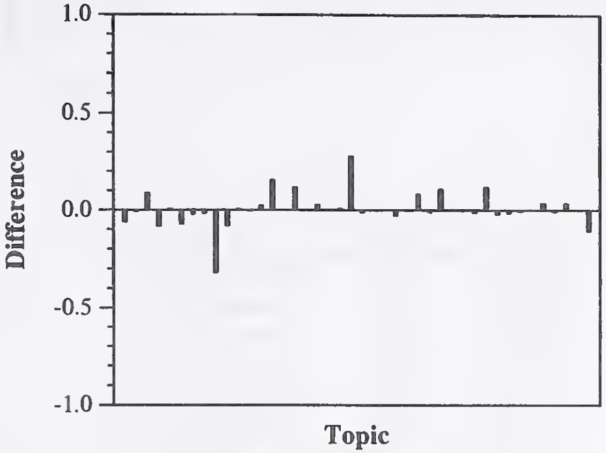


Difference from median in Document based R-precision per topic

Run ID	UIHARD1
Run Description	not a baseline, used form; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2624	0.0545
At 10 passages	0.2442	0.0776
At 15 passages	0.2399	0.0904
At 20 passages	0.2234	0.1044
At 30 passages	0.2017	0.1209
At 50 passages	0.1894	0.1407
At 100 passages	0.1732	0.1662
R-Precision	0.1625	

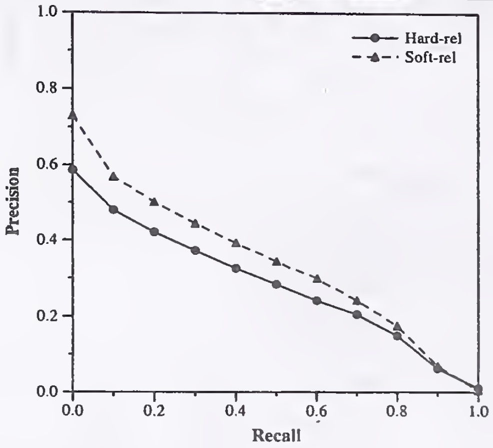


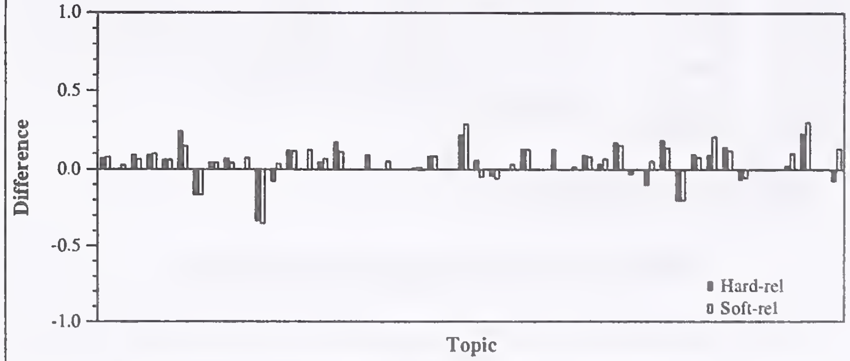
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3727	5264
MAP	0.2727	0.3286

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4292	0.5792
At 10 docs	0.4000	0.5292
At 15 docs	0.4097	0.5236
At 20 docs	0.3906	0.5062
At 30 docs	0.3694	0.4771
At 100 docs	0.2994	0.3925
At 200 docs	0.2243	0.3024
At 500 docs	0.1302	0.1793
At 1000 docs	0.0776	0.1097
R-Precision	0.2908	0.3512



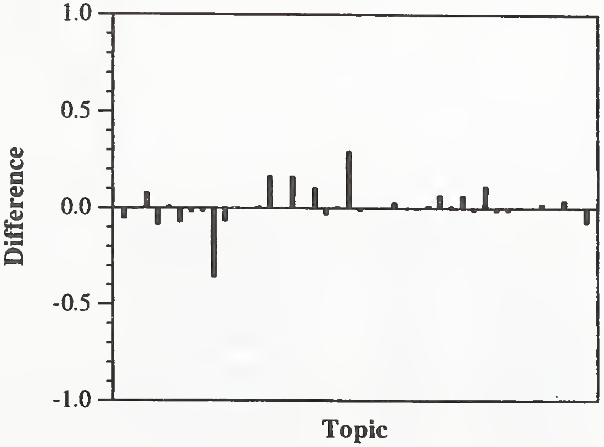


Difference from median in Document based R-precision per topic

Run ID	UIHARD2
Run Description	not a baseline, used form; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2515	0.0515
At 10 passages	0.2527	0.0822
At 15 passages	0.2358	0.0918
At 20 passages	0.2211	0.1052
At 30 passages	0.1992	0.1222
At 50 passages	0.1859	0.1396
At 100 passages	0.1797	0.1706
R-Precision	0.1672	

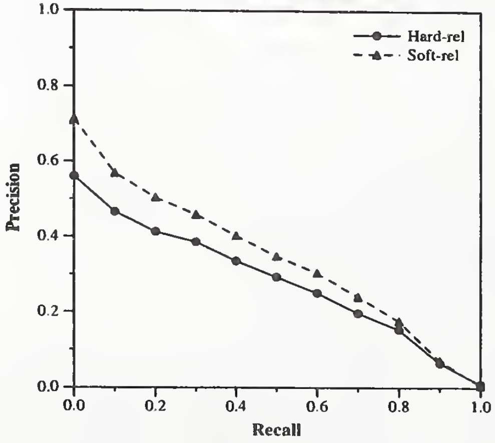


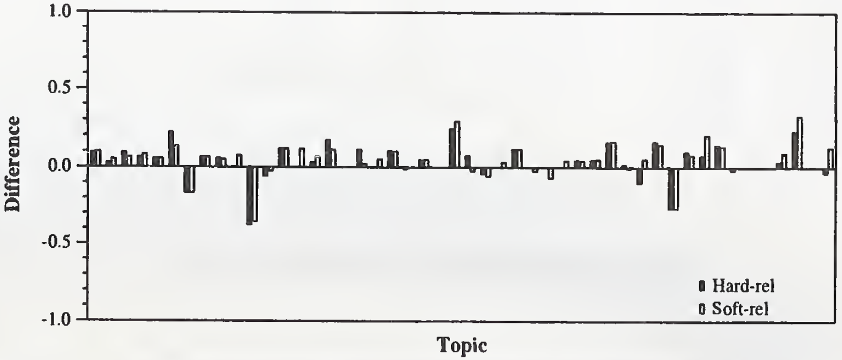
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3710	5280
MAP	0.2737	0.3321

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.3833	0.5542
At 10 docs	0.4021	0.5312
At 15 docs	0.4028	0.5181
At 20 docs	0.3885	0.5031
At 30 docs	0.3667	0.4715
At 100 docs	0.2981	0.3915
At 200 docs	0.2227	0.3024
At 500 docs	0.1287	0.1786
At 1000 docs	0.0773	0.1100
R-Precision	0.2888	0.3497



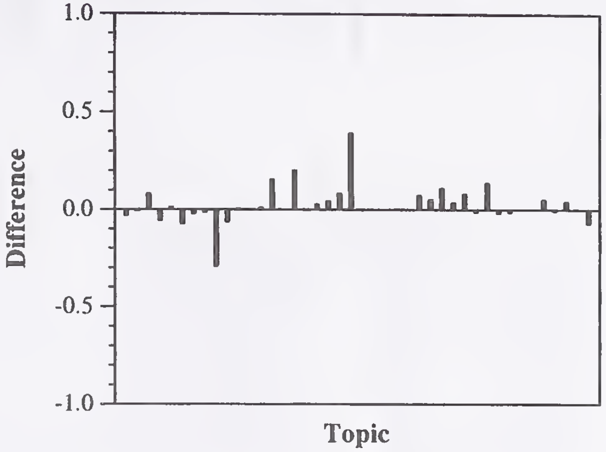


Difference from median in Document based R-precision per topic

Run ID	UIHARD3
Run Description	not a baseline, used form; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3034	0.0639
At 10 passages	0.2705	0.0915
At 15 passages	0.2455	0.0973
At 20 passages	0.2301	0.1111
At 30 passages	0.2165	0.1291
At 50 passages	0.2023	0.1509
At 100 passages	0.1809	0.1759
R-Precision	0.1792	

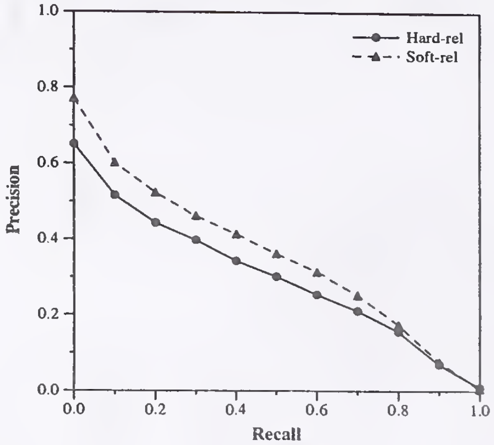


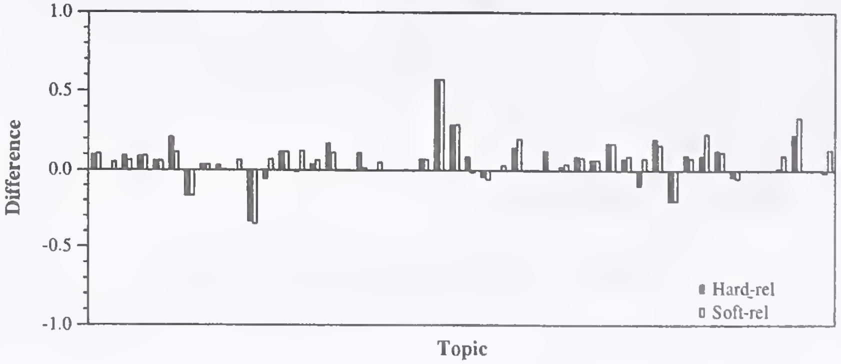
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3730	5270
MAP	0.2898	0.3465

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4417	0.6000
At 10 docs	0.4312	0.5625
At 15 docs	0.4153	0.5375
At 20 docs	0.4000	0.5219
At 30 docs	0.3771	0.4944
At 100 docs	0.3050	0.4004
At 200 docs	0.2270	0.3073
At 500 docs	0.1302	0.1798
At 1000 docs	0.0777	0.1098
R-Precision	0.3087	0.3701



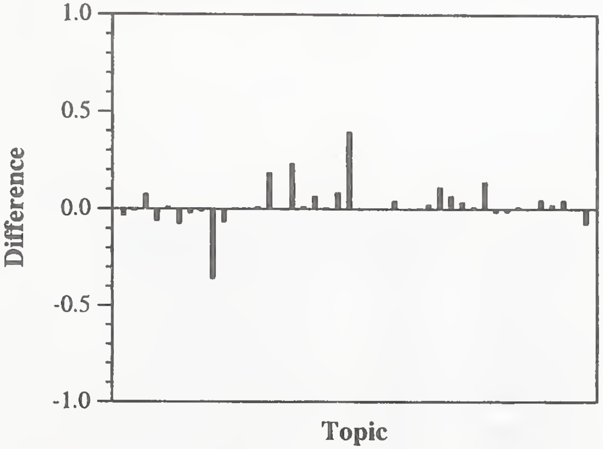


Difference from median in Document based R-precision per topic

Run ID	UIHARD4
Run Description	not a baseline, used form; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2625	0.0611
At 10 passages	0.2662	0.0908
At 15 passages	0.2395	0.0938
At 20 passages	0.2317	0.1105
At 30 passages	0.2215	0.1345
At 50 passages	0.2065	0.1547
At 100 passages	0.1784	0.1758
R-Precision	0.1796	

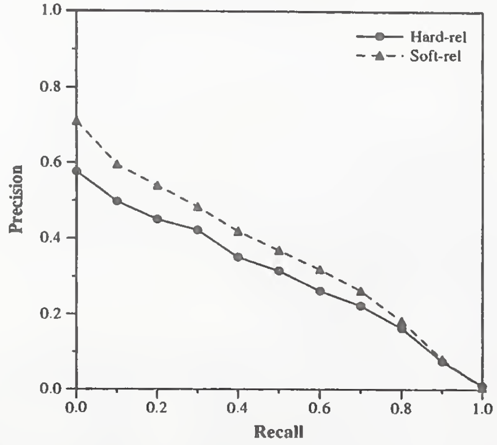


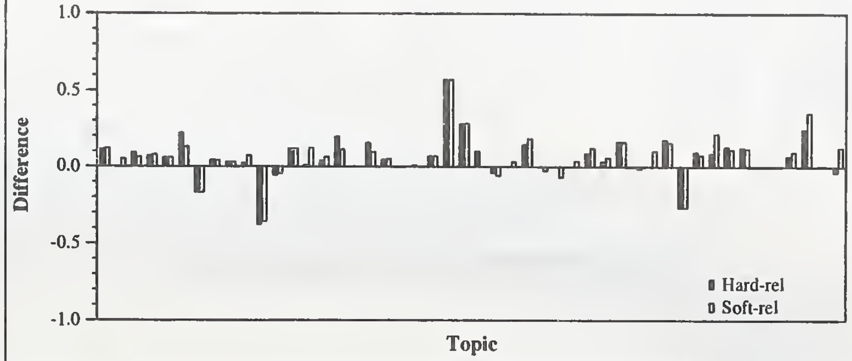
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3726	5282
MAP	0.2939	0.3510

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4083	0.5792
At 10 docs	0.4208	0.5604
At 15 docs	0.4111	0.5292
At 20 docs	0.4052	0.5167
At 30 docs	0.3826	0.4889
At 100 docs	0.3065	0.4019
At 200 docs	0.2296	0.3101
At 500 docs	0.1289	0.1792
At 1000 docs	0.0776	0.1100
R-Precision	0.3109	0.3687



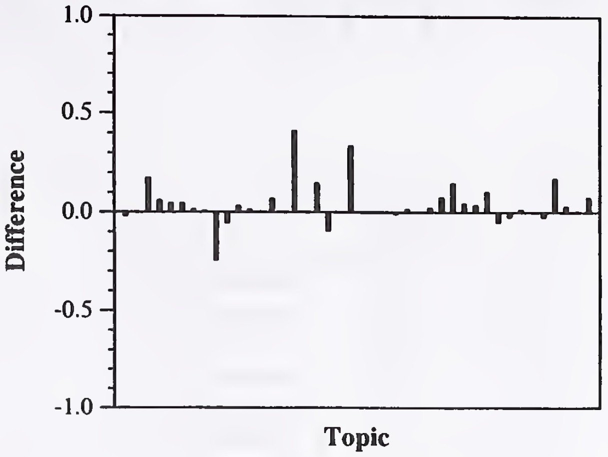


Difference from median in Document based R-precision per topic

Run ID	umdbsne2tit
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2759	0.0609
At 10 passages	0.2862	0.0968
At 15 passages	0.2488	0.0962
At 20 passages	0.2539	0.1130
At 30 passages	0.2289	0.1249
At 50 passages	0.2052	0.1443
At 100 passages	0.1945	0.1810
R-Precision	0.1972	

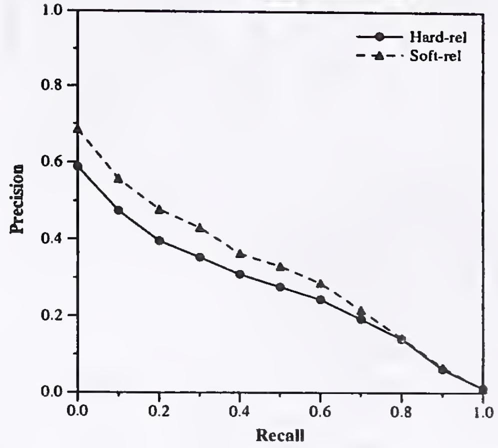


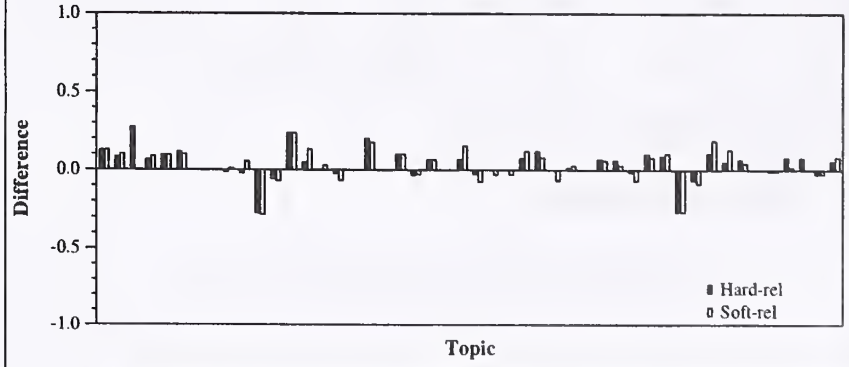
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3606	5098
MAP	0.2617	0.3110

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4042	0.5208
At 10 docs	0.3979	0.5000
At 15 docs	0.3764	0.4847
At 20 docs	0.3719	0.4802
At 30 docs	0.3479	0.4542
At 100 docs	0.2854	0.3773
At 200 docs	0.2161	0.2893
At 500 docs	0.1249	0.1723
At 1000 docs	0.0751	0.1062
R-Precision	0.2888	0.3317



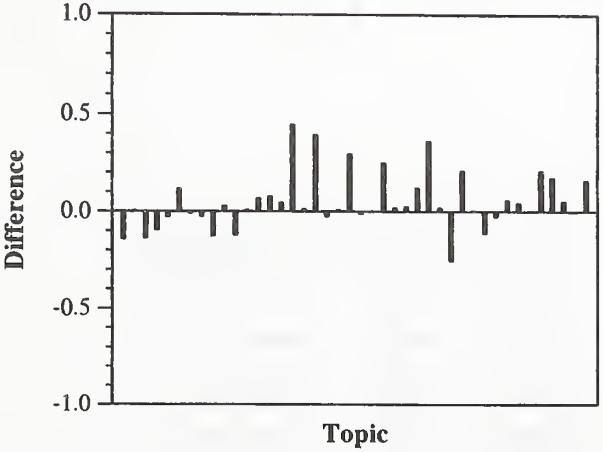


Difference from median in Document based R-precision per topic

Run ID	UMARIPVR4
Run Description	not a baseline, used form; title+desc+narr

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3752	0.0844
At 10 passages	0.3593	0.1220
At 15 passages	0.3145	0.1242
At 20 passages	0.3019	0.1350
At 30 passages	0.2838	0.1576
At 50 passages	0.2396	0.1707
At 100 passages	0.1918	0.1876
R-Precision	0.2167	

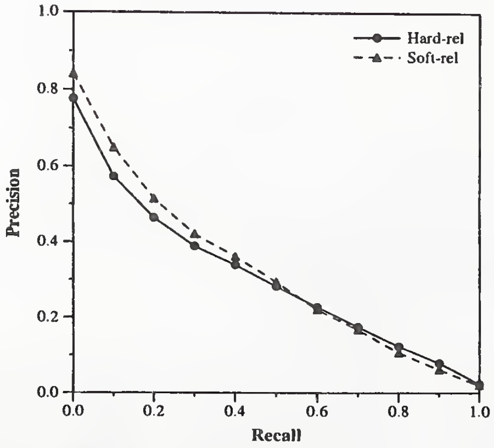


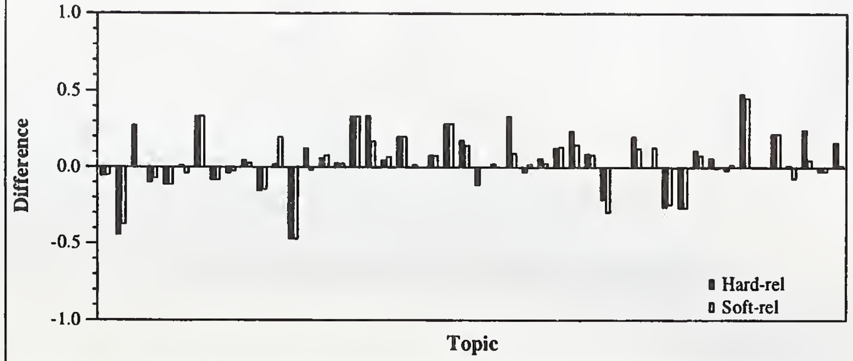
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3813	5213
MAP	0.2901	0.3103

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5292	0.6375
At 10 docs	0.4875	0.5896
At 15 docs	0.4542	0.5514
At 20 docs	0.4365	0.5375
At 30 docs	0.4104	0.5056
At 100 docs	0.2854	0.3698
At 200 docs	0.2077	0.2723
At 500 docs	0.1169	0.1577
At 1000 docs	0.0794	0.1086
R-Precision	0.3042	0.3300



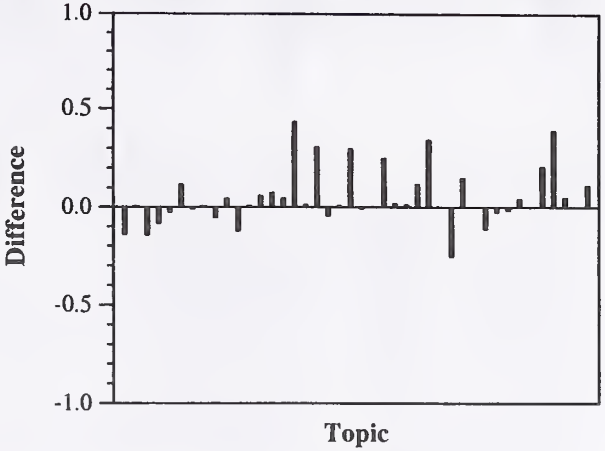


Difference from median in Document based R-precision per topic

Run ID	UMARIPVR5
Run Description	not a baseline, used both; title+desc+narr

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3407	0.0727
At 10 passages	0.3504	0.1159
At 15 passages	0.3160	0.1249
At 20 passages	0.3053	0.1353
At 30 passages	0.2758	0.1513
At 50 passages	0.2498	0.1760
At 100 passages	0.1958	0.1900
R-Precision	0.2142	

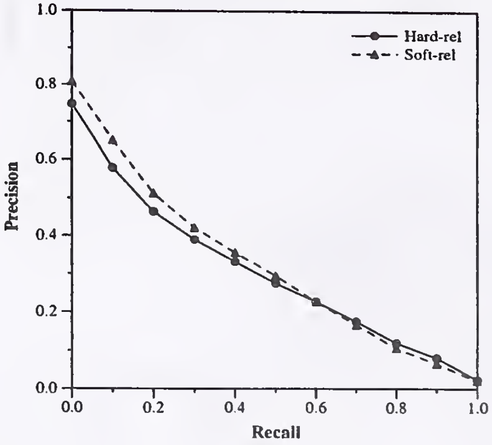


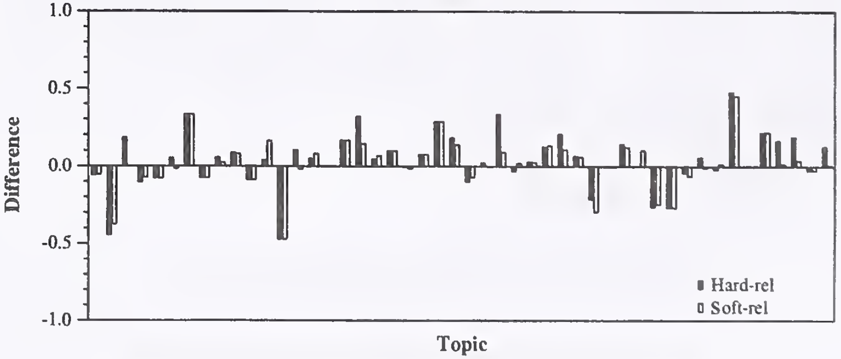
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3809	5221
MAP	0.2895	0.3111

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5000	0.5958
At 10 docs	0.4771	0.5646
At 15 docs	0.4542	0.5528
At 20 docs	0.4313	0.5260
At 30 docs	0.3965	0.4910
At 100 docs	0.2898	0.3744
At 200 docs	0.2076	0.2691
At 500 docs	0.1175	0.1570
At 1000 docs	0.0794	0.1088
R-Precision	0.2969	0.3222



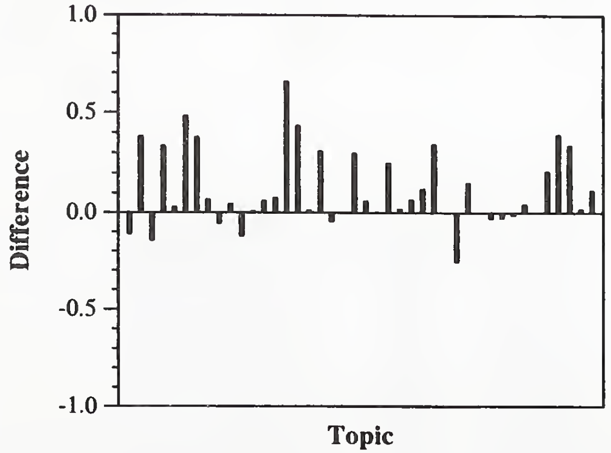


Difference from median in Document based R-precision per topic

Run ID	UMARIPVR7
Run Description	not a baseline, used both; title+desc+narr

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.4232	0.0665
At 10 passages	0.3950	0.1038
At 15 passages	0.3491	0.1104
At 20 passages	0.3390	0.1206
At 30 passages	0.3147	0.1370
At 50 passages	0.2960	0.1630
At 100 passages	0.2492	0.1887
R-Precision	0.3024	

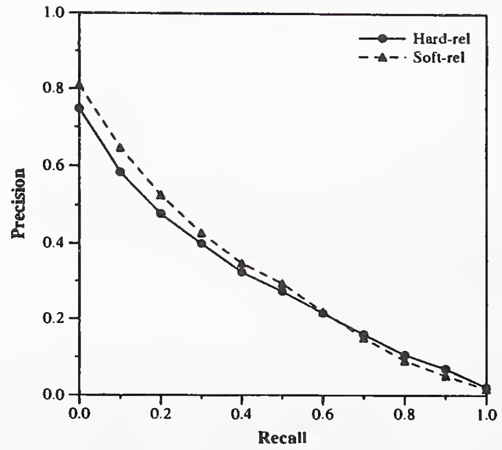


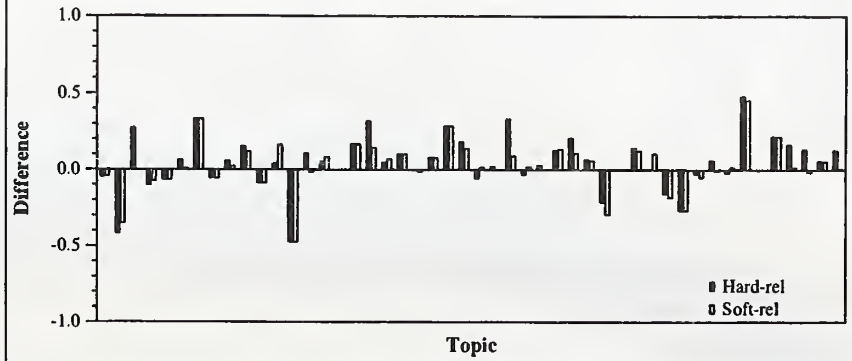
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	37892	37892
Total relevant	5123	7576
Total rel. ret.	3450	4781
MAP	0.2875	0.3072

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5125	0.6083
At 10 docs	0.4771	0.5667
At 15 docs	0.4514	0.5514
At 20 docs	0.4365	0.5313
At 30 docs	0.3979	0.4910
At 100 docs	0.2954	0.3781
At 200 docs	0.2123	0.2755
At 500 docs	0.1158	0.1547
At 1000 docs	0.0719	0.0996
R-Precision	0.3058	0.3282



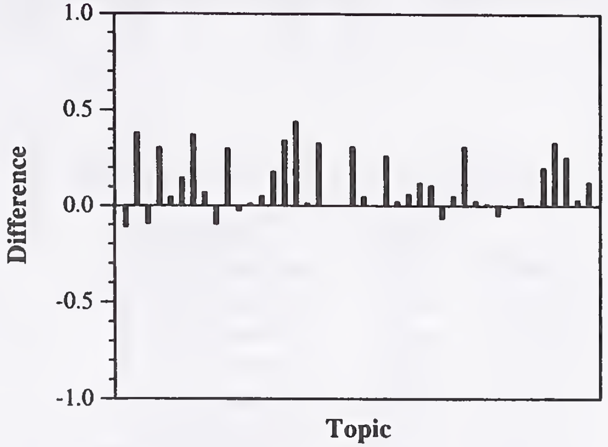


Difference from median in Document based R-precision per topic

Run ID	UMARIPVR8
Run Description	not a baseline, used both; title+desc+narr

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.4348	0.0664
At 10 passages	0.3973	0.1034
At 15 passages	0.3543	0.0992
At 20 passages	0.3449	0.1154
At 30 passages	0.3176	0.1355
At 50 passages	0.2858	0.1561
At 100 passages	0.2587	0.1968
R-Precision	0.2928	

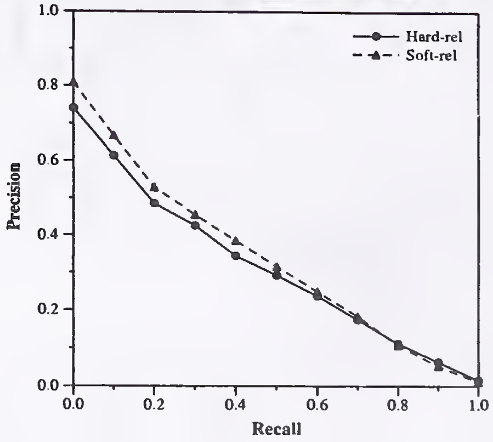


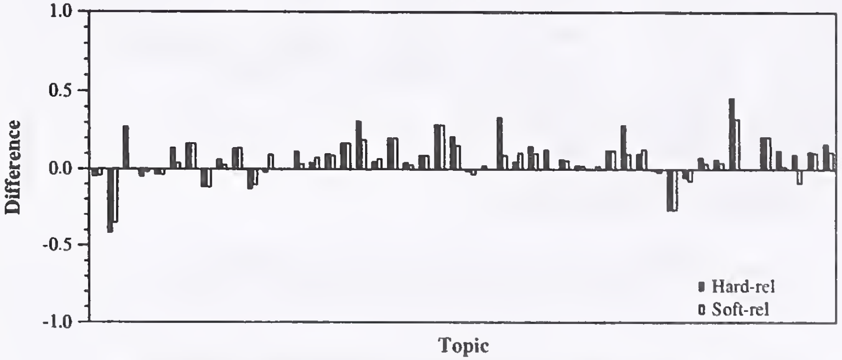
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	37868	37868
Total relevant	5123	7576
Total rel. ret.	3447	4776
MAP	0.3001	0.3253

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5417	0.6250
At 10 docs	0.5063	0.5917
At 15 docs	0.4681	0.5528
At 20 docs	0.4417	0.5323
At 30 docs	0.4083	0.5014
At 100 docs	0.3060	0.3908
At 200 docs	0.2248	0.2989
At 500 docs	0.1235	0.1672
At 1000 docs	0.0718	0.0995
R-Precision	0.3347	0.3521



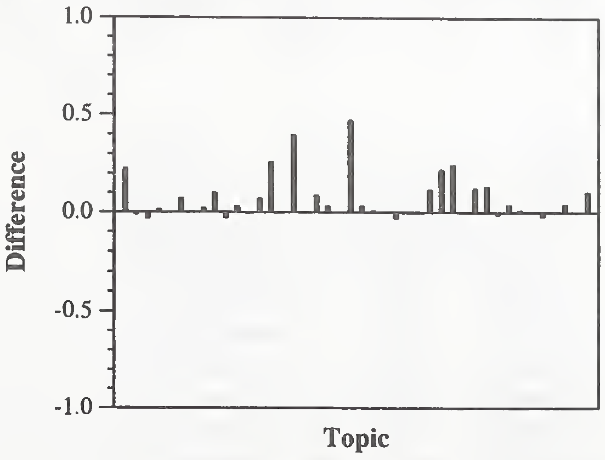


Difference from median in Document based R-precision per topic

Run ID	ciirtbas
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3094	0.0764
At 10 passages	0.3025	0.1087
At 15 passages	0.2859	0.1161
At 20 passages	0.2801	0.1306
At 30 passages	0.2669	0.1569
At 50 passages	0.2497	0.1849
At 100 passages	0.2121	0.2090
R-Precision	0.2261	

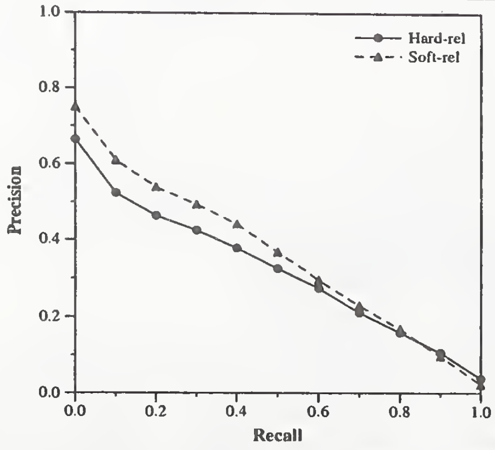


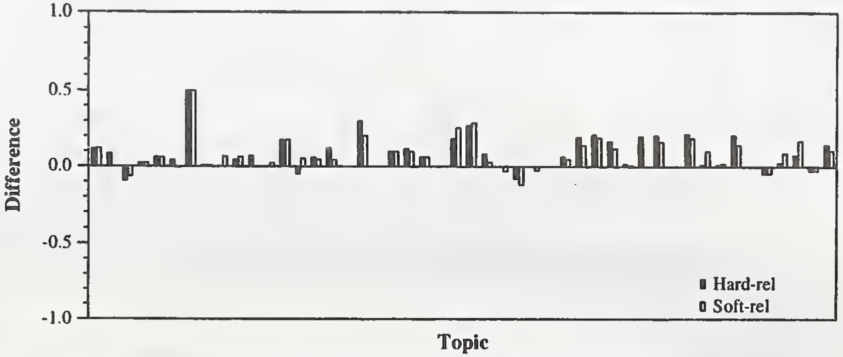
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3613	5062
MAP	0.3091	0.3518

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4750	0.6375
At 10 docs	0.4500	0.5792
At 15 docs	0.4375	0.5528
At 20 docs	0.4250	0.5365
At 30 docs	0.4090	0.5181
At 100 docs	0.3221	0.4244
At 200 docs	0.2291	0.3070
At 500 docs	0.1278	0.1756
At 1000 docs	0.0753	0.1055
R-Precision	0.3368	0.3754



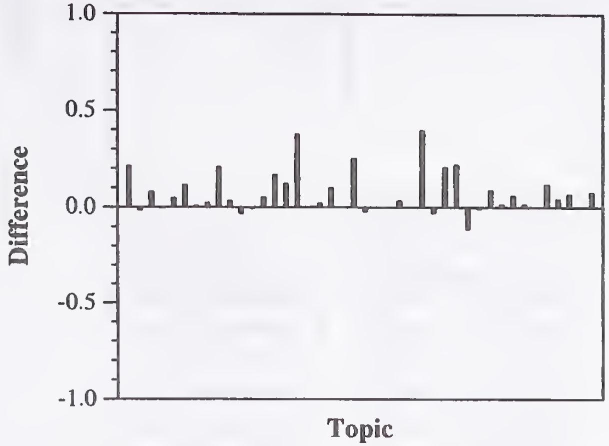


Difference from median in Document based R-precision per topic

Run ID	ciirtdbas
Run Description	baseline run; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3503	0.0769
At 10 passages	0.3341	0.1122
At 15 passages	0.2893	0.1205
At 20 passages	0.2820	0.1380
At 30 passages	0.2738	0.1668
At 50 passages	0.2533	0.1870
At 100 passages	0.2125	0.2063
R-Precision	0.2301	

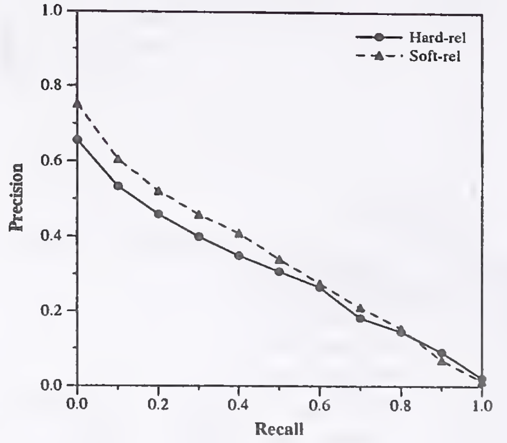


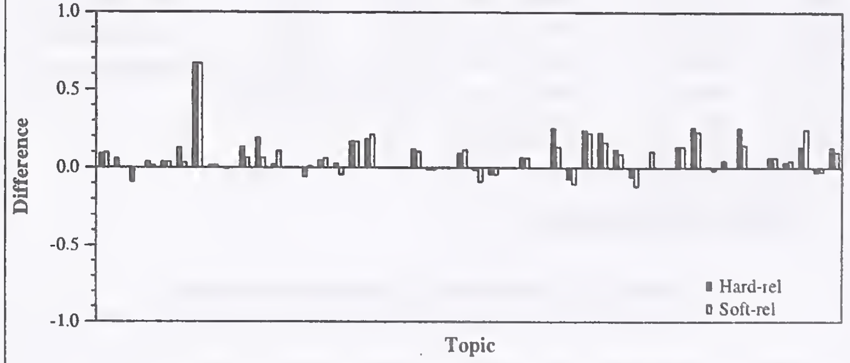
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3579	4834
MAP	0.2969	0.3314

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5042	0.6208
At 10 docs	0.4604	0.5771
At 15 docs	0.4278	0.5403
At 20 docs	0.4156	0.5198
At 30 docs	0.4104	0.5049
At 100 docs	0.3071	0.3881
At 200 docs	0.2239	0.2876
At 500 docs	0.1249	0.1640
At 1000 docs	0.0746	0.1007
R-Precision	0.3300	0.3677



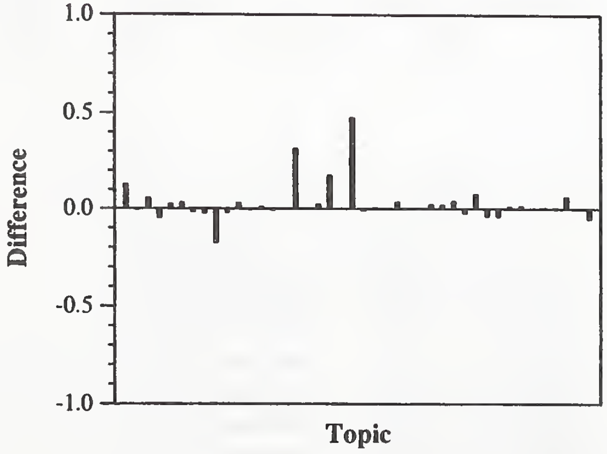


Difference from median in Document based R-precision per topic

Run ID	ciirtpsgbas
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3006	0.0629
At 10 passages	0.2676	0.0909
At 15 passages	0.2291	0.0887
At 20 passages	0.2260	0.1007
At 30 passages	0.2037	0.1146
At 50 passages	0.1964	0.1364
At 100 passages	0.1705	0.1576
R-Precision	0.1853	

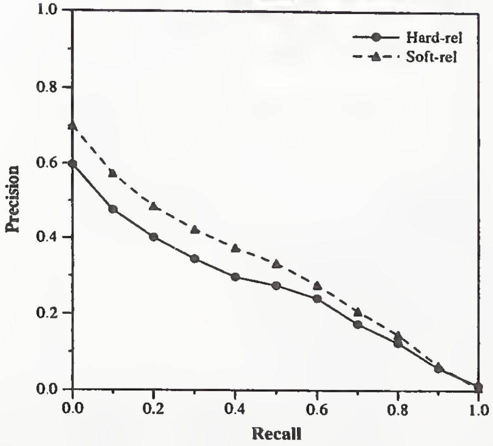


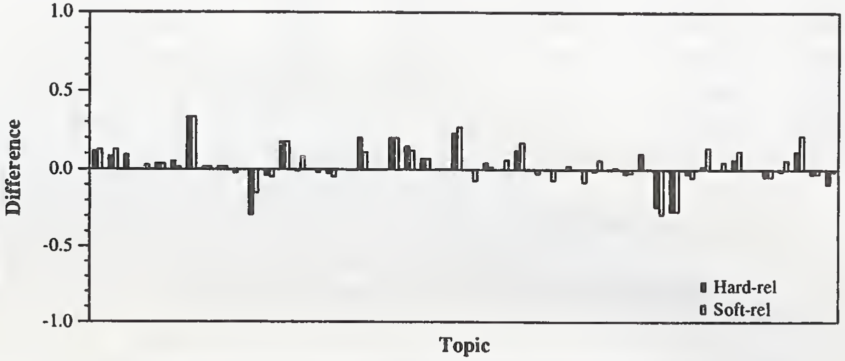
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3613	5062
MAP	0.2529	0.3052

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4125	0.5167
At 10 docs	0.3646	0.4750
At 15 docs	0.3556	0.4708
At 20 docs	0.3438	0.4719
At 30 docs	0.3292	0.4500
At 100 docs	0.2615	0.3583
At 200 docs	0.2111	0.2878
At 500 docs	0.1248	0.1743
At 1000 docs	0.0753	0.1055
R-Precision	0.2788	0.3344



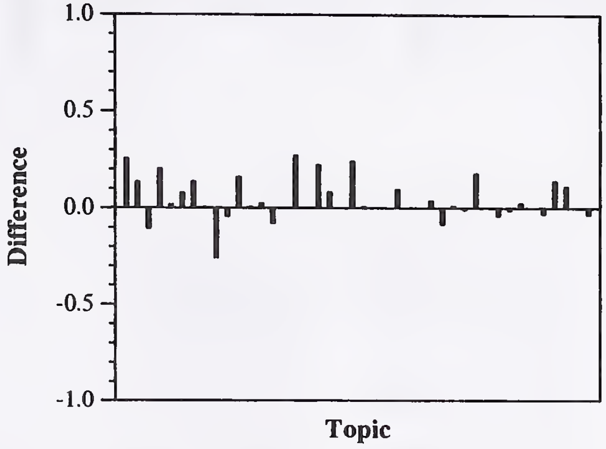


Difference from median in Document based R-precision per topic

Run ID	ciirtp
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3257	0.0368
At 10 passages	0.2999	0.0519
At 15 passages	0.2738	0.0525
At 20 passages	0.2563	0.0590
At 30 passages	0.2377	0.0676
At 50 passages	0.2187	0.0751
At 100 passages	0.1835	0.0929
R-Precision	0.2093	

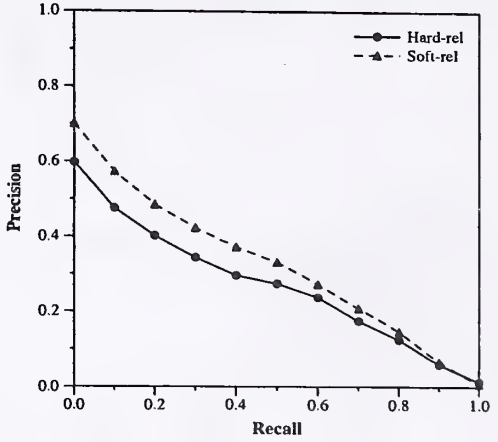


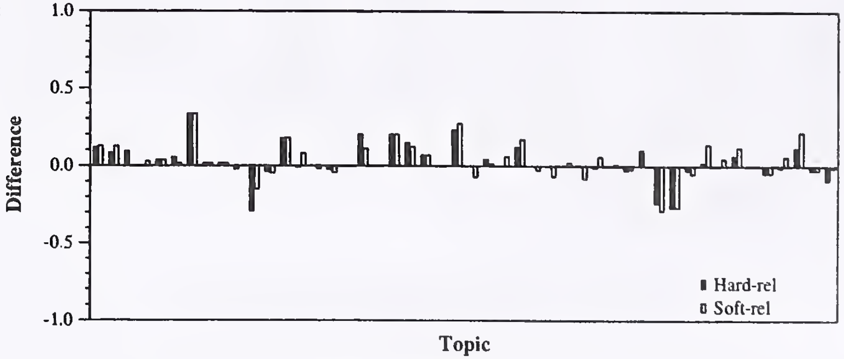
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3613	5062
MAP	0.2526	0.3049

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4125	0.5167
At 10 docs	0.3646	0.4750
At 15 docs	0.3542	0.4694
At 20 docs	0.3438	0.4719
At 30 docs	0.3292	0.4500
At 100 docs	0.2615	0.3583
At 200 docs	0.2111	0.2878
At 500 docs	0.1248	0.1743
At 1000 docs	0.0753	0.1055
R-Precision	0.2788	0.3344



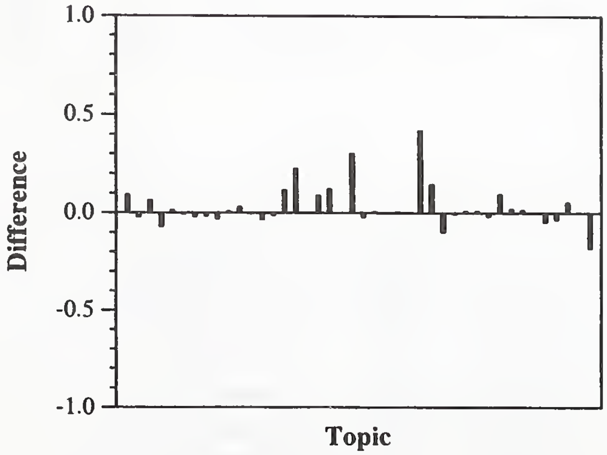


Difference from median in Document based R-precision per topic

Run ID	ciirtdpsgbas
Run Description	baseline run; title+desc

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3186	0.0869
At 10 passages	0.3027	0.1091
At 15 passages	0.2661	0.1068
At 20 passages	0.2474	0.1180
At 30 passages	0.2276	0.1340
At 50 passages	0.1936	0.1416
At 100 passages	0.1698	0.1627
R-Precision	0.1942	

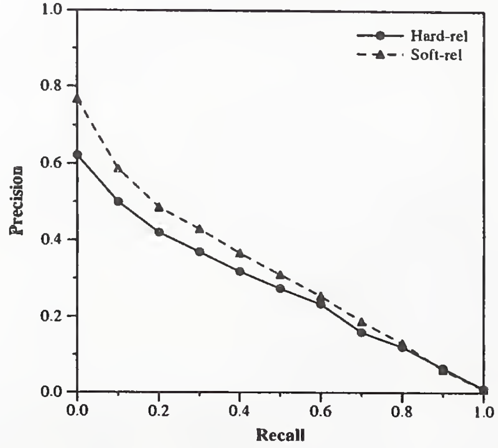


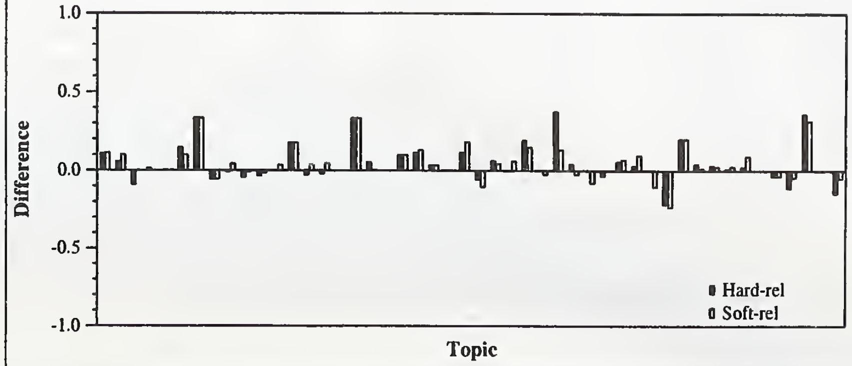
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3579	4834
MAP	0.2640	0.3029

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4417	0.5792
At 10 docs	0.4167	0.5188
At 15 docs	0.3931	0.4944
At 20 docs	0.3708	0.4708
At 30 docs	0.3431	0.4403
At 100 docs	0.2646	0.3460
At 200 docs	0.2130	0.2811
At 500 docs	0.1282	0.1707
At 1000 docs	0.0746	0.1007
R-Precision	0.3004	0.3510



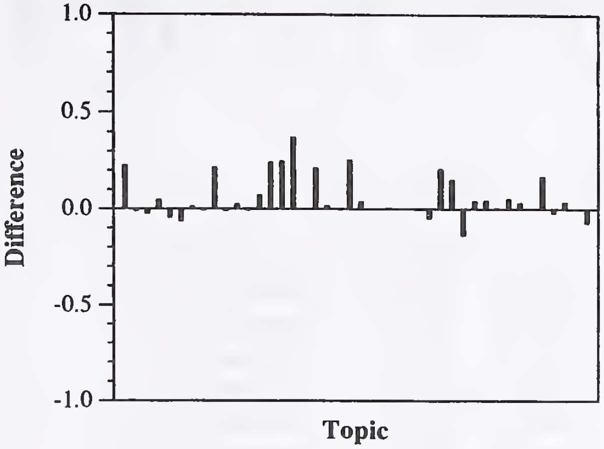


Difference from median in Document based R-precision per topic

Run ID	ciirtcftt
Run Description	not a baseline, used form; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3522	0.0927
At 10 passages	0.3225	0.1249
At 15 passages	0.3111	0.1281
At 20 passages	0.3144	0.1469
At 30 passages	0.2930	0.1676
At 50 passages	0.2536	0.1817
At 100 passages	0.2041	0.1974
R-Precision	0.2229	

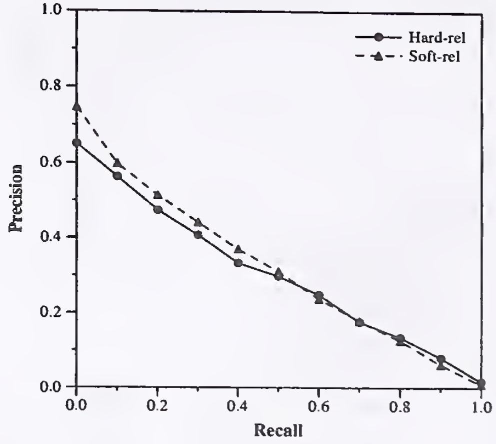


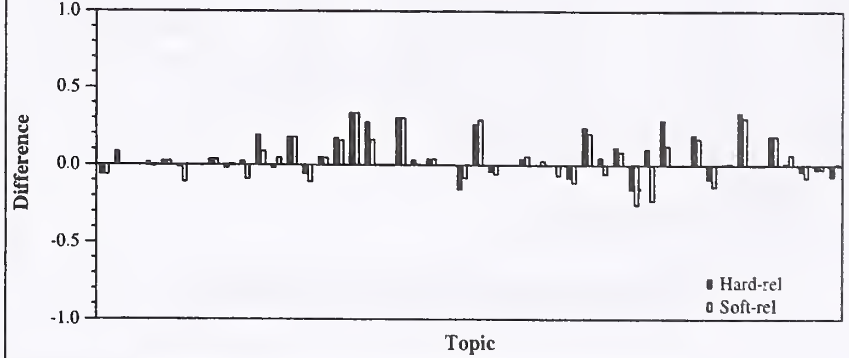
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3261	4397
MAP	0.2942	0.3146

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5125	0.6500
At 10 docs	0.4729	0.5896
At 15 docs	0.4569	0.5653
At 20 docs	0.4469	0.5448
At 30 docs	0.4090	0.5014
At 100 docs	0.2950	0.3602
At 200 docs	0.2102	0.2645
At 500 docs	0.1144	0.1515
At 1000 docs	0.0679	0.0916
R-Precision	0.3123	0.3345



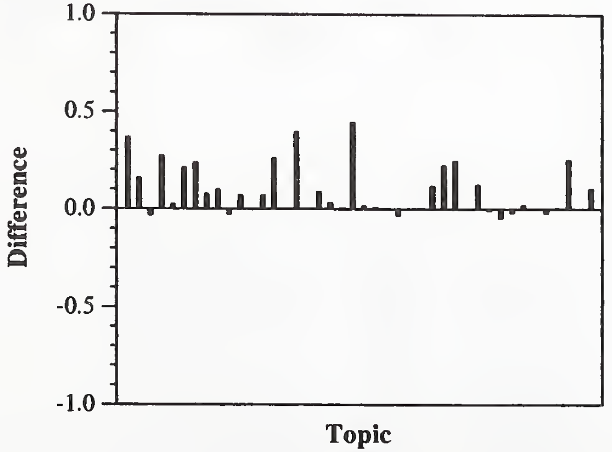


Difference from median in Document based R-precision per topic

Run ID	ciirtmdap
Run Description	not a baseline, used metadata; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3459	0.0573
At 10 passages	0.3120	0.0880
At 15 passages	0.2984	0.0903
At 20 passages	0.2920	0.1007
At 30 passages	0.2804	0.1268
At 50 passages	0.2648	0.1578
At 100 passages	0.2330	0.1943
R-Precision	0.2542	

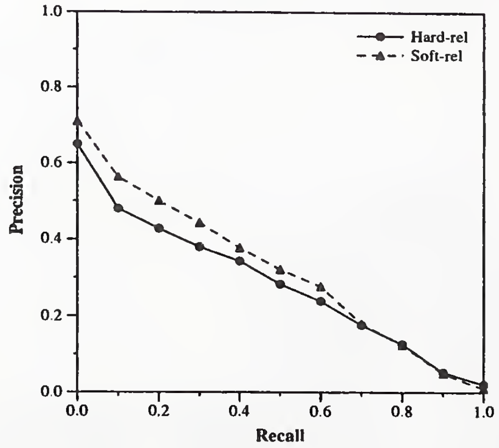


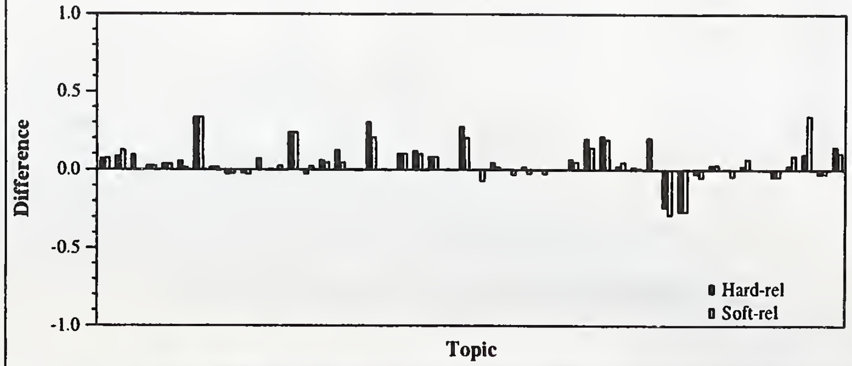
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	31049	31049
Total relevant	5123	7576
Total rel. ret.	3104	4246
MAP	0.2682	0.3056

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4375	0.5500
At 10 docs	0.3958	0.5125
At 15 docs	0.3903	0.5083
At 20 docs	0.3844	0.5031
At 30 docs	0.3688	0.4840
At 100 docs	0.2894	0.3896
At 200 docs	0.2158	0.2886
At 500 docs	0.1198	0.1626
At 1000 docs	0.0647	0.0885
R-Precision	0.3070	0.3430



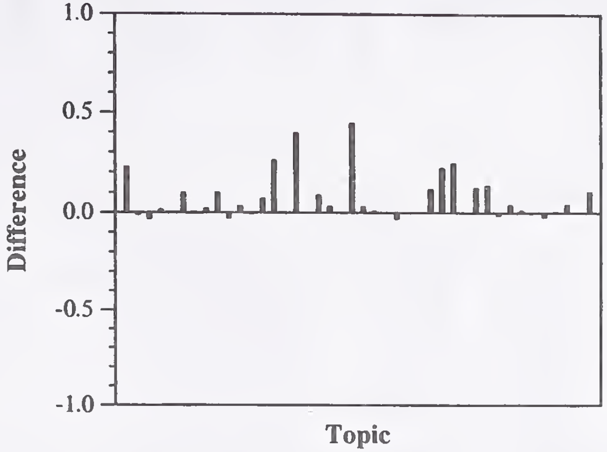


Difference from median in Document based R-precision per topic

Run ID	ciirtmda
Run Description	not a baseline, used metadata; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3345	0.0820
At 10 passages	0.3026	0.1088
At 15 passages	0.2874	0.1170
At 20 passages	0.2805	0.1307
At 30 passages	0.2687	0.1581
At 50 passages	0.2497	0.1854
At 100 passages	0.2126	0.2110
R-Precision	0.2261	

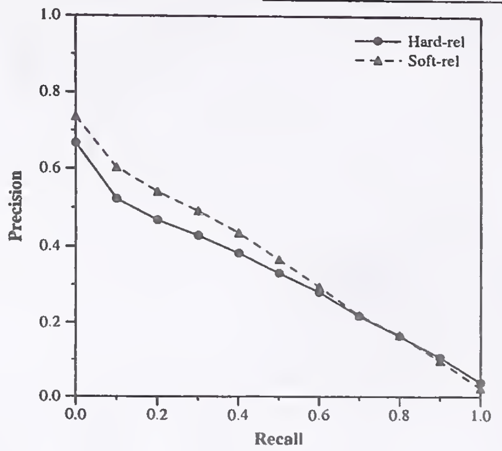


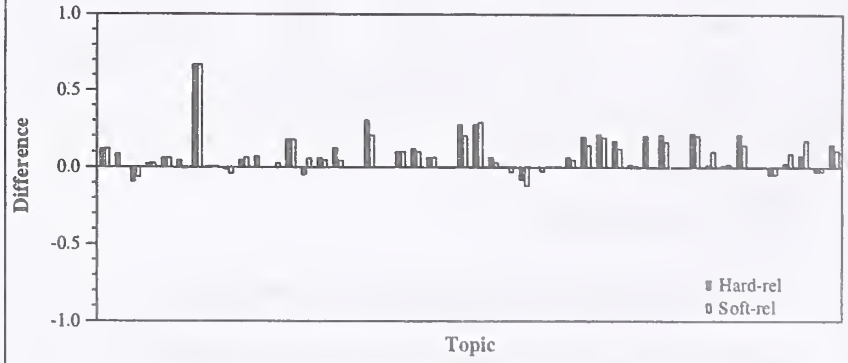
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3586	4955
MAP	0.3136	0.3500

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4750	0.6042
At 10 docs	0.4542	0.5646
At 15 docs	0.4403	0.5542
At 20 docs	0.4260	0.5344
At 30 docs	0.4132	0.5167
At 100 docs	0.3260	0.4246
At 200 docs	0.2304	0.3047
At 500 docs	0.1271	0.1724
At 1000 docs	0.0747	0.1032
R-Precision	0.3415	0.3759



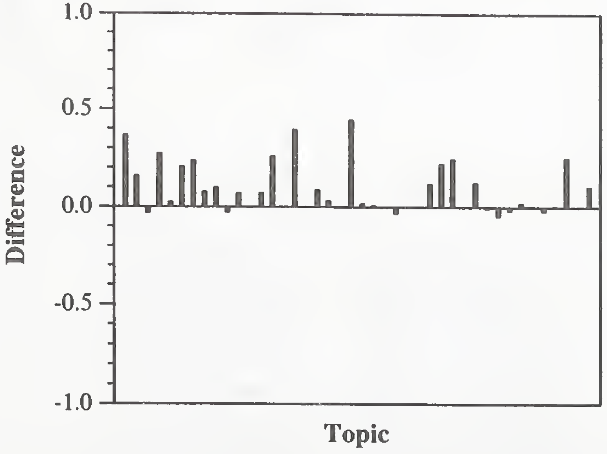


Difference from median in Document based R-precision per topic

Run ID	ciirtmdgp
Run Description	not a baseline, used metadata; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3459	0.0573
At 10 passages	0.3119	0.0879
At 15 passages	0.2987	0.0901
At 20 passages	0.2917	0.1007
At 30 passages	0.2810	0.1267
At 50 passages	0.2643	0.1575
At 100 passages	0.2330	0.1943
R-Precision	0.2541	

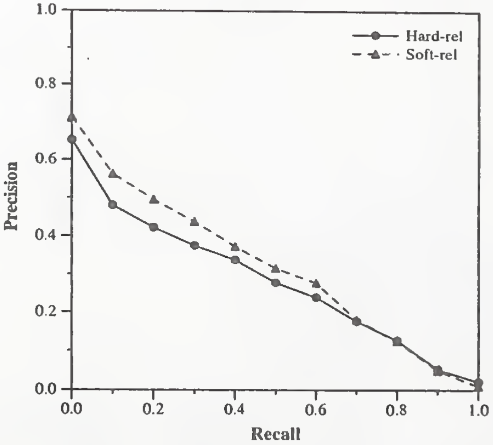


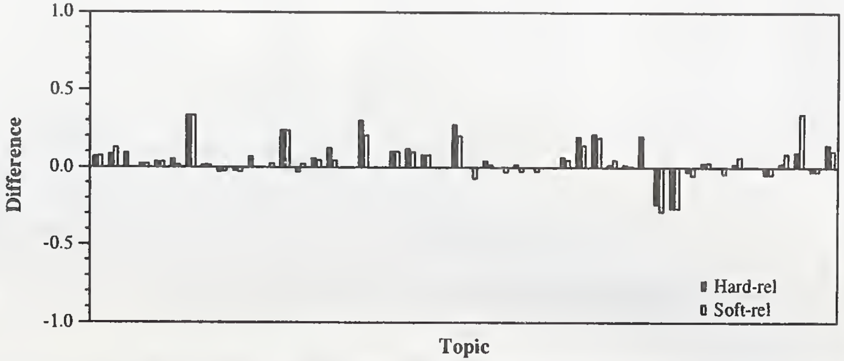
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	31048	31048
Total relevant	5123	7576
Total rel. ret.	3104	4245
MAP	0.2662	0.3036

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4375	0.5542
At 10 docs	0.3958	0.5146
At 15 docs	0.3903	0.5097
At 20 docs	0.3844	0.5031
At 30 docs	0.3688	0.4847
At 100 docs	0.2894	0.3894
At 200 docs	0.2158	0.2886
At 500 docs	0.1198	0.1626
At 1000 docs	0.0647	0.0884
R-Precision	0.3069	0.3430



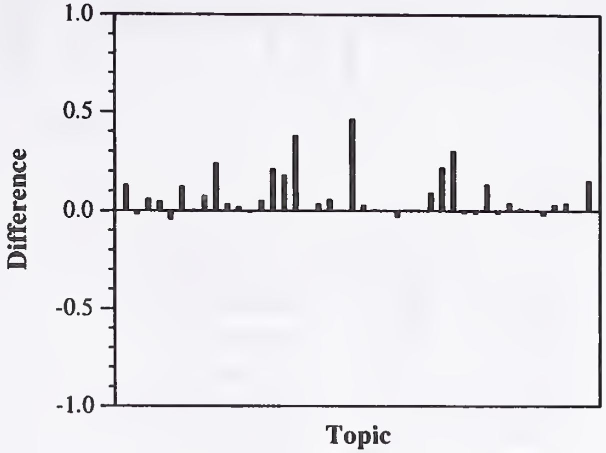


Difference from median in Document based R-precision per topic

Run ID	ciirtrt
Run Description	not a baseline, used metadata; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3103	0.0708
At 10 passages	0.3031	0.1035
At 15 passages	0.2837	0.1203
At 20 passages	0.2842	0.1409
At 30 passages	0.2824	0.1669
At 50 passages	0.2588	0.1892
At 100 passages	0.2156	0.2105
R-Precision	0.2327	

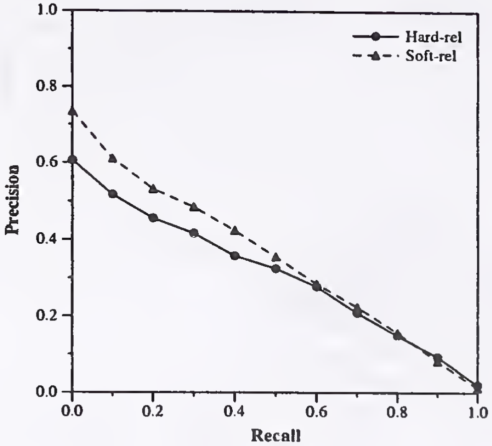


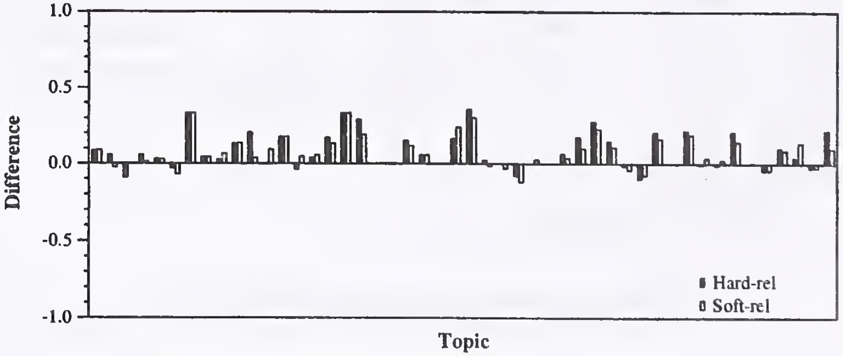
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3584	4986
MAP	0.3016	0.3430

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4333	0.6083
At 10 docs	0.4250	0.5708
At 15 docs	0.4292	0.5569
At 20 docs	0.4250	0.5469
At 30 docs	0.4090	0.5229
At 100 docs	0.3190	0.4140
At 200 docs	0.2270	0.3024
At 500 docs	0.1257	0.1720
At 1000 docs	0.0747	0.1039
R-Precision	0.3378	0.3759



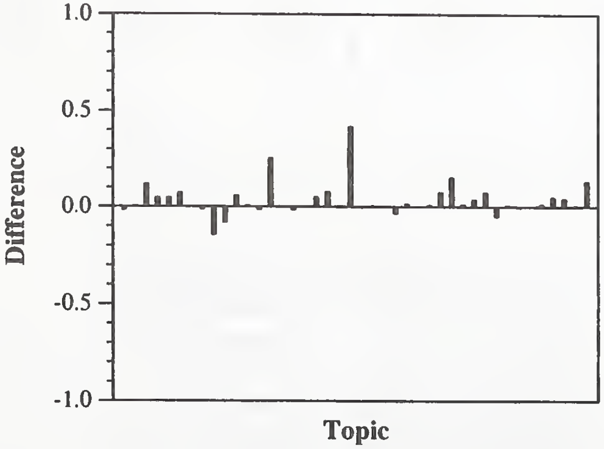


Difference from median in Document based R-precision per topic

Run ID	UWAThard1
Run Description	baseline run; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.2879	0.0601
At 10 passages	0.2668	0.0862
At 15 passages	0.2519	0.0947
At 20 passages	0.2320	0.1029
At 30 passages	0.2334	0.1255
At 50 passages	0.2044	0.1409
At 100 passages	0.1830	0.1696
R-Precision	0.1908	

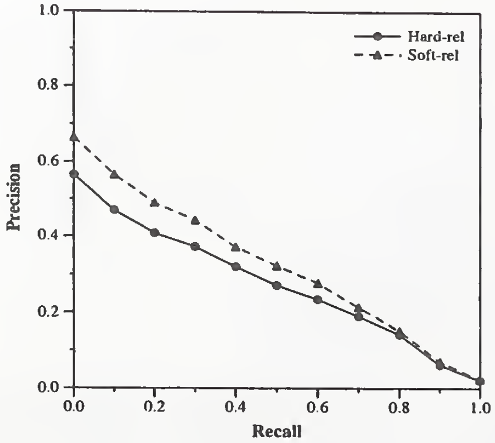


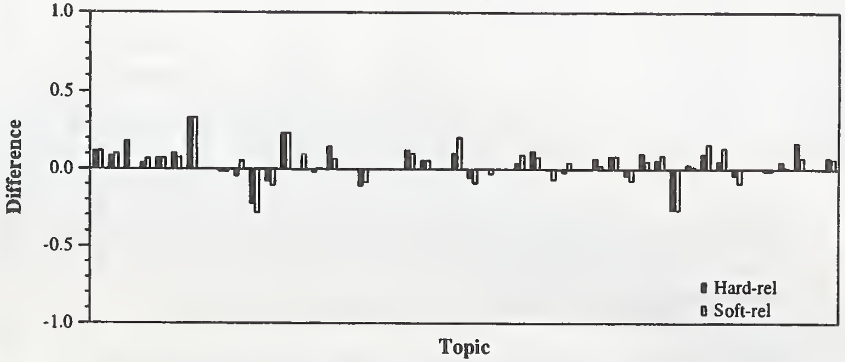
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	48000	48000
Total relevant	5123	7576
Total rel. ret.	3613	5052
MAP	0.2638	0.3134

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4000	0.5167
At 10 docs	0.3875	0.4875
At 15 docs	0.3903	0.4986
At 20 docs	0.3729	0.4823
At 30 docs	0.3597	0.4688
At 100 docs	0.2765	0.3669
At 200 docs	0.2114	0.2854
At 500 docs	0.1257	0.1717
At 1000 docs	0.0753	0.1052
R-Precision	0.2893	0.3336



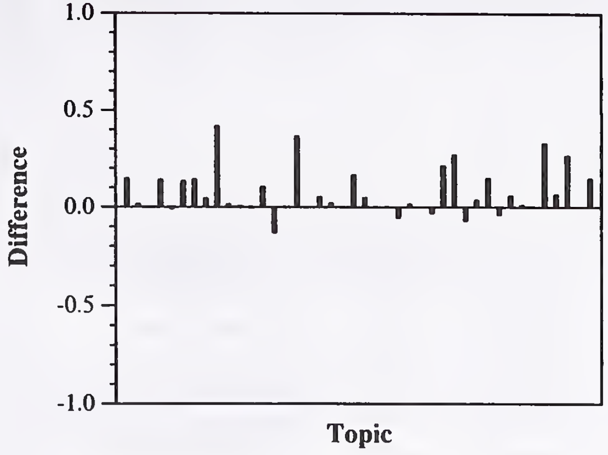


Difference from median in Document based R-precision per topic

Run ID	UWAThard2
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3215	0.0674
At 10 passages	0.3305	0.1005
At 15 passages	0.3108	0.1062
At 20 passages	0.3044	0.1204
At 30 passages	0.2925	0.1454
At 50 passages	0.2694	0.1744
At 100 passages	0.2321	0.2072
R-Precision	0.2359	

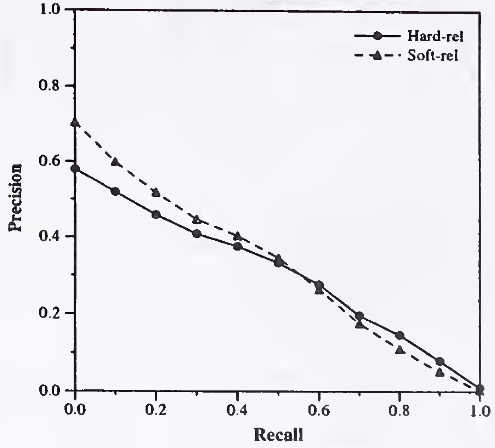


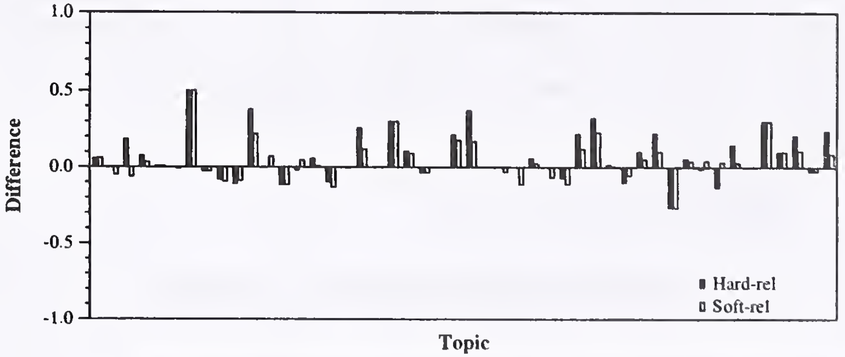
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	47242	47242
Total relevant	5123	7576
Total rel. ret.	3514	4784
MAP	0.2978	0.3150

Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.4417	0.5708
At 10 docs	0.4354	0.5479
At 15 docs	0.4222	0.5292
At 20 docs	0.4146	0.5198
At 30 docs	0.3903	0.4986
At 100 docs	0.3048	0.3890
At 200 docs	0.2166	0.2776
At 500 docs	0.1205	0.1594
At 1000 docs	0.0732	0.0997
R-Precision	0.3263	0.3417



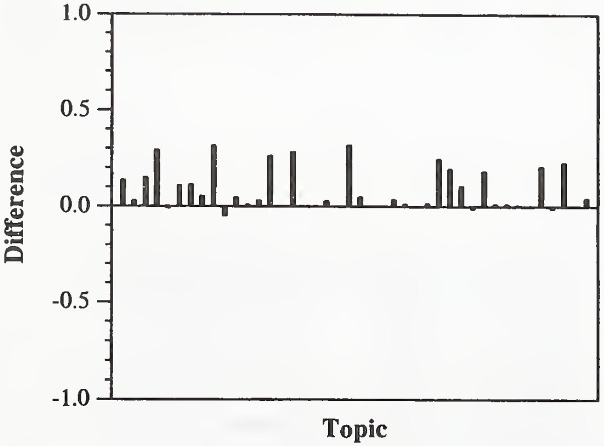


Difference from median in Document based R-precision per topic

Run ID	UWAThard3
Run Description	not a baseline, used both; title only

Passages-based Evaluation (42 topics)

Passage Level Averages		
	Precision	F Score
At 5 passages	0.3815	0.0691
At 10 passages	0.3617	0.1077
At 15 passages	0.3429	0.1191
At 20 passages	0.3324	0.1343
At 30 passages	0.3104	0.1559
At 50 passages	0.2800	0.1836
At 100 passages	0.2380	0.2166
R-Precision	0.2426	

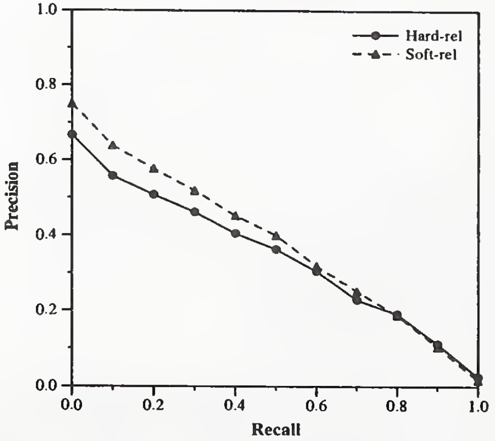


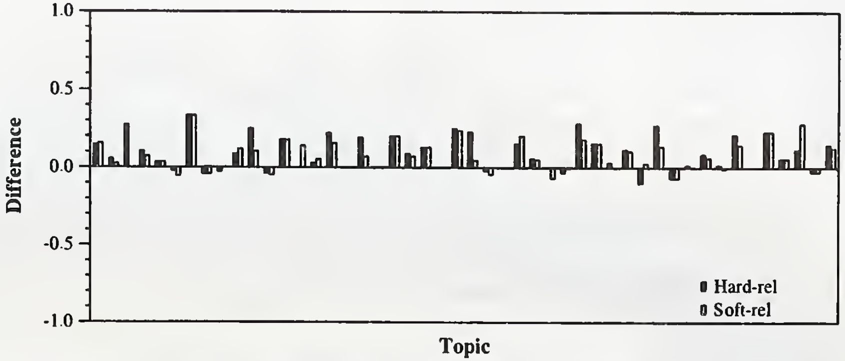
Difference from median passage R-precision

Document-based Evaluation (48 topics)

	Hard-rel	Soft-rel
Total retrieved	45808	45808
Total relevant	5123	.7576
Total rel. ret.	3814	5219
MAP	0.3335	0.3719

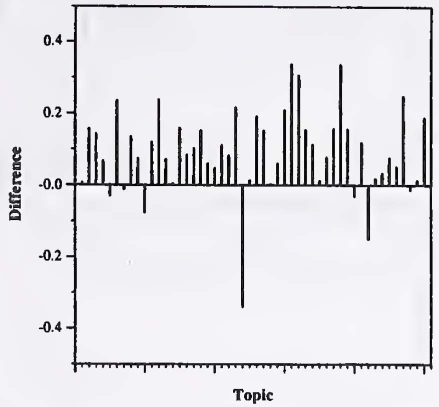
Document Level Averages		
	Precision	
	Hard-rel	Soft-rel
At 5 docs	0.5042	0.6208
At 10 docs	0.4854	0.5958
At 15 docs	0.4750	0.5833
At 20 docs	0.4646	0.5698
At 30 docs	0.4326	0.5424
At 100 docs	0.3267	0.4129
At 200 docs	0.2390	0.3101
At 500 docs	0.1363	0.1818
At 1000 docs	0.0795	0.1087
R-Precision	0.3466	0.3780



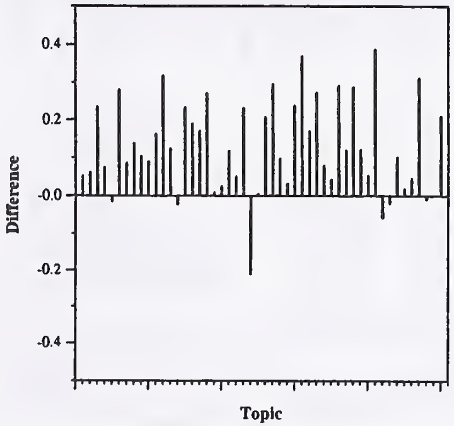


Difference from median in Document based R-precision per topic

Summary Statistics		
Run ID	ICT03NOV1BSL	
Num topics		50
	Relevant	New
Average precision	0.62	0.41
Average recall	0.51	0.48
Average F	0.486	0.379

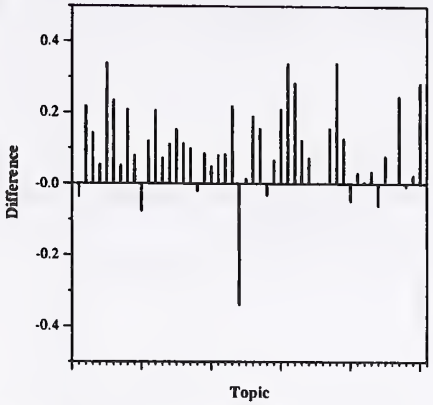


Difference from Median in F score per topic for relevant sentences

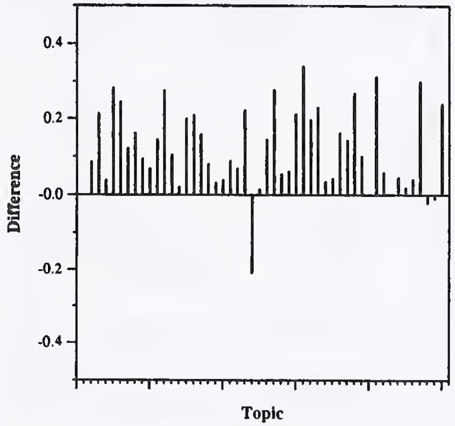


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	ICT03NOV1DTH	
Num topics		50
	Relevant	New
Average precision	0.63	0.42
Average recall	0.50	0.44
Average F	0.489	0.370

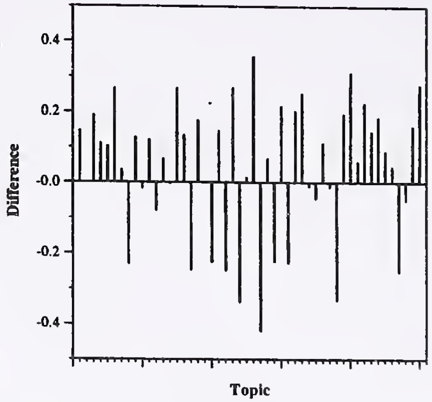


Difference from Median in F score per topic for relevant sentences

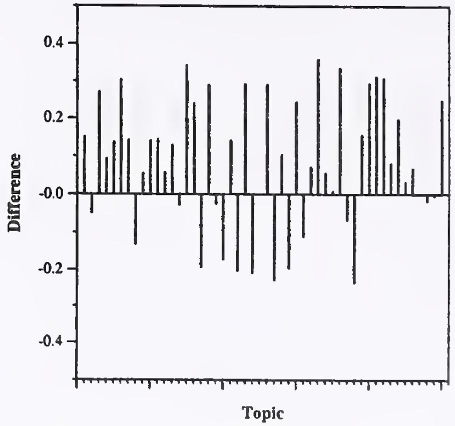


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	ICT03NOV1NAR	
Num topics		50
	Relevant	New
Average precision	0.58	0.37
Average recall	0.46	0.44
Average F	0.434	0.334

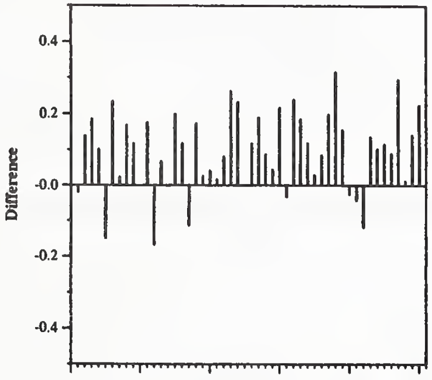


Difference from Median in F score per topic for relevant sentences

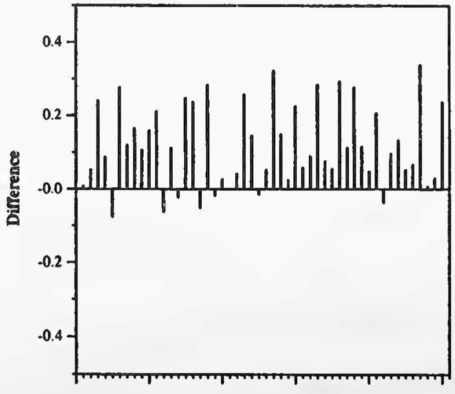


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	ICT03NOV1SQR	
Num topics		50
	Relevant	New
Average precision	0.63	0.40
Average recall	0.54	0.49
Average F	0.489	0.368

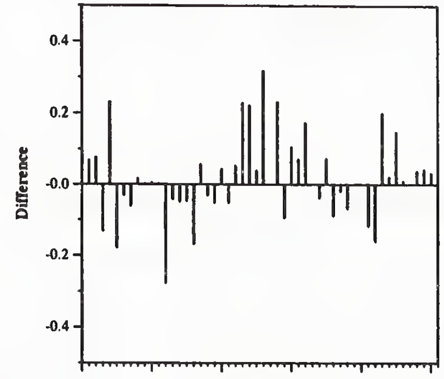


Difference from Median in F score per topic for relevant sentences

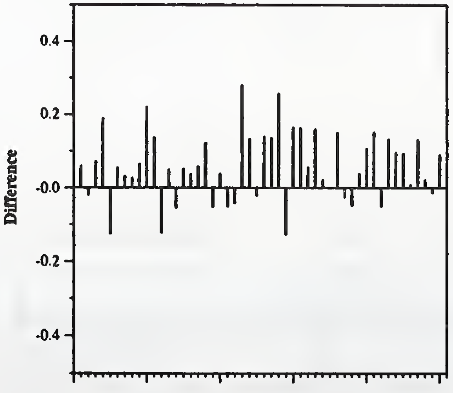


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	ICT03NOV1XTD	
Num topics		50
	Relevant	New
Average precision	0.61	0.39
Average recall	0.39	0.36
Average F	0.408	0.310



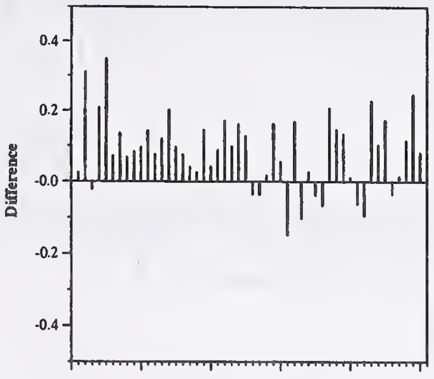
Difference from Median in F score per topic for relevant sentences



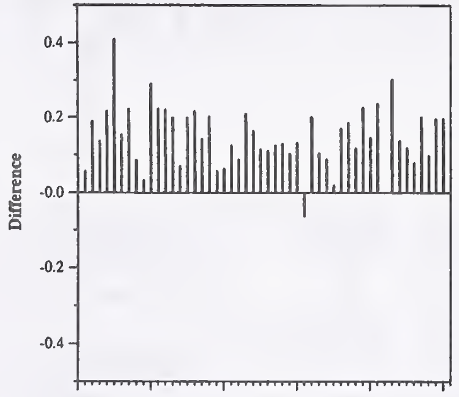
Difference from Median in F score per topic for novel sentences

Novelty track results, task 1 — Chinese Academy of Sciences (CAS-NLPR)

Summary Statistics		
Run ID		NLPR03n1f1
Num topics		50
	Relevant	New
Average precision	0.73	0.51
Average recall	0.42	0.41
Average F	0.477	0.399

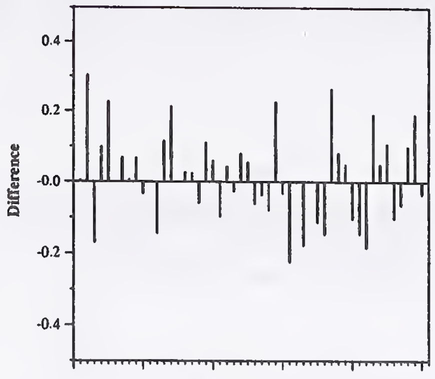


Difference from Median in F score per topic for relevant sentences

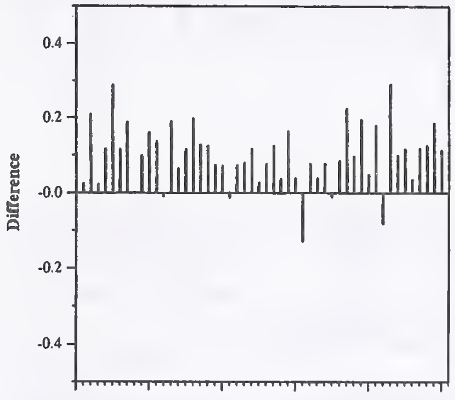


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NLPR03n1f2
Num topics		50
	Relevant	New
Average precision	0.75	0.53
Average recall	0.32	0.32
Average F	0.407	0.349

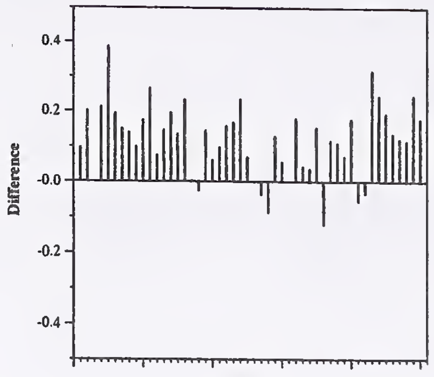


Difference from Median in F score per topic for relevant sentences

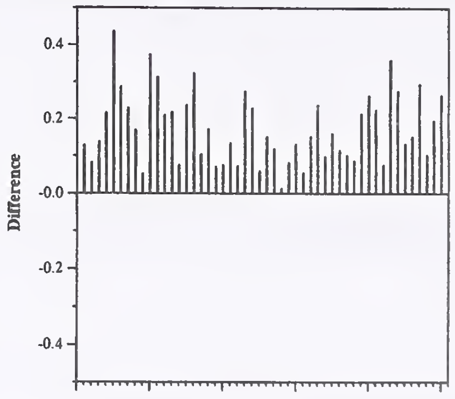


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NLPR03n1w1
Num topics		50
	Relevant	New
Average precision	0.71	0.51
Average recall	0.48	0.47
Average F	0.510	0.425

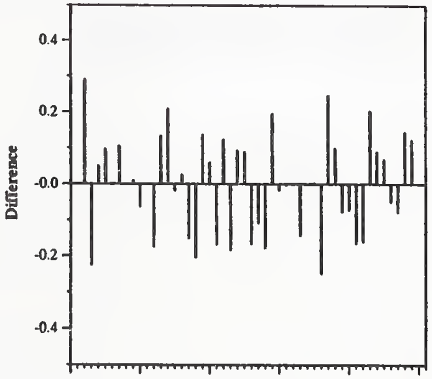


Difference from Median in F score per topic for relevant sentences

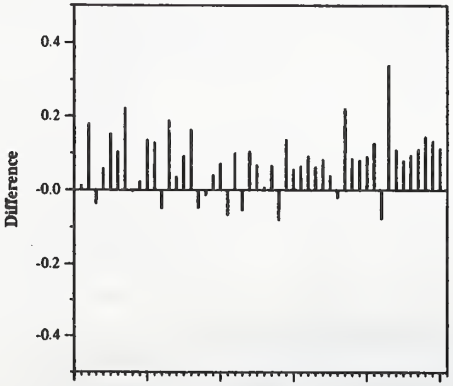


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	NLPR03n1w2	
Num topics	50	
	Relevant	New
Average precision	0.76	0.53
Average recall	0.31	0.29
Average F	0.391	0.325

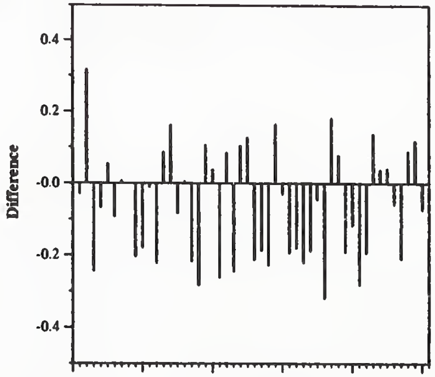


Difference from Median in F score per topic for relevant sentences

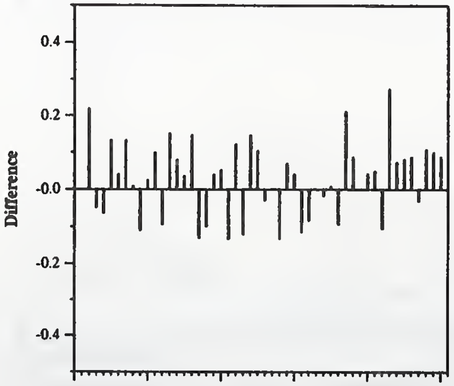


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	NLPR03n1w3	
Num topics	50	
	Relevant	New
Average precision	0.77	0.54
Average recall	0.24	0.23
Average F	0.330	0.279

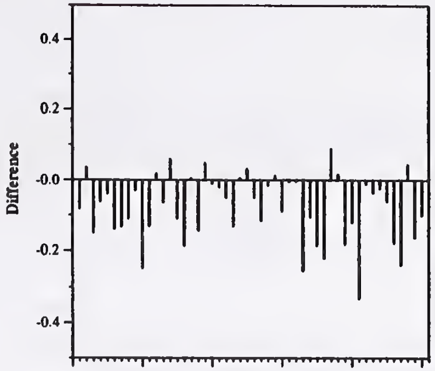


Difference from Median in F score per topic for relevant sentences

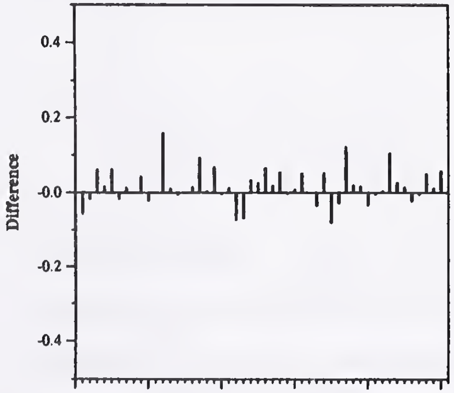


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		ccsumlaqr
Num topics		50
	Relevant	New
Average precision	0.65	0.47
Average recall	0.25	0.24
Average F	0.313	0.267

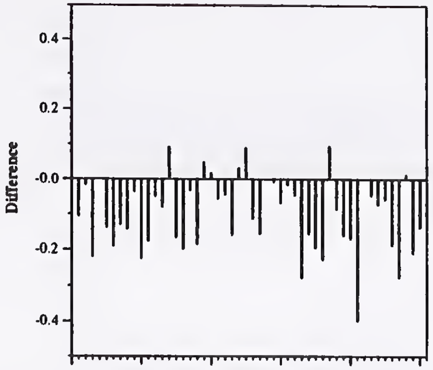


Difference from Median in F score per topic for relevant sentences

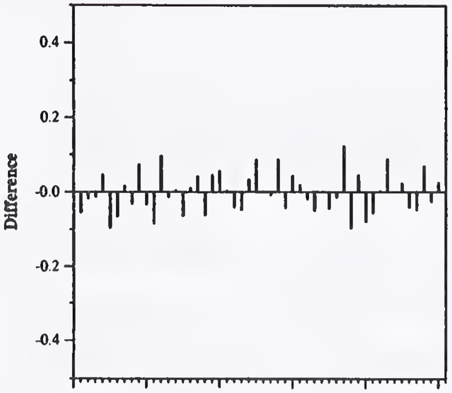


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		ccsummeoqr
Num topics		50
	Relevant	New
Average precision	0.67	0.49
Average recall	0.22	0.21
Average F	0.291	0.248

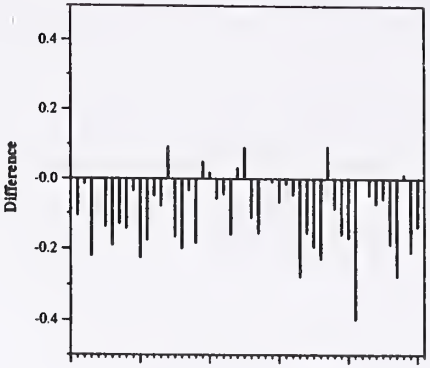


Difference from Median in F score per topic for relevant sentences

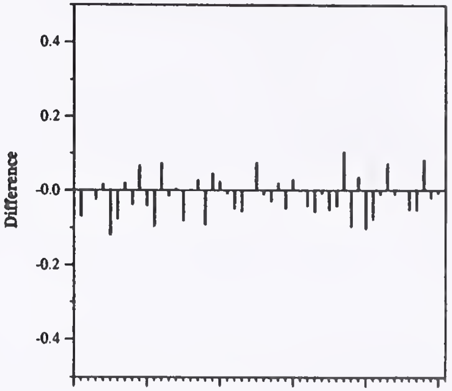


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		ccsummeosvd
Num topics		50
	Relevant	New
Average precision	0.67	0.48
Average recall	0.22	0.19
Average F	0.291	0.234

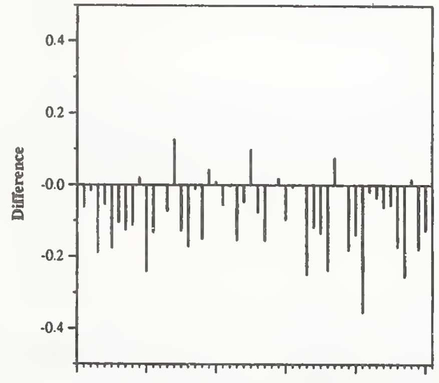


Difference from Median in F score per topic for relevant sentences

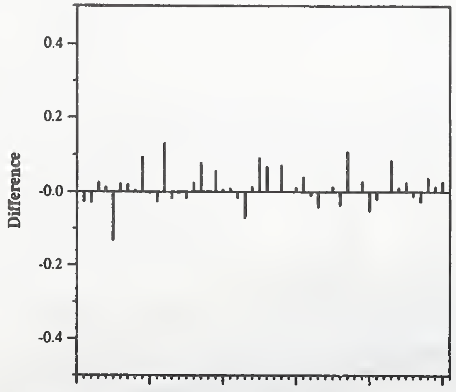


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		ccsumrelqr
Num topics		50
	Relevant	New
Average precision	0.66	0.48
Average recall	0.24	0.23
Average F	0.307	0.262

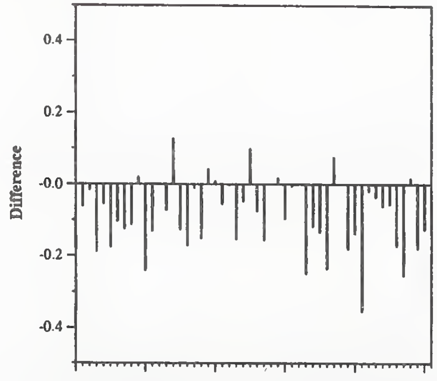


Difference from Median in F score per topic for relevant sentences

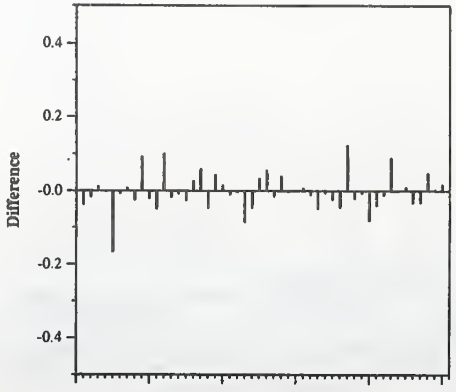


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		ccsumrelsvd
Num topics		50
	Relevant	New
Average precision	0.66	0.47
Average recall	0.24	0.21
Average F	0.307	0.247



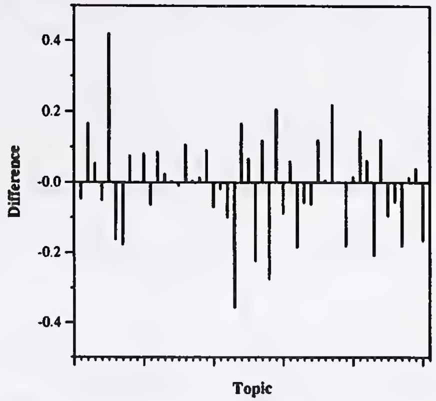
Difference from Median in F score per topic for relevant sentences



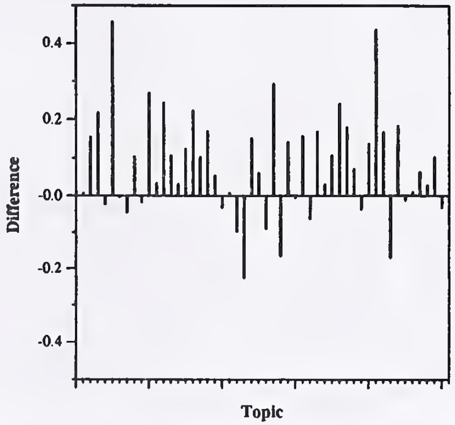
Difference from Median in F score per topic for novel sentences

Novelty track results, task 1 — CL Research

Summary Statistics		
Run ID		clr03n1d
Num topics		50
	Relevant	New
Average precision	0.72	0.51
Average recall	0.32	0.31
Average F	0.385	0.331

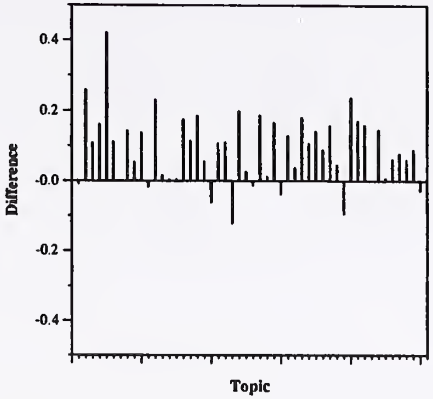


Difference from Median in F score per topic for relevant sentences

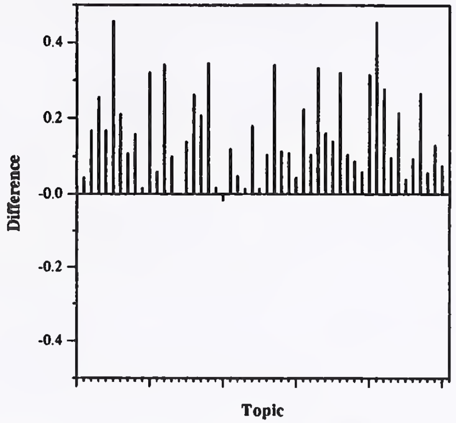


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		clr03n1n2
Num topics		50
	Relevant	New
Average precision	0.69	0.50
Average recall	0.45	0.44
Average F	0.483	0.410

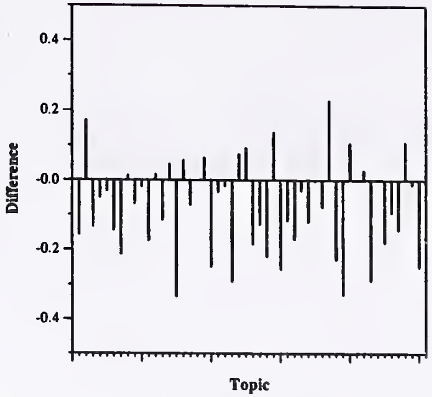


Difference from Median in F score per topic for relevant sentences

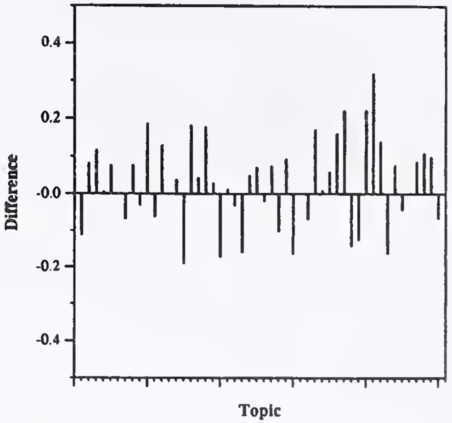


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		clr03n1n3
Num topics		50
	Relevant	New
Average precision	0.72	0.51
Average recall	0.24	0.24
Average F	0.316	0.278

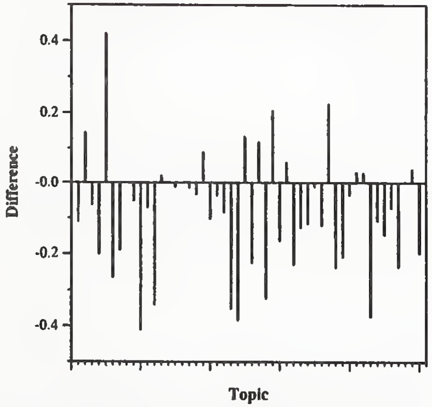


Difference from Median in F score per topic for relevant sentences

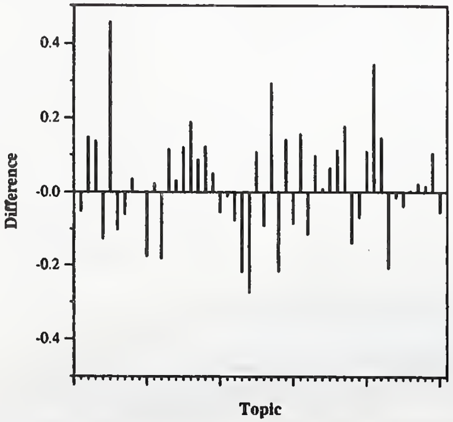


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		clr03n1t
Num topics		50
	Relevant	New
Average precision	0.71	0.51
Average recall	0.25	0.24
Average F	0.309	0.272

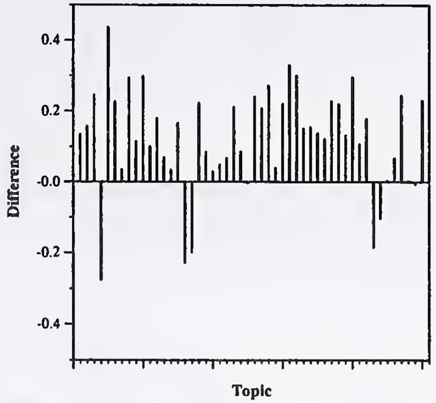


Difference from Median in F score per topic for relevant sentences

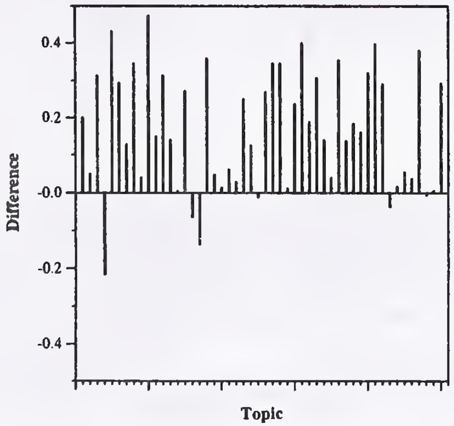


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		IRITf2bis
Num topics		50
	Relevant	New
Average precision	0.63	0.45
Average recall	0.57	0.55
Average F	0.516	0.421

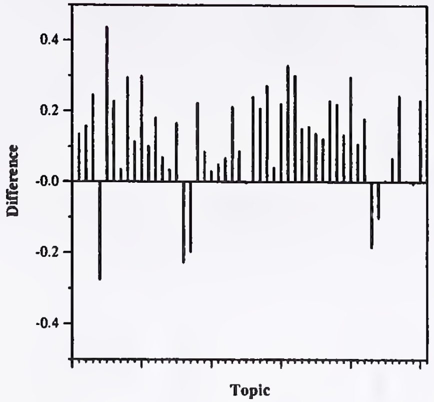


Difference from Median in F score per topic for relevant sentences

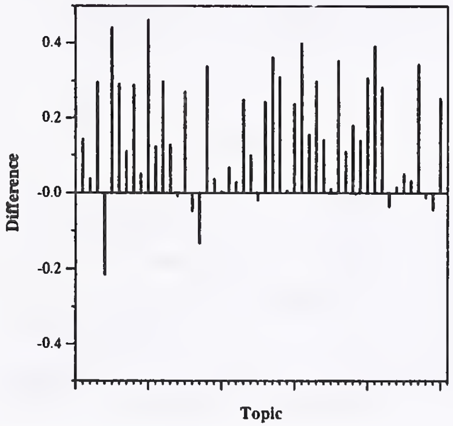


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		IRITfNegR2
Num topics		50
	Relevant	New
Average precision	0.63	0.41
Average recall	0.57	0.57
Average F	0.516	0.408

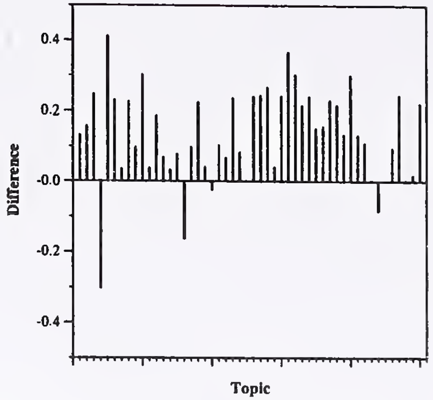


Difference from Median in F score per topic for relevant sentences

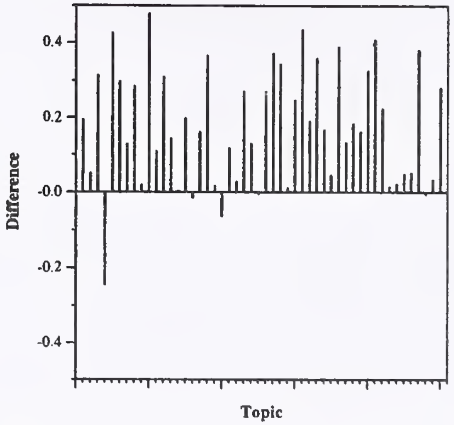


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		IRITfb1MtmIb
Num topics		50
	Relevant	New
Average precision	0.64	0.45
Average recall	0.58	0.55
Average F	0.526	0.426

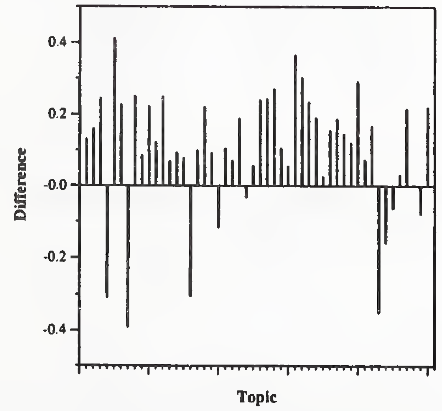


Difference from Median in F score per topic for relevant sentences

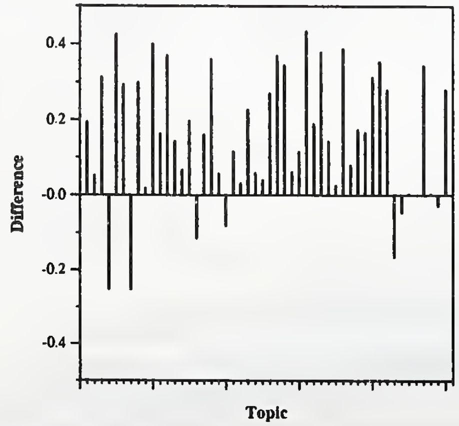


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		IRITnb1MtmI4
Num topics		50
	Relevant	New
Average precision	0.66	0.46
Average recall	0.50	0.48
Average F	0.493	0.406

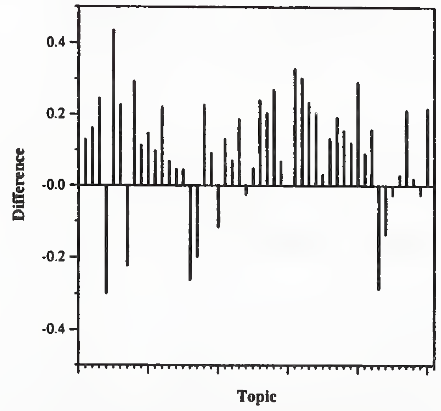


Difference from Median in F score per topic for relevant sentences

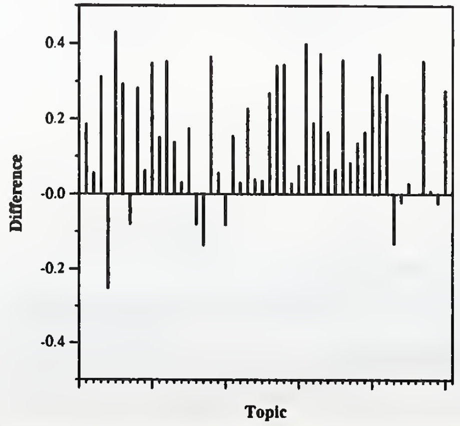


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		IRITnip2bis
Num topics		50
	Relevant	New
Average precision	0.65	0.44
Average recall	0.50	0.48
Average F	0.491	0.402

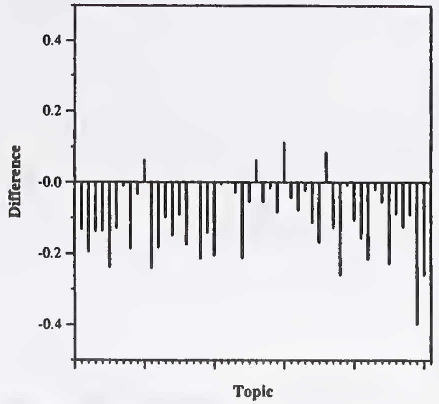


Difference from Median in F score per topic for relevant sentences

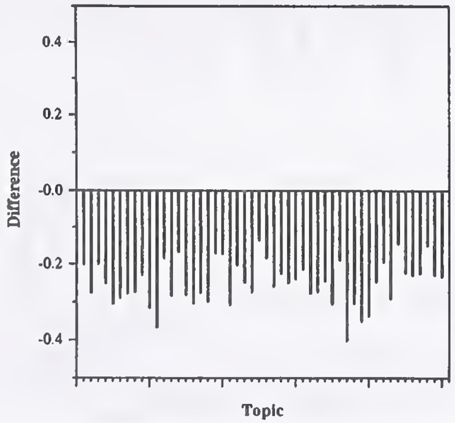


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		lexiclone03
Num topics		50
	Relevant	New
Average precision	0.48	0.00
Average recall	0.24	0.00
Average F	0.285	0.000

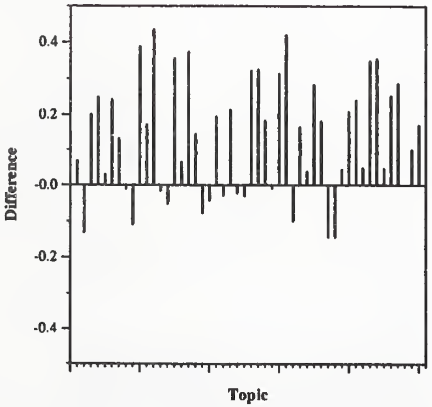


Difference from Median in F score per topic for relevant sentences

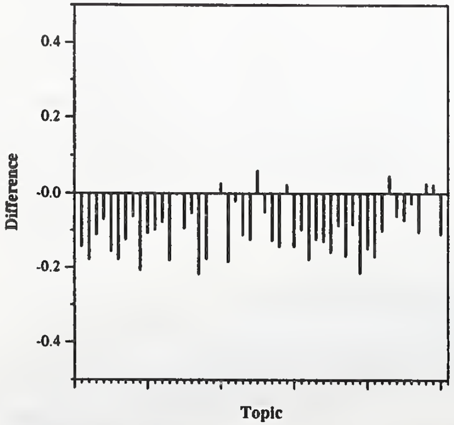


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		MeijiHilF11
Num topics		50
	Relevant	New
Average precision	0.53	0.52
Average recall	0.64	0.10
Average F	0.526	0.151

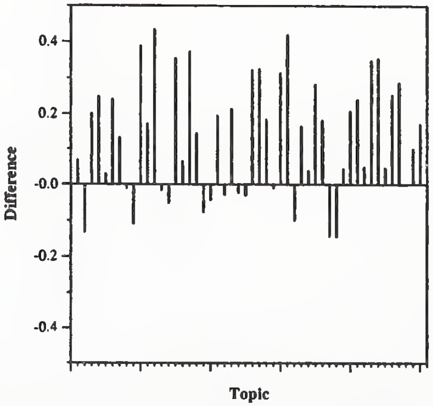


Difference from Median in F score per topic for relevant sentences

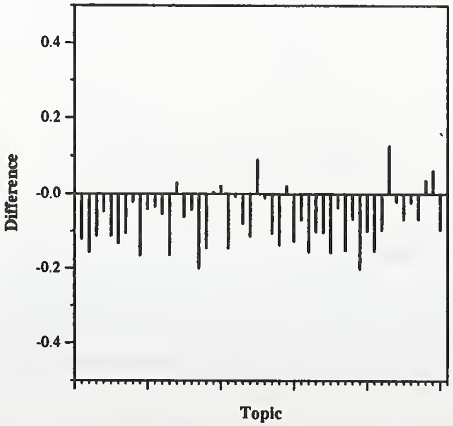


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		MeijiHilF12
Num topics		50
	Relevant	New
Average precision	0.53	0.51
Average recall	0.64	0.13
Average F	0.526	0.176

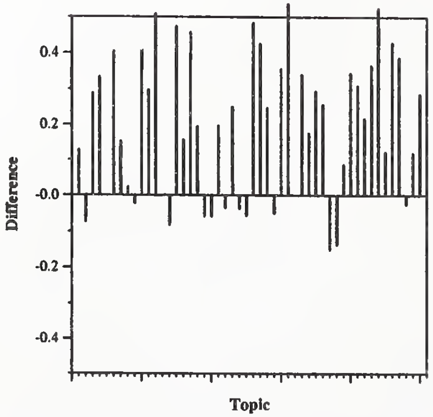


Difference from Median in F score per topic for relevant sentences

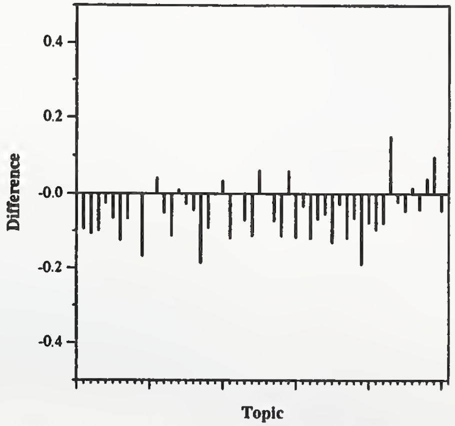


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		MeijiHilF13
Num topics		50
	Relevant	New
Average precision	0.52	0.50
Average recall	0.84	0.15
Average F	0.589	0.199

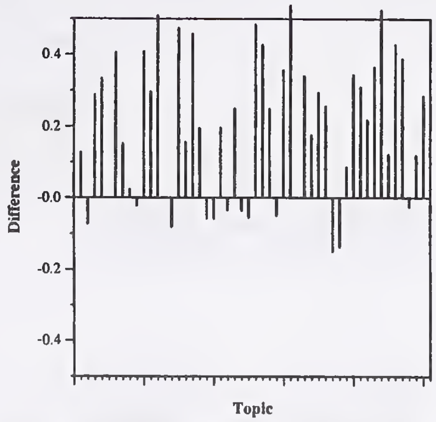


Difference from Median in F score per topic for relevant sentences

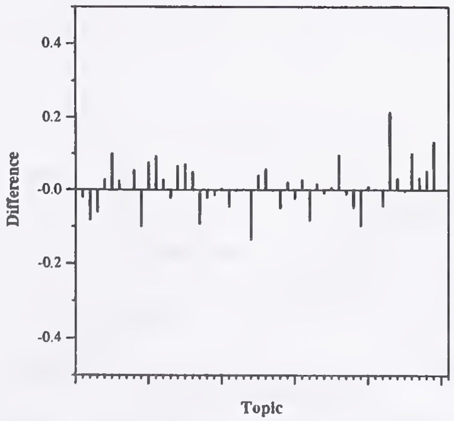


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		MeijiHilF14
Num topics		50
	Relevant	New
Average precision	0.52	0.49
Average recall	0.84	0.22
Average F	0.589	0.260

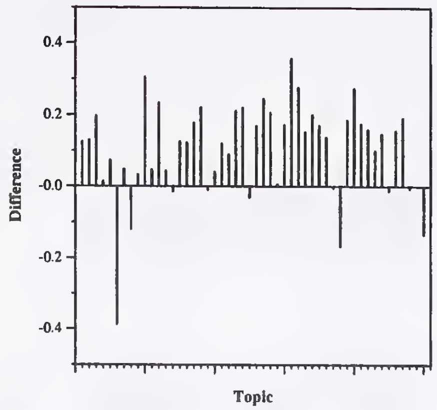


Difference from Median in F score per topic for relevant sentences

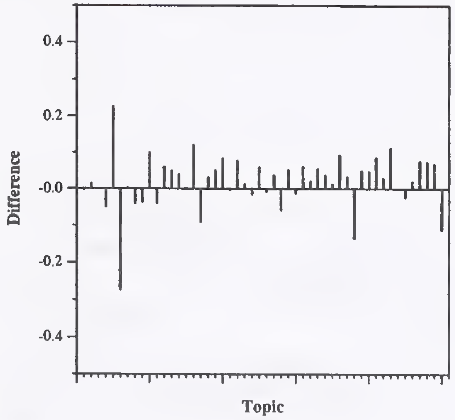


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		MeijiHilF15
Num topics		50
	Relevant	New
Average precision	0.57	0.55
Average recall	0.55	0.22
Average F	0.496	0.270

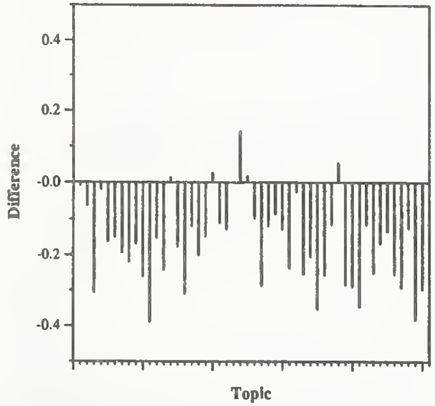


Difference from Median in F score per topic for relevant sentences

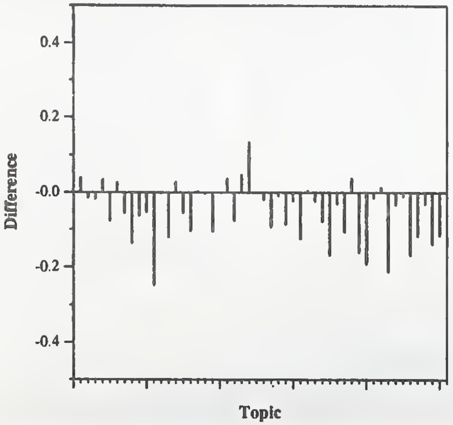


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NTU11
Num topics		50
	Relevant	New
Average precision	0.59	0.43
Average recall	0.16	0.15
Average F	0.225	0.197

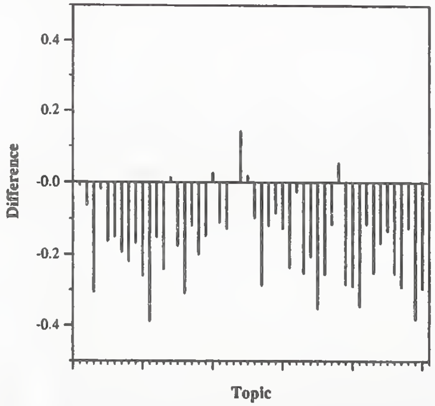


Difference from Median in F score per topic for relevant sentences

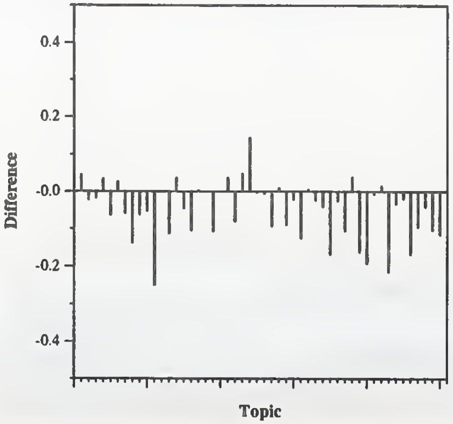


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NTU12
Num topics		50
	Relevant	New
Average precision	0.59	0.43
Average recall	0.16	0.15
Average F	0.225	0.200

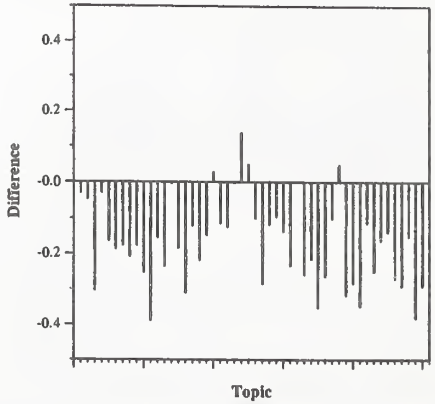


Difference from Median in F score per topic for relevant sentences

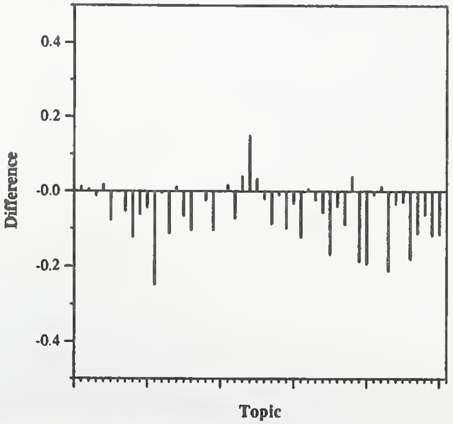


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NTU13
Num topics		50
	Relevant	New
Average precision	0.58	0.43
Average recall	0.16	0.15
Average F	0.223	0.195

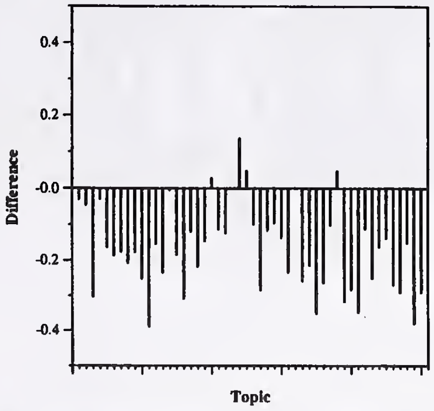


Difference from Median in F score per topic for relevant sentences

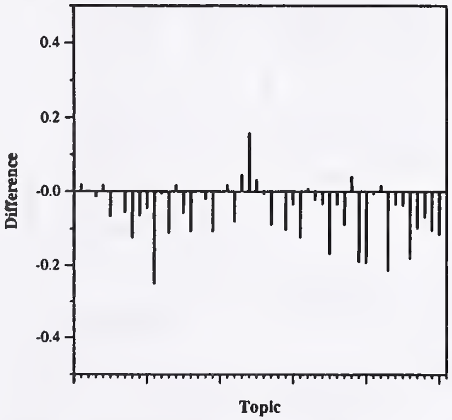


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NTU14
Num topics		50
	Relevant	New
Average precision	0.58	0.42
Average recall	0.16	0.15
Average F	0.223	0.197

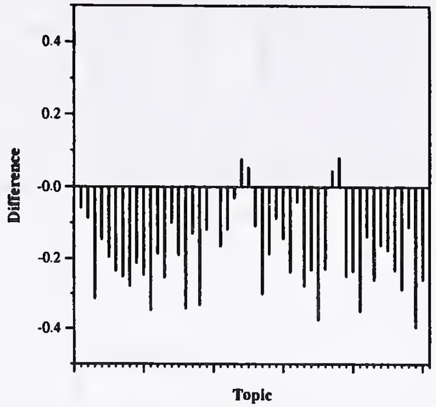


Difference from Median in F score per topic for relevant sentences

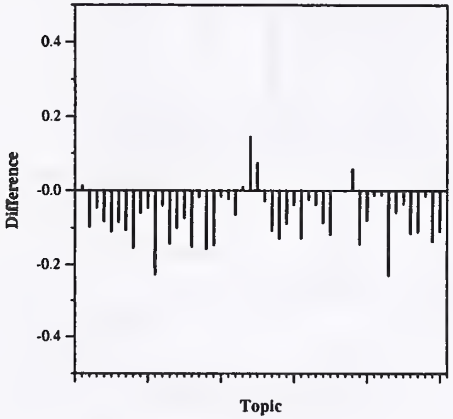


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NTU15
Num topics		50
	Relevant	New
Average precision	0.57	0.42
Average recall	0.14	0.14
Average F	0.209	0.180

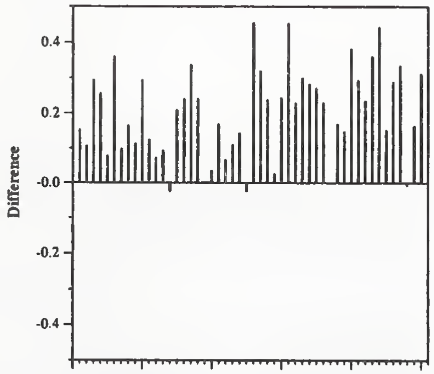


Difference from Median in F score per topic for relevant sentences

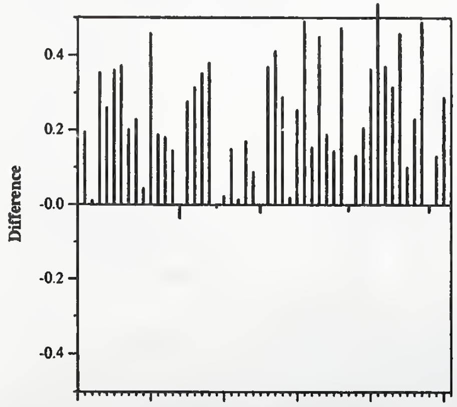


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	THUIRnv0311	
Num topics		50
	Relevant	New
Average precision	0.61	0.47
Average recall	0.73	0.66
Average F	0.593	0.481

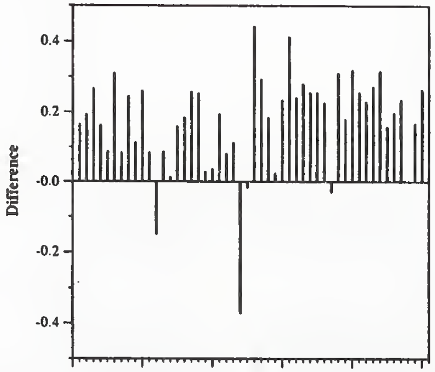


Difference from Median in F score per topic for relevant sentences

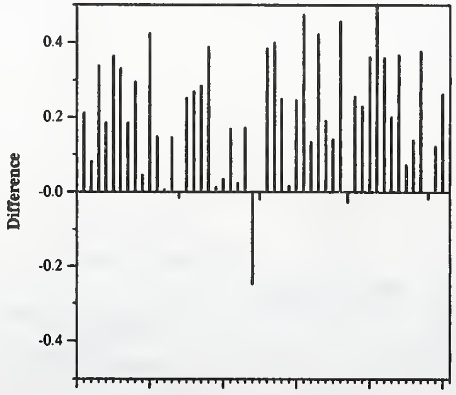


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	THUIRnv0312	
Num topics		50
	Relevant	New
Average precision	0.62	0.48
Average recall	0.67	0.61
Average F	0.564	0.459

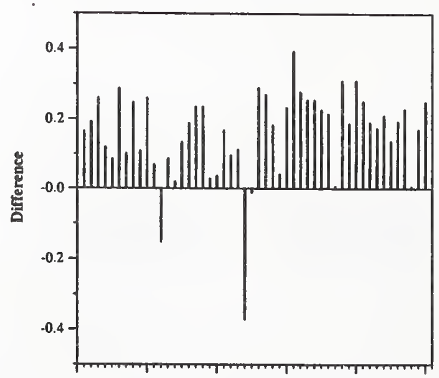


Difference from Median in F score per topic for relevant sentences

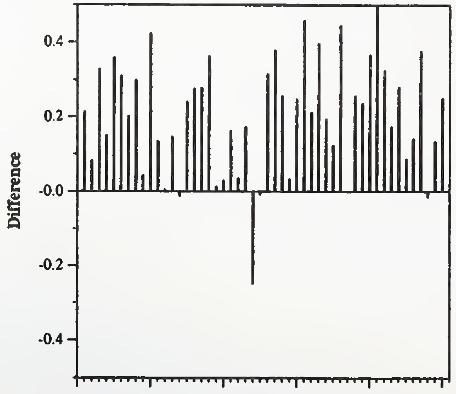


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	THUIRnv0313	
Num topics		50
	Relevant	New
Average precision	0.63	0.49
Average recall	0.64	0.58
Average F	0.552	0.453

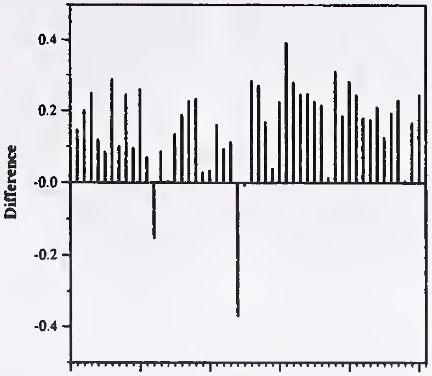


Difference from Median in F score per topic for relevant sentences

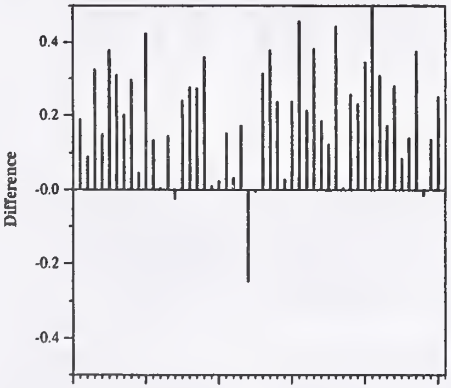


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		THUIRnv0314
Num topics		50
	Relevant	New
Average precision	0.63	0.49
Average recall	0.63	0.57
Average F	0.548	0.451

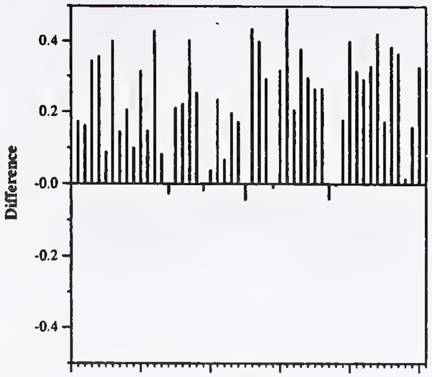


Difference from Median in F score per topic for relevant sentences

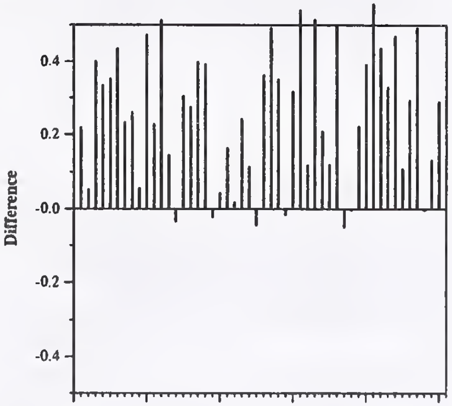


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		THUIRnv0315
Num topics		50
	Relevant	New
Average precision	0.60	0.46
Average recall	0.79	0.74
Average F	0.619	0.505

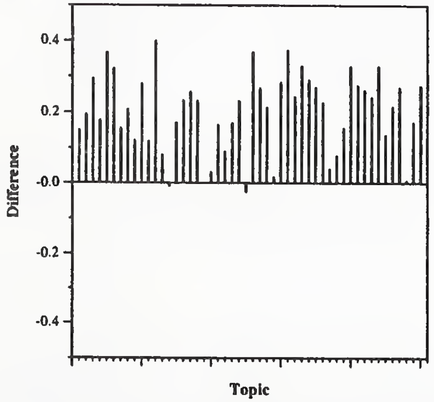


Difference from Median in F score per topic for relevant sentences

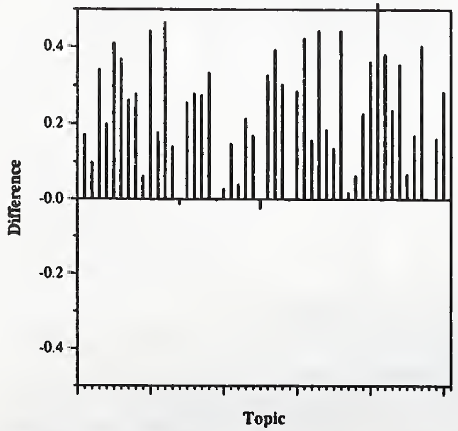


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		UIowa03Nov01
Num topics		50
	Relevant	New
Average precision	0.64	0.47
Average recall	0.70	0.65
Average F	0.594	0.480

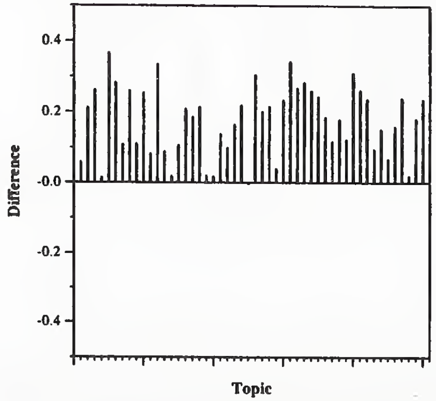


Difference from Median in F score per topic for relevant sentences

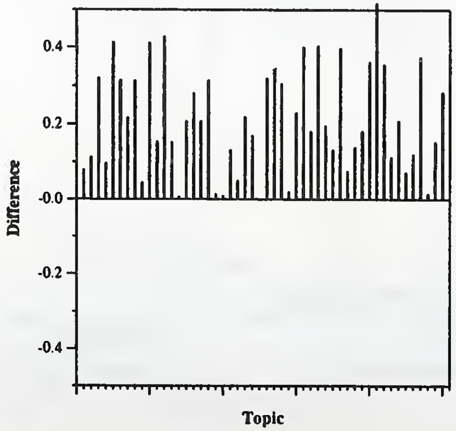


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		UIowa03Nov02
Num topics		50
	Relevant	New
Average precision	0.65	0.48
Average recall	0.64	0.59
Average F	0.568	0.461

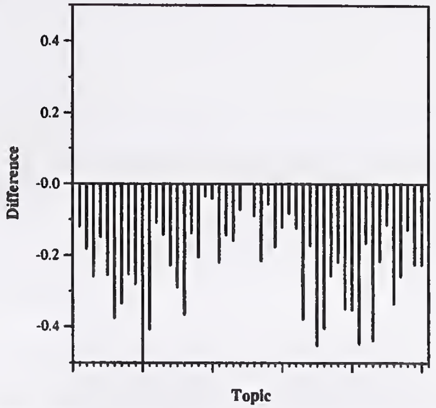


Difference from Median in F score per topic for relevant sentences

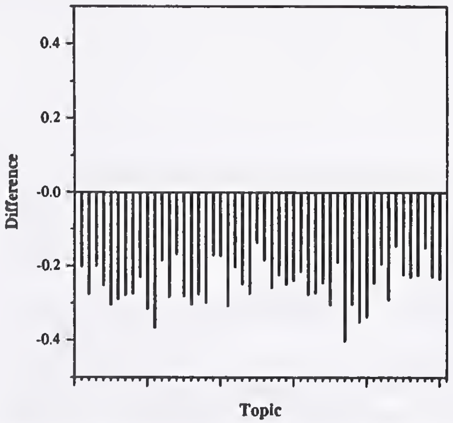


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		umbcrun1
Num topics		50
	Relevant	New
Average precision	0.40	0.00
Average recall	0.12	0.00
Average F	0.166	0.000

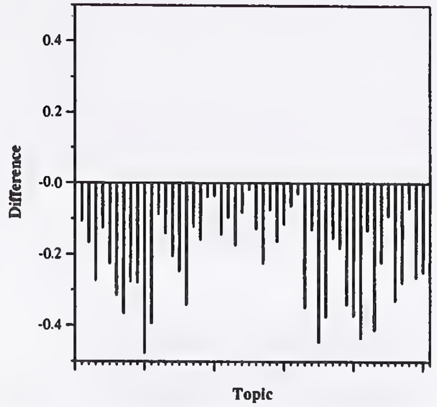


Difference from Median in F score per topic for relevant sentences

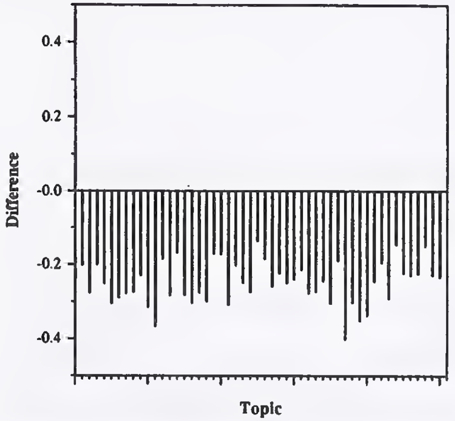


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		umbcrun2
Num topics		50
	Relevant	New
Average precision	0.41	0.00
Average recall	0.14	0.00
Average F	0.181	0.000

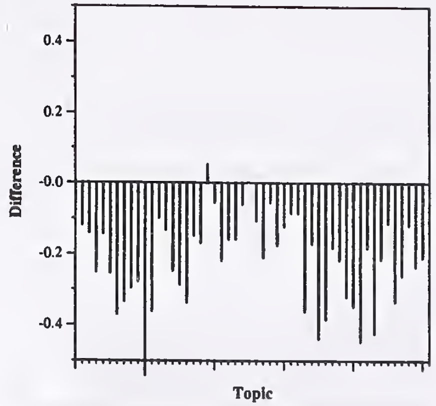


Difference from Median in F score per topic for relevant sentences

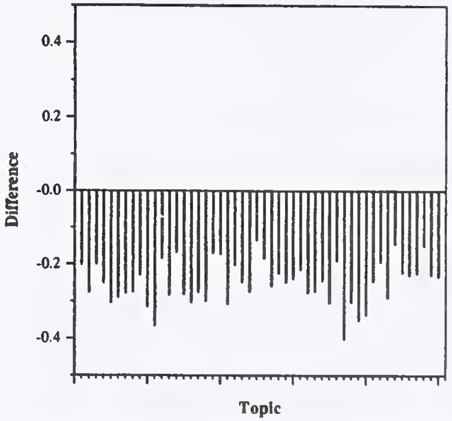


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		umbcrun3
Num topics		50
	Relevant	New
Average precision	0.40	0.00
Average recall	0.13	0.00
Average F	0.173	0.000

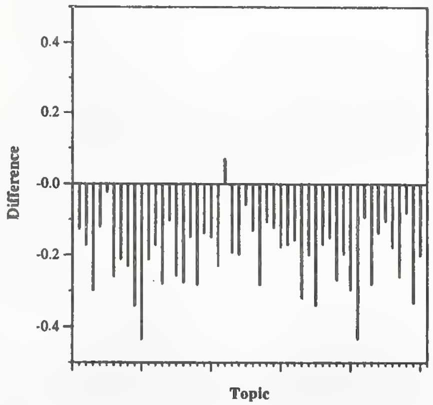


Difference from Median in F score per topic for relevant sentences

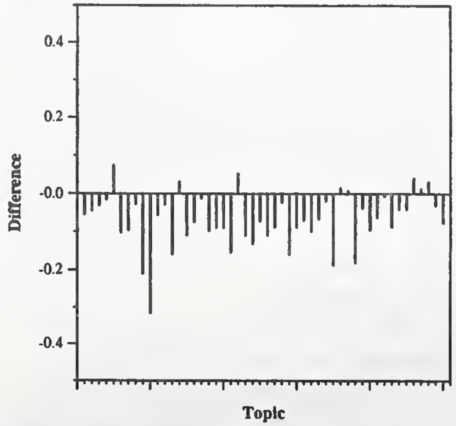


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		umich1
Num topics		50
	Relevant	New
Average precision	0.58	0.46
Average recall	0.13	0.13
Average F	0.192	0.182

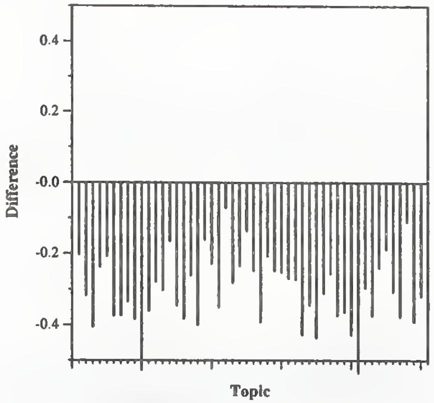


Difference from Median in F score per topic for relevant sentences

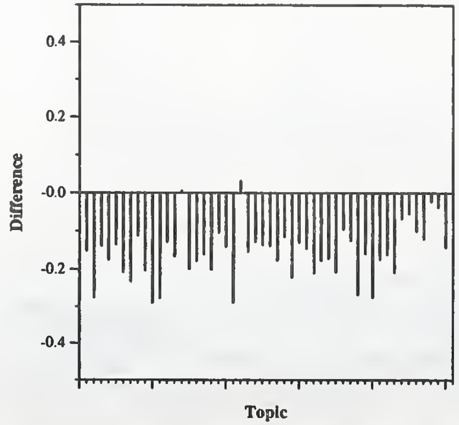


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		umich2
Num topics		50
	Relevant	New
Average precision	0.58	0.53
Average recall	0.05	0.06
Average F	0.086	0.093

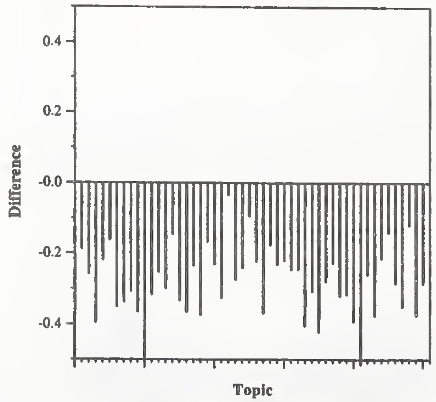


Difference from Median in F score per topic for relevant sentences

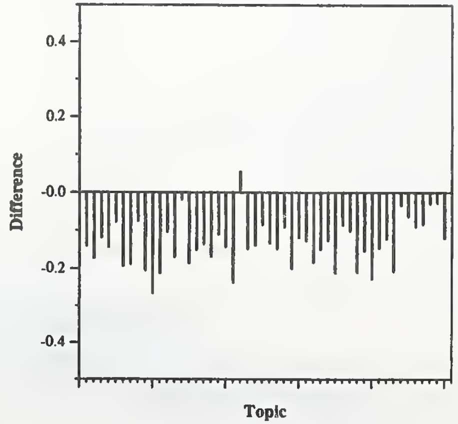


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		umich3
Num topics		50
	Relevant	New
Average precision	0.59	0.52
Average recall	0.06	0.07
Average F	0.110	0.115

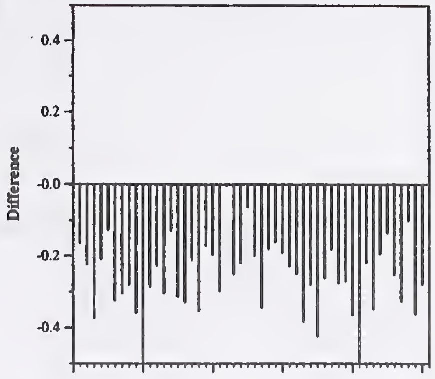


Difference from Median in F score per topic for relevant sentences

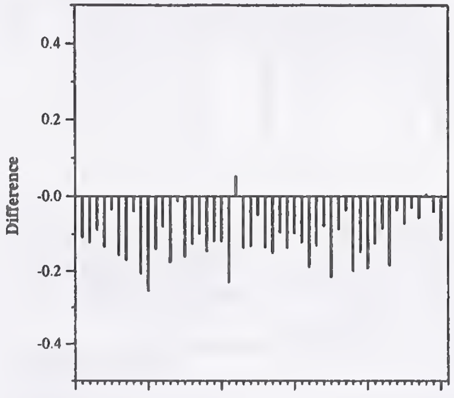


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		umich4
Num topics		50
	Relevant	New
Average precision	0.58	0.51
Average recall	0.08	0.09
Average F	0.134	0.136

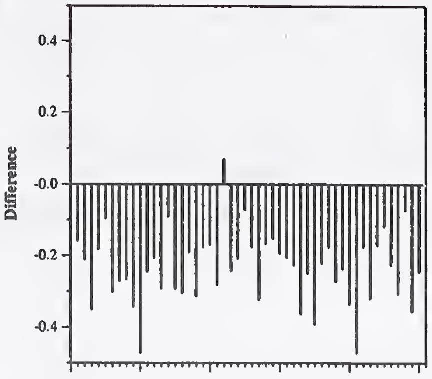


Difference from Median in F score per topic for relevant sentences

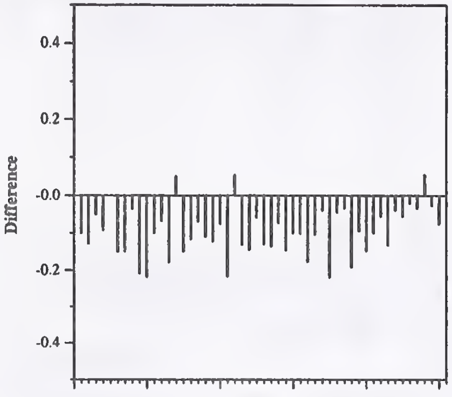


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		umich5
Num topics		50
	Relevant	New
Average precision	0.59	0.50
Average recall	0.10	0.10
Average F	0.156	0.155

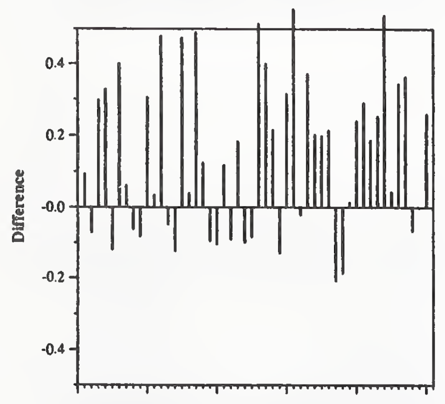


Difference from Median in F score per topic for relevant sentences

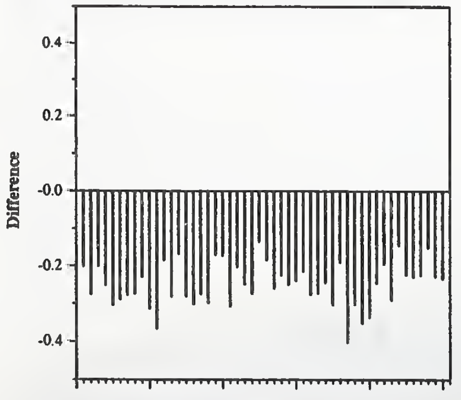


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		ISIALL03
Num topics		50
	Relevant	New
Average precision	0.41	0.00
Average recall	1.00	0.00
Average F	0.540	0.000

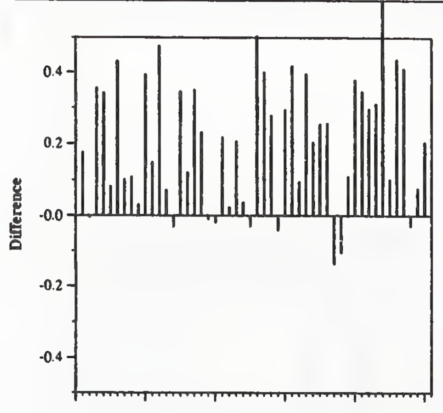


Difference from Median in F score per topic for relevant sentences

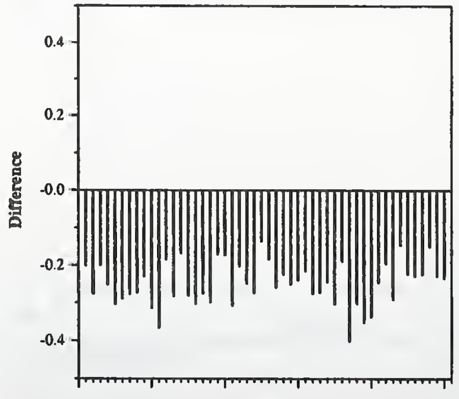


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		ISIDSCm203
Num topics		50
	Relevant	New
Average precision	0.53	0.00
Average recall	0.83	0.00
Average F	0.597	0.000

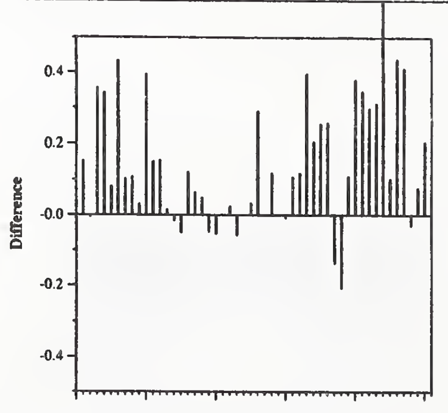


Difference from Median in F score per topic for relevant sentences

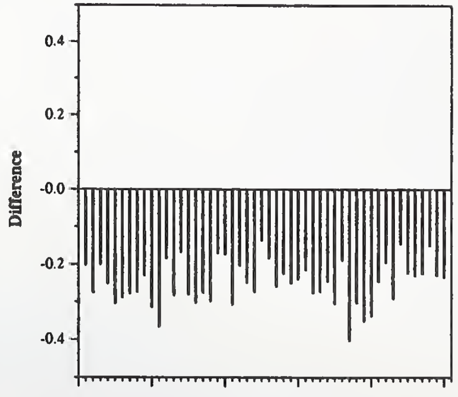


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		ISIDSm203
Num topics		50
	Relevant	New
Average precision	0.54	0.00
Average recall	0.67	0.00
Average F	0.533	0.000

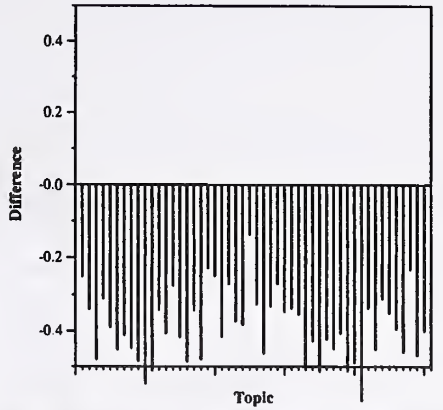


Difference from Median in F score per topic for relevant sentences

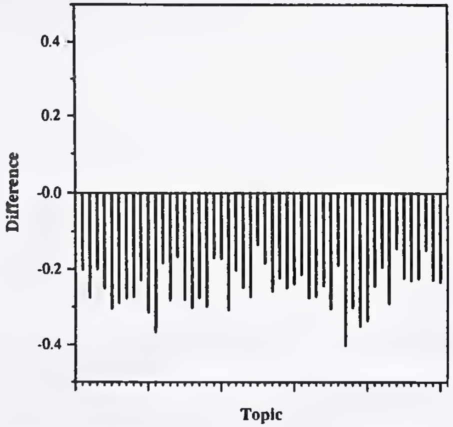


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		ISINONE03
Num topics		50
	Relevant	New
Average precision	0.00	0.00
Average recall	0.00	0.00
Average F	0.000	0.000

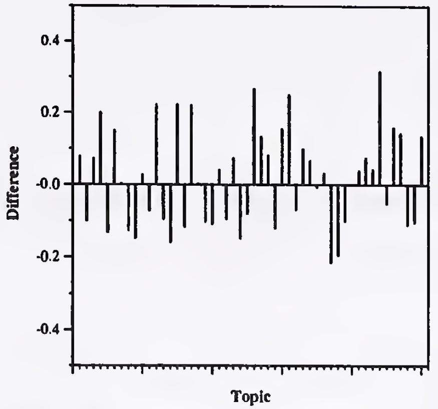


Difference from Median in F score per topic for relevant sentences

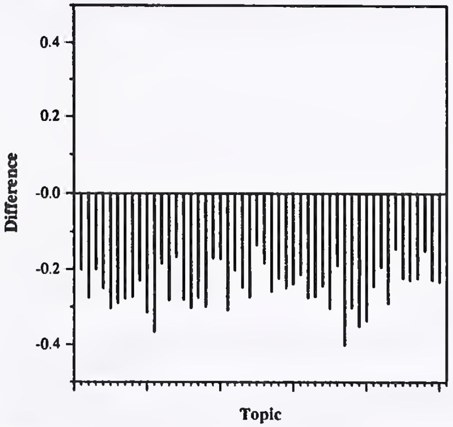


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		ISIRAND03
Num topics		50
	Relevant	New
Average precision	0.41	0.00
Average recall	0.50	0.00
Average F	0.409	0.000



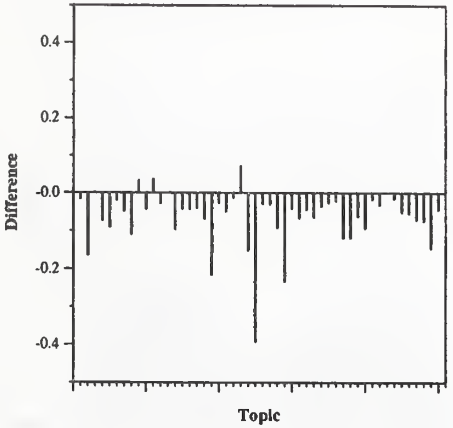
Difference from Median in F score per topic for relevant sentences



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ICT03NOV2CUR
Num topics	

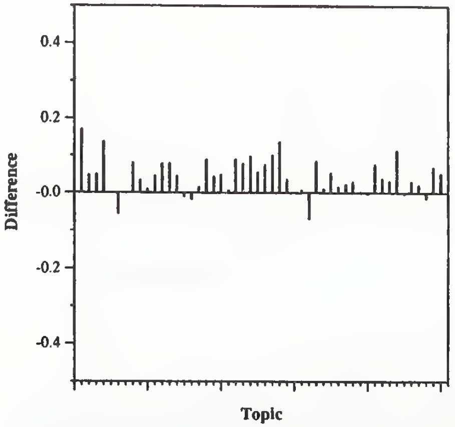
	New
Average precision	0.65
Average recall	0.73
Average F	0.677



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ICT03NOV2LPA
Num topics	

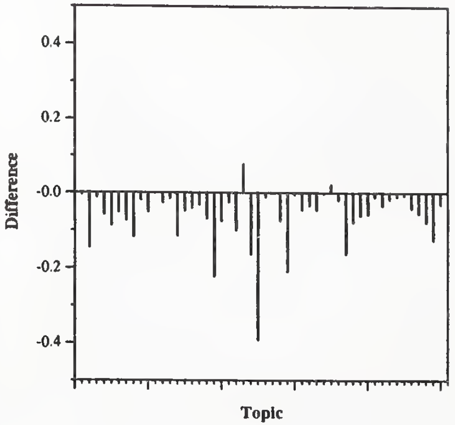
	New
Average precision	0.73
Average recall	0.87
Average F	0.783



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ICT03NOV2LPP
Num topics	

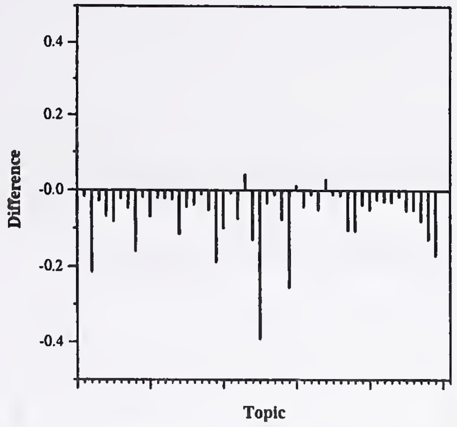
	New
Average precision	0.65
Average recall	0.74
Average F	0.679



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ICT03NOV2PNK
Num topics	

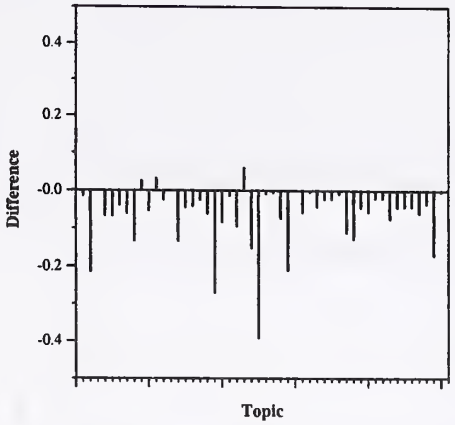
	New
Average precision	0.65
Average recall	0.73
Average F	0.676



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ICT03NOV2SQR
Num topics	

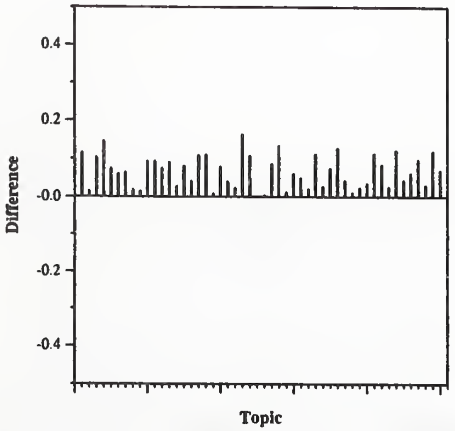
	New
Average precision	0.65
Average recall	0.74
Average F	0.677



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NLPR03n2d1
Num topics	

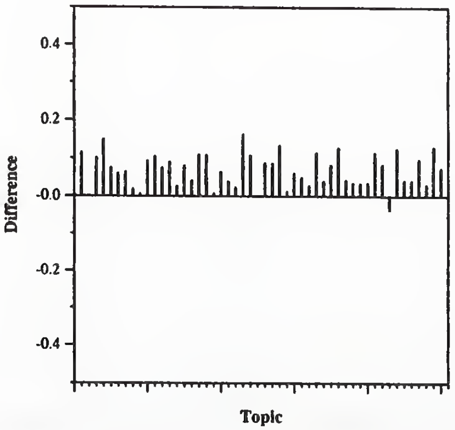
	New
Average precision	0.71
Average recall	0.95
Average F	0.807



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NLPR03n2d2
Num topics	

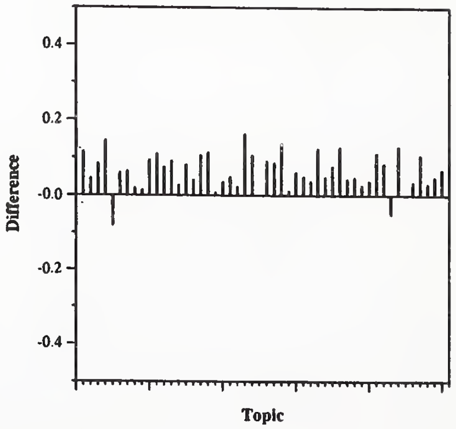
	New
Average precision	0.72
Average recall	0.94
Average F	0.808



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NLPR03n2d3
Num topics	

	New
Average precision	0.70
Average recall	0.97
Average F	0.803

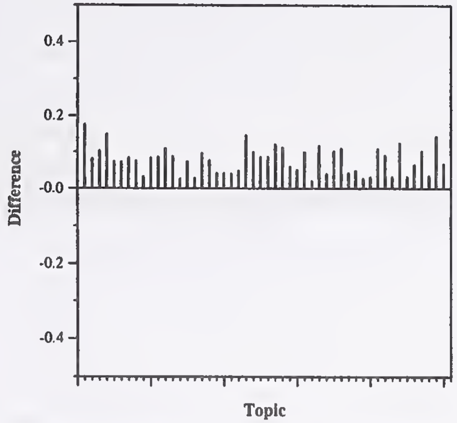


Difference from Median in F score per topic for novel sentences

Novelty track results, task 2 — Chinese Academy of Sciences (CAS-NLPR)

Summary Statistics	
Run ID	NLPR03n2s1
Num topics	

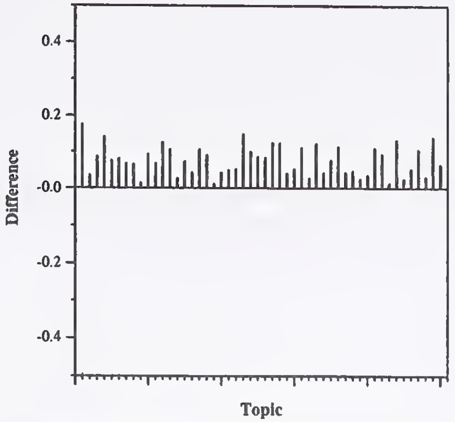
	New
Average precision	0.73
Average recall	0.96
Average F	0.819



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NLPR03n2s2
Num topics	

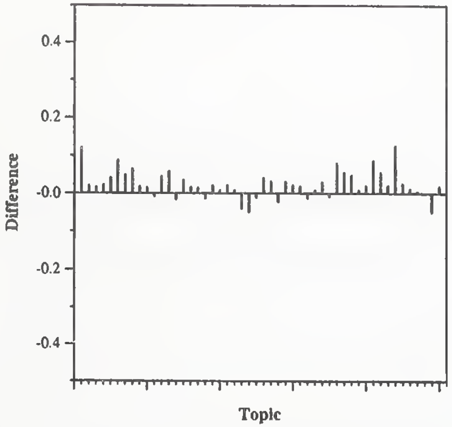
	New
Average precision	0.72
Average recall	0.97
Average F	0.817



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ccsum2svdpqr
Num topics	

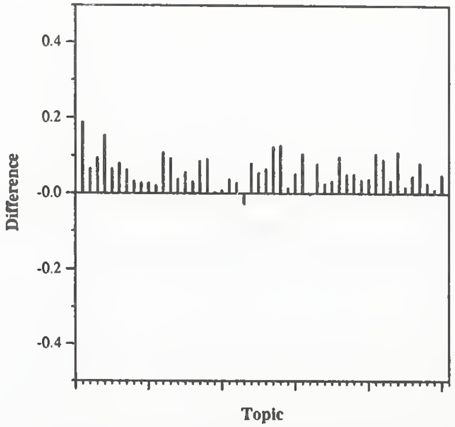
	New
Average precision	0.70
Average recall	0.87
Average F	0.764



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ccsumt2pqr
Num topics	

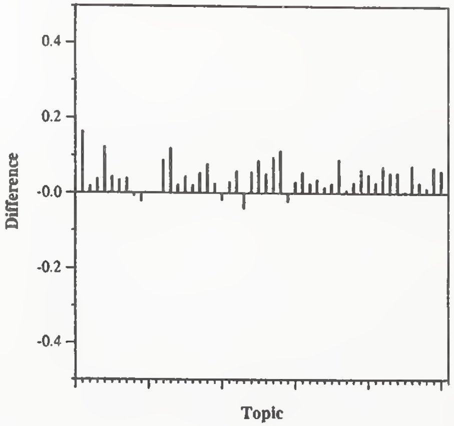
	New
Average precision	0.72
Average recall	0.94
Average F	0.800



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ccsumt2qr
Num topics	

	New
Average precision	0.71
Average recall	0.91
Average F	0.784

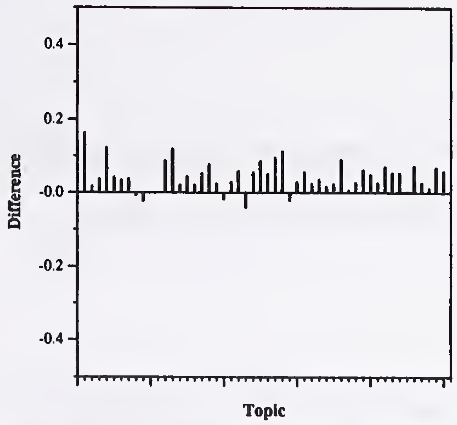


Difference from Median in F score per topic for novel sentences

Novelty track results, task 2 — Center for Computing Science and U. Maryland

Summary Statistics	
Run ID	ccsumt2svdqr
Num topics	

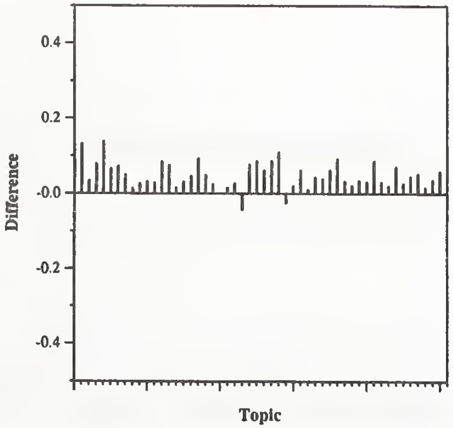
	New
Average precision	0.71
Average recall	0.91
Average F	0.784



Difference from Median in F score per topic for novel sentences

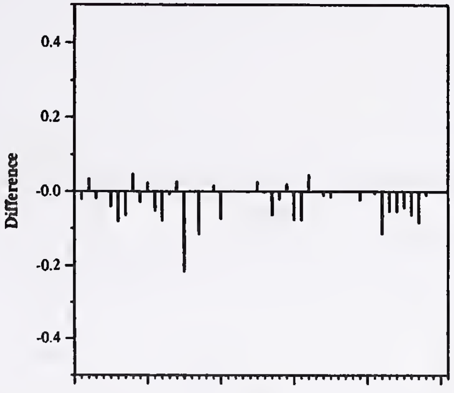
Summary Statistics	
Run ID	clr03n2
Num topics	

	New
Average precision	0.71
Average recall	0.91
Average F	0.788



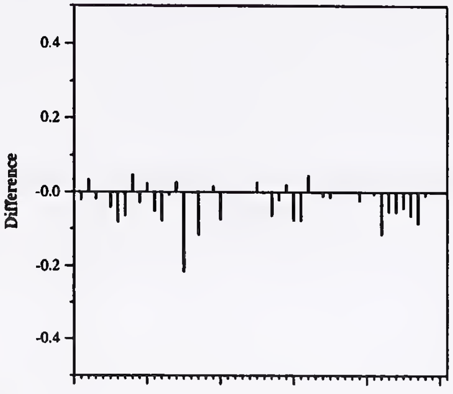
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	Irit1
Num topics	
	New
Average precision	0.71
Average recall	0.76
Average F	0.715



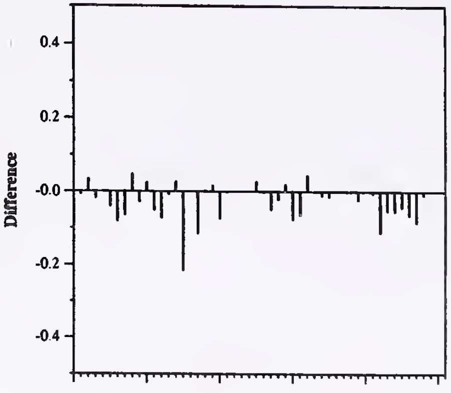
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	Irit5q
Num topics	
	New
Average precision	0.71
Average recall	0.76
Average F	0.715



Difference from Median in F score per topic for novel sentences

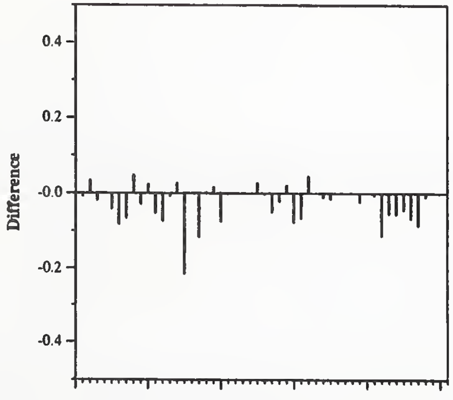
Summary Statistics	
Run ID	IritMtm4
Num topics	
	New
Average precision	0.71
Average recall	0.76
Average F	0.716



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	IritMtm5
Num topics	

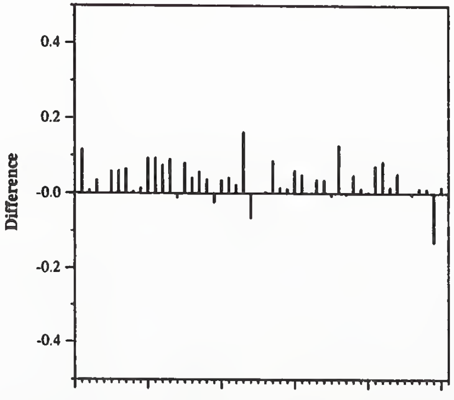
	New
Average precision	0.71
Average recall	0.76
Average F	0.716



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	Irito
Num topics	

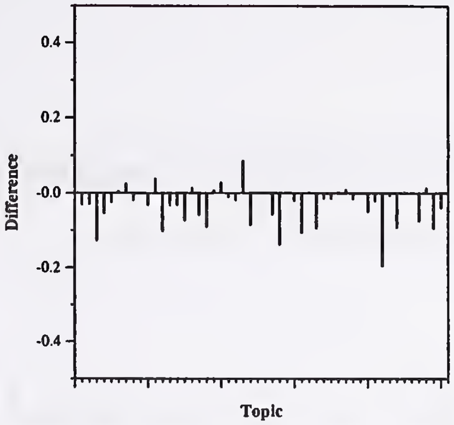
	New
Average precision	0.65
Average recall	1.00
Average F	0.774



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	MeijiHilF21
Num topics	

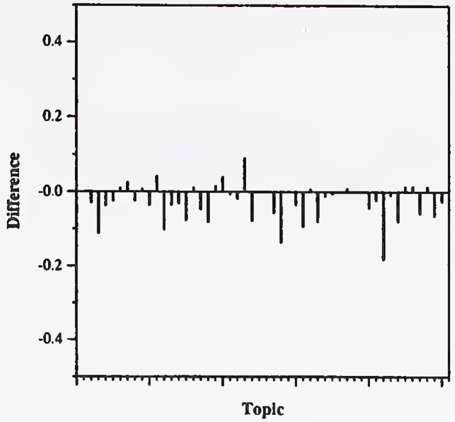
	New
Average precision	0.68
Average recall	0.77
Average F	0.708



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	MeijiHilF22
Num topics	

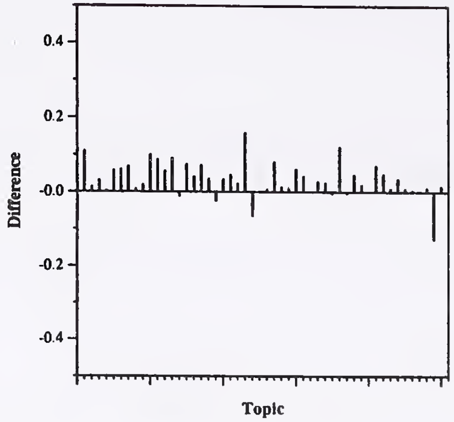
	New
Average precision	0.69
Average recall	0.78
Average F	0.713



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	MeijiHilF23
Num topics	

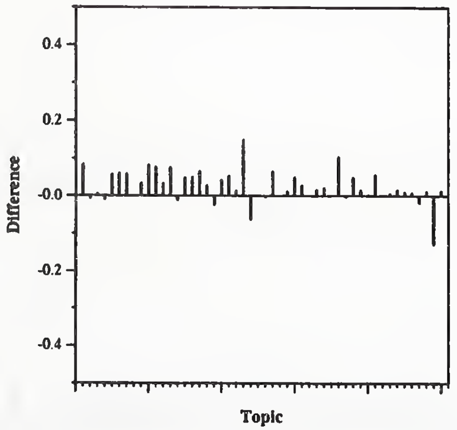
	New
Average precision	0.65
Average recall	0.99
Average F	0.773



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	MeijiHilF24
Num topics	

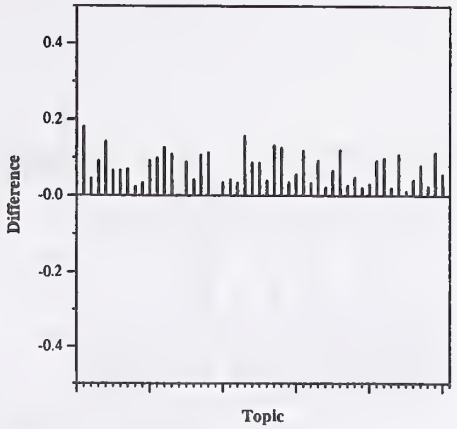
	New
Average precision	0.65
Average recall	0.96
Average F	0.765



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NTU21
Num topics	

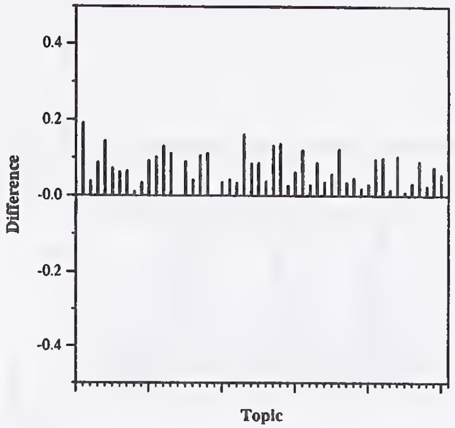
	New
Average precision	0.71
Average recall	0.98
Average F	0.812



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NTU22
Num topics	

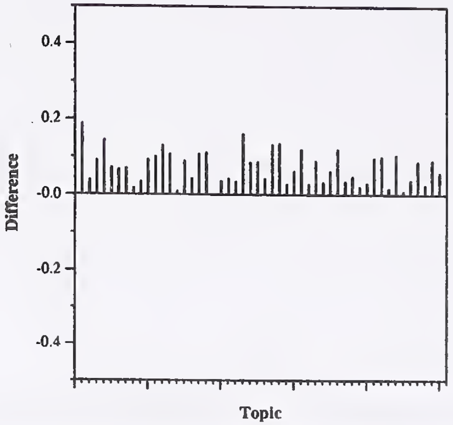
	New
Average precision	0.70
Average recall	0.99
Average F	0.811



Difference from Median in F score per topic for novel sentences

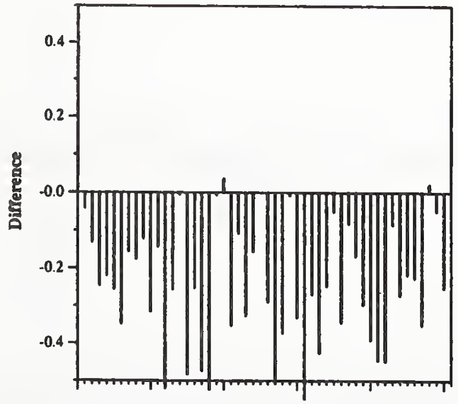
Summary Statistics	
Run ID	NTU23
Num topics	

	New
Average precision	0.70
Average recall	0.99
Average F	0.812



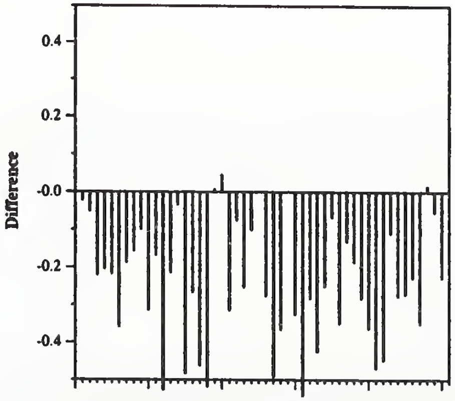
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NTU24
Num topics	
	New
Average precision	0.74
Average recall	0.42
Average F	0.495



Difference from Median in F score per topic for novel sentences

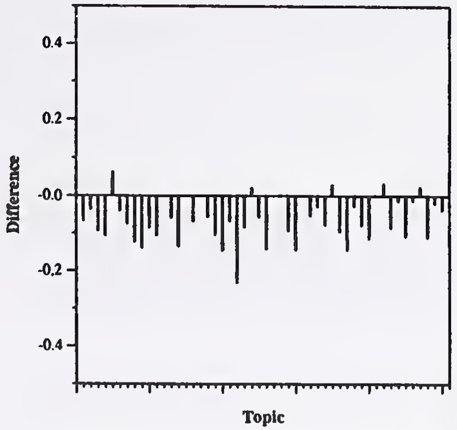
Summary Statistics	
Run ID	NTU25
Num topics	
	New
Average precision	0.74
Average recall	0.42
Average F	0.501



Difference from Median in F score per topic for novel sentences

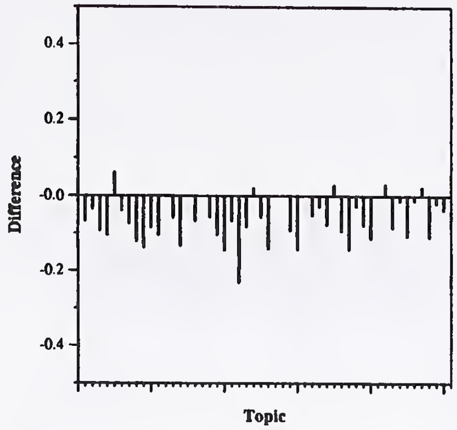
Novelty track results, task 2 — Tsinghua University (Ma)

Summary Statistics	
Run ID	THUIRnv0321
Num topics	
	New
Average precision	0.69
Average recall	0.69
Average F	0.679



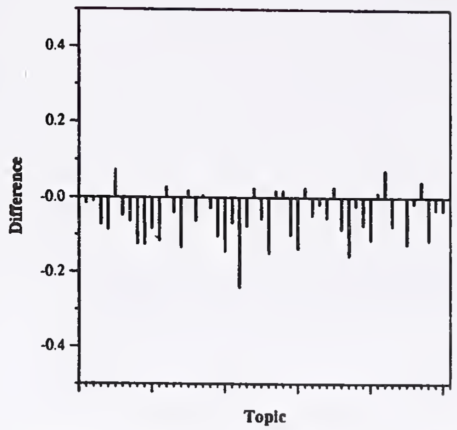
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	THUIRnv0322
Num topics	
	New
Average precision	0.69
Average recall	0.69
Average F	0.679



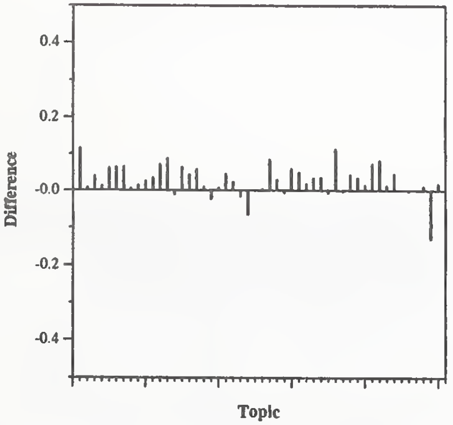
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	THUIRnv0323
Num topics	
	New
Average precision	0.68
Average recall	0.72
Average F	0.687



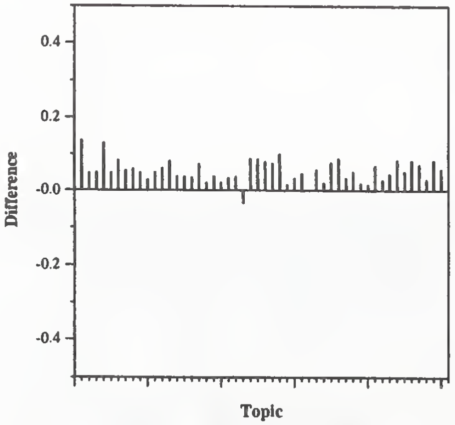
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	UIowa03Nov03
Num topics	
	New
Average precision	0.65
Average recall	0.98
Average F	0.767



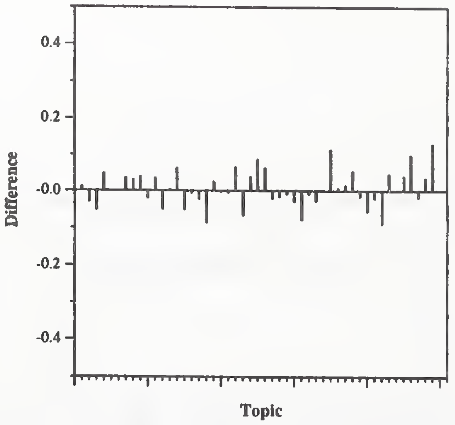
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	UIowa03Nov04
Num topics	
	New
Average precision	0.73
Average recall	0.90
Average F	0.794



Difference from Median in F score per topic for novel sentences

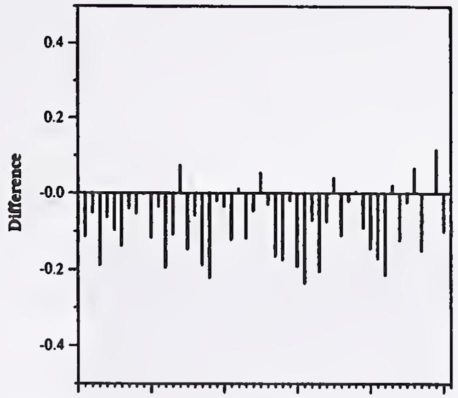
Summary Statistics	
Run ID	UIowa03Nov05
Num topics	
	New
Average precision	0.76
Average recall	0.77
Average F	0.746



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	UIowa03Nov06
Num topics	

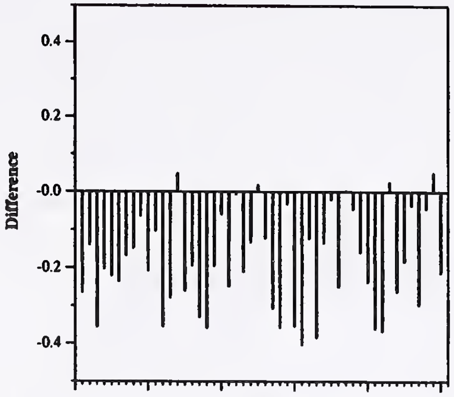
	New
Average precision	0.78
Average recall	0.60
Average F	0.659



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	UIowa03Nov07
Num topics	

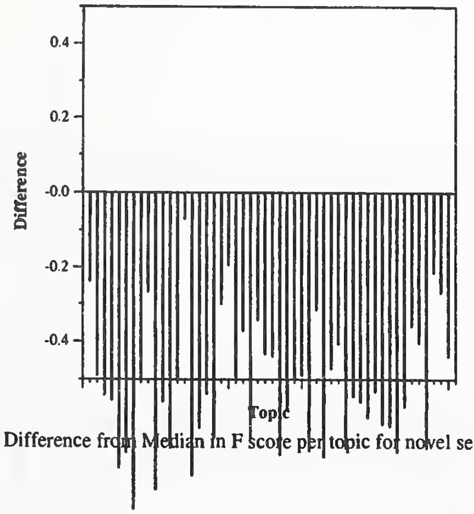
	New
Average precision	0.80
Average recall	0.45
Average F	0.555



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	umbcnew1
Num topics	

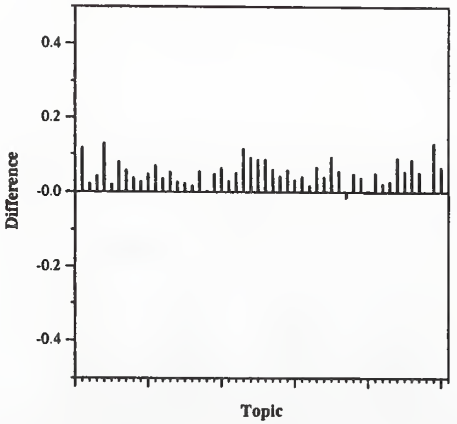
	New
Average precision	0.56
Average recall	0.15
Average F	0.222



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	umbcnew2
Num topics	

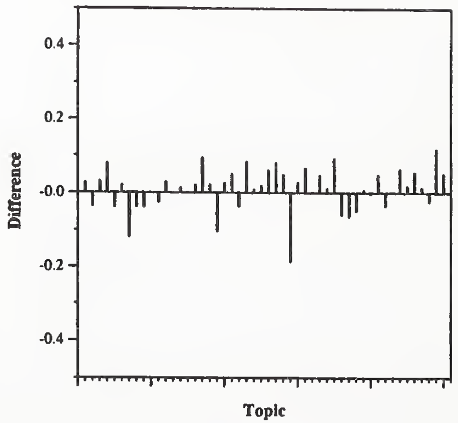
	New
Average precision	0.76
Average recall	0.85
Average F	0.792



Difference from Median in F score per topic for novel sentences

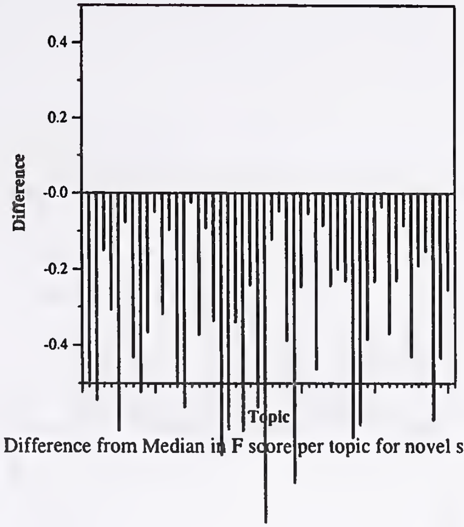
Summary Statistics	
Run ID	umbcnew3
Num topics	

	New
Average precision	0.75
Average recall	0.77
Average F	0.750



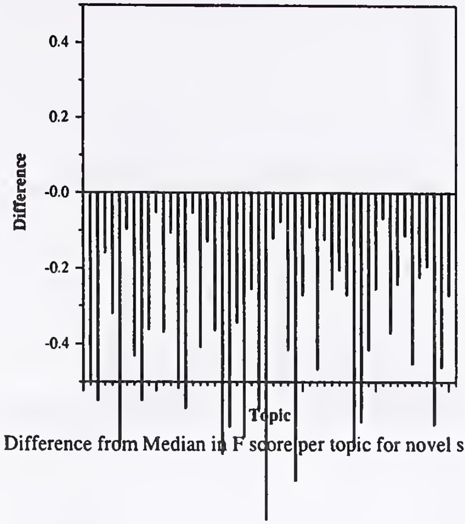
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	umich21
Num topics	
	New
Average precision	0.27
Average recall	1.00
Average F	0.394



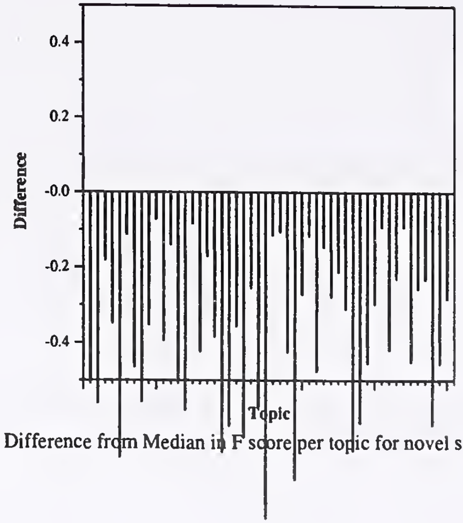
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	umich22
Num topics	
	New
Average precision	0.26
Average recall	0.91
Average F	0.377



Difference from Median in F score per topic for novel sentences

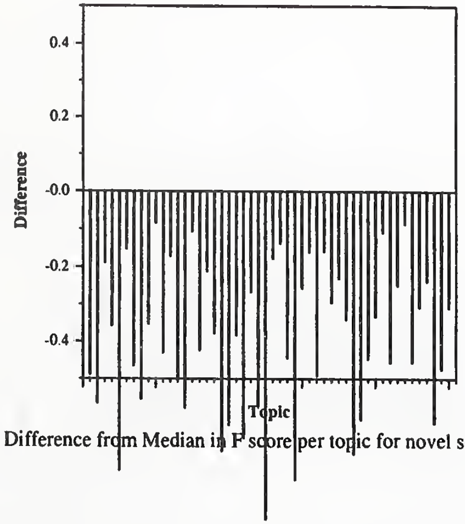
Summary Statistics	
Run ID	umich23
Num topics	
	New
Average precision	0.26
Average recall	0.84
Average F	0.361



Difference from Median in F score per topic for novel sentences

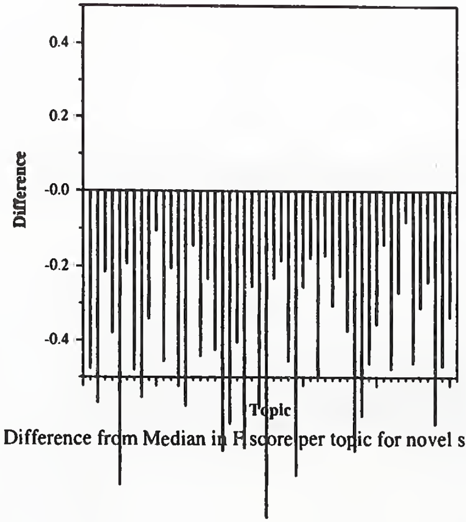
Novelty track results, task 2 — University of Michigan

Summary Statistics	
Run ID	umich24
Num topics	
	New
Average precision	0.25
Average recall	0.77
Average F	0.347



Difference from Median in F score per topic for novel sentences

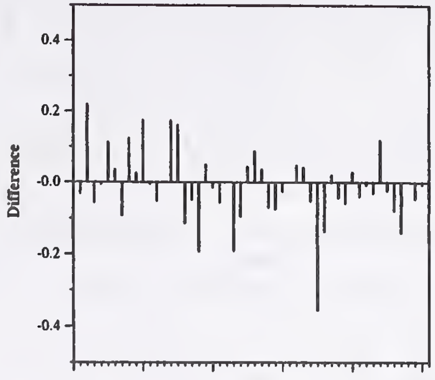
Summary Statistics	
Run ID	umich25
Num topics	
	New
Average precision	0.24
Average recall	0.70
Average F	0.332



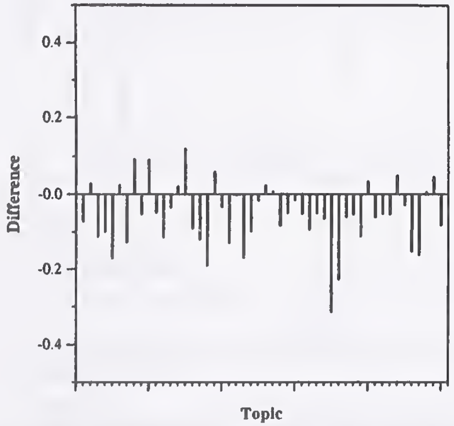
Difference from Median in F score per topic for novel sentences

Novelty track results, task 3 — Chinese Academy of Sciences (CAS-ICT)

Summary Statistics		
Run ID	ICT03NOV3IKK	
Num topics		50
	Relevant	New
Average precision	0.57	0.36
Average recall	0.58	0.39
Average F	0.548	0.348

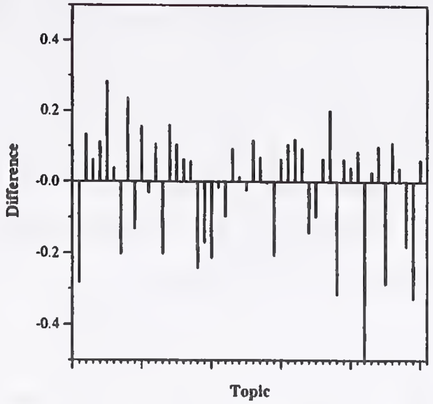


Difference from Median in F score per topic for relevant sentences

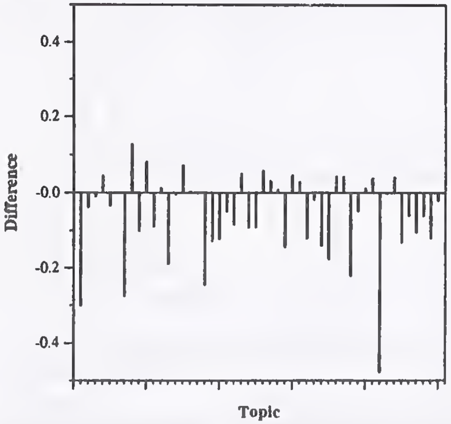


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	ICT03NOV3KNN	
Num topics		50
	Relevant	New
Average precision	0.57	0.35
Average recall	0.56	0.37
Average F	0.547	0.346

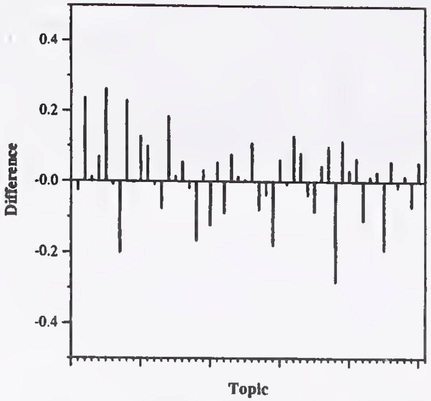


Difference from Median in F score per topic for relevant sentences

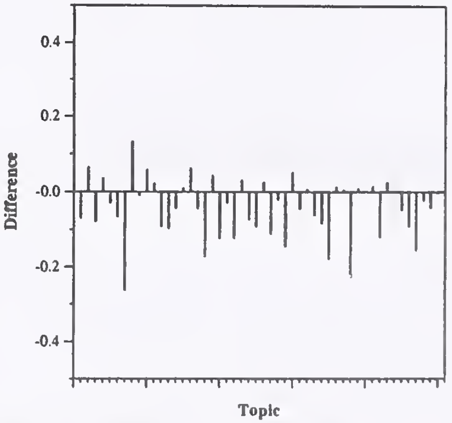


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	ICT03NOV3KNS	
Num topics		50
	Relevant	New
Average precision	0.60	0.37
Average recall	0.58	0.39
Average F	0.572	0.362

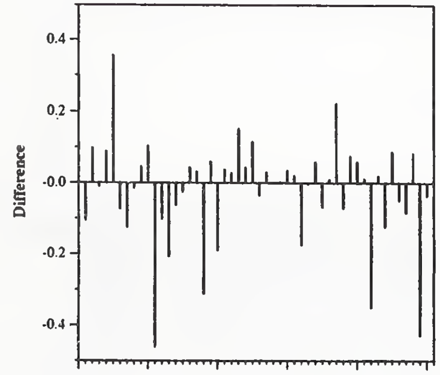


Difference from Median in F score per topic for relevant sentences

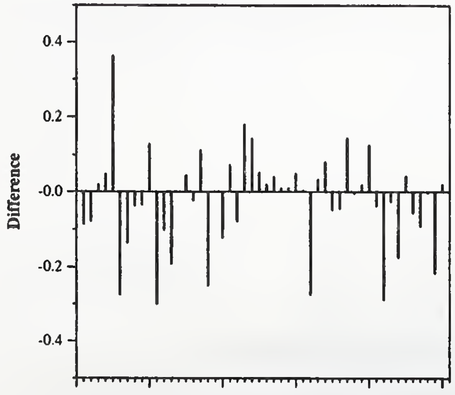


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	ICT03NOV3WN3	
Num topics		50
	Relevant	New
Average precision	0.68	0.43
Average recall	0.49	0.41
Average F	0.537	0.381

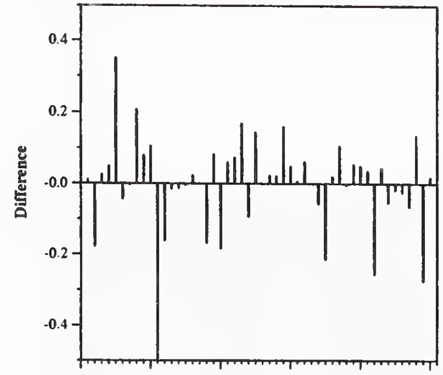


Difference from Median in F score per topic for relevant sentences

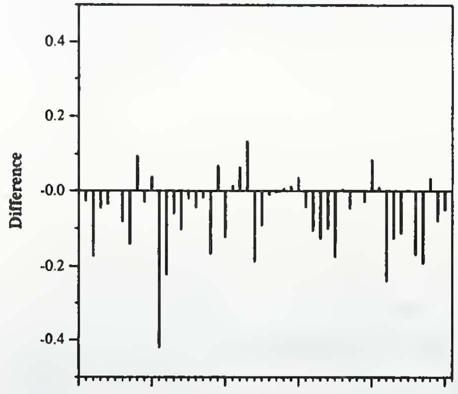


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	ICT03NOV3WND	
Num topics		50
	Relevant	New
Average precision	0.65	0.39
Average recall	0.53	0.35
Average F	0.557	0.346



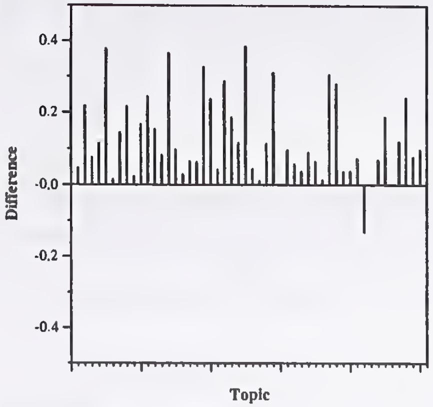
Difference from Median in F score per topic for relevant sentences



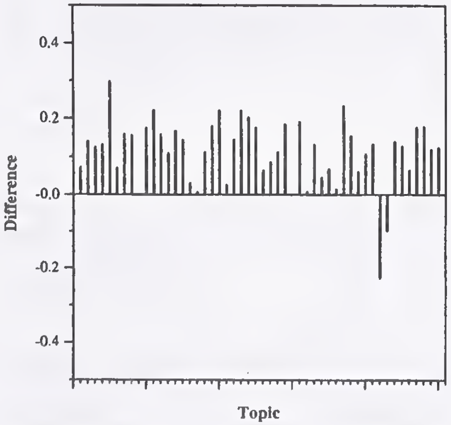
Difference from Median in F score per topic for novel sentences

Novelty track results, task 3 — Chinese Academy of Sciences (CAS-NLPR)

Summary Statistics		
Run ID		NLPR03n3d1
Num topics		50
	Relevant	New
Average precision	0.60	0.44
Average recall	0.88	0.75
Average F	0.687	0.518

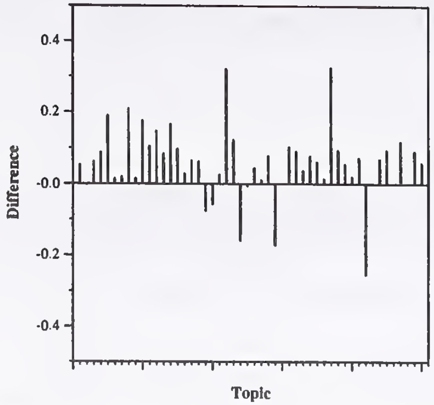


Difference from Median in F score per topic for relevant sentences

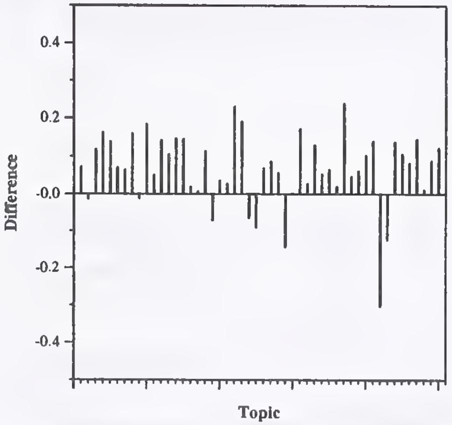


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NLPR03n3d2
Num topics		50
	Relevant	New
Average precision	0.60	0.43
Average recall	0.75	0.65
Average F	0.618	0.472

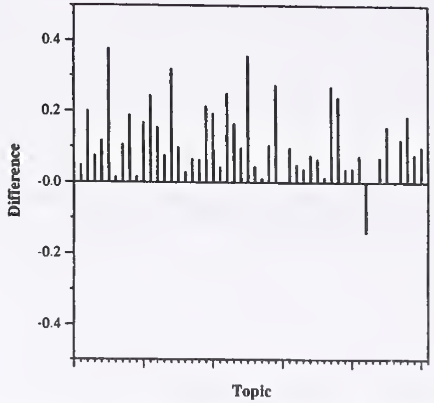


Difference from Median in F score per topic for relevant sentences

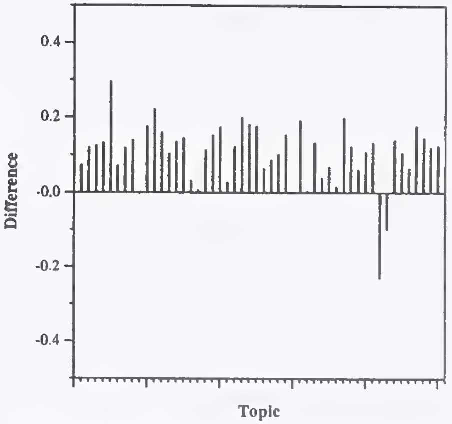


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NLPR03n3d3
Num topics		50
	Relevant	New
Average precision	0.58	0.43
Average recall	0.87	0.75
Average F	0.674	0.509

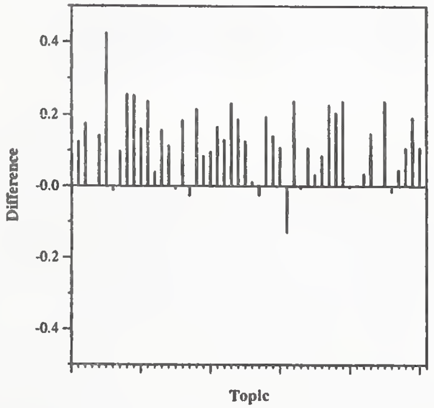


Difference from Median in F score per topic for relevant sentences

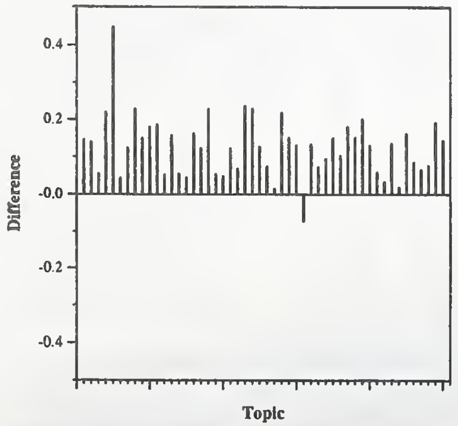


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	NLPR03n3s1	
Num topics		50
	Relevant	New
Average precision	0.66	0.48
Average recall	0.82	0.78
Average F	0.677	0.532

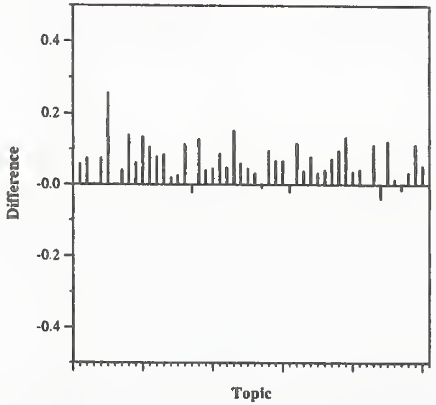


Difference from Median in F score per topic for relevant sentences

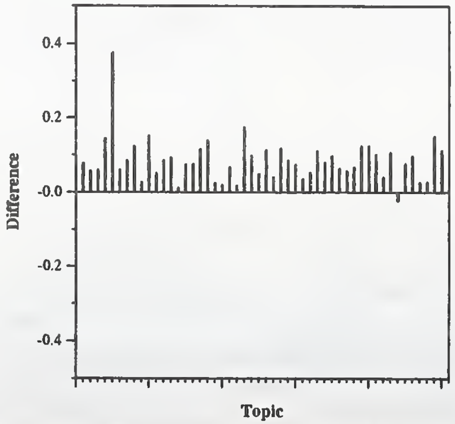


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	NLPR03n3s2	
Num topics		50
	Relevant	New
Average precision	0.58	0.42
Average recall	0.83	0.79
Average F	0.624	0.489

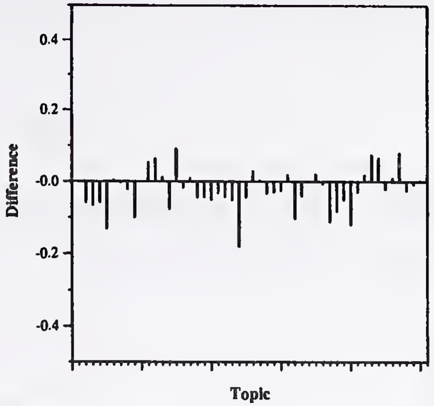


Difference from Median in F score per topic for relevant sentences

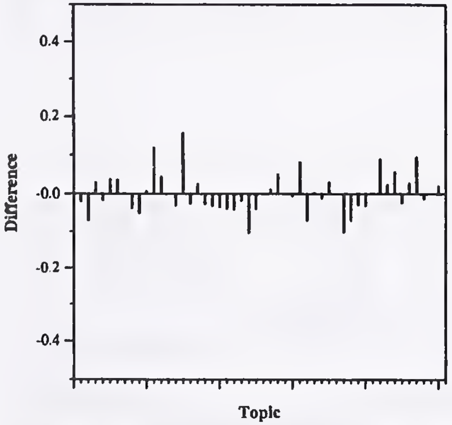


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		ccsum3pqr
Num topics		50
	Relevant	New
Average precision	0.43	0.30
Average recall	0.92	0.90
Average F	0.538	0.405

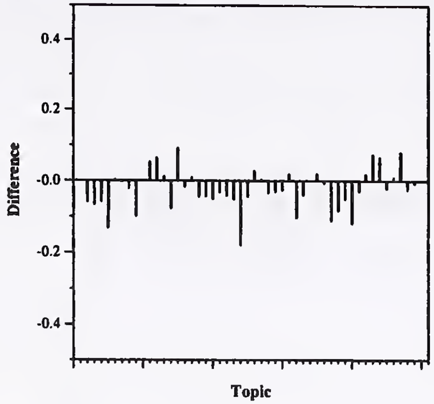


Difference from Median in F score per topic for relevant sentences

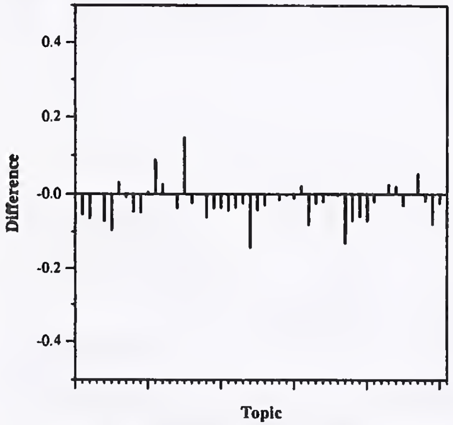


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		ccsum3qr
Num topics		50
	Relevant	New
Average precision	0.43	0.27
Average recall	0.92	0.92
Average F	0.538	0.382

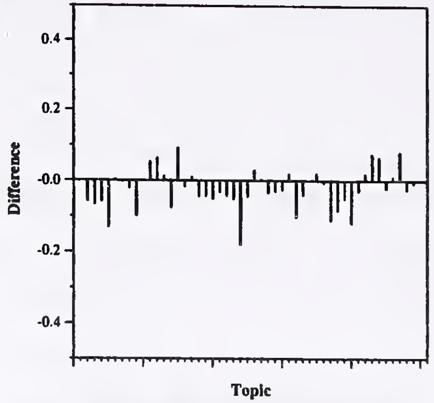


Difference from Median in F score per topic for relevant sentences

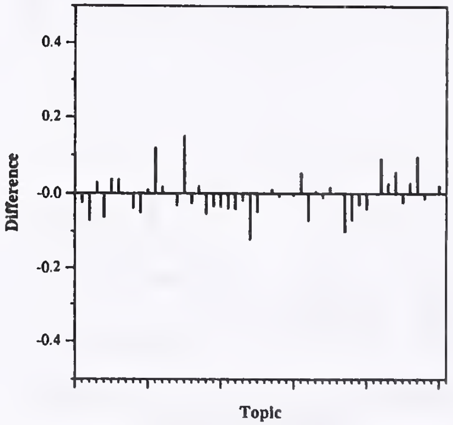


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		ccsum3svdpqr
Num topics		50
	Relevant	New
Average precision	0.43	0.29
Average recall	0.92	0.89
Average F	0.538	0.401

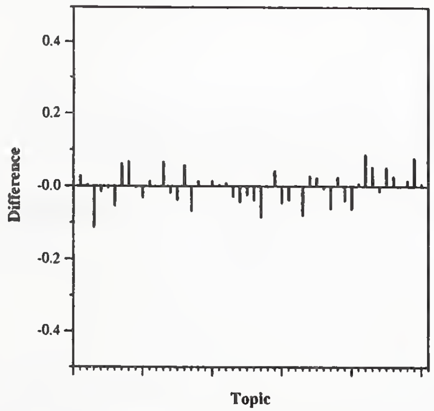


Difference from Median in F score per topic for relevant sentences

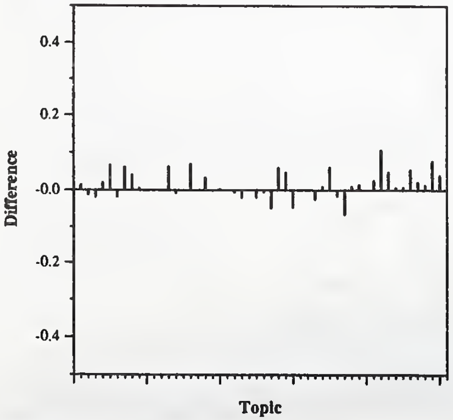


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		clr03n3f01
Num topics		50
	Relevant	New
Average precision	0.48	0.33
Average recall	0.84	0.79
Average F	0.558	0.419

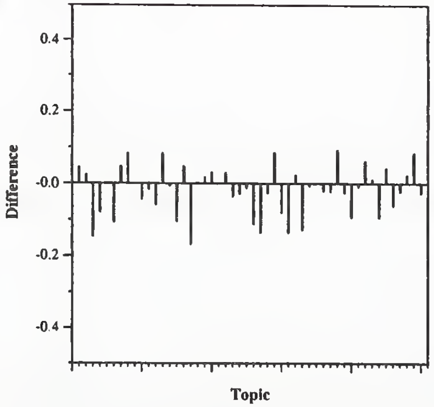


Difference from Median in F score per topic for relevant sentences

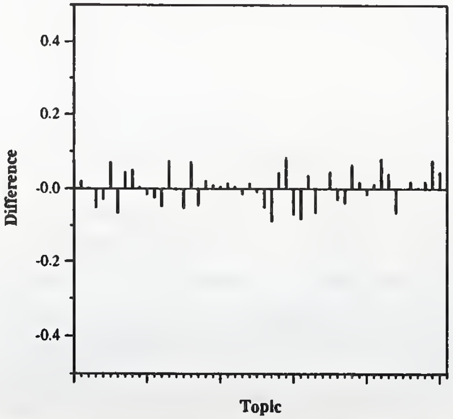


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		clr03n3f02
Num topics		50
	Relevant	New
Average precision	0.48	0.33
Average recall	0.77	0.73
Average F	0.541	0.408

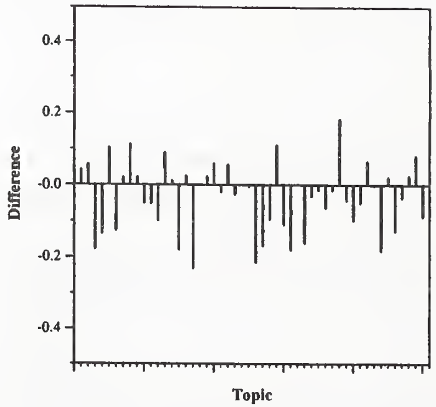


Difference from Median in F score per topic for relevant sentences

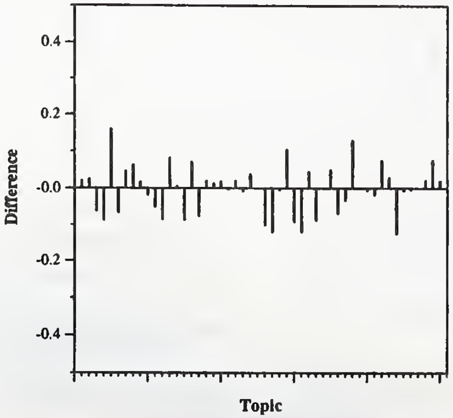


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		clr03n3f03
Num topics		50
	Relevant	New
Average precision	0.48	0.33
Average recall	0.72	0.69
Average F	0.527	0.401

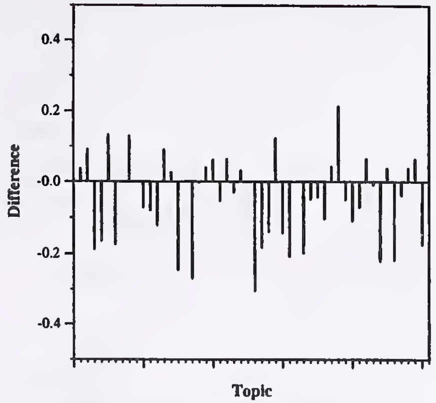


Difference from Median in F score per topic for relevant sentences

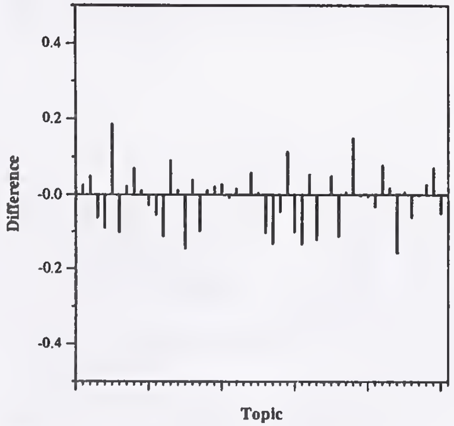


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		clr03n3f04
Num topics		50
	Relevant	New
Average precision	0.48	0.33
Average recall	0.68	0.65
Average F	0.513	0.395

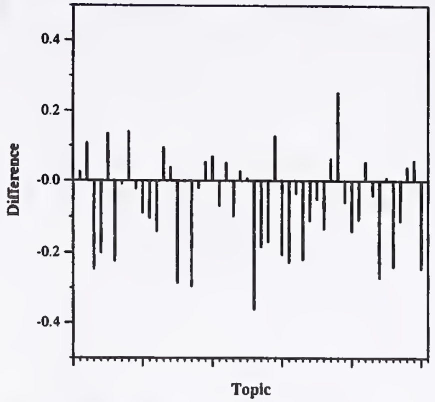


Difference from Median in F score per topic for relevant sentences

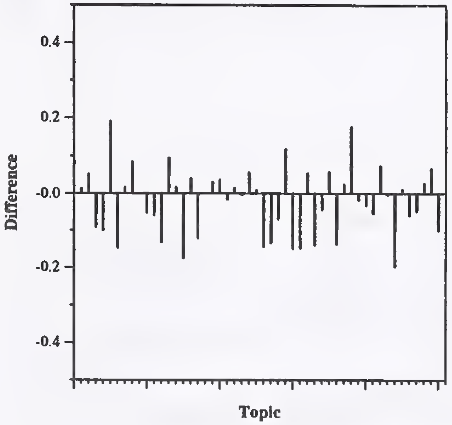


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		clr03n3f05
Num topics		50
	Relevant	New
Average precision	0.48	0.33
Average recall	0.63	0.61
Average F	0.493	0.383

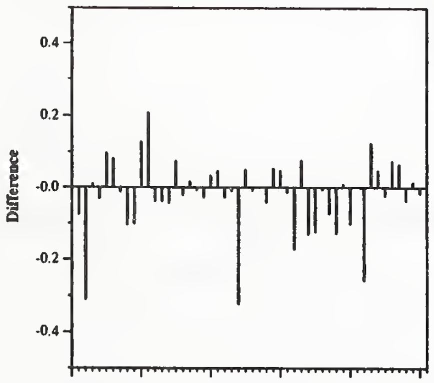


Difference from Median in F score per topic for relevant sentences

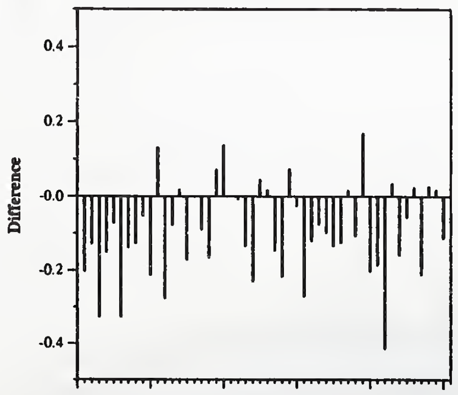


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		MeijiHilF31
Num topics		50
	Relevant	New
Average precision	0.54	0.44
Average recall	0.58	0.26
Average F	0.540	0.310

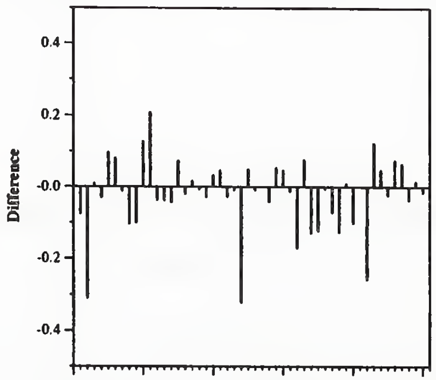


Difference from Median in F score per topic for relevant sentences

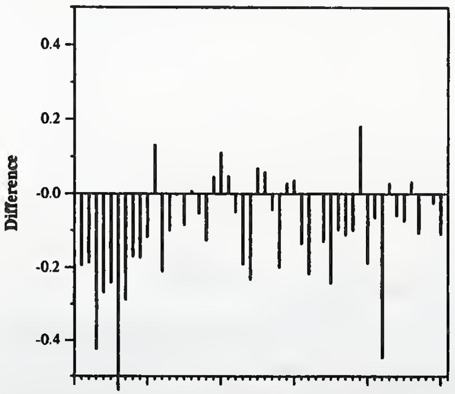


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		MeijiHilF32
Num topics		50
	Relevant	New
Average precision	0.54	0.46
Average recall	0.58	0.25
Average F	0.540	0.301

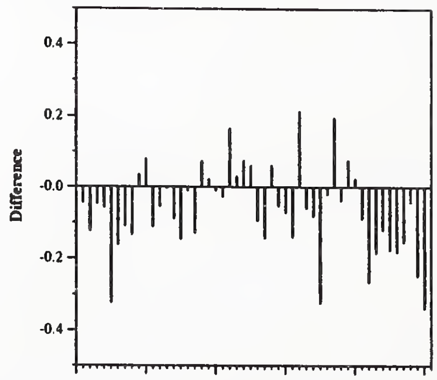


Difference from Median in F score per topic for relevant sentences

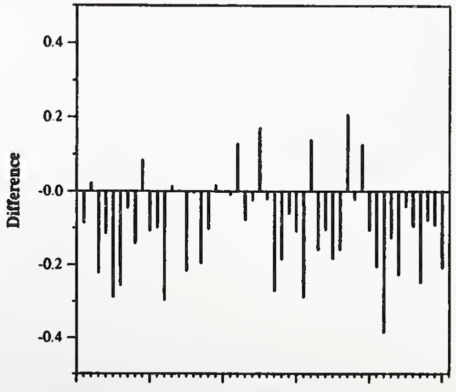


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		MeijiHilF33
Num topics		50
	Relevant	New
Average precision	0.54	0.46
Average recall	0.49	0.26
Average F	0.495	0.310

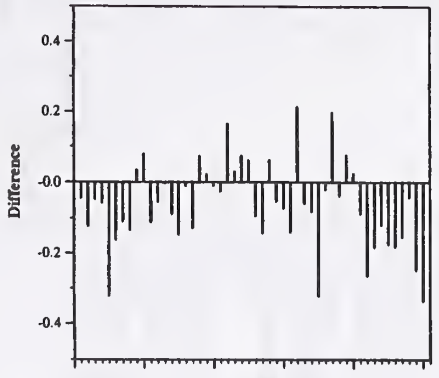


Difference from Median in F score per topic for relevant sentences

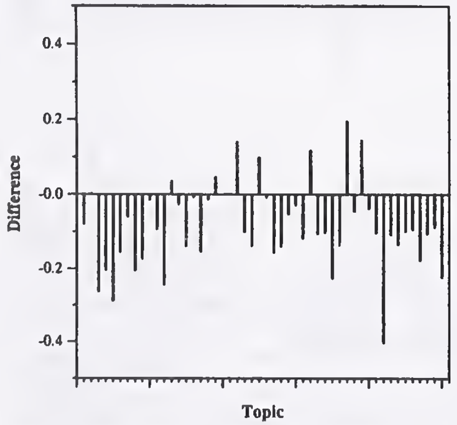


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		MeijiHilF34
Num topics		50
	Relevant	New
Average precision	0.54	0.47
Average recall	0.49	0.27
Average F	0.495	0.320

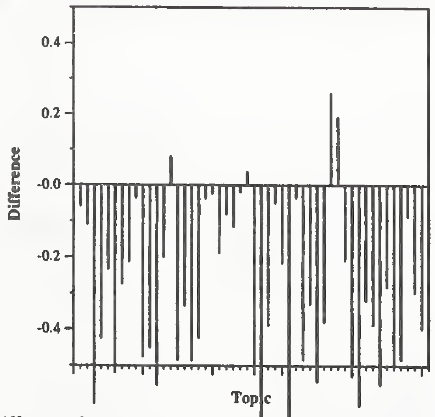


Difference from Median in F score per topic for relevant sentences

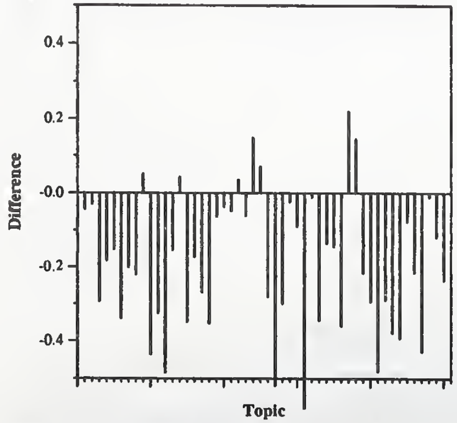


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NTU31
Num topics		50
	Relevant	New
Average precision	0.56	0.39
Average recall	0.20	0.19
Average F	0.266	0.217

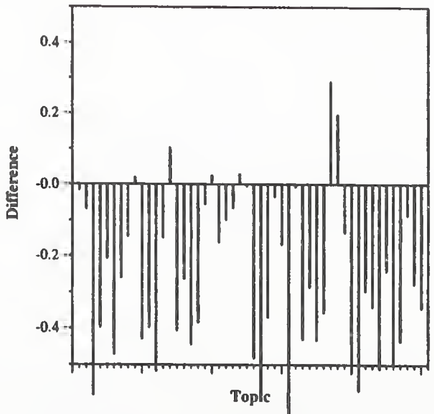


Difference from Median in F score per topic for relevant sentences

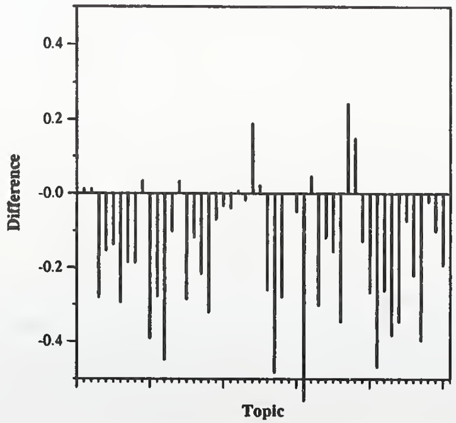


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NTU32
Num topics		50
	Relevant	New
Average precision	0.57	0.39
Average recall	0.25	0.23
Average F	0.301	0.241

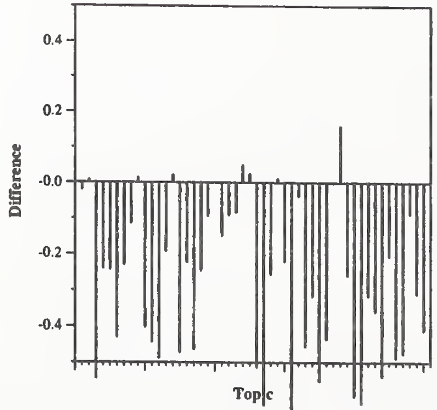


Difference from Median in F score per topic for relevant sentences

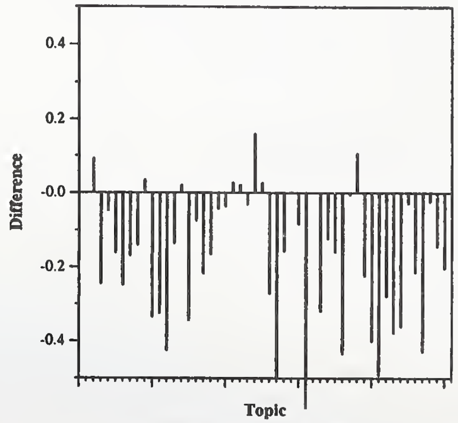


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NTU33
Num topics		50
	Relevant	New
Average precision	0.58	0.40
Average recall	0.22	0.21
Average F	0.287	0.236

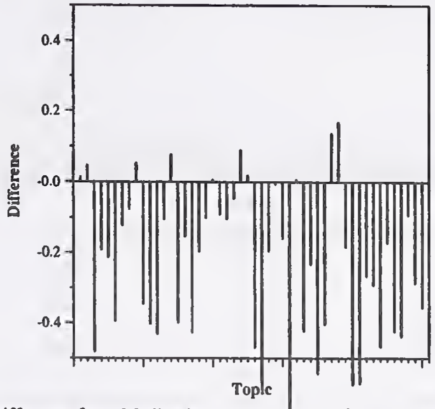


Difference from Median in F score per topic for relevant sentences

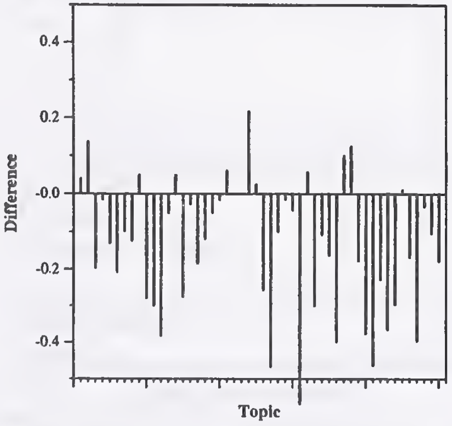


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NTU34
Num topics		50
	Relevant	New
Average precision	0.58	0.40
Average recall	0.27	0.26
Average F	0.330	0.270

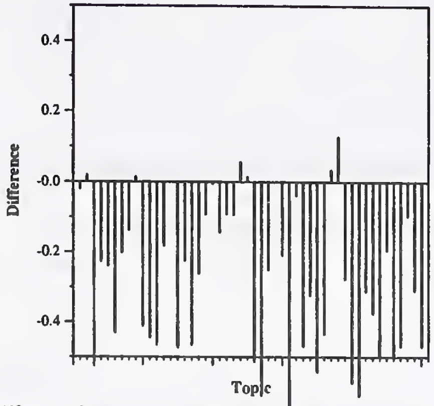


Difference from Median in F score per topic for relevant sentences

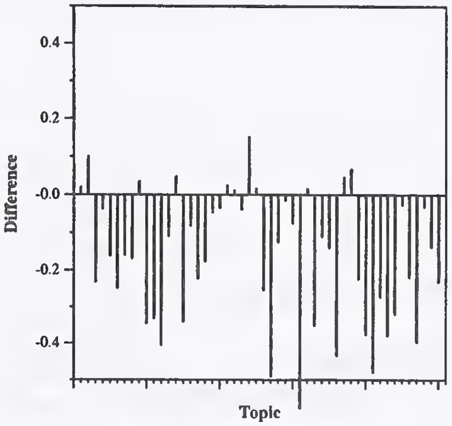


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		NTU35
Num topics		50
	Relevant	New
Average precision	0.57	0.39
Average recall	0.22	0.21
Average F	0.287	0.240

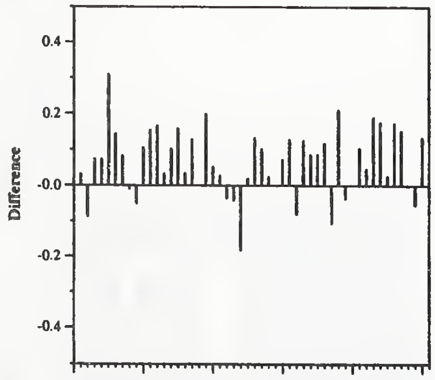


Difference from Median in F score per topic for relevant sentences

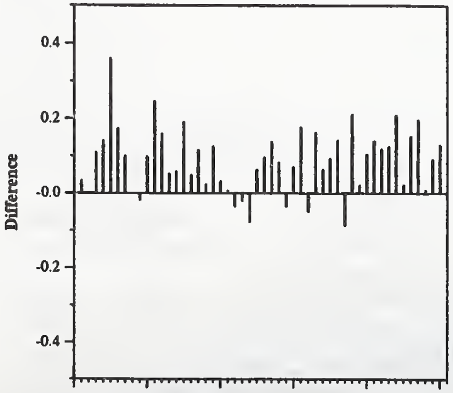


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	THUIRnv0331	
Num topics	50	
	Relevant	New
Average precision	0.57	0.42
Average recall	0.87	0.82
Average F	0.627	0.493

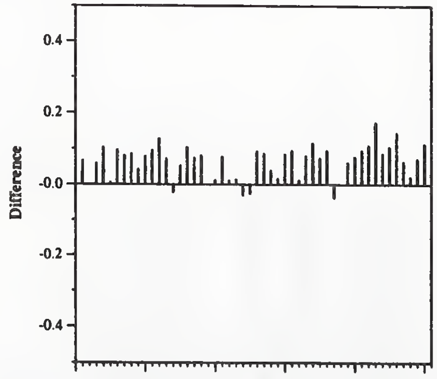


Difference from Median in F score per topic for relevant sentences

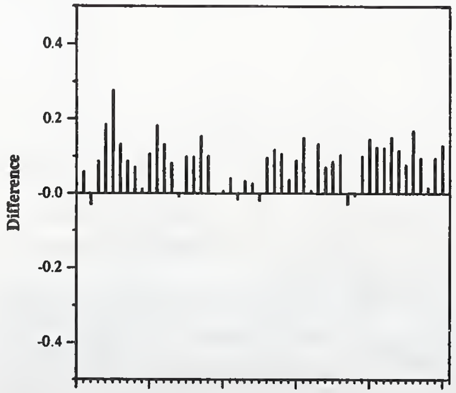


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	THUIRnv0332	
Num topics	50	
	Relevant	New
Average precision	0.56	0.40
Average recall	0.89	0.86
Average F	0.623	0.489

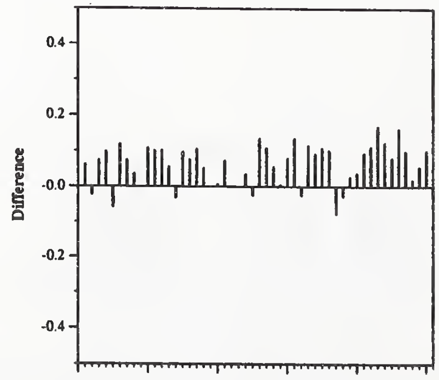


Difference from Median in F score per topic for relevant sentences

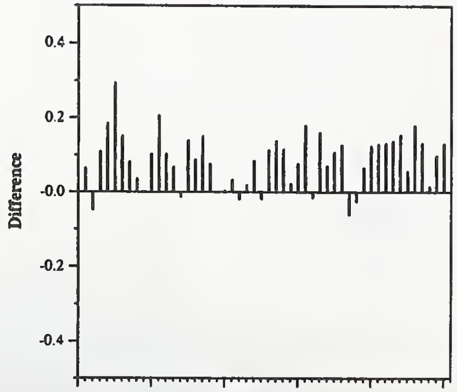


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID	THUIRnv0333	
Num topics	50	
	Relevant	New
Average precision	0.54	0.39
Average recall	0.92	0.88
Average F	0.621	0.491



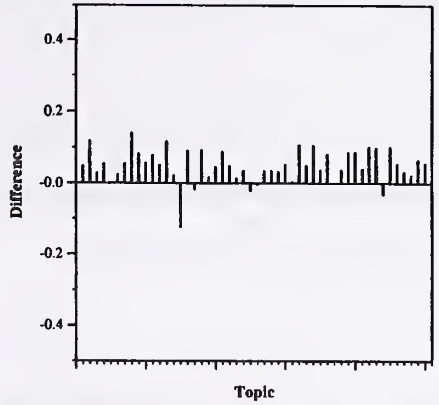
Difference from Median in F score per topic for relevant sentences



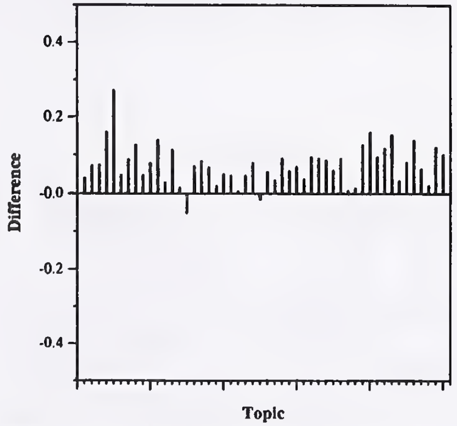
Difference from Median in F score per topic for novel sentences

Novelty track results, task 3 — Tsinghua University (Ma)

Summary Statistics		
Run ID	THUIRnv0334	
Num topics	50	
	Relevant	New
Average precision	0.58	0.43
Average recall	0.81	0.73
Average F	0.610	0.479

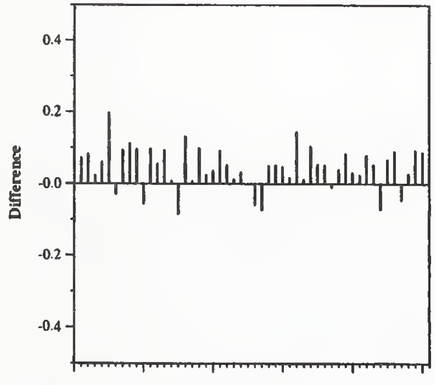


Difference from Median in F score per topic for relevant sentences

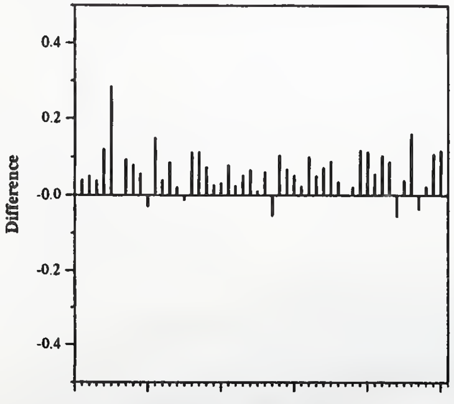


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		UIowa03Nov08
Num topics		50
	Relevant	New
Average precision	0.60	0.43
Average recall	0.78	0.71
Average F	0.606	0.466

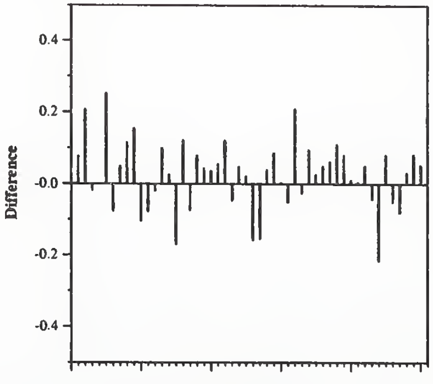


Difference from Median in F score per topic for relevant sentences

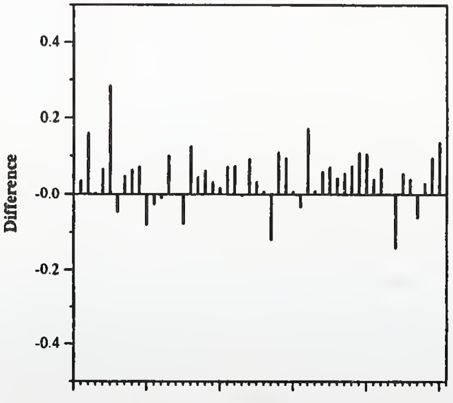


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		UIowa03Nov09
Num topics		50
	Relevant	New
Average precision	0.62	0.44
Average recall	0.69	0.62
Average F	0.585	0.448

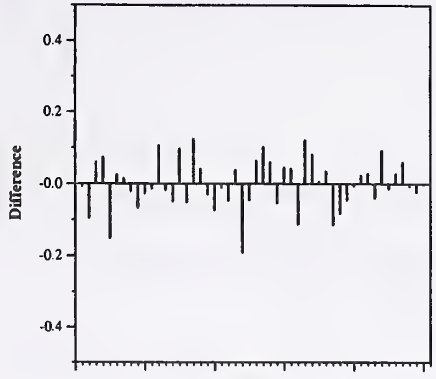


Difference from Median in F score per topic for relevant sentences

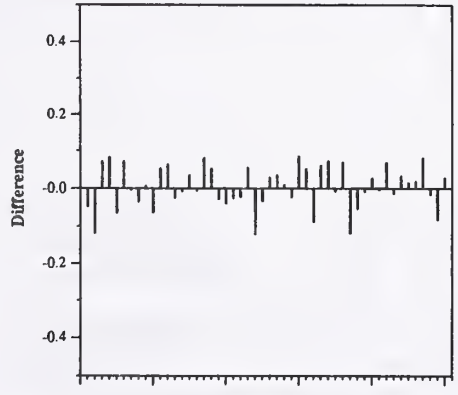


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		umich31
Num topics		50
	Relevant	New
Average precision	0.46	0.31
Average recall	0.92	0.83
Average F	0.560	0.409

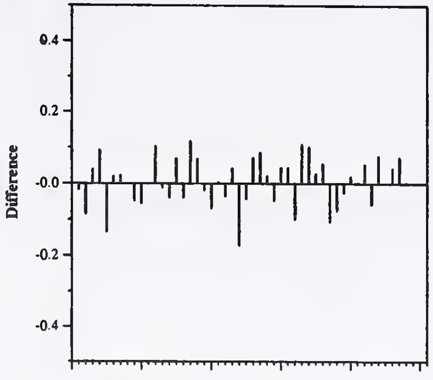


Difference from Median in F score per topic for relevant sentences

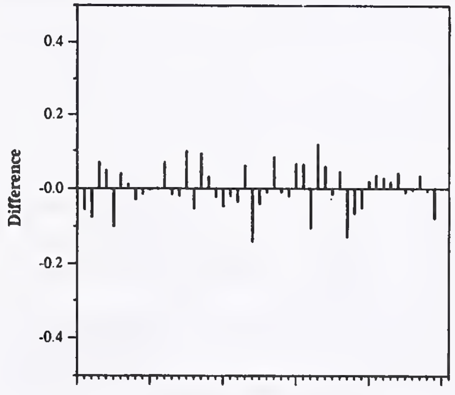


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		umich32
Num topics		50
	Relevant	New
Average precision	0.47	0.30
Average recall	0.90	0.90
Average F	0.565	0.405

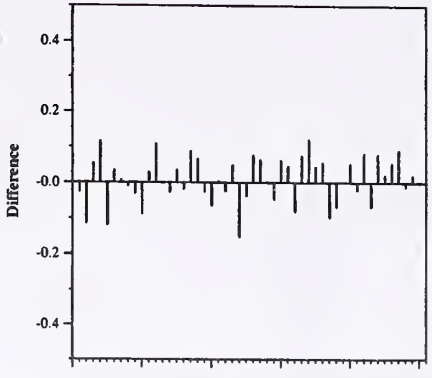


Difference from Median in F score per topic for relevant sentences

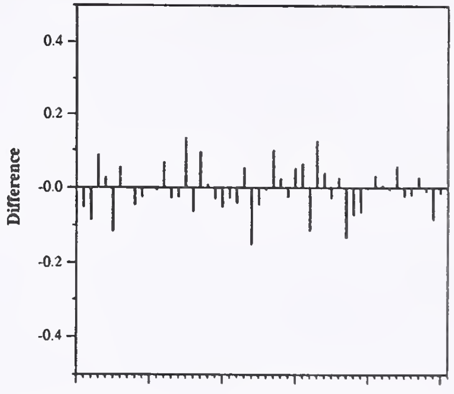


Difference from Median in F score per topic for novel sentences

Summary Statistics		
Run ID		umich33
Num topics		50
	Relevant	New
Average precision	0.49	0.29
Average recall	0.87	0.92
Average F	0.569	0.399



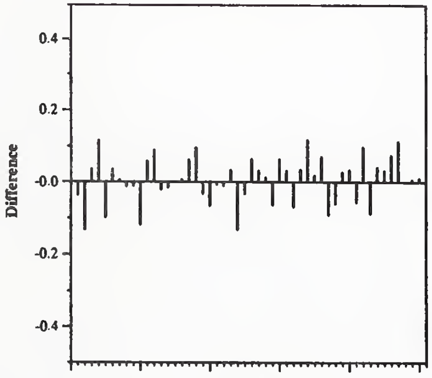
Difference from Median in F score per topic for relevant sentences



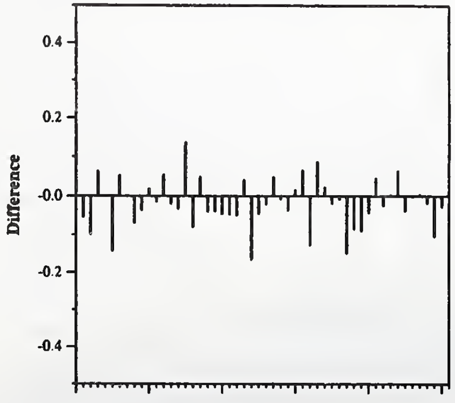
Difference from Median in F score per topic for novel sentences

Summary Statistics

Run ID		umich34
Num topics		50
	Relevant	New
Average precision	0.50	0.27
Average recall	0.83	0.96
Average F	0.566	0.385



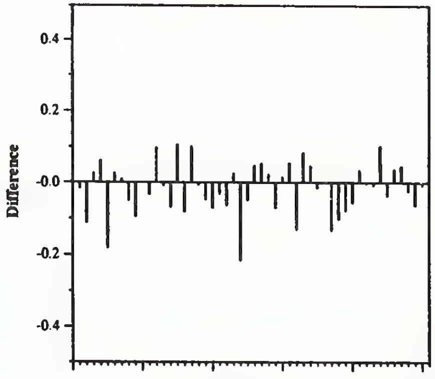
Difference from Median in F score per topic for relevant sentences



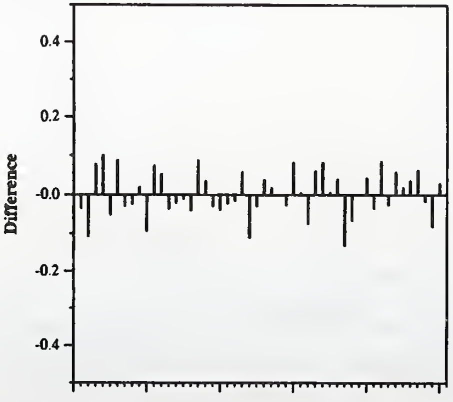
Difference from Median in F score per topic for novel sentences

Summary Statistics

Run ID		umich35
Num topics		50
	Relevant	New
Average precision	0.43	0.32
Average recall	0.96	0.78
Average F	0.543	0.408



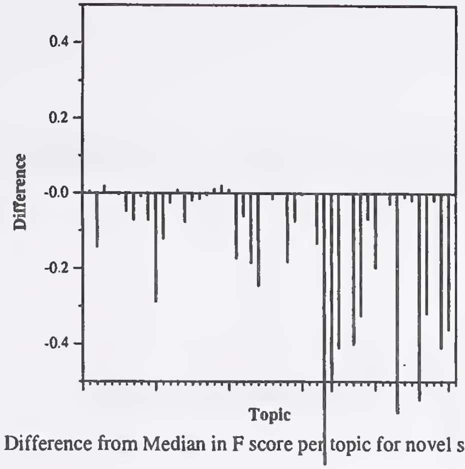
Difference from Median in F score per topic for relevant sentences



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ICT03NOV4ALL
Num topics	

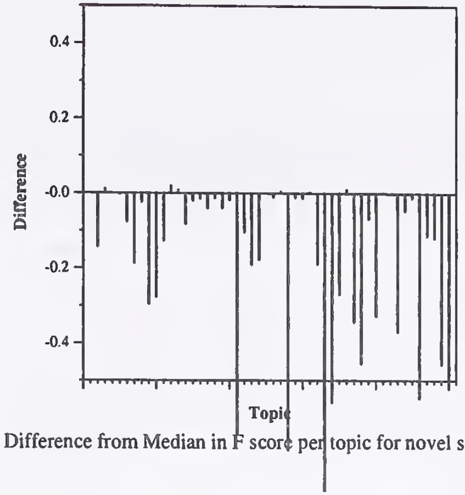
	New
Average precision	0.60
Average recall	0.68
Average F	0.598



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ICT03NOV4LFF
Num topics	

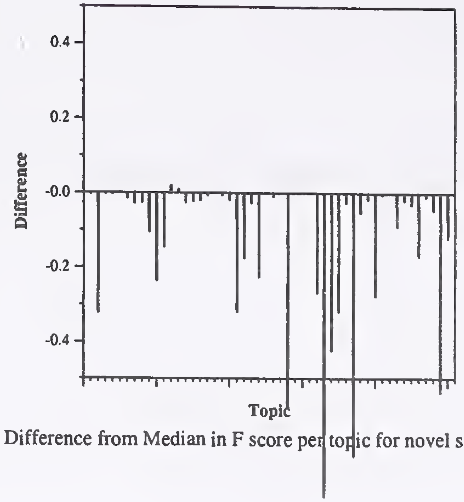
	New
Average precision	0.59
Average recall	0.64
Average F	0.568



Difference from Median in F score per topic for novel sentences

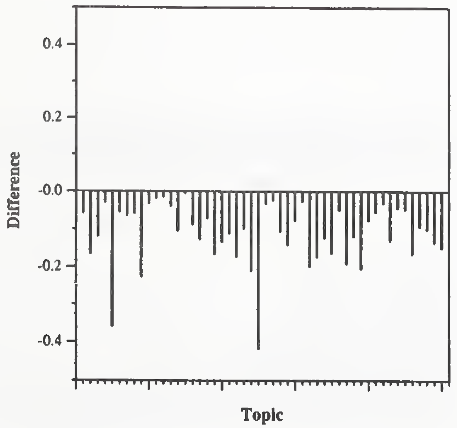
Summary Statistics	
Run ID	ICT03NOV4OTP
Num topics	

	New
Average precision	0.59
Average recall	0.70
Average F	0.610



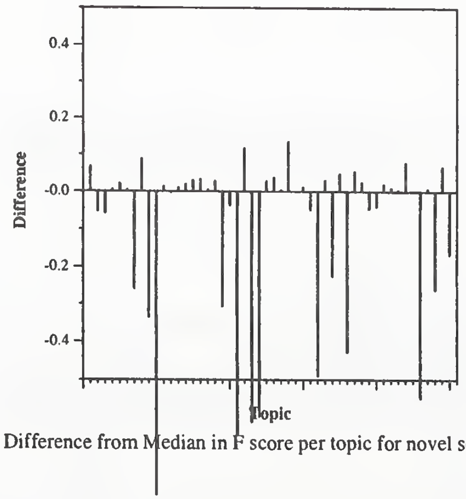
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ICT03NOV4SQR
Num topics	
	New
Average precision	0.61
Average recall	0.66
Average F	0.623



Difference from Median in F score per topic for novel sentences

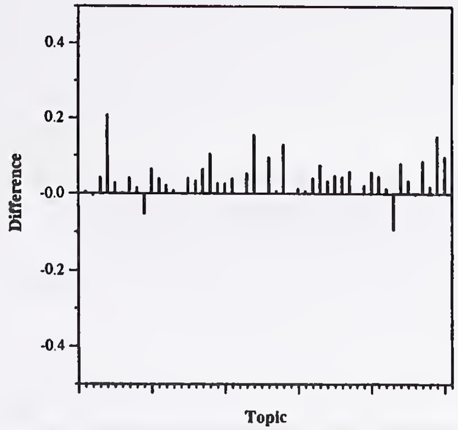
Summary Statistics	
Run ID	ICT03NOV4WNW
Num topics	
	New
Average precision	0.65
Average recall	0.72
Average F	0.636



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NLPR03n4d1
Num topics	

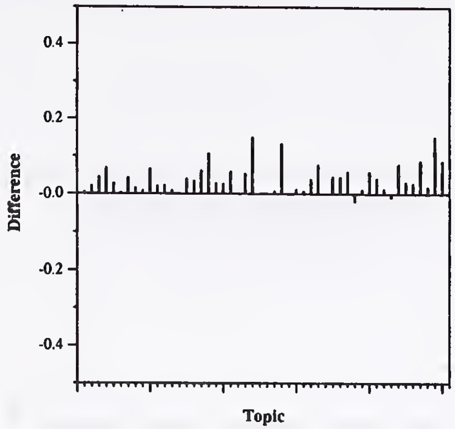
	New
Average precision	0.68
Average recall	0.93
Average F	0.775



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NLPR03n4d2
Num topics	

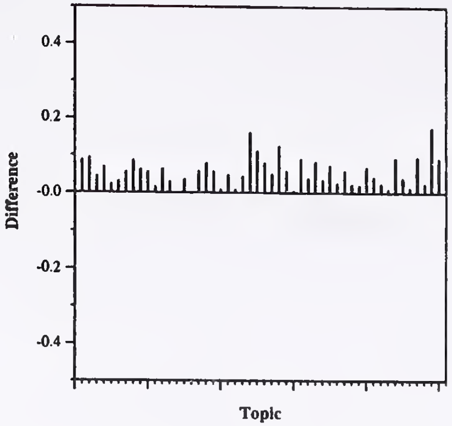
	New
Average precision	0.67
Average recall	0.94
Average F	0.773



Difference from Median in F score per topic for novel sentences

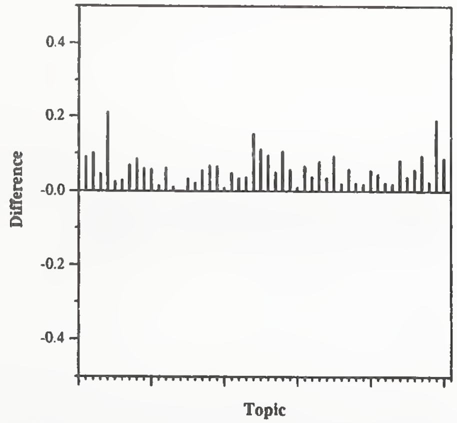
Summary Statistics	
Run ID	NLPR03n4s1
Num topics	

	New
Average precision	0.69
Average recall	0.96
Average F	0.789



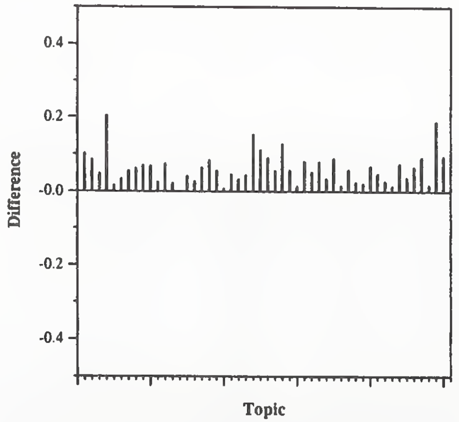
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NLPR03n4s2
Num topics	
	New
Average precision	0.70
Average recall	0.95
Average F	0.794



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NLPR03n4s3
Num topics	
	New
Average precision	0.70
Average recall	0.97
Average F	0.796

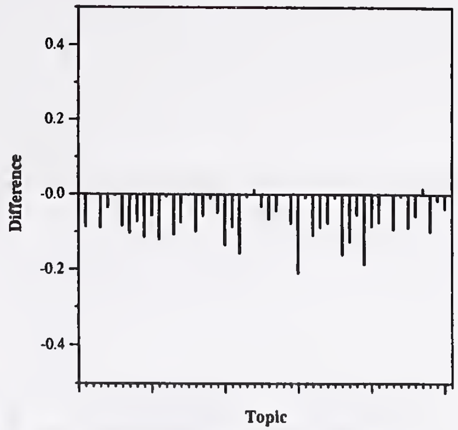


Difference from Median in F score per topic for novel sentences

Novelty track results, task 4 — Center for Computing Science and U. Maryland

Summary Statistics	
Run ID	ccsum4spq001
Num topics	

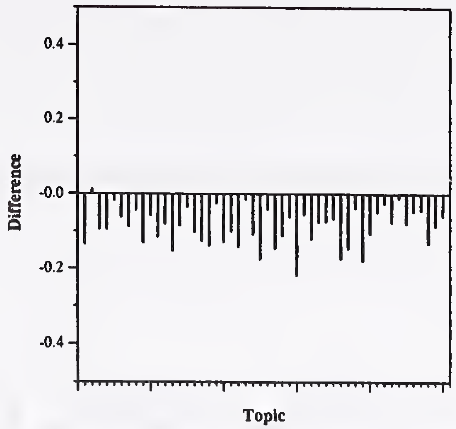
	New
Average precision	0.53
Average recall	0.98
Average F	0.668



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ccsum4svdpqr
Num topics	

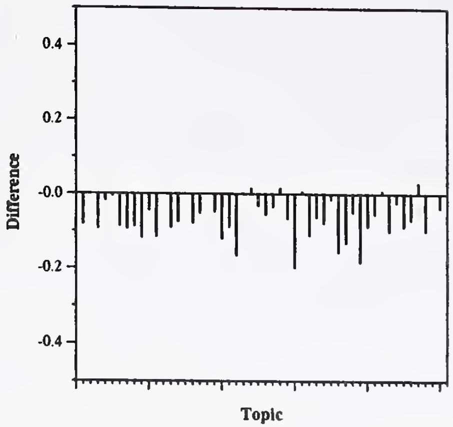
	New
Average precision	0.52
Average recall	0.91
Average F	0.646



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ccsumt4pqr
Num topics	

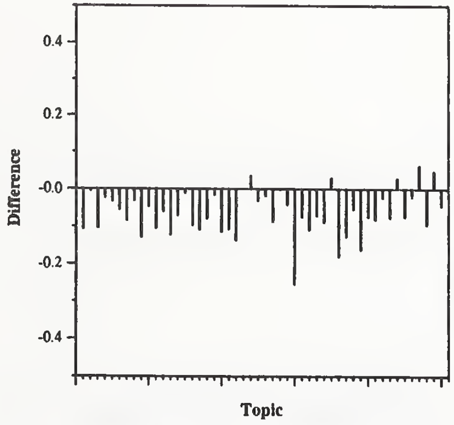
	New
Average precision	0.53
Average recall	0.99
Average F	0.672



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ccsumt4qr
Num topics	

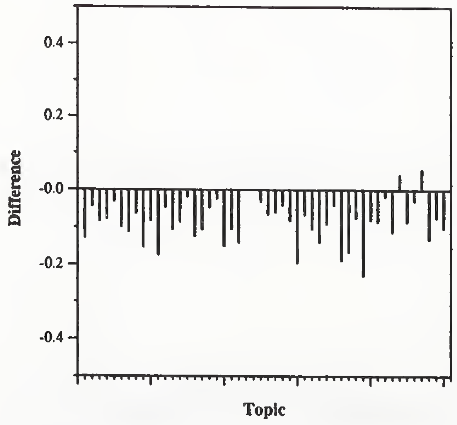
	New
Average precision	0.55
Average recall	0.92
Average F	0.670



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	ccsumt4sqr01
Num topics	

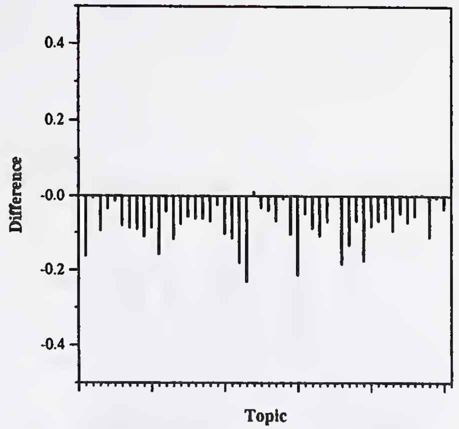
	New
Average precision	0.52
Average recall	0.91
Average F	0.651



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	clr03n4
Num topics	

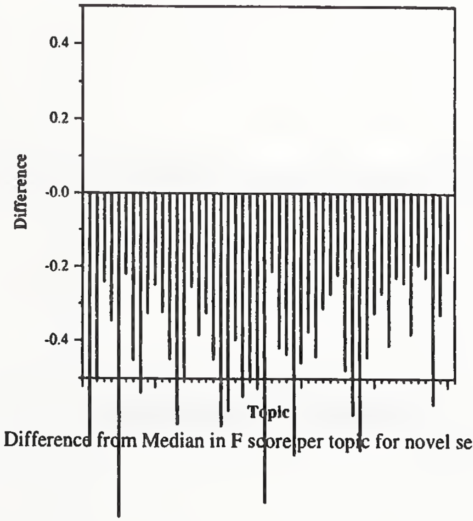
	New
Average precision	0.53
Average recall	0.91
Average F	0.655



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	IITBN1
Num topics	

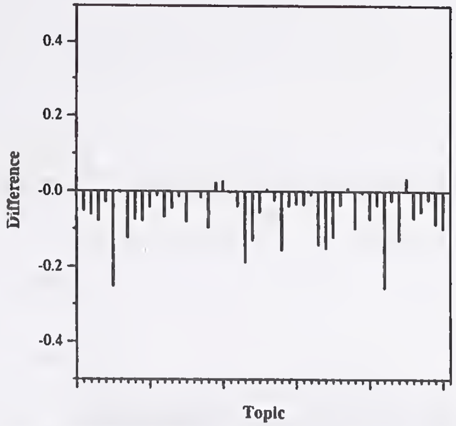
	New
Average precision	0.28
Average recall	0.49
Average F	0.311



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	MeijiHilF41
Num topics	

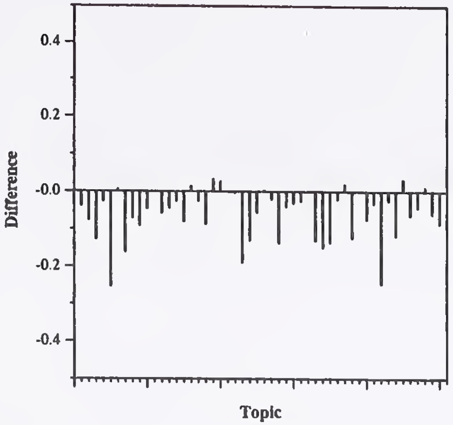
	New
Average precision	0.65
Average recall	0.73
Average F	0.672



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	MeijiHilF42
Num topics	

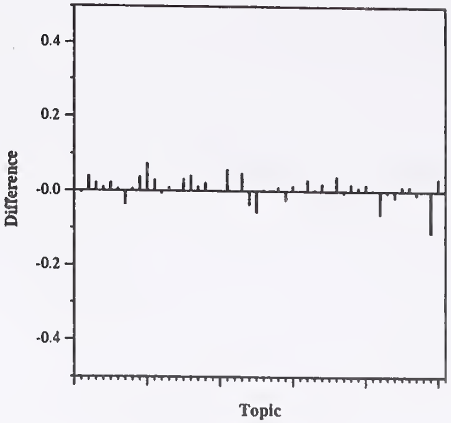
	New
Average precision	0.66
Average recall	0.72
Average F	0.675



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	MeijiHilF43
Num topics	

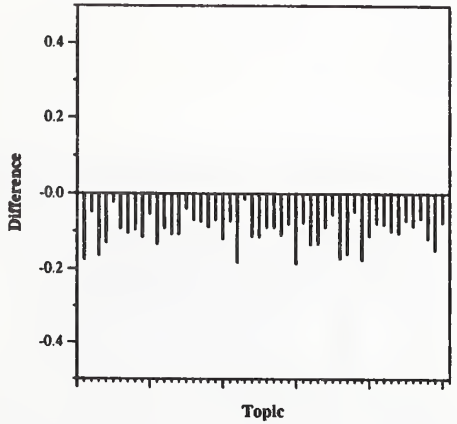
	New
Average precision	0.62
Average recall	0.98
Average F	0.741



Difference from Median in F score per topic for novel sentences

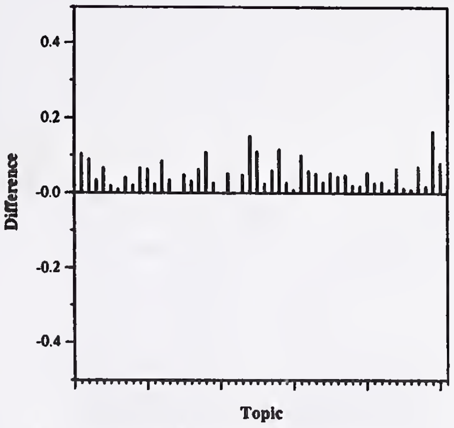
Summary Statistics	
Run ID	MeijiHilF44
Num topics	

	New
Average precision	0.49
Average recall	0.96
Average F	0.634



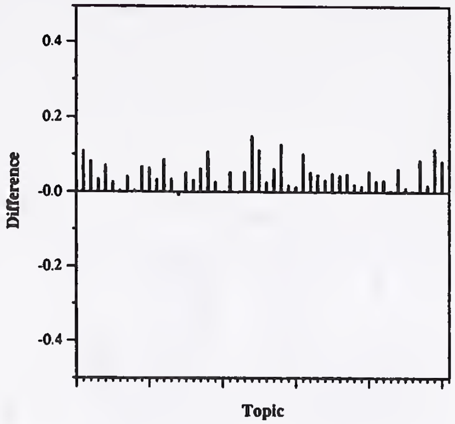
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NTU41
Num topics	
	New
Average precision	0.67
Average recall	0.98
Average F	0.785



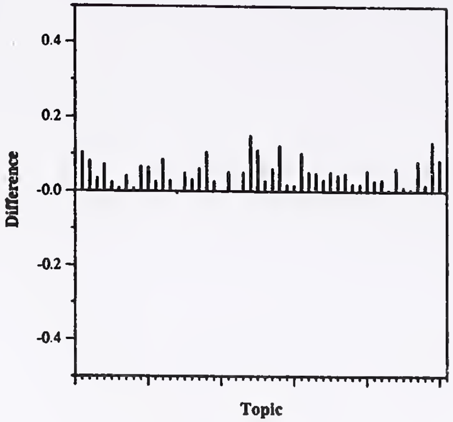
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NTU42
Num topics	
	New
Average precision	0.67
Average recall	0.99
Average F	0.784



Difference from Median in F score per topic for novel sentences

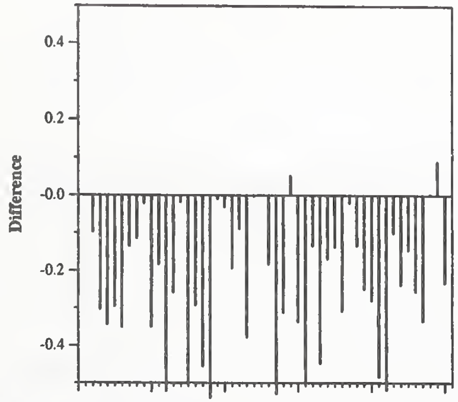
Summary Statistics	
Run ID	NTU43
Num topics	
	New
Average precision	0.67
Average recall	0.99
Average F	0.784



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	NTU44
Num topics	

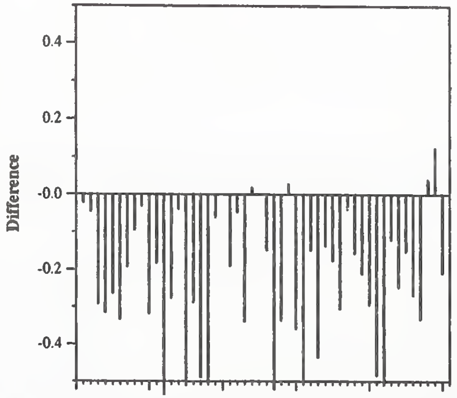
	New
Average precision	0.68
Average recall	0.46
Average F	0.507



Difference from Median in F score per topic for novel sentences

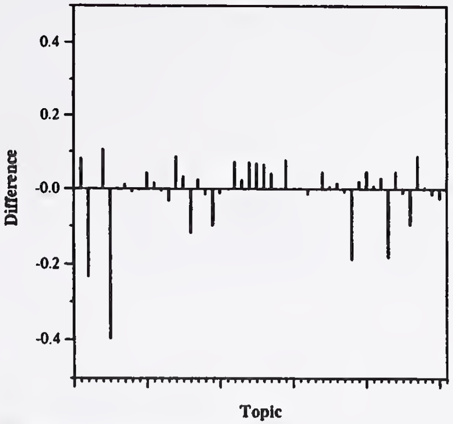
Summary Statistics	
Run ID	NTU45
Num topics	

	New
Average precision	0.68
Average recall	0.47
Average F	0.509



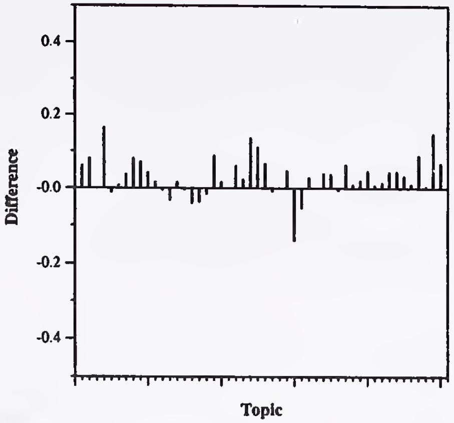
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	THUIRnv0341
Num topics	
	New
Average precision	0.72
Average recall	0.78
Average F	0.728



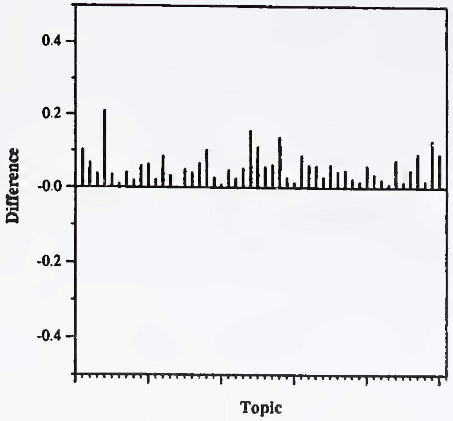
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	THUIRnv0342
Num topics	
	New
Average precision	0.70
Average recall	0.88
Average F	0.765



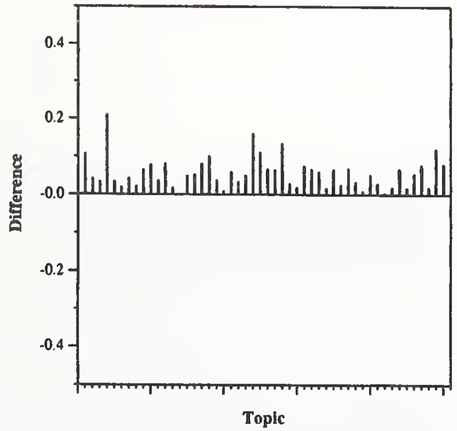
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	THUIRnv0343
Num topics	
	New
Average precision	0.68
Average recall	0.99
Average F	0.791



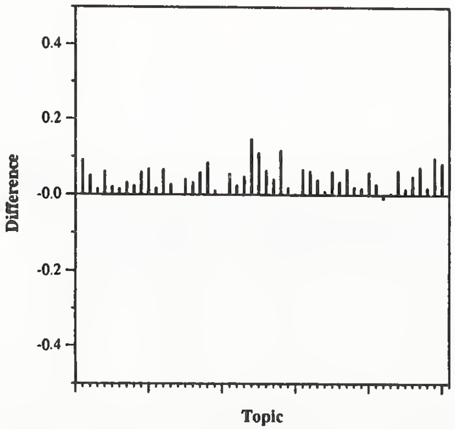
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	THUIRnv0344
Num topics	
	New
Average precision	0.68
Average recall	0.98
Average F	0.790



Difference from Median in F score per topic for novel sentences

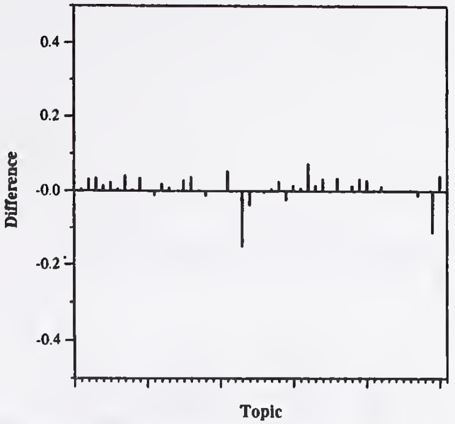
Summary Statistics	
Run ID	THUIRnv0345
Num topics	
	New
Average precision	0.68
Average recall	0.96
Average F	0.780



Difference from Median in F score per topic for novel sentences

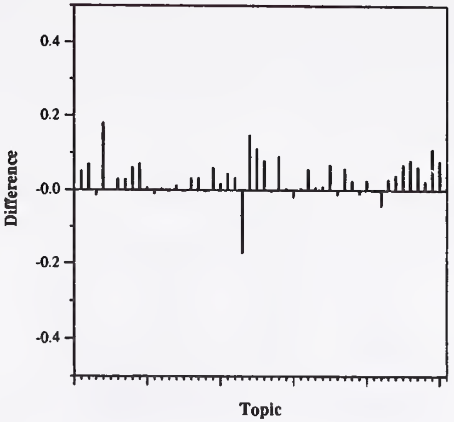
Novelty track results, task 4 — University of Iowa

Summary Statistics	
Run ID	UIowa03Nov10
Num topics	
	New
Average precision	0.62
Average recall	0.98
Average F	0.741



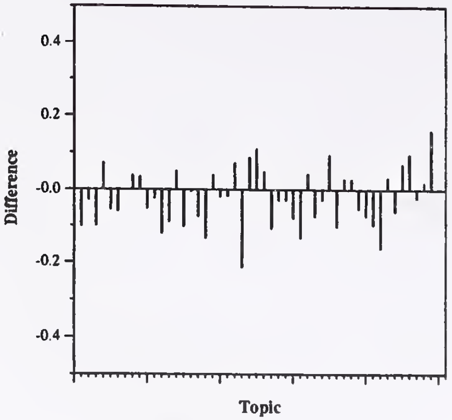
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	UIowa03Nov11
Num topics	
	New
Average precision	0.70
Average recall	0.89
Average F	0.767



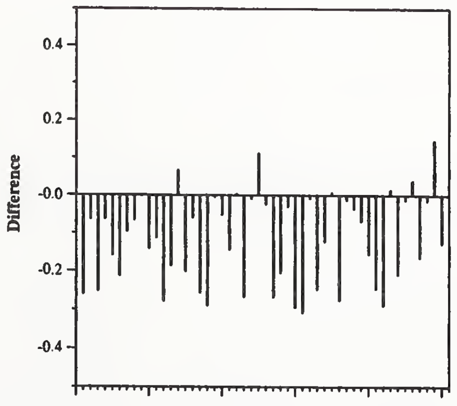
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	UIowa03Nov12
Num topics	
	New
Average precision	0.73
Average recall	0.74
Average F	0.712



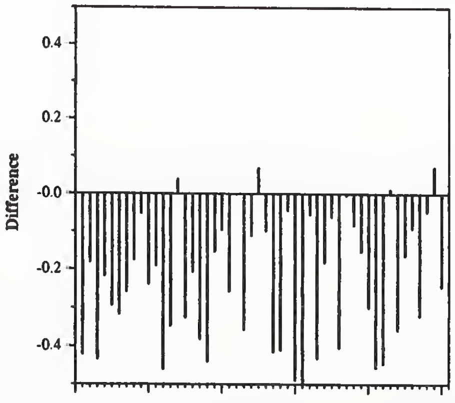
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	UIowa03Nov13
Num topics	
	New
Average precision	0.75
Average recall	0.56
Average F	0.617



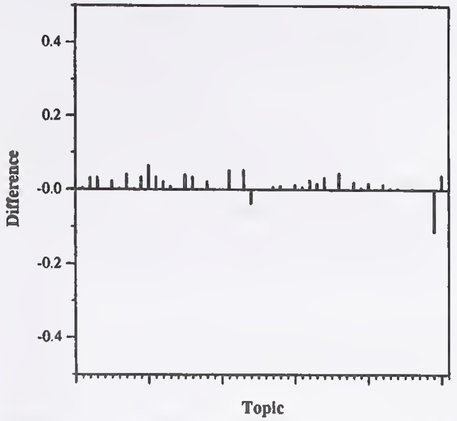
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	UIowa03Nov14
Num topics	
	New
Average precision	0.78
Average recall	0.40
Average F	0.505



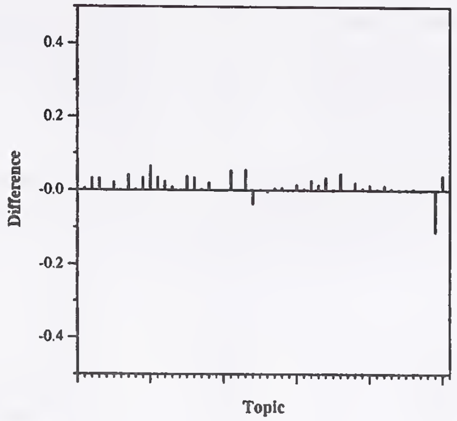
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	umich41
Num topics	
	New
Average precision	0.61
Average recall	1.00
Average F	0.747



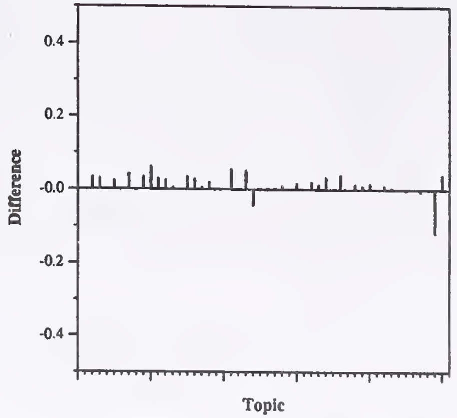
Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	umich42
Num topics	
	New
Average precision	0.61
Average recall	1.00
Average F	0.747



Difference from Median in F score per topic for novel sentences

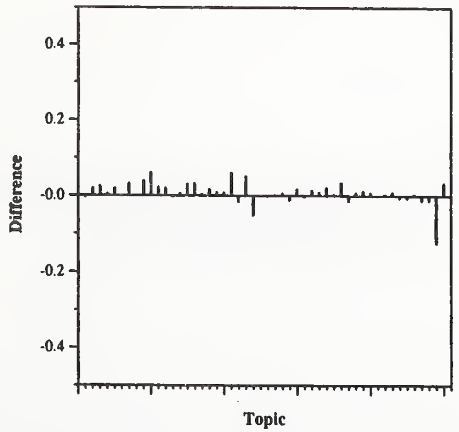
Summary Statistics	
Run ID	umich43
Num topics	
	New
Average precision	0.61
Average recall	0.99
Average F	0.745



Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	umich44
Num topics	

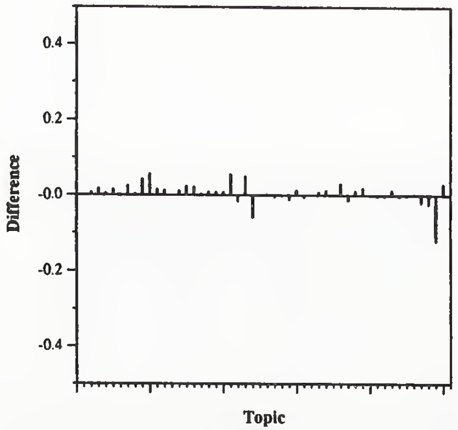
	New
Average precision	0.61
Average recall	0.98
Average F	0.742



Difference from Median in F score per topic for novel sentences

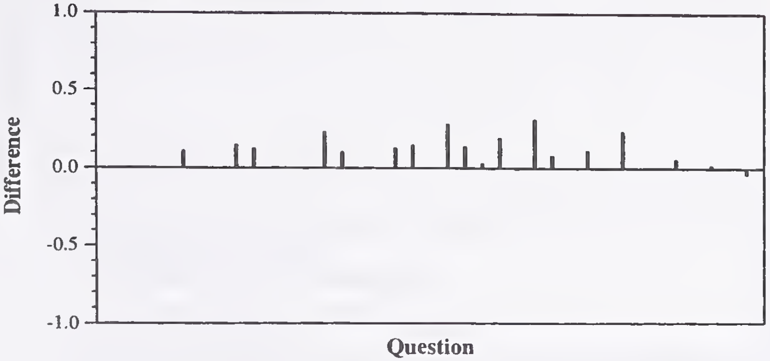
Summary Statistics	
Run ID	umich45
Num topics	

	New
Average precision	0.61
Average recall	0.97
Average F	0.740

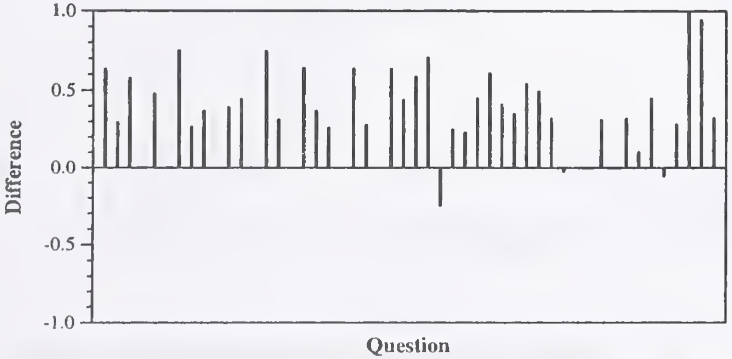


Difference from Median in F score per topic for novel sentences

Summary Statistics	
Run ID	BBN2003A
Num questions	413 factoid + 37 list + 50 def
Number factoid right	73
Number factoid inexact	2
Number factoid unsupported	6
Number factoid wrong	332
Precision of recognizing no answer	3 / 46 = 0.065
Recall of recognizing no answer	3 / 30 = 0.100
Factoid accuracy	0.177
Average F score for lists	0.087
Average F score for definintions	0.521
Final combined score	0.240

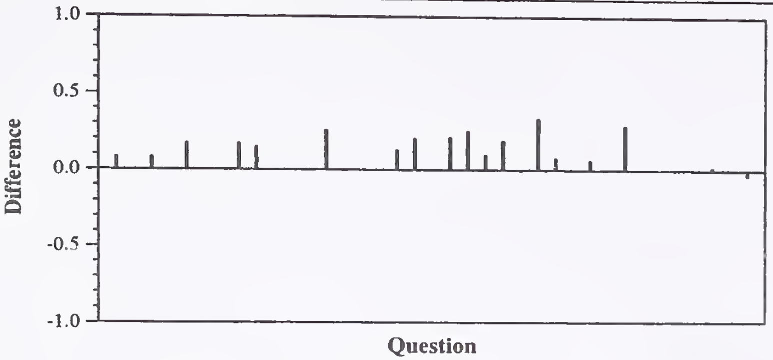


Difference from Median in F score for list questions

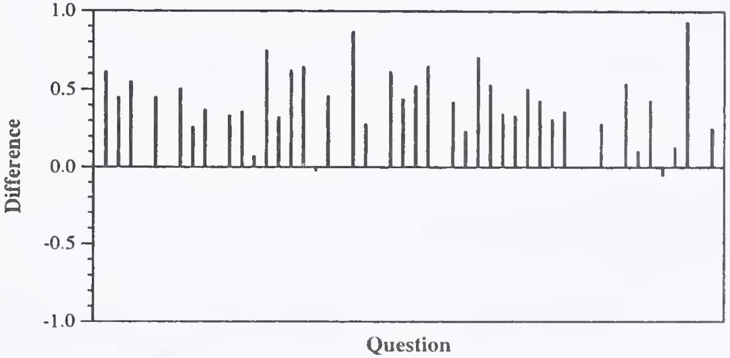


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	BBN2003B
Num questions	413 factoid + 37 list + 50 def
Number factoid right	86
Number factoid inexact	4
Number factoid unsupported	29
Number factoid wrong	294
Precision of recognizing no answer	3 / 44 = 0.068
Recall of recognizing no answer	3 / 30 = 0.100
Factoid accuracy	0.208
Average F score for lists	0.097
Average F score for definintions	0.520
Final combined score	0.258

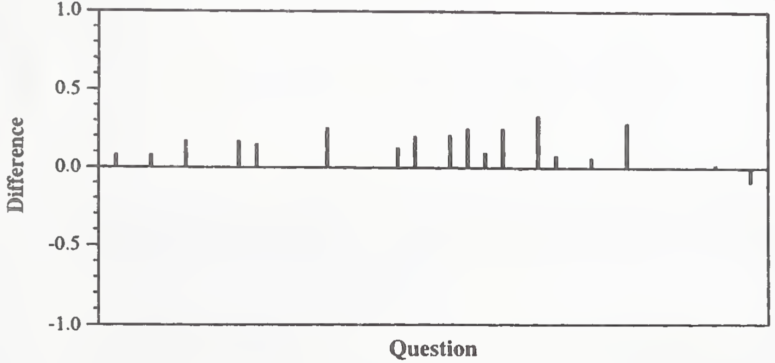


Difference from Median in F score for list questions

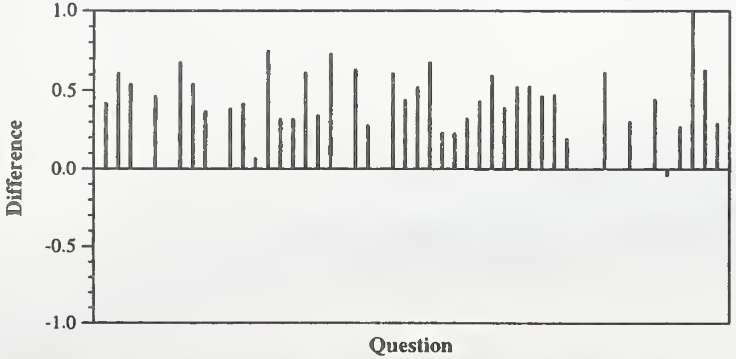


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	BBN2003C
Num questions	413 factoid + 37 list + 50 def
Number factoid right	85
Number factoid inexact	3
Number factoid unsupported	18
Number factoid wrong	307
Precision of recognizing no answer	3 / 46 = 0.065
Recall of recognizing no answer	3 / 30 = 0.100
Factoid accuracy	0.206
Average F score for lists	0.097
Average F score for definintions	0.555
Final combined score	0.266

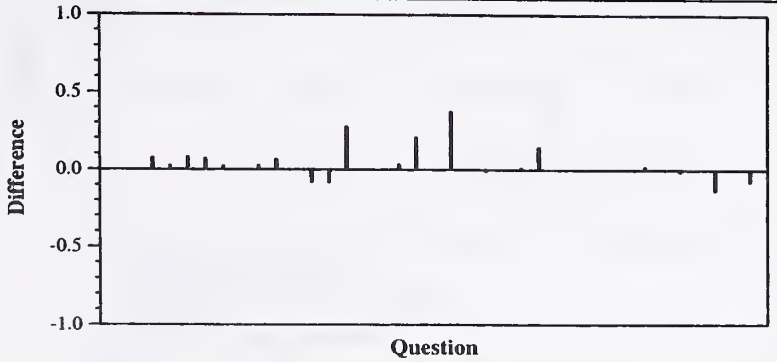


Difference from Median in F score for list questions

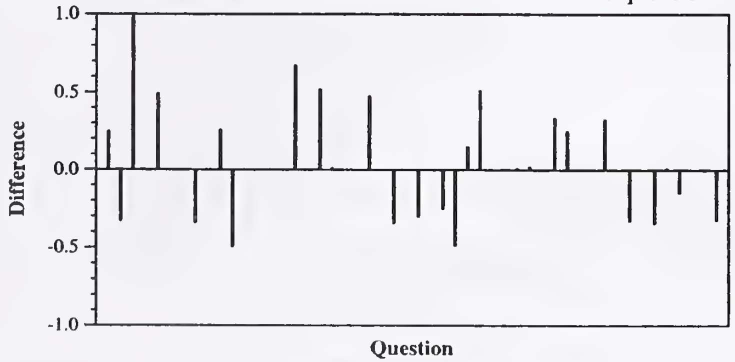


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	CMUJAV2003
Num questions	413 factoid + 37 list + 50 def
Number factoid right	55
Number factoid inexact	3
Number factoid unsupported	3
Number factoid wrong	352
Precision of recognizing no answer	5 / 35 = 0.143
Recall of recognizing no answer	5 / 30 = 0.167
Factoid accuracy	0.133
Average F score for lists	0.052
Average F score for definintions	0.216
Final combined score	0.134

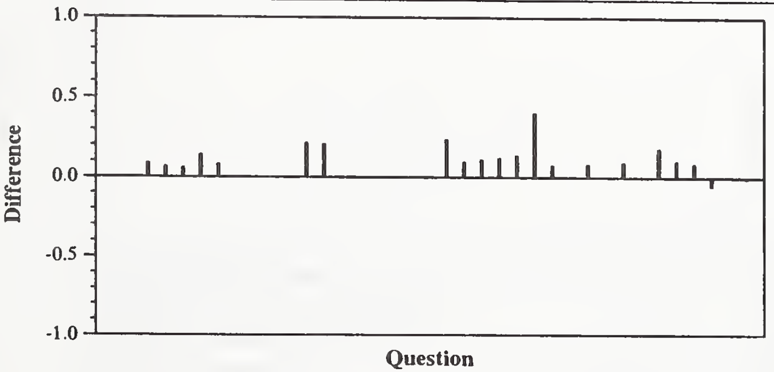


Difference from Median in F score for list questions

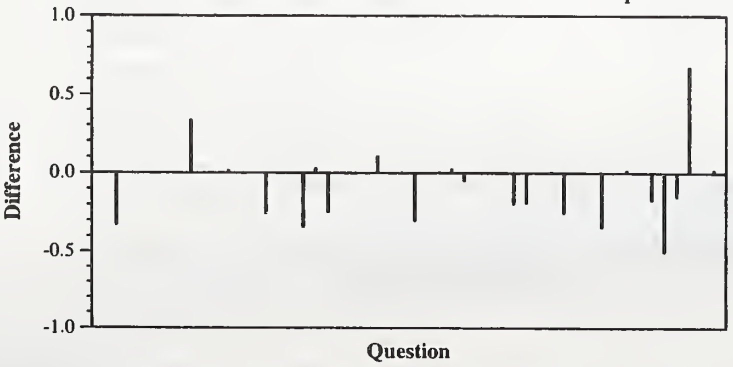


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	ICTQA2003A
Num questions	413 factoid + 37 list + 50 def
Number factoid right	53
Number factoid inexact	6
Number factoid unsupported	6
Number factoid wrong	348
Precision of recognizing no answer	21 /249 = 0.084
Recall of recognizing no answer	21 / 30 = 0.700
Factoid accuracy	0.128
Average F score for lists	0.091
Average F score for definintions	0.142
Final combined score	0.122

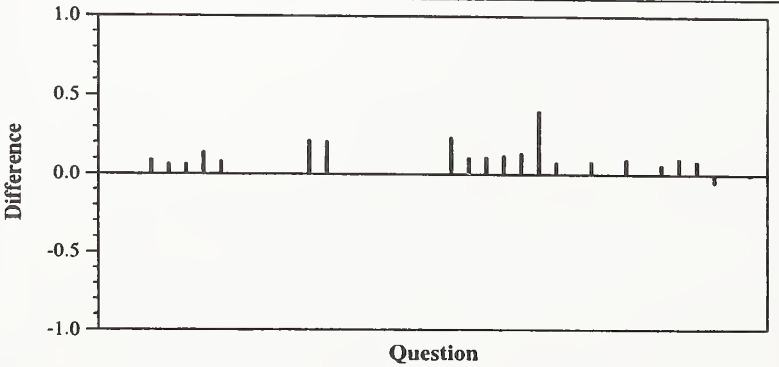


Difference from Median in F score for list questions

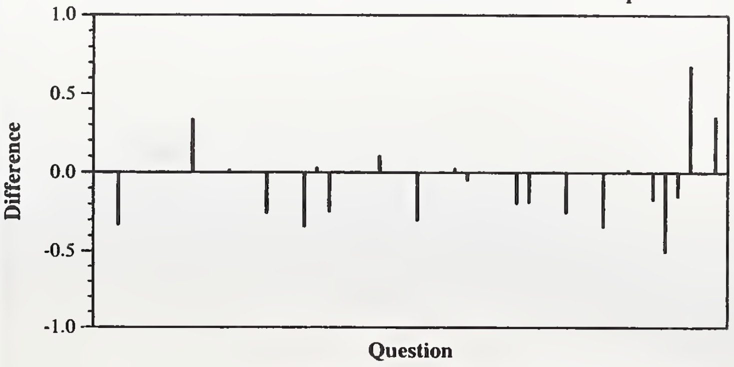


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	ICTQA2003B
Num questions	413 factoid + 37 list + 50 def
Number factoid right	58
Number factoid inexact	5
Number factoid unsupported	8
Number factoid wrong	342
Precision of recognizing no answer	17 /193 = 0.088
Recall of recognizing no answer	17 / 30 = 0.567
Factoid accuracy	0.140
Average F score for lists	0.089
Average F score for definintions	0.149
Final combined score	0.130



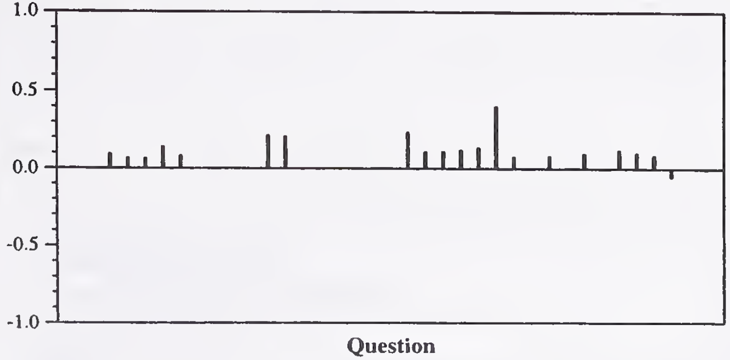
Difference from Median in F score for list questions



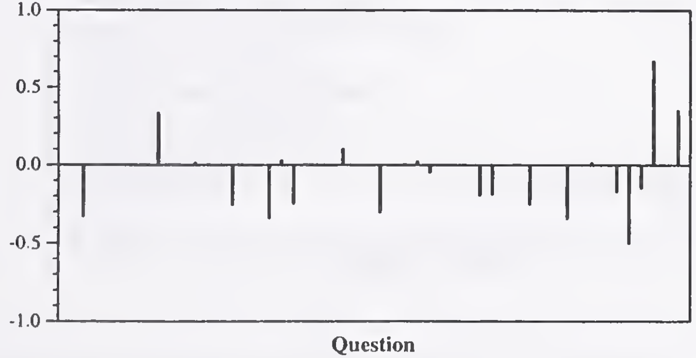
Difference from Median in F score for definition questions

Question Answering track, main task results — Chinese Academy of Sciences (CAS-ICT)

Summary Statistics	
Run ID	ICTQA2003C
Num questions	413 factoid + 37 list + 50 def
Number factoid right	60
Number factoid inexact	8
Number factoid unsupported	10
Number factoid wrong	335
Precision of recognizing no answer	14 /133 = 0.105
Recall of recognizing no answer	14 / 30 = 0.467
Factoid accuracy	0.145
Average F score for lists	0.091
Average F score for definintions	0.149
Final combined score	0.133

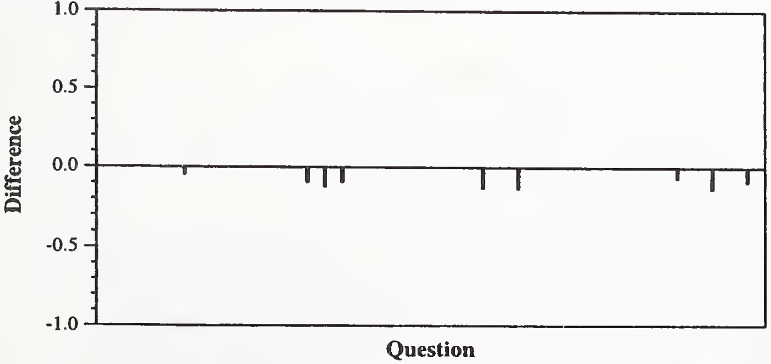


Difference from Median in F score for list questions

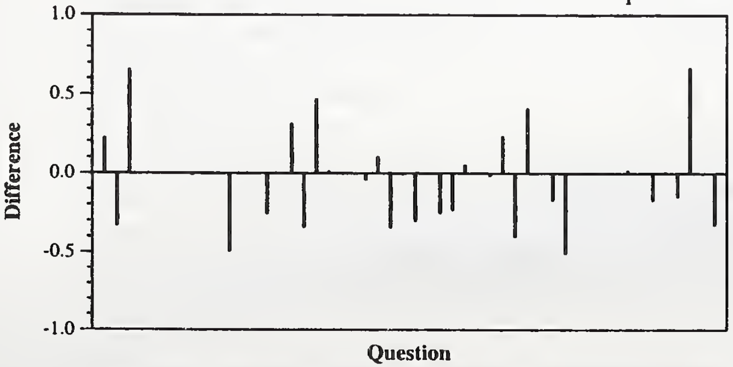


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	clr03m1
Num questions	413 factoid + 37 list + 50 def
Number factoid right	29
Number factoid inexact	12
Number factoid unsupported	3
Number factoid wrong	369
Precision of recognizing no answer	7 / 64 = 0.109
Recall of recognizing no answer	7 / 30 = 0.233
Factoid accuracy	0.070
Average F score for lists	0.000
Average F score for definintions	0.160
Final combined score	0.075

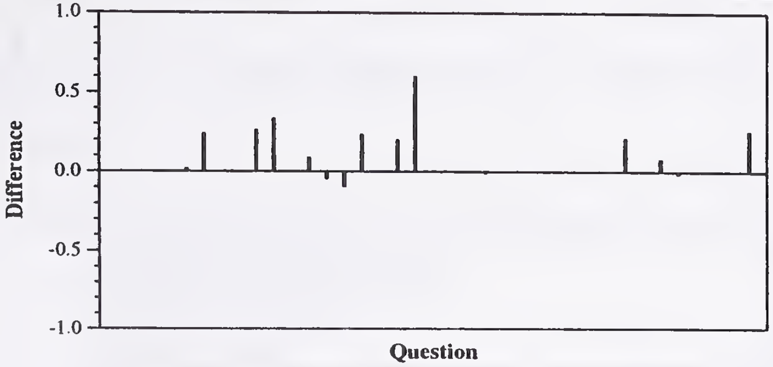


Difference from Median in F score for list questions

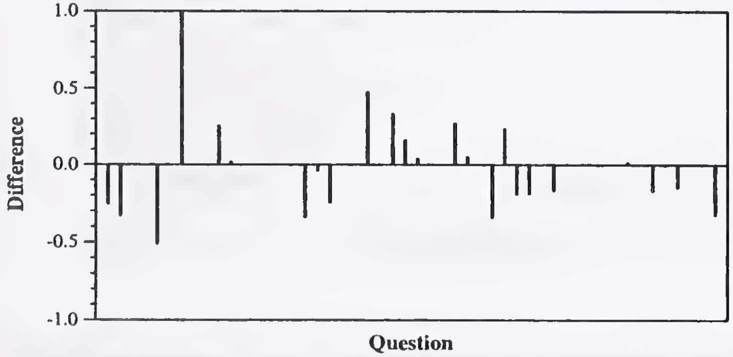


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	FDUT12QA1
Num questions	413 factoid + 37 list + 50 def
Number factoid right	80
Number factoid inexact	6
Number factoid unsupported	28
Number factoid wrong	299
Precision of recognizing no answer	7 / 91 = 0.077
Recall of recognizing no answer	7 / 30 = 0.233
Factoid accuracy	0.194
Average F score for lists	0.088
Average F score for definintions	0.176
Final combined score	0.163

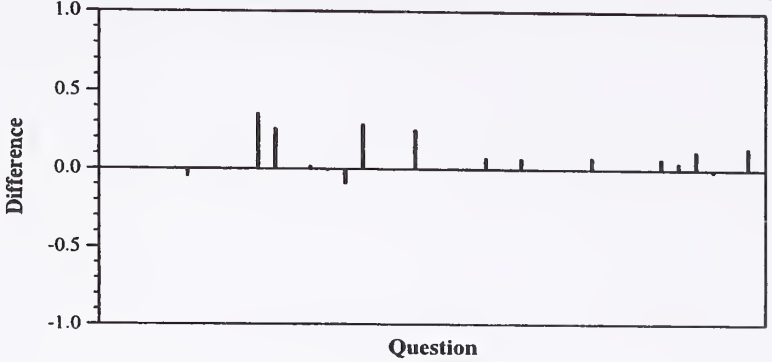


Difference from Median in F score for list questions

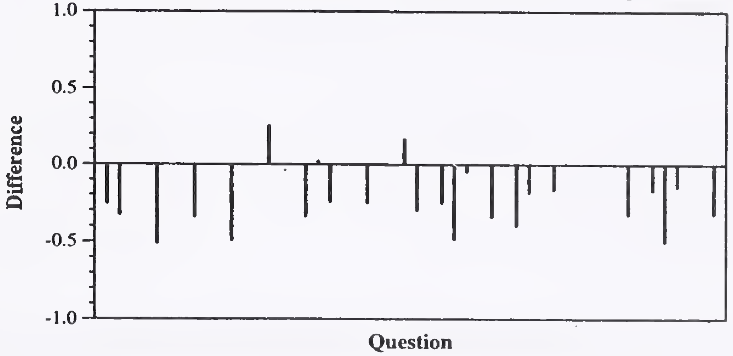


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	FDUT12QA2
Num questions	413 factoid + 37 list + 50 def
Number factoid right	74
Number factoid inexact	9
Number factoid unsupported	27
Number factoid wrong	303
Precision of recognizing no answer	13 /135 = 0.096
Recall of recognizing no answer	13 / 30 = 0.433
Factoid accuracy	0.179
Average F score for lists	0.067
Average F score for definintions	0.065
Final combined score	0.122

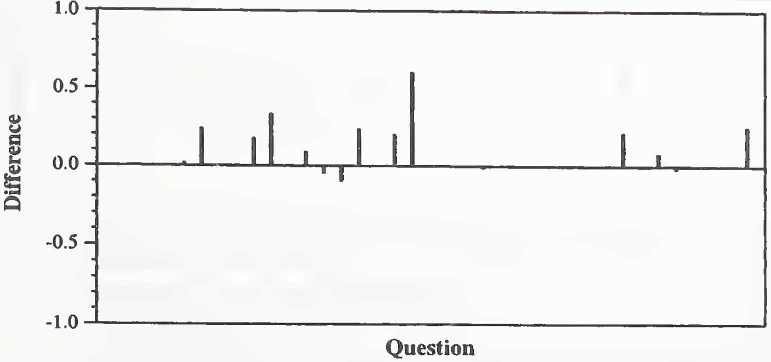


Difference from Median in F score for list questions

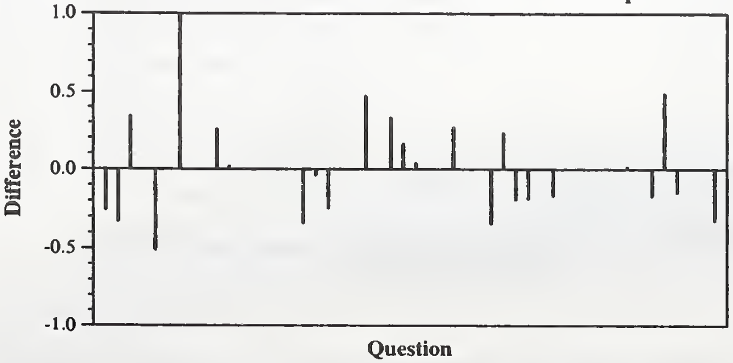


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	FDUT12QA3
Num questions	413 factoid + 37 list + 50 def
Number factoid right	79
Number factoid inexact	6
Number factoid unsupported	27
Number factoid wrong	301
Precision of recognizing no answer	7 / 91 = 0.077
Recall of recognizing no answer	7 / 30 = 0.233
Factoid accuracy	0.191
Average F score for lists	0.086
Average F score for definintions	0.192
Final combined score	0.165

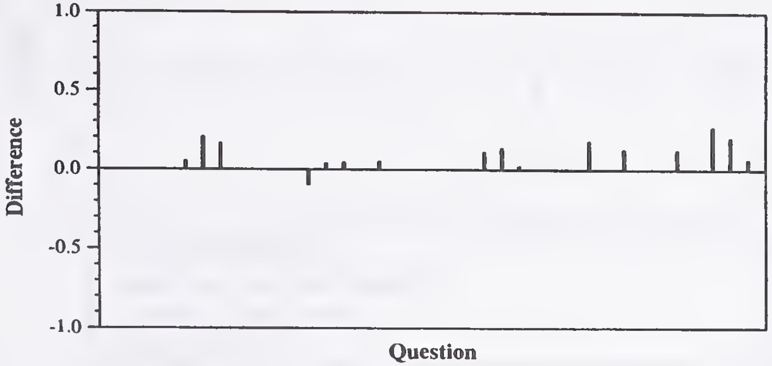


Difference from Median in F score for list questions

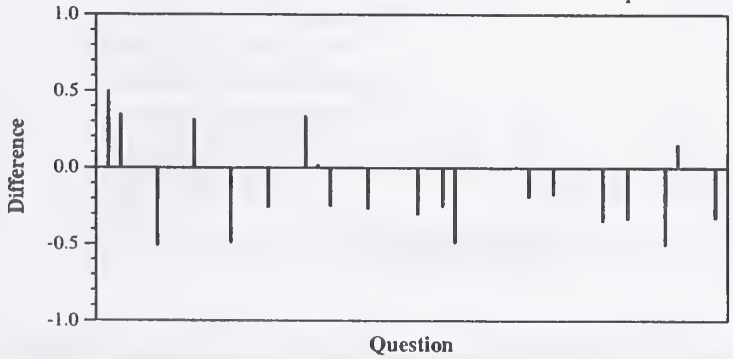


Difference from Median in F score for definition questįons

Summary Statistics	
Run ID	IBM2003a
Num questions	413 factoid + 37 list + 50 def
Number factoid right	123
Number factoid inexact	8
Number factoid unsupported	11
Number factoid wrong	271
Precision of recognizing no answer	7 / 85 = 0.082
Recall of recognizing no answer	7 / 30 = 0.233
Factoid accuracy	0.298
Average F score for lists	0.070
Average F score for definintions	0.124
Final combined score	0.197

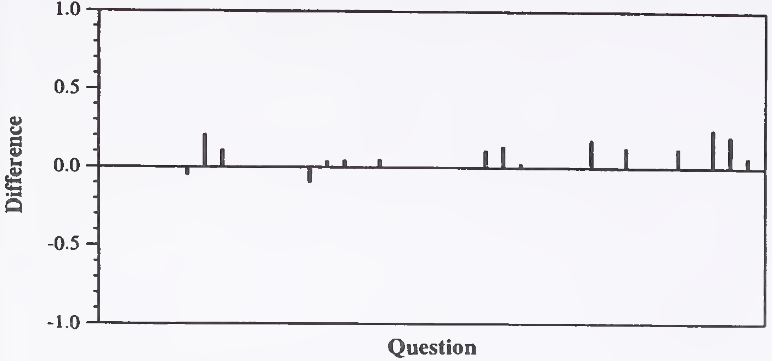


Difference from Median in F score for list questions

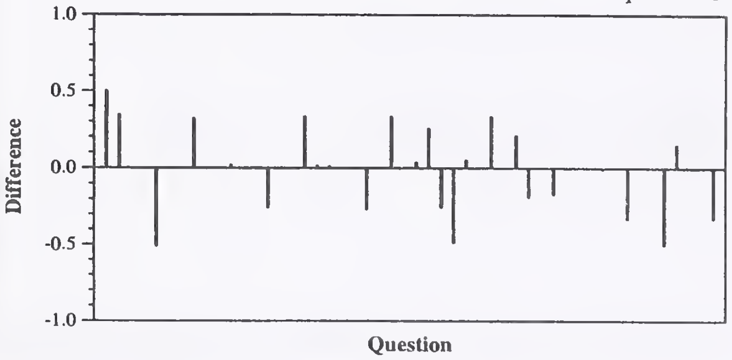


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	IBM2003b
Num questions	413 factoid + 37 list + 50 def
Number factoid right	123
Number factoid inexact	8
Number factoid unsupported	11
Number factoid wrong	271
Precision of recognizing no answer	7 / 85 = 0.082
Recall of recognizing no answer	7 / 30 = 0.233
Factoid accuracy	0.298
Average F score for lists	0.065
Average F score for definintions	0.177
Final combined score	0.210

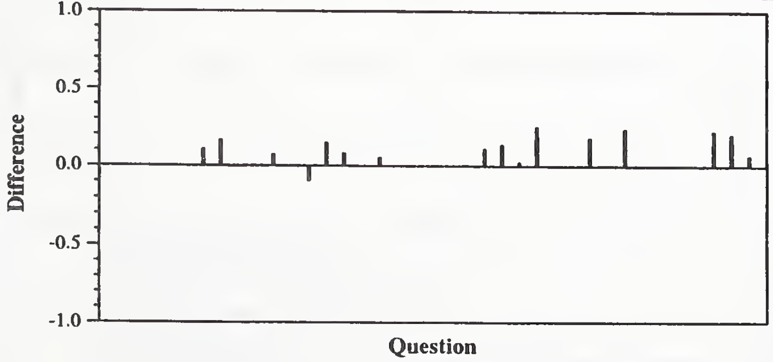


Difference from Median in F score for list questions

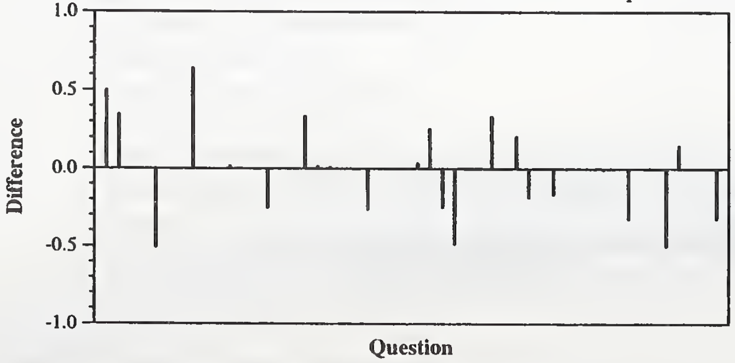


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	IBM2003c
Num questions	413 factoid + 37 list + 50 def
Number factoid right	123
Number factoid inexact	8
Number factoid unsupported	10
Number factoid wrong	272
Precision of recognizing no answer	7 / 85 = 0.082
Recall of recognizing no answer	7 / 30 = 0.233
Factoid accuracy	0.298
Average F score for lists	0.077
Average F score for definintions	0.175
Final combined score	0.212

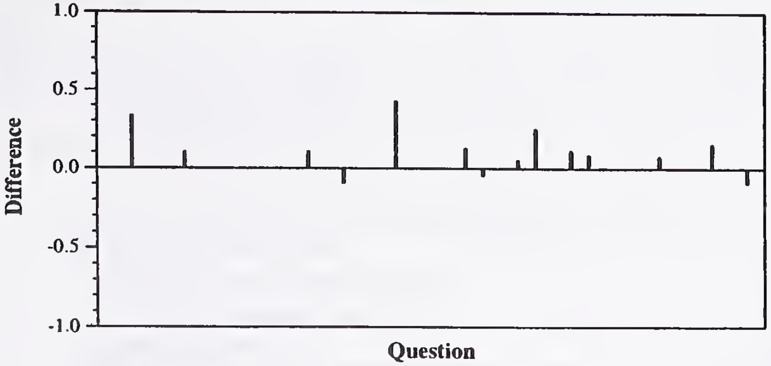


Difference from Median in F score for list questions

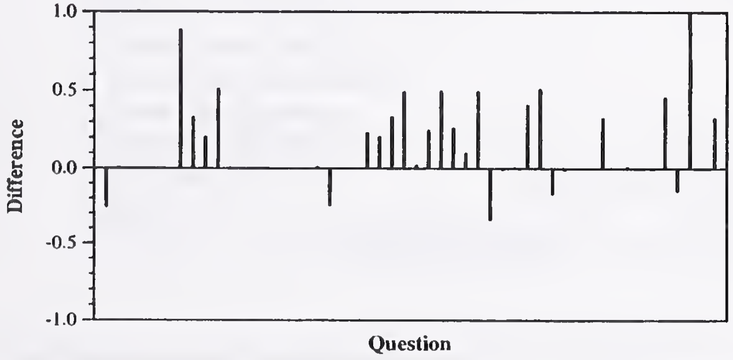


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	irstqa2003d
Num questions	413 factoid + 37 list + 50 def
Number factoid right	62
Number factoid inexact	4
Number factoid unsupported	4
Number factoid wrong	343
Precision of recognizing no answer	2 / 18 = 0.111
Recall of recognizing no answer	2 / 30 = 0.067
Factoid accuracy	0.150
Average F score for lists	0.067
Average F score for definintions	0.318
Final combined score	0.171

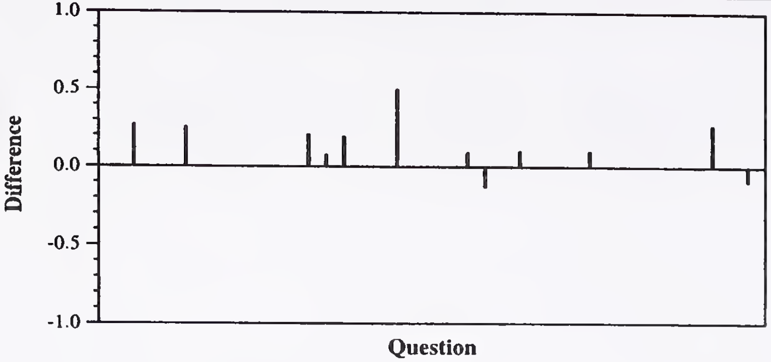


Difference from Median in F score for list questions

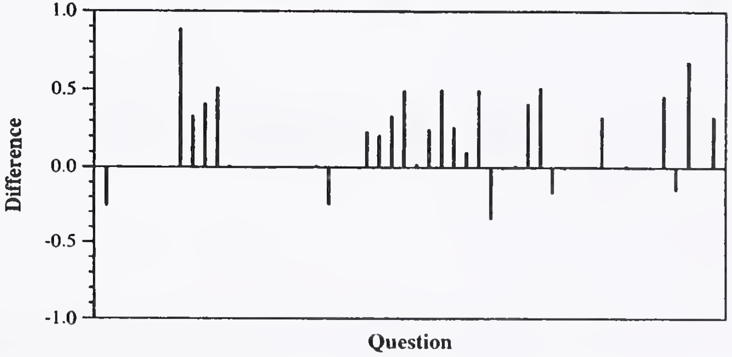


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	irstqa2003p
Num questions	413 factoid + 37 list + 50 def
Number factoid right	92
Number factoid inexact	5
Number factoid unsupported	11
Number factoid wrong	305
Precision of recognizing no answer	5 / 38 = 0.132
Recall of recognizing no answer	5 / 30 = 0.167
Factoid accuracy	0.223
Average F score for lists	0.074
Average F score for definintions	0.315
Final combined score	0.209

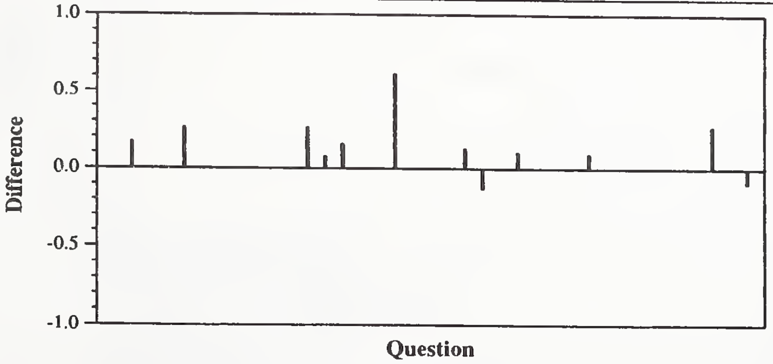


Difference from Median in F score for list questions

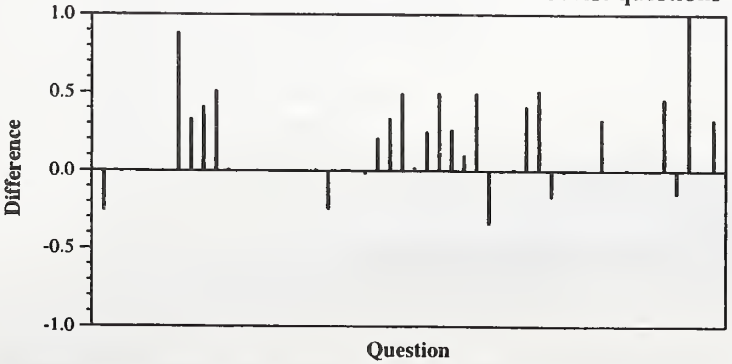


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	irstqa2003w
Num questions	413 factoid + 37 list + 50 def
Number factoid right	97
Number factoid inexact	6
Number factoid unsupported	10
Number factoid wrong	300
Precision of recognizing no answer	8 / 66 = 0.121
Recall of recognizing no answer	8 / 30 = 0.267
Factoid accuracy	0.235
Average F score for lists	0.076
Average F score for definintions	0.317
Final combined score	0.216

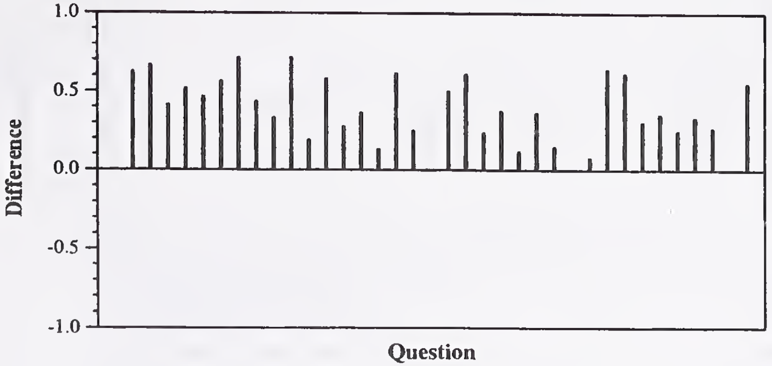


Difference from Median in F score for list questions

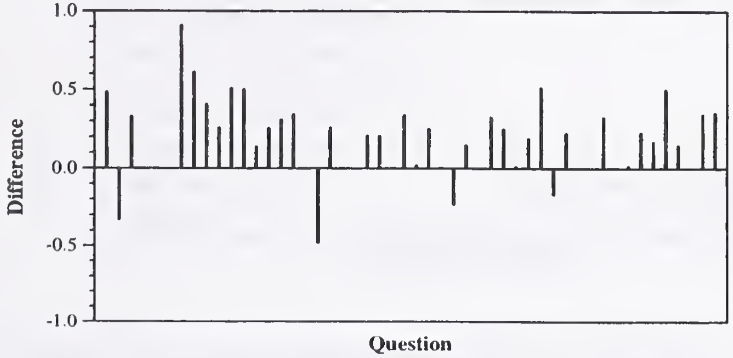


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	LCCmainE03
Num questions	413 factoid + 37 list + 50 def
Number factoid right	289
Number factoid inexact	7
Number factoid unsupported	12
Number factoid wrong	105
Precision of recognizing no answer	24 / 63 = 0.381
Recall of recognizing no answer	24 / 30 = 0.800
Factoid accuracy	0.700
Average F score for lists	0.392
Average F score for definintions	0.361
Final combined score	0.538

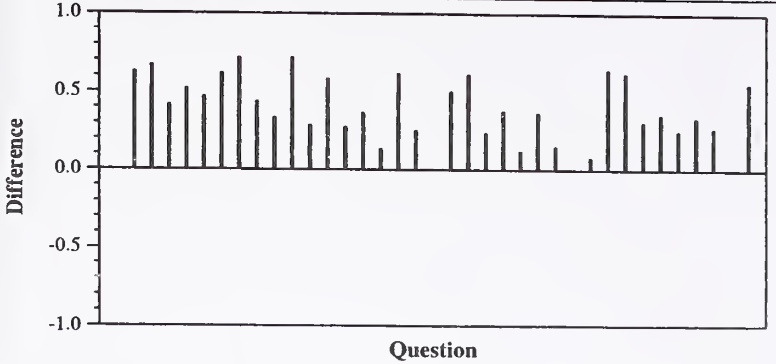


Difference from Median in F score for list questions

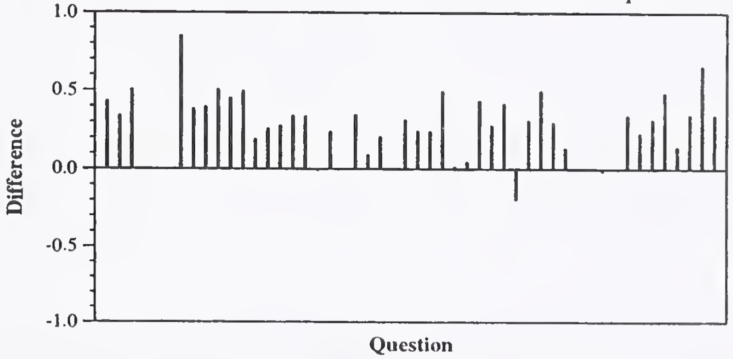


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	LCCmainS03
Num questions	413 factoid + 37 list + 50 def
Number factoid right	289
Number factoid inexact	7
Number factoid unsupported	12
Number factoid wrong	105
Precision of recognizing no answer	24 / 63 = 0.381
Recall of recognizing no answer	24 / 30 = 0.800
Factoid accuracy	0.700
Average F score for lists	0.396
Average F score for definintions	0.442
Final combined score	0.559

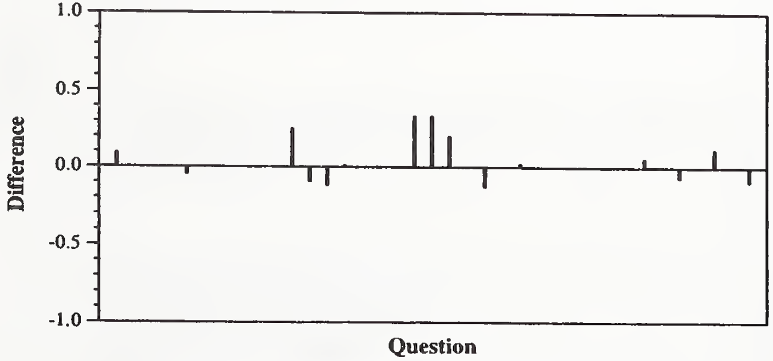


Difference from Median in F score for list questions

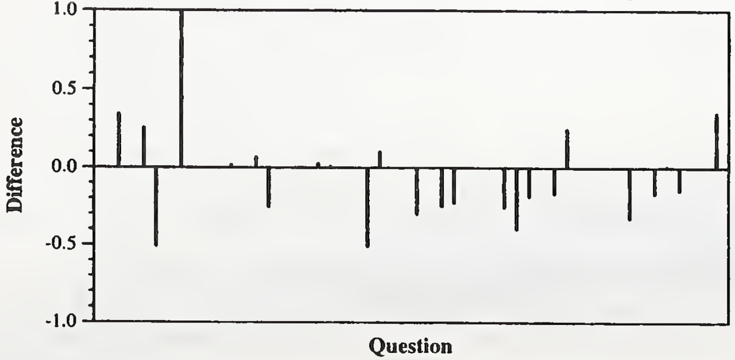


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	lexiclone92
Num questions	413 factoid + 37 list + 50 def
Number factoid right	257
Number factoid inexact	15
Number factoid unsupported	12
Number factoid wrong	129
NIL response never returned	
Factoid accuracy	0.622
Average F score for lists	0.048
Average F score for definintions	0.159
Final combined score	0.363

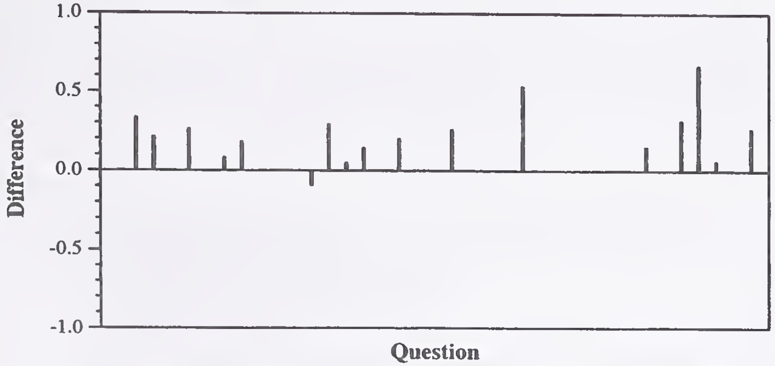


Difference from Median in F score for list questions

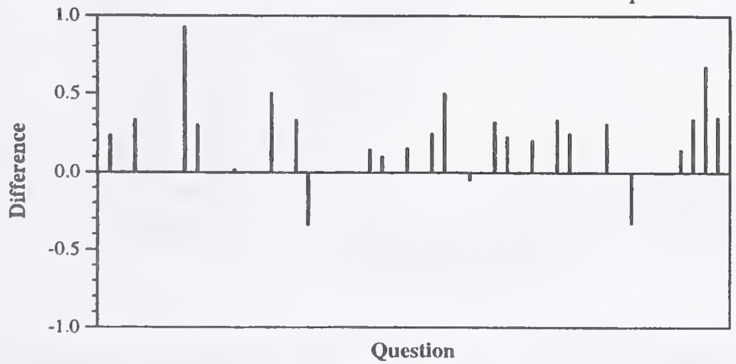


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	MITCSAIL03a
Num questions	413 factoid + 37 list + 50 def
Number factoid right	121
Number factoid inexact	18
Number factoid unsupported	26
Number factoid wrong	248
Precision of recognizing no answer	8 / 35 = 0.229
Recall of recognizing no answer	8 / 30 = 0.267
Factoid accuracy	0.293
Average F score for lists	0.130
Average F score for definintions	0.309
Final combined score	0.256

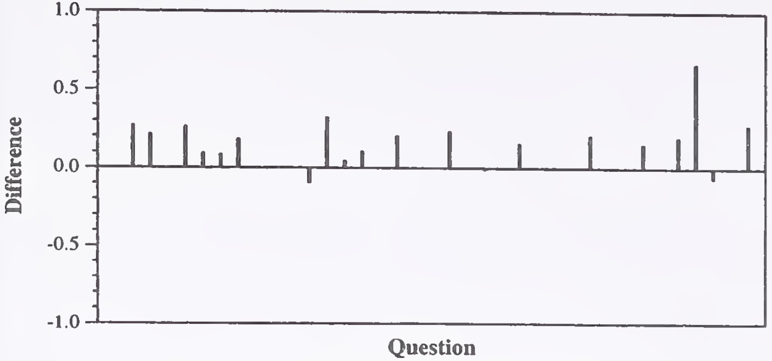


Difference from Median in F score for list questions

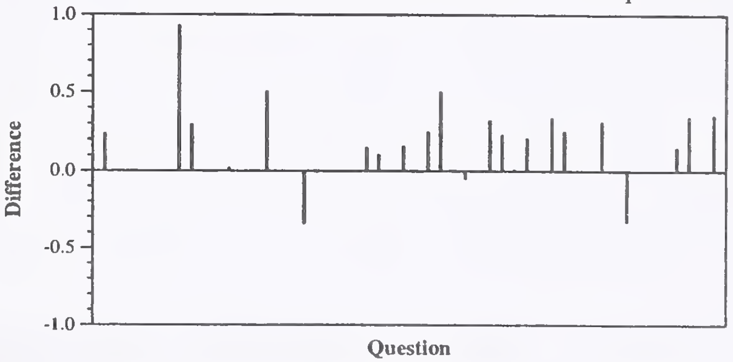


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	MITCSAIL03b
Num questions	413 factoid + 37 list + 50 def
Number factoid right	122
Number factoid inexact	15
Number factoid unsupported	21
Number factoid wrong	255
Precision of recognizing no answer	8 / 31 = 0.258
Recall of recognizing no answer	8 / 30 = 0.267
Factoid accuracy	0.295
Average F score for lists	0.118
Average F score for definintions	0.282
Final combined score	0.247

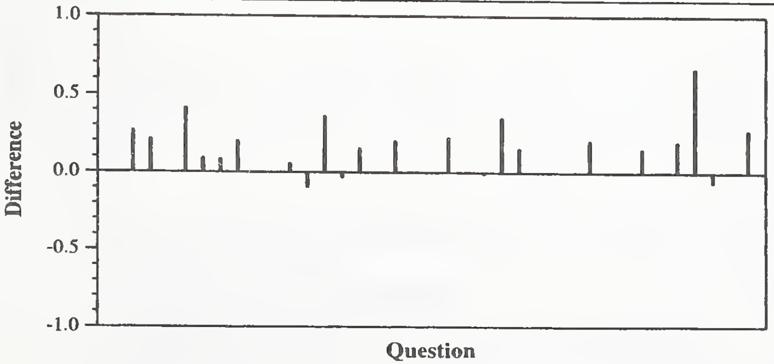


Difference from Median in F score for list questions

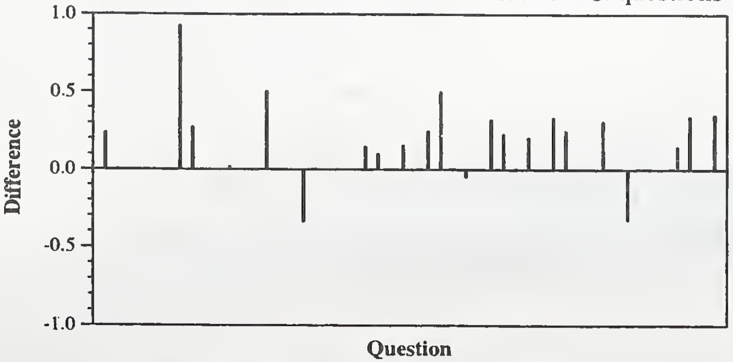


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	MITCSAIL03c
Num questions	413 factoid + 37 list + 50 def
Number factoid right	120
Number factoid inexact	15
Number factoid unsupported	21
Number factoid wrong	257
Precision of recognizing no answer	9 / 46 = 0.196
Recall of recognizing no answer	9 / 30 = 0.300
Factoid accuracy	0.291
Average F score for lists	0.134
Average F score for definintions	0.282
Final combined score	0.249

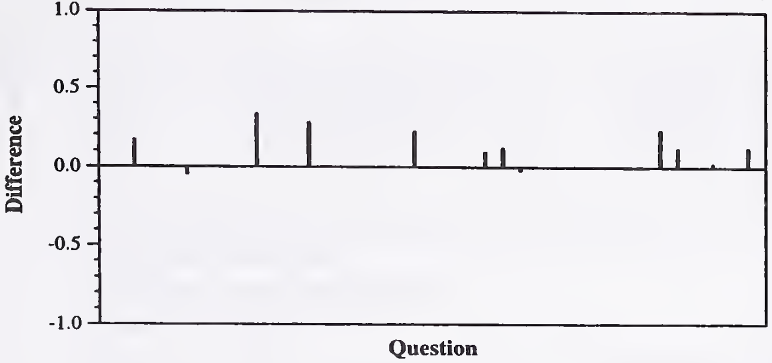


Difference from Median in F score for list questions

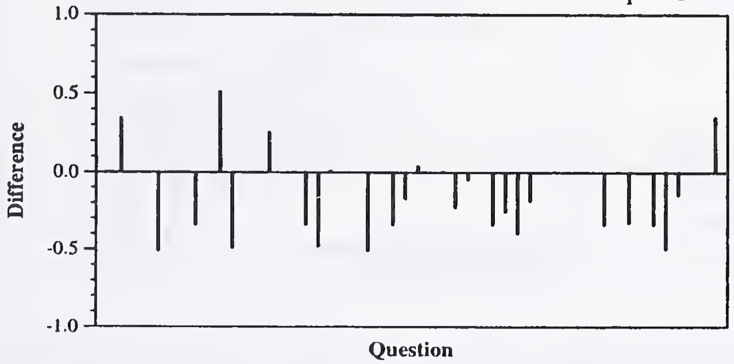


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	MITRE2003A
Num questions	413 factoid + 37 list + 50 def
Number factoid right	61
Number factoid inexact	7
Number factoid unsupported	10
Number factoid wrong	335
Precision of recognizing no answer	0 / 9 = 0.000
Recall of recognizing no answer	0 / 30 = 0.000
Factoid accuracy	0.148
Average F score for lists	0.069
Average F score for definintions	0.088
Final combined score	0.113

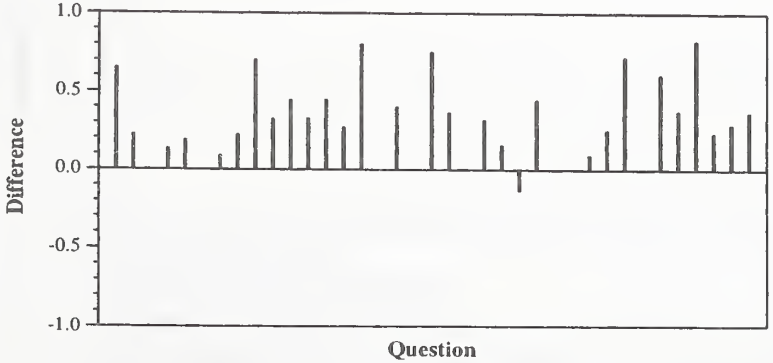


Difference from Median in F score for list questions

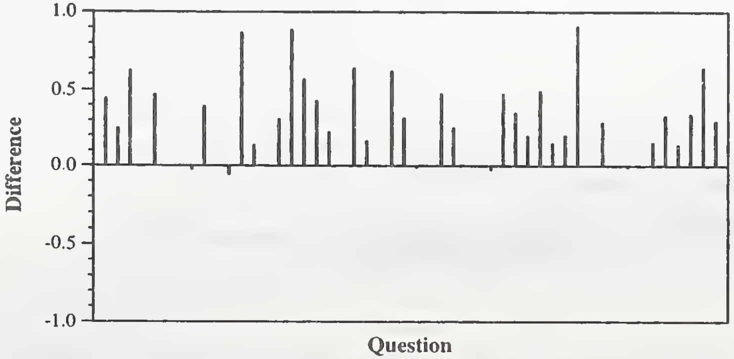


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	nusmml03r1
Num questions	413 factoid + 37 list + 50 def
Number factoid right	232
Number factoid inexact	13
Number factoid unsupported	24
Number factoid wrong	144
Precision of recognizing no answer	12 / 75 = 0.160
Recall of recognizing no answer	12 / 30 = 0.400
Factoid accuracy	0.562
Average F score for lists	0.318
Average F score for definintions	0.441
Final combined score	0.471

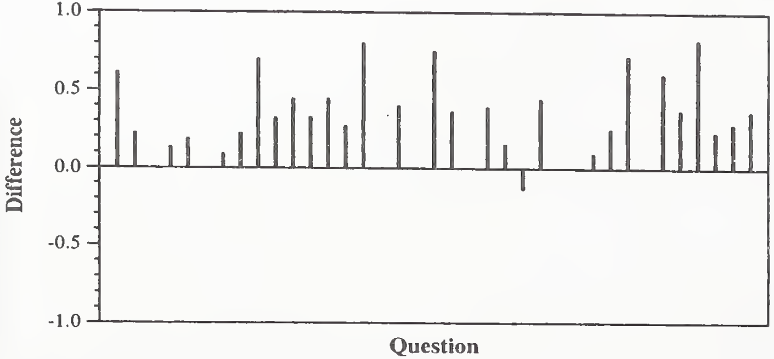


Difference from Median in F score for list questions

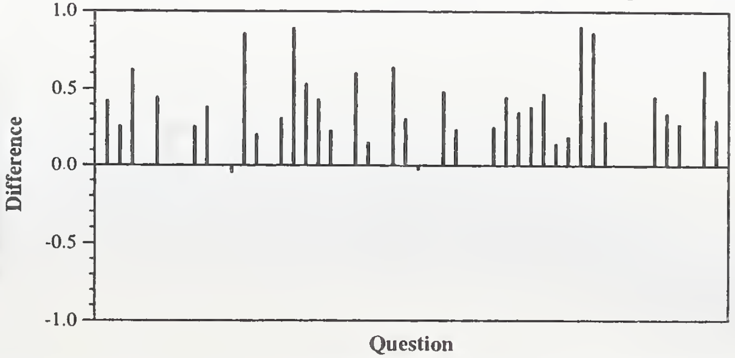


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	nusmml03r2
Num questions	413 factoid + 37 list + 50 def
Number factoid right	232
Number factoid inexact	13
Number factoid unsupported	24
Number factoid wrong	144
Precision of recognizing no answer	12 / 75 = 0.160
Recall of recognizing no answer	12 / 30 = 0.400
Factoid accuracy	0.562
Average F score for lists	0.319
Average F score for definintions	0.473
Final combined score	0.479

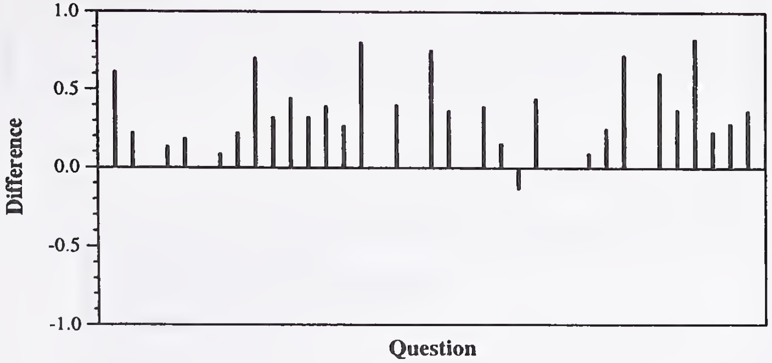


Difference from Median in F score for list questions

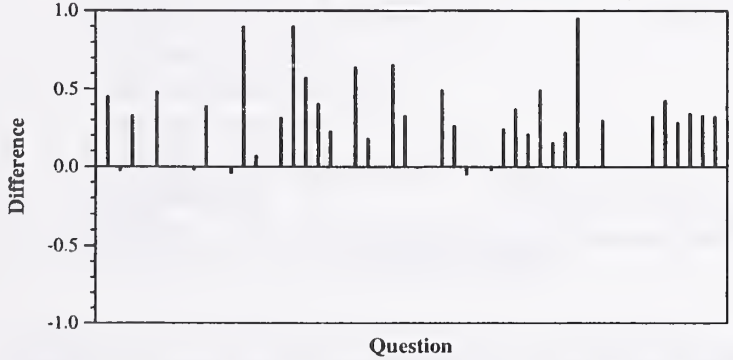


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	nusmml03r3
Num questions	413 factoid + 37 list + 50 def
Number factoid right	225
Number factoid inexact	12
Number factoid unsupported	20
Number factoid wrong	156
Precision of recognizing no answer	23 /146 = 0.158
Recall of recognizing no answer	23 / 30 = 0.767
Factoid accuracy	0.545
Average F score for lists	0.317
Average F score for definintions	0.432
Final combined score	0.460

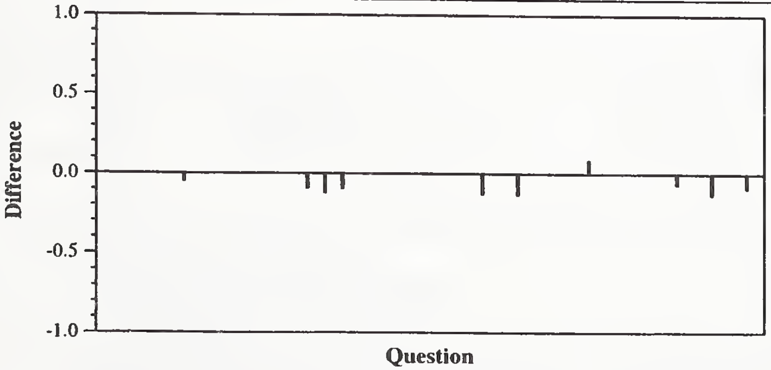


Difference from Median in F score for list questions

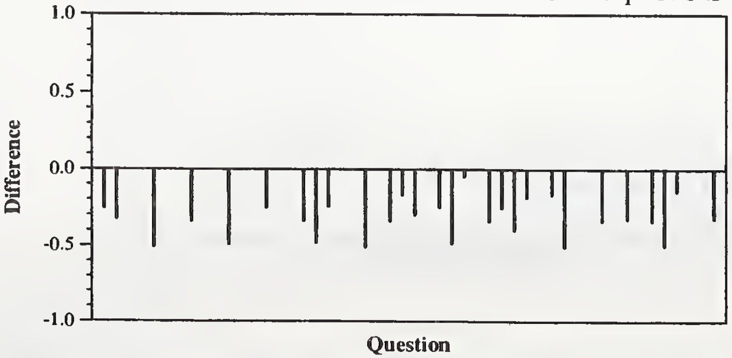


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	CRL2003
Num questions	413 factoid + 37 list + 50 def
Number factoid right	31
Number factoid inexact	0
Number factoid unsupported	0
Number factoid wrong	382
Precision of recognizing no answer	25 /317 = 0.079
Recall of recognizing no answer	25 / 30 = 0.833
Factoid accuracy	0.075
Average F score for lists	0.002
Average F score for definintions	0.000
Final combined score	0.038

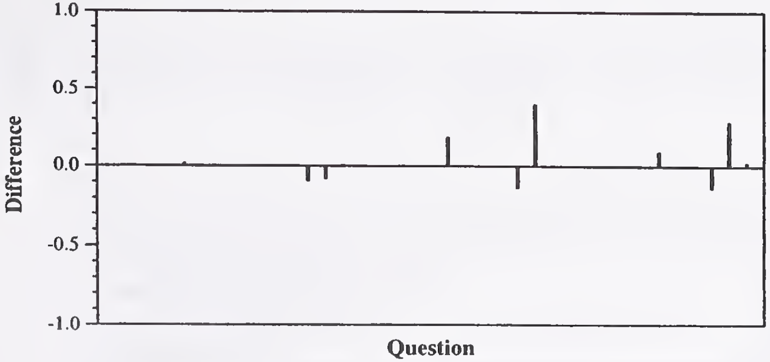


Difference from Median in F score for list questions

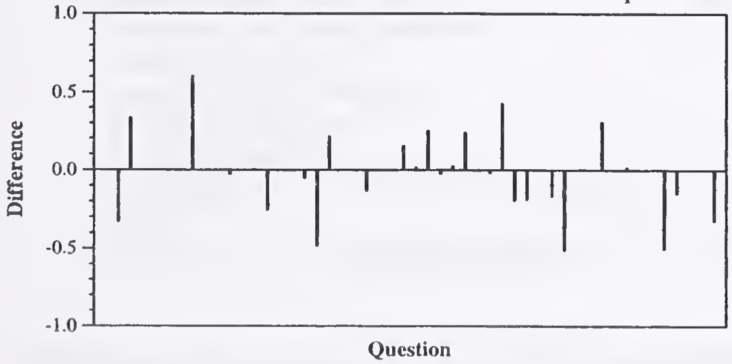


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	ntt2003qam1
Num questions	413 factoid + 37 list + 50 def
Number factoid right	62
Number factoid inexact	8
Number factoid unsupported	4
Number factoid wrong	339
Precision of recognizing no answer	8 / 89 = 0.090
Recall of recognizing no answer	8 / 30 = 0.267
Factoid accuracy	0.150
Average F score for lists	0.040
Average F score for definintions	0.169
Final combined score	0.127

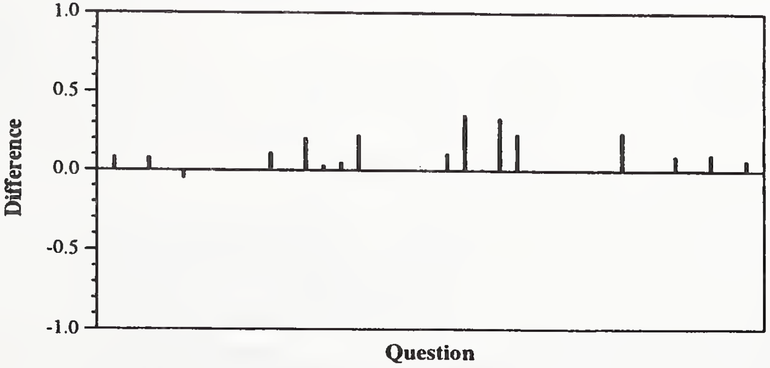


Difference from Median in F score for list questions

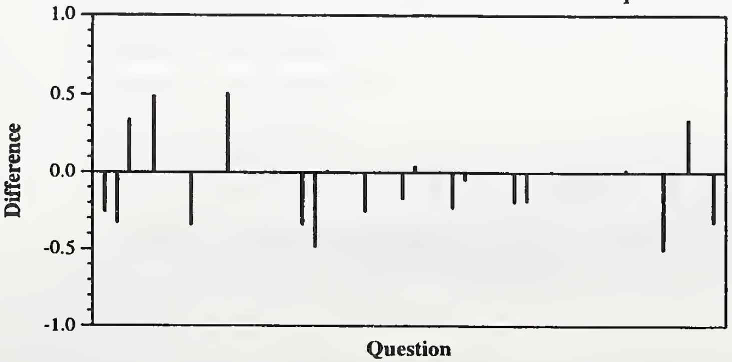


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	Albany03I2
Num questions	413 factoid + 37 list + 50 def
Number factoid right	99
Number factoid inexact	8
Number factoid unsupported	3
Number factoid wrong	303
Precision of recognizing no answer	6 / 55 = 0.109
Recall of recognizing no answer	6 / 30 = 0.200
Factoid accuracy	0.240
Average F score for lists	0.085
Average F score for definintions	0.146
Final combined score	0.178

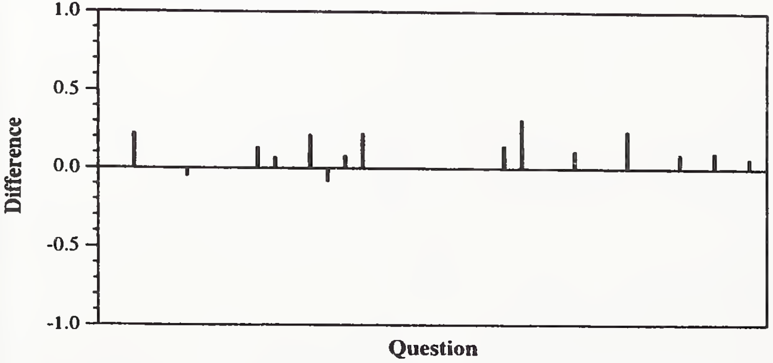


Difference from Median in F score for list questions

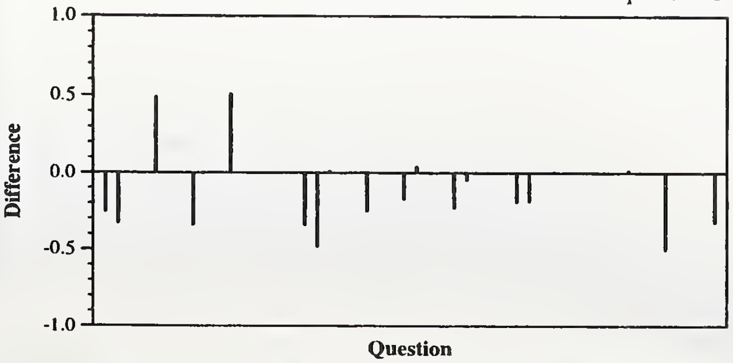


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	Albany03I3
Num questions	413 factoid + 37 list + 50 def
Number factoid right	85
Number factoid inexact	5
Number factoid unsupported	5
Number factoid wrong	318
Precision of recognizing no answer	3 / 40 = 0.075
Recall of recognizing no answer	3 / 30 = 0.100
Factoid accuracy	0.206
Average F score for lists	0.075
Average F score for definintions	0.133
Final combined score	0.155

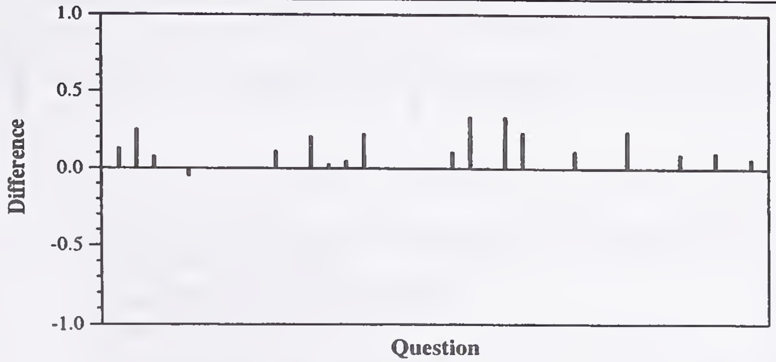


Difference from Median in F score for list questions

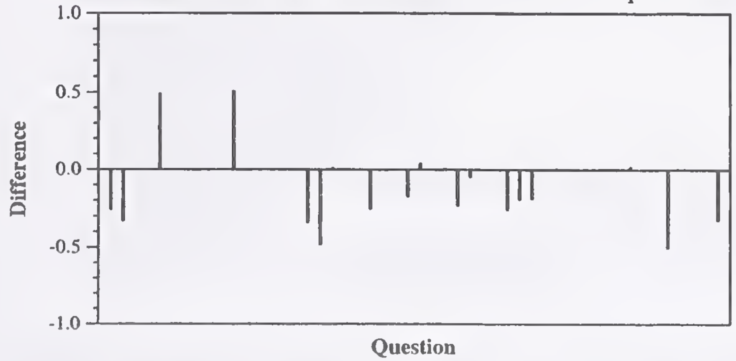


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	Albany03I4
Num questions	413 factoid + 37 list + 50 def
Number factoid right	94
Number factoid inexact	8
Number factoid unsupported	3
Number factoid wrong	308
Precision of recognizing no answer	6 / 52 = 0.115
Recall of recognizing no answer	6 / 30 = 0.200
Factoid accuracy	0.228
Average F score for lists	0.096
Average F score for definintions	0.134
Final combined score	0.172

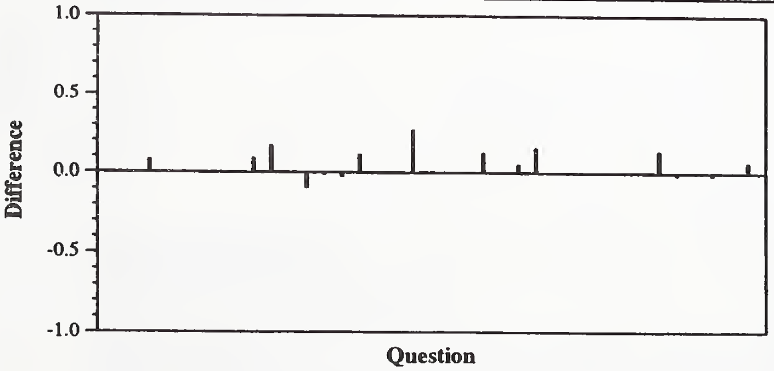


Difference from Median in F score for list questions

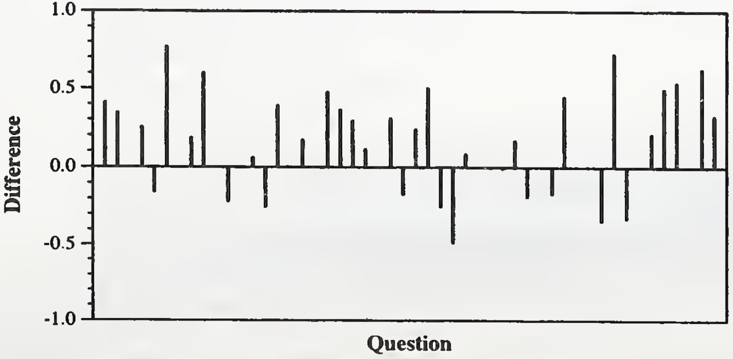


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	UAmsT03M1
Num questions	413 factoid + 37 list + 50 def
Number factoid right	56
Number factoid inexact	32
Number factoid unsupported	22
Number factoid wrong	303
Precision of recognizing no answer	1 / 3 = 0.333
Recall of recognizing no answer	1 / 30 = 0.033
Factoid accuracy	0.136
Average F score for lists	0.054
Average F score for definintions	0.315
Final combined score	0.160

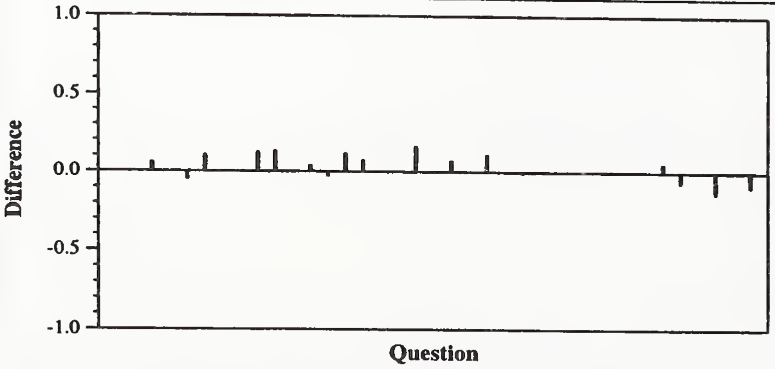


Difference from Median in F score for list questions

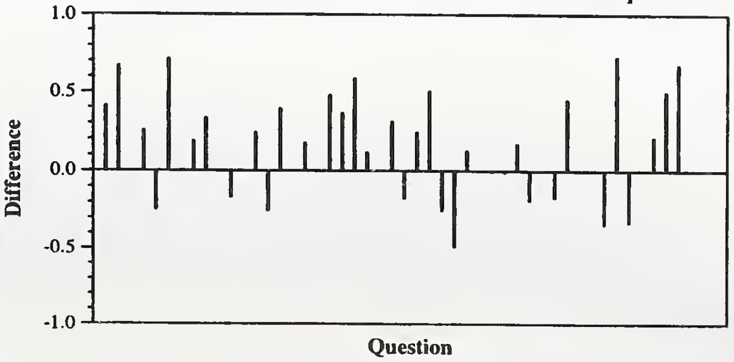


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	UAmsT03M2
Num questions	413 factoid + 37 list + 50 def
Number factoid right	60
Number factoid inexact	26
Number factoid unsupported	20
Number factoid wrong	307
Precision of recognizing no answer	3 / 11 = 0.273
Recall of recognizing no answer	3 / 30 = 0.100
Factoid accuracy	0.145
Average F score for lists	0.042
Average F score for definintions	0.308
Final combined score	0.160

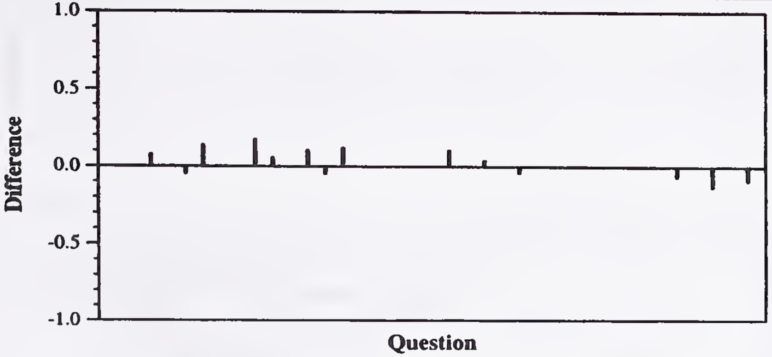


Difference from Median in F score for list questions

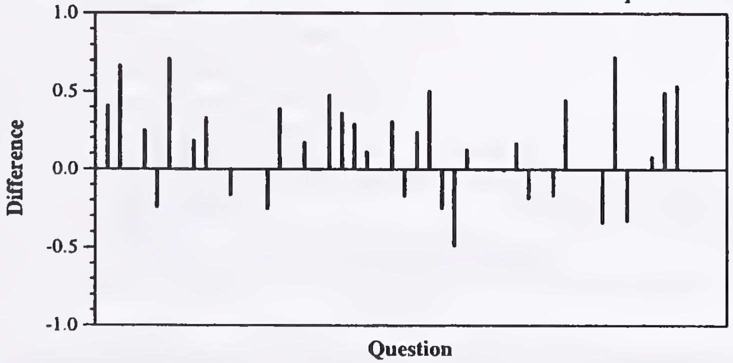


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	UAmsT03M3
Num questions	413 factoid + 37 list + 50 def
Number factoid right	53
Number factoid inexact	30
Number factoid unsupported	24
Number factoid wrong	306
Precision of recognizing no answer	1 / 3 = 0.333
Recall of recognizing no answer	1 / 30 = 0.033
Factoid accuracy	0.128
Average F score for lists	0.035
Average F score for definintions	0.292
Final combined score	0.146

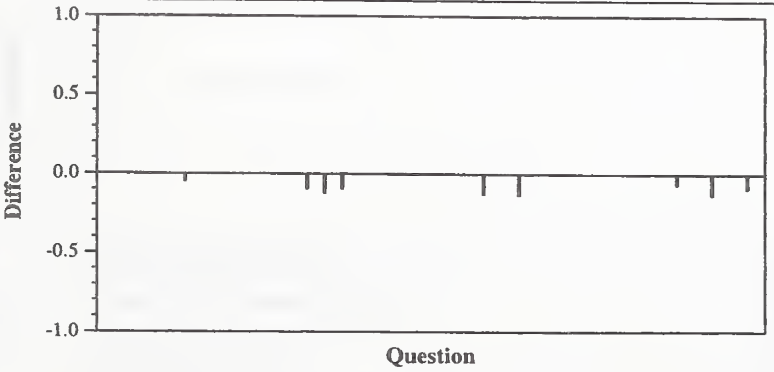


Difference from Median in F score for list questions

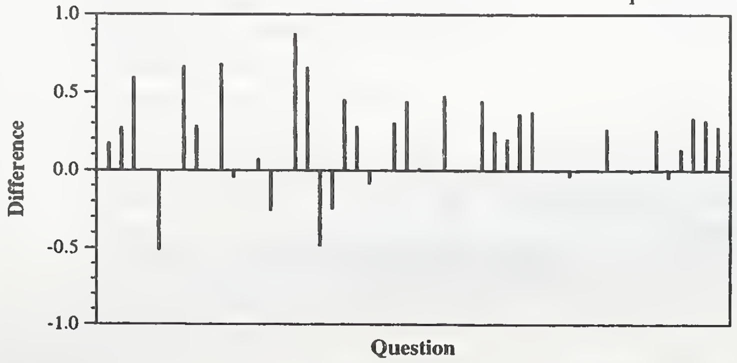


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	cuaqdef2003
Num questions	413 factoid + 37 list + 50 def
Number factoid right	30
Number factoid inexact	0
Number factoid unsupported	0
Number factoid wrong	383
Precision of recognizing no answer	30 /413 = 0.073
Recall of recognizing no answer	30 / 30 = 1.000
Factoid accuracy	0.073
Average F score for lists	0.000
Average F score for definintions	0.338
Final combined score	0.121

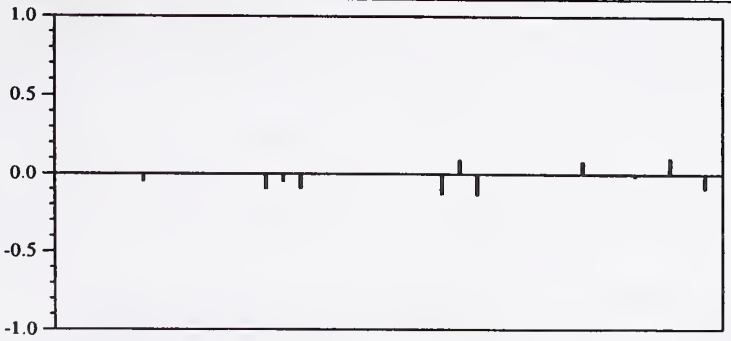


Difference from Median in F score for list questions

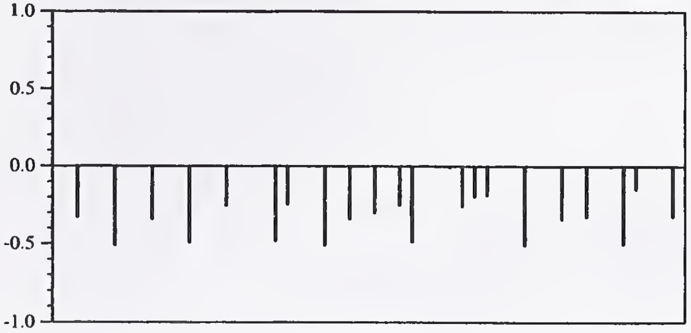


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	EdinInf2003A
Num questions	413 factoid + 37 list + 50 def
Number factoid right	30
Number factoid inexact	6
Number factoid unsupported	5
Number factoid wrong	372
NIL response never returned	
Factoid accuracy	0.073
Average F score for lists	0.014
Average F score for definintions	0.037
Final combined score	0.049

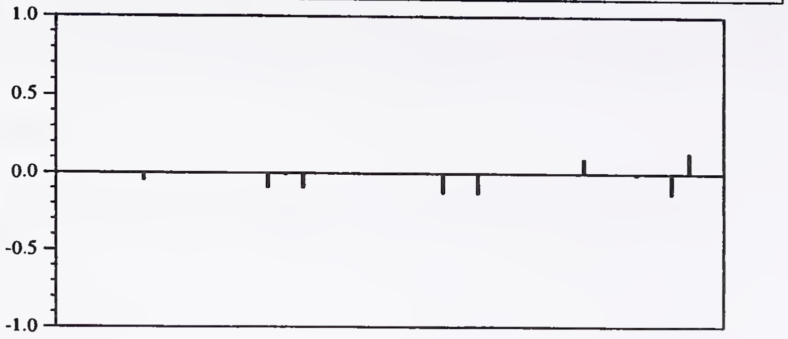


Difference from Median in F score for list questions

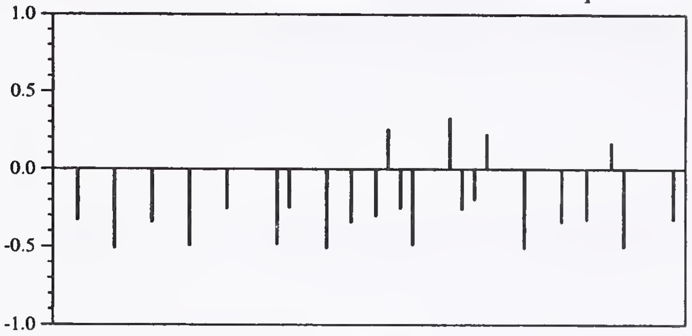


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	EdinInf2003B
Num questions	413 factoid + 37 list + 50 def
Number factoid right	30
Number factoid inexact	7
Number factoid unsupported	5
Number factoid wrong	371
NIL response never returned	
Factoid accuracy	0.073
Average F score for lists	0.013
Average F score for definintions	0.063
Final combined score	0.056

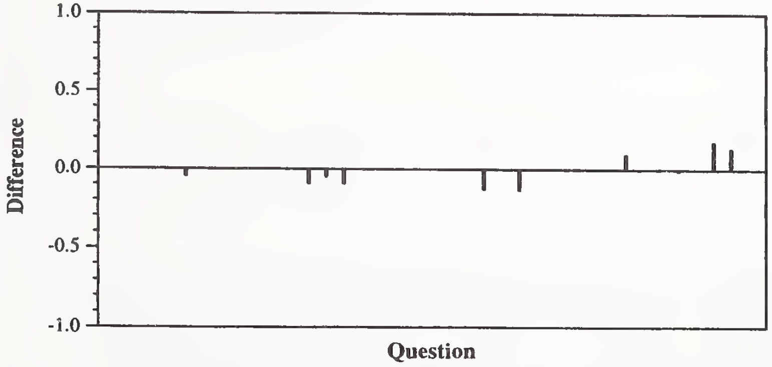


Difference from Median in F score for list questions

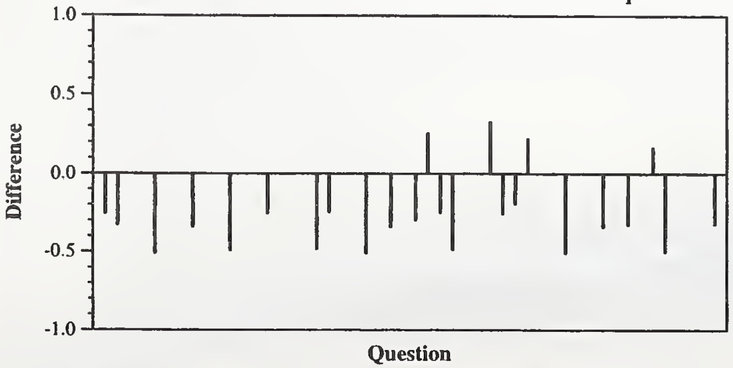


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	EdinInf2003C
Num questions	413 factoid + 37 list + 50 def
Number factoid right	24
Number factoid inexact	11
Number factoid unsupported	3
Number factoid wrong	375
NIL response never returned	
Factoid accuracy	0.058
Average F score for lists	0.020
Average F score for definintions	0.058
Final combined score	0.049

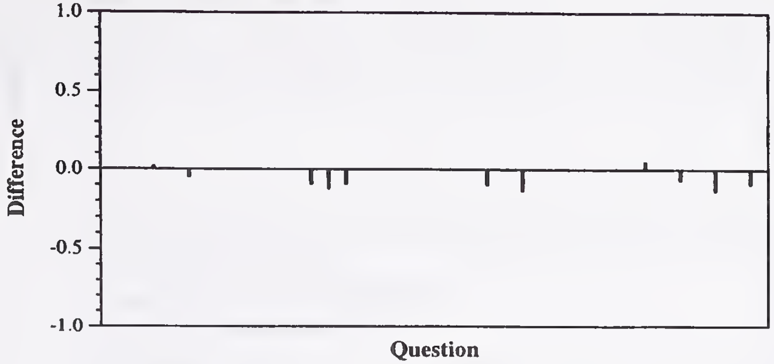


Difference from Median in F score for list questions

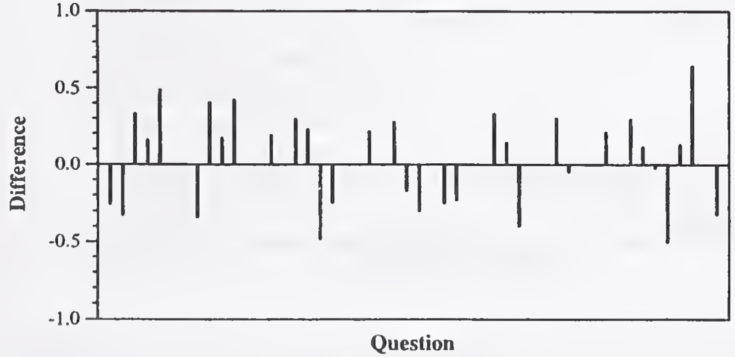


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	UIowaQA0301
Num questions	413 factoid + 37 list + 50 def
Number factoid right	14
Number factoid inexact	4
Number factoid unsupported	3
Number factoid wrong	392
Precision of recognizing no answer	10 /100 = 0.100
Recall of recognizing no answer	10 / 30 = 0.333
Factoid accuracy	0.034
Average F score for lists	0.002
Average F score for definintions	0.214
Final combined score	0.071

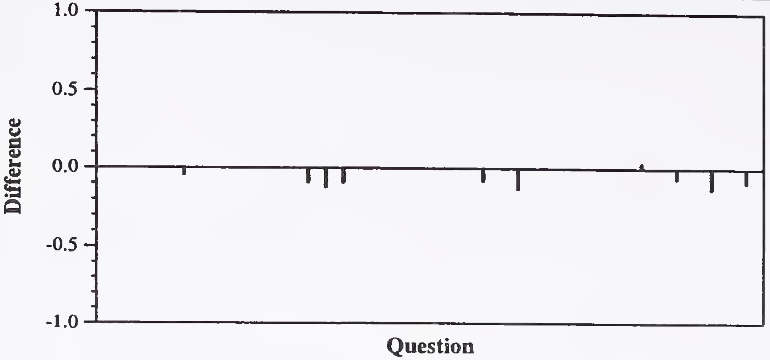


Difference from Median in F score for list questions

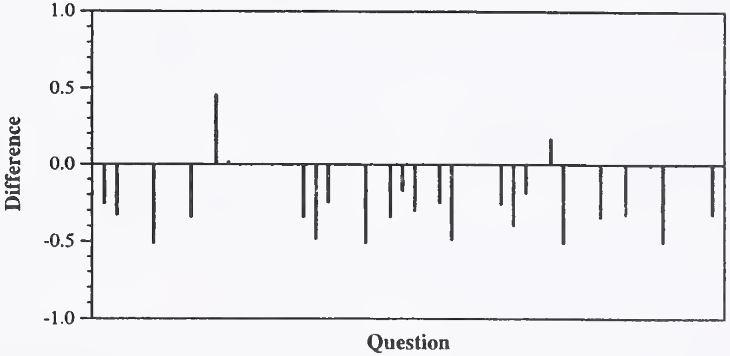


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	UIowaQA0302
Num questions	413 factoid + 37 list + 50 def
Number factoid right	17
Number factoid inexact	2
Number factoid unsupported	2
Number factoid wrong	392
Precision of recognizing no answer	15 /173 = 0.087
Recall of recognizing no answer	15 / 30 = 0.500
Factoid accuracy	0.041
Average F score for lists	0.002
Average F score for definintions	0.048
Final combined score	0.033

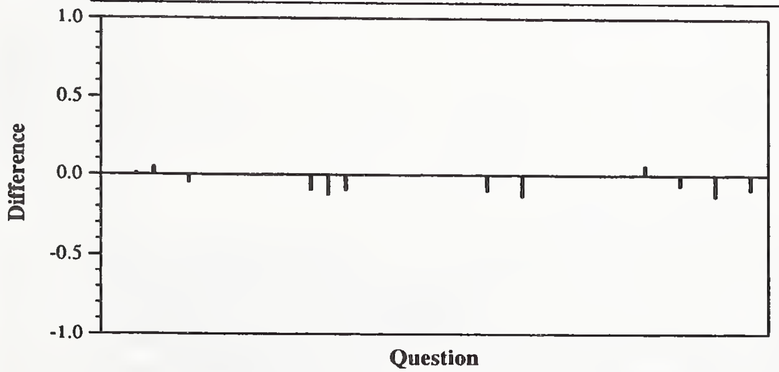


Difference from Median in F score for list questions

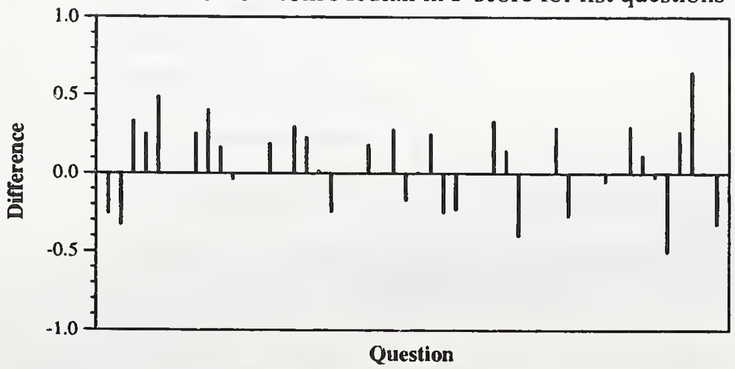


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	UIowaQA0303
Num questions	413 factoid + 37 list + 50 def
Number factoid right	17
Number factoid inexact	2
Number factoid unsupported	3
Number factoid wrong	391
Precision of recognizing no answer	10 / 98 = 0.102
Recall of recognizing no answer	10 / 30 = 0.333
Factoid accuracy	0.041
Average F score for lists	0.004
Average F score for definintions	0.231
Final combined score	0.079

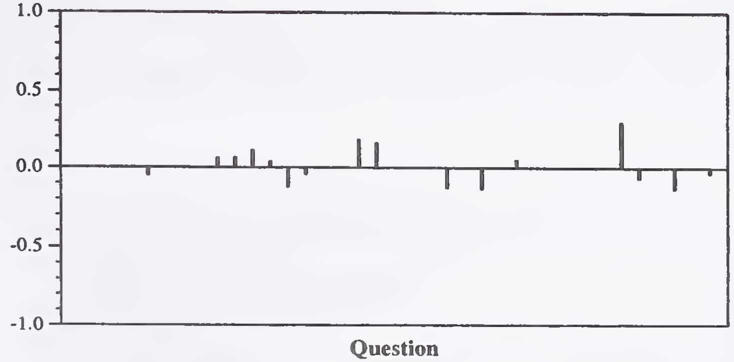


Difference from Median in F score for list questions

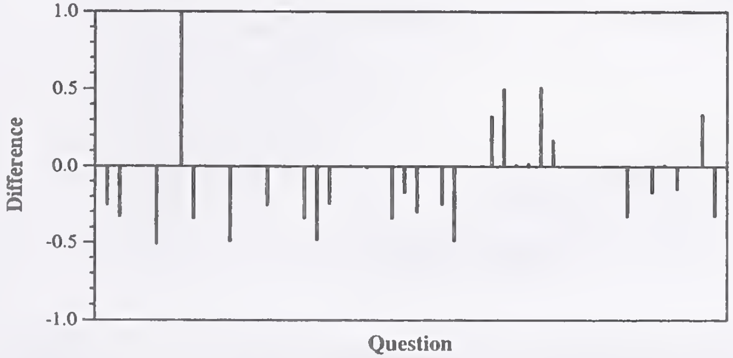


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	DLT03QA01
Num questions	413 factoid + 37 list + 50 def
Number factoid right	37
Number factoid inexact	9
Number factoid unsupported	0
Number factoid wrong	367
Precision of recognizing no answer	7 /119 = 0.059
Recall of recognizing no answer	7 / 30 = 0.233
Factoid accuracy	0.090
Average F score for lists	0.032
Average F score for definintions	0.126
Final combined score	0.084

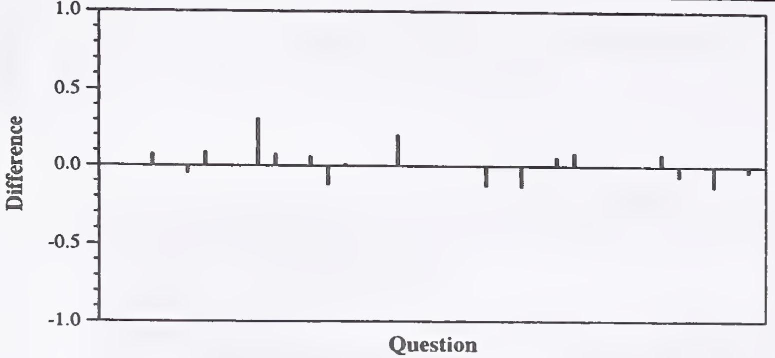


Difference from Median in F score for list questions

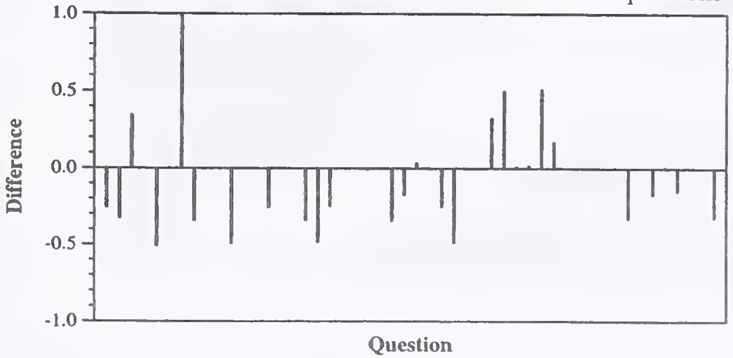


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	DLT03QA02
Num questions	413 factoid + 37 list + 50 def
Number factoid right	33
Number factoid inexact	3
Number factoid unsupported	1
Number factoid wrong	376
Precision of recognizing no answer	16 /188 = 0.085
Recall of recognizing no answer	16 / 30 = 0.533
Factoid accuracy	0.080
Average F score for lists	0.034
Average F score for definintions	0.133
Final combined score	0.082

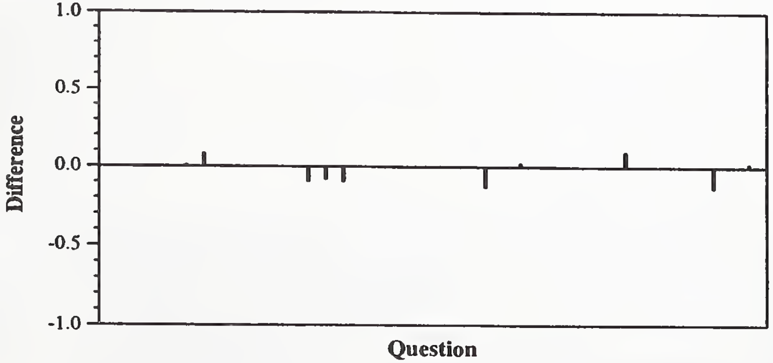


Difference from Median in F score for list questions

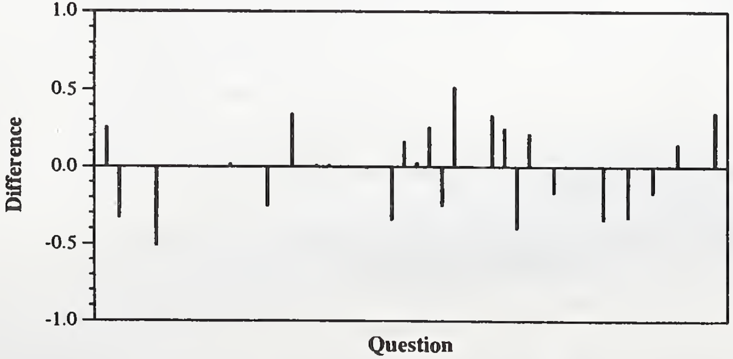


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	piq001
Num questions	413 factoid + 37 list + 50 def
Number factoid right	38
Number factoid inexact	6
Number factoid unsupported	2
Number factoid wrong	367
Precision of recognizing no answer	9 / 73 = 0.123
Recall of recognizing no answer	9 / 30 = 0.300
Factoid accuracy	0.092
Average F score for lists	0.017
Average F score for definintions	0.180
Final combined score	0.095

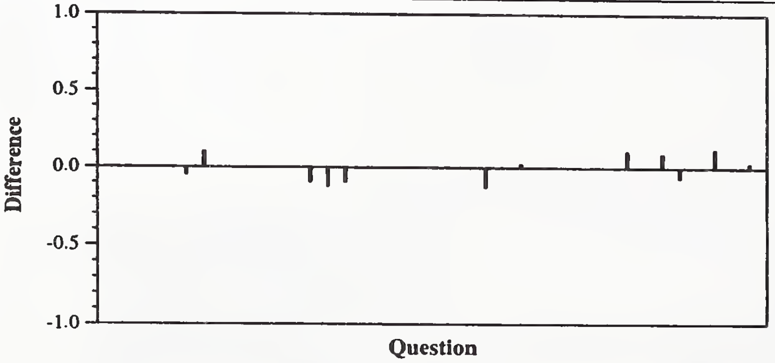


Difference from Median in F score for list questions

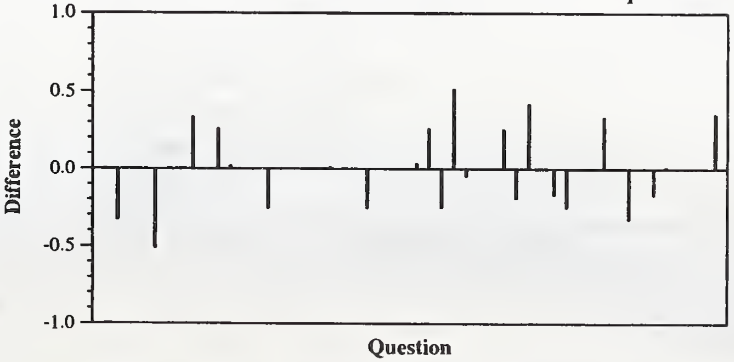


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	piq002
Num questions	413 factoid + 37 list + 50 def
Number factoid right	49
Number factoid inexact	5
Number factoid unsupported	3
Number factoid wrong	356
Precision of recognizing no answer	19 /193 = 0.098
Recall of recognizing no answer	19 / 30 = 0.633
Factoid accuracy	0.119
Average F score for lists	0.022
Average F score for definintions	0.185
Final combined score	0.111



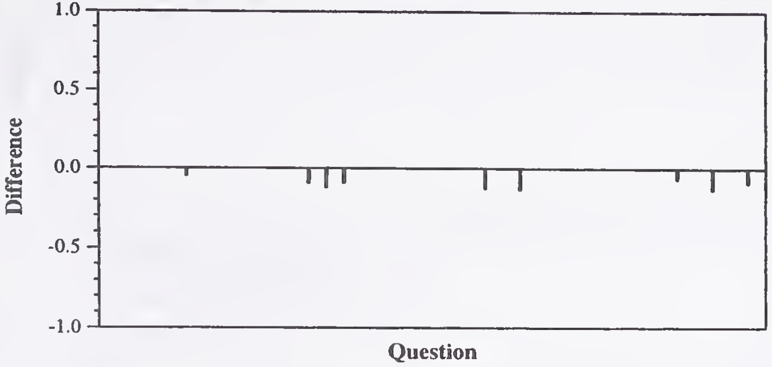
Difference from Median in F score for list questions



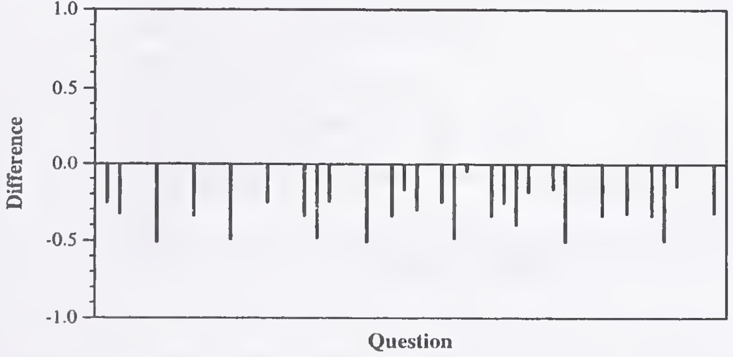
Difference from Median in F score for definition questions

Question Answering track, main task results — Universitat Politecnica de Catalunya (UPC) and Universitat de Girona (UdG)

Summary Statistics	
Run ID	UPCUdGsys1
Num questions	413 factoid + 37 list + 50 def
Number factoid right	22
Number factoid inexact	2
Number factoid unsupported	0
Number factoid wrong	389
Precision of recognizing no answer	13 /141 = 0.092
Recall of recognizing no answer	13 / 30 = 0.433
Factoid accuracy	0.053
Average F score for lists	0.000
Average F score for definintions	0.000
Final combined score	0.026

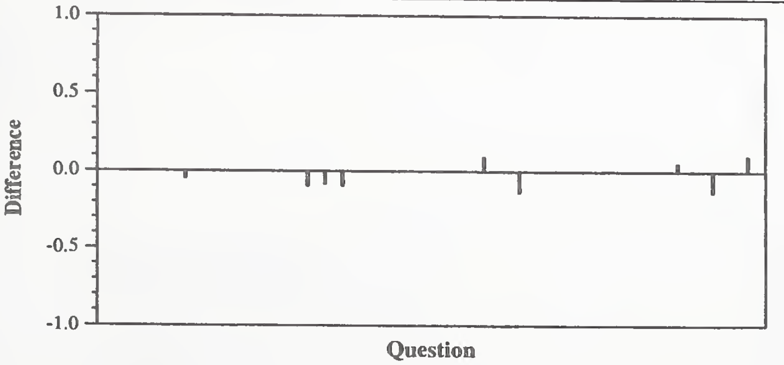


Difference from Median in F score for list questions

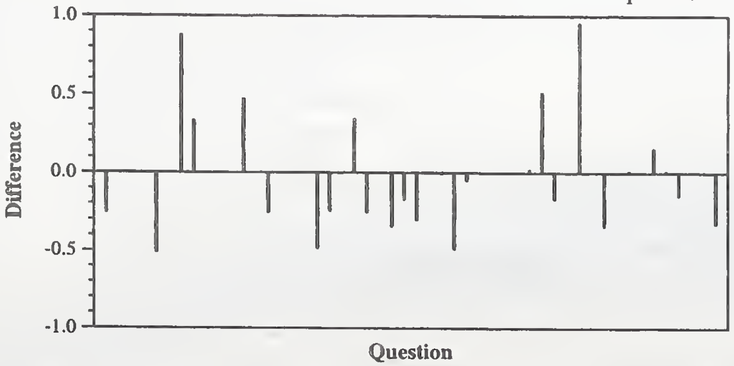


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	shef12madcow
Num questions	413 factoid + 37 list + 50 def
Number factoid right	26
Number factoid inexact	32
Number factoid unsupported	10
Number factoid wrong	345
Precision of recognizing no answer	4 / 36 = 0.111
Recall of recognizing no answer	4 / 30 = 0.133
Factoid accuracy	0.063
Average F score for lists	0.015
Average F score for definintions	0.171
Final combined score	0.078

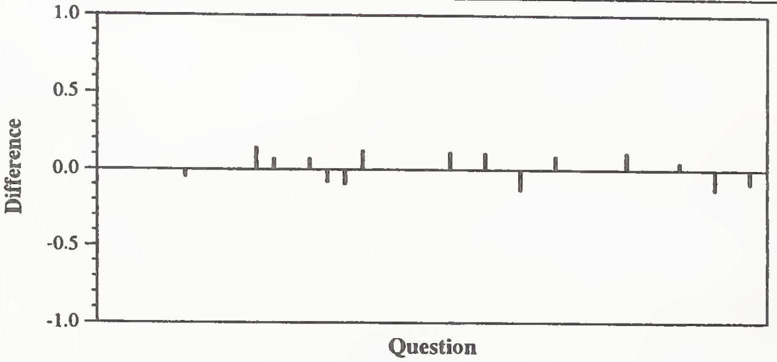


Difference from Median in F score for list questions

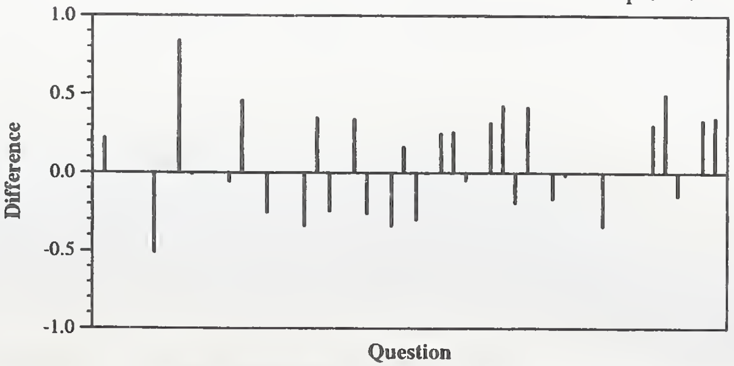


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	shef12okapi
Num questions	413 factoid + 37 list + 50 def
Number factoid right	19
Number factoid inexact	27
Number factoid unsupported	8
Number factoid wrong	359
Precision of recognizing no answer	1 / 10 = 0.100
Recall of recognizing no answer	1 / 30 = 0.033
Factoid accuracy	0.046
Average F score for lists	0.033
Average F score for definintions	0.230
Final combined score	0.089

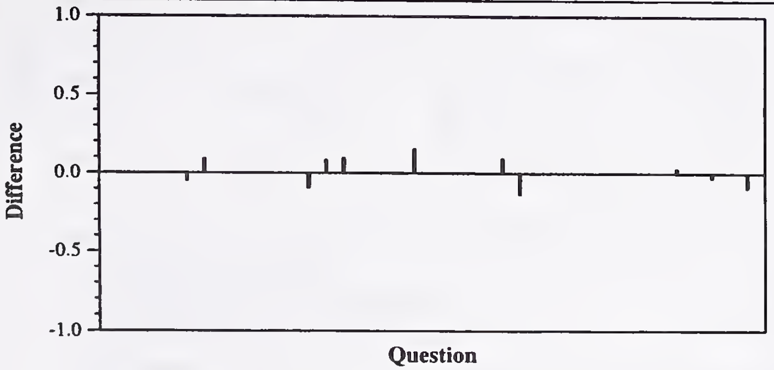


Difference from Median in F score for list questions

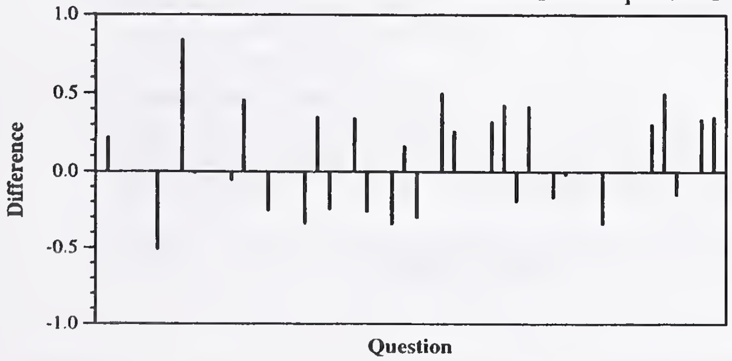


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	shef12simple
Num questions	413 factoid + 37 list + 50 def
Number factoid right	57
Number factoid inexact	6
Number factoid unsupported	12
Number factoid wrong	338
Precision of recognizing no answer	15 /191 = 0.079
Recall of recognizing no answer	15 / 30 = 0.500
Factoid accuracy	0.138
Average F score for lists	0.029
Average F score for definintions	0.236
Final combined score	0.135

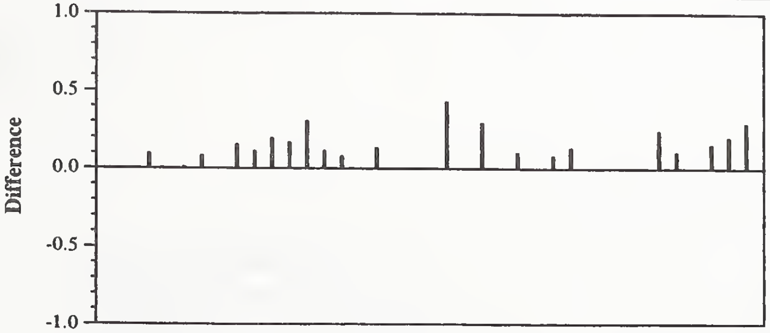


Difference from Median in F score for list questions

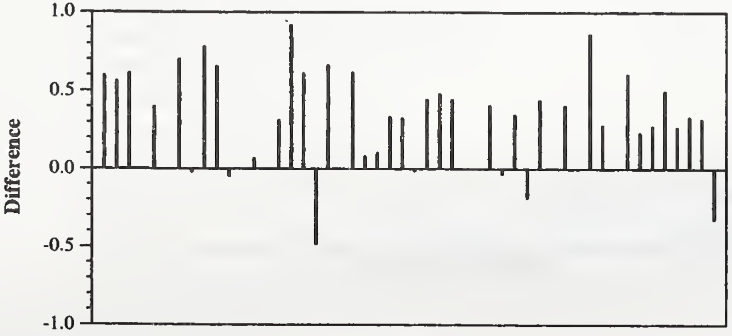


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	isi03a
Num questions	413 factoid + 37 list + 50 def
Number factoid right	139
Number factoid inexact	5
Number factoid unsupported	21
Number factoid wrong	248
Precision of recognizing no answer	6 / 84 = 0.071
Recall of recognizing no answer	6 / 30 = 0.200
Factoid accuracy	0.337
Average F score for lists	0.118
Average F score for definintions	0.461
Final combined score	0.313

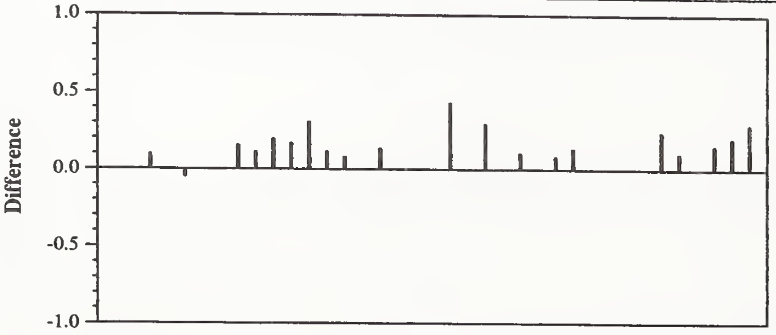


Difference from Median in F score for list questions

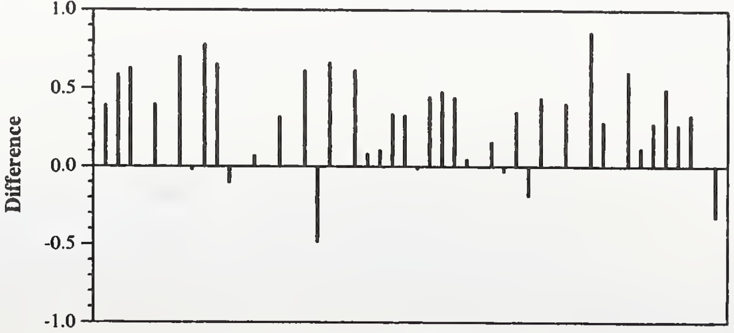


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	isi03b
Num questions	413 factoid + 37 list + 50 def
Number factoid right	119
Number factoid inexact	7
Number factoid unsupported	21
Number factoid wrong	266
NIL response never returned	
Factoid accuracy	0.288
Average F score for lists	0.115
Average F score for definintions	0.426
Final combined score	0.279

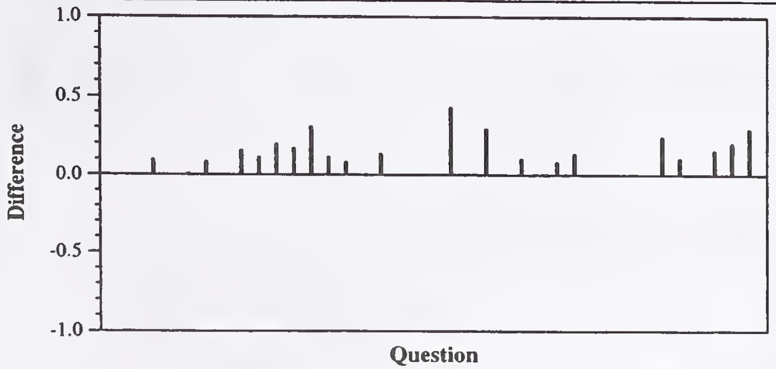


Difference from Median in F score for list questions

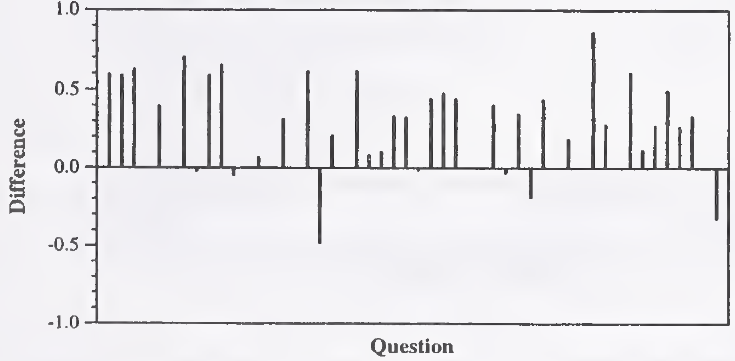


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	isi03c
Num questions	413 factoid + 37 list + 50 def
Number factoid right	136
Number factoid inexact	5
Number factoid unsupported	21
Number factoid wrong	251
Precision of recognizing no answer	6 / 90 = 0.067
Recall of recognizing no answer	6 / 30 = 0.200
Factoid accuracy	0.329
Average F score for lists	0.118
Average F score for definintions	0.418
Final combined score	0.298

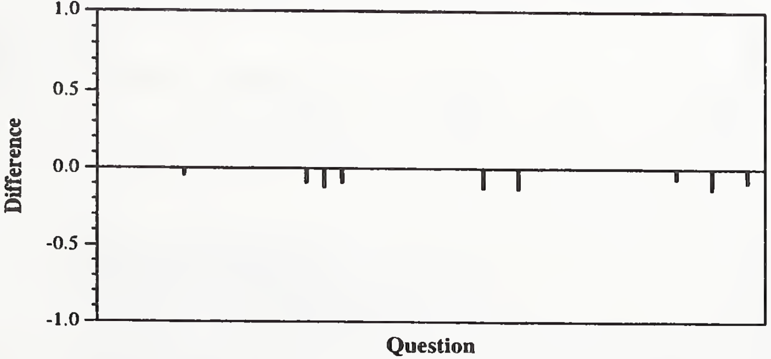


Difference from Median in F score for list questions

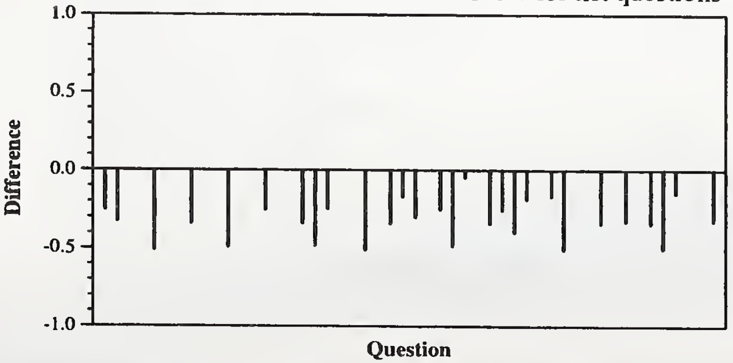


Difference from Median in F score for definition questions

Summary Statistics	
Run ID	uwbqitekat03
Num questions	413 factoid + 37 list + 50 def
Number factoid right	107
Number factoid inexact	6
Number factoid unsupported	2
Number factoid wrong	298
Precision of recognizing no answer	29 /316 = 0.092
Recall of recognizing no answer	29 / 30 = 0.967
Factoid accuracy	0.259
Average F score for lists	0.000
Average F score for definintions	0.000
Final combined score	0.130



Difference from Median in F score for list questions



Difference from Median in F score for definition questions

Summary Statistics	
Run ID	clr03p1
Num questions	413
Number right	36
Number unsupported	1
Number wrong	376
Accuracy	0.087
Precision of recognizing no answer	7 / 64 = 0.109
Recall of recognizing no answer	7 / 30 = 0.233

Summary Statistics	
Run ID	clr03p2
Num questions	413
Number right	49
Number unsupported	1
Number wrong	363
Accuracy	0.119
Precision of recognizing no answer	7 / 64 = 0.109
Recall of recognizing no answer	7 / 30 = 0.233

Summary Statistics	
Run ID	IITBQA
Num questions	413
Number right	44
Number unsupported	1
Number wrong	368
Accuracy	0.107
Precision of recognizing no answer	22 /303 = 0.073
Recall of recognizing no answer	22 / 30 = 0.733

Summary Statistics	
Run ID	IITBQA1
Num questions	413
Number right	55
Number unsupported	5
Number wrong	353
Accuracy	0.133
Precision of recognizing no answer	3 / 66 = 0.045
Recall of recognizing no answer	3 / 30 = 0.100

Summary Statistics	
Run ID	LCCpass03
Num questions	413
Number right	283
Number unsupported	8
Number wrong	122
Accuracy	0.685
Precision of recognizing no answer	24 / 63 = 0.381
Recall of recognizing no answer	24 / 30 = 0.800

Summary Statistics	
Run ID	answfind1
Num questions	413
Number right	79
Number unsupported	1
Number wrong	333
Accuracy	0.191
NIL response never returned	

Summary Statistics	
Run ID	answfind2
Num questions	413
Number right	77
Number unsupported	1
Number wrong	335
Accuracy	0.186
NIL response never returned	

Summary Statistics	
Run ID	answfind3
Num questions	413
Number right	75
Number unsupported	2
Number wrong	336
Accuracy	0.182
NIL response never returned	

Summary Statistics	
Run ID	nuslamp03
Num questions	413
Number right	168
Number unsupported	2
Number wrong	243
Accuracy	0.407
Precision of recognizing no answer	25 /207 = 0.121
Recall of recognizing no answer	25 / 30 = 0.833

Summary Statistics	
Run ID	nuslamp03a
Num questions	413
Number right	173
Number unsupported	9
Number wrong	231
Accuracy	0.419
Precision of recognizing no answer	10 / 64 = 0.156
Recall of recognizing no answer	10 / 30 = 0.333

Summary Statistics	
Run ID	nuslamp03b
Num questions	413
Number right	164
Number unsupported	8
Number wrong	241
Accuracy	0.397
Precision of recognizing no answer	17 /144 = 0.118
Recall of recognizing no answer	17 / 30 = 0.567

Summary Statistics	
Run ID	pircsqa1
Num questions	413
Number right	37
Number unsupported	0
Number wrong	376
Accuracy	0.090
Precision of recognizing no answer	7 / 82 = 0.085
Recall of recognizing no answer	7 / 30 = 0.233

Summary Statistics	
Run ID	pircsqa2
Num questions	413
Number right	39
Number unsupported	2
Number wrong	372
Accuracy	0.094
Precision of recognizing no answer	8 / 82 = 0.098
Recall of recognizing no answer	8 / 30 = 0.267

Summary Statistics	
Run ID	pircsqa3
Num questions	413
Number right	40
Number unsupported	5
Number wrong	368
Accuracy	0.097
Precision of recognizing no answer	0 / 3 = 0.000
Recall of recognizing no answer	0 / 30 = 0.000

Question Answering track, passages task results — Saarland University

Summary Statistics	
Run ID	Saarland
Num questions	413
Number right	70
Number unsupported	4
Number wrong	339
Accuracy	0.169
Precision of recognizing no answer	11 /113 = 0.097
Recall of recognizing no answer	11 / 30 = 0.367

Summary Statistics	
Run ID	UAmsT03P1
Num questions	413
Number right	46
Number unsupported	6
Number wrong	361
Accuracy	0.111
Precision of recognizing no answer	10 / 78 = 0.128
Recall of recognizing no answer	10 / 30 = 0.333

Summary Statistics	
Run ID	umassql
Num questions	413
Number right	83
Number unsupported	6
Number wrong	324
Accuracy	0.201
NIL response never returned	

Question Answering track, passages task results — University of Michigan

Summary Statistics	
Run ID	NSIR
Num questions	413
Number right	35
Number unsupported	4
Number wrong	374
Accuracy	0.085
Precision of recognizing no answer	3 / 40 = 0.075
Recall of recognizing no answer	3 / 30 = 0.100

Summary Statistics	
Run ID	uwmtCQ0
Num questions	413
Number right	132
Number unsupported	15
Number wrong	266
Accuracy	0.320
NIL response never returned	

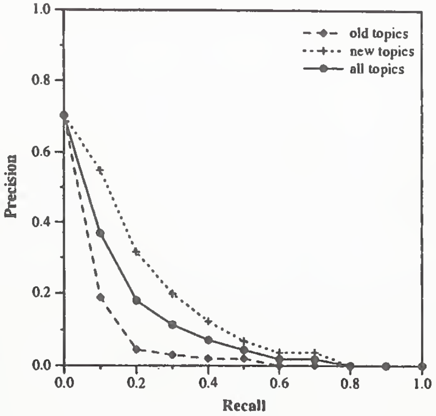
Summary Statistics	
Run ID	uwmtCQ1
Num questions	413
Number right	122
Number unsupported	14
Number wrong	277
Accuracy	0.295
NIL response never returned	

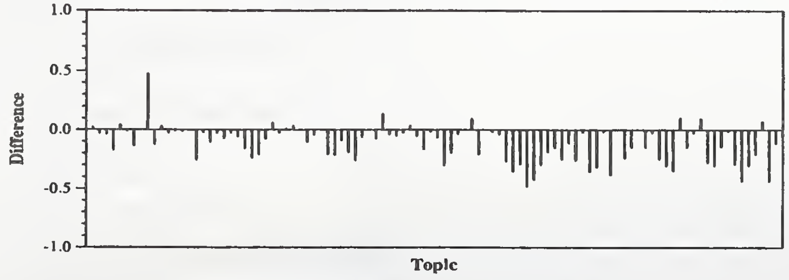
Summary Statistics	
Run ID	uwmtCQ2
Num questions	413
Number right	145
Number unsupported	11
Number wrong	257
Accuracy	0.351
NIL response never returned	

Summary Statistics	
Run ID:	NLPR03vb10
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	500	504	1004
Total relevant	4416	1658	6074
Total relevant retrieved	167	231	398
Mean average precision	0.0533	0.1577	0.1055
% topics with no relevant in top 10	6.0	8.0	7.0
Integral of MAP(X) curve for worst 1/4 topics	0.0014	0.0042	0.0023

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3800	0.5160	0.4480
At 10 docs	0.3340	0.4600	0.3970
At 15 docs	0.2227	0.3080	0.2653
At 20 docs	0.1670	0.2310	0.1990
At 30 docs	0.1113	0.1540	0.1327
At 100 docs	0.0334	0.0462	0.0398
At 200 docs	0.0167	0.0231	0.0199
At 500 docs	0.0067	0.0092	0.0080
At 1000 docs	0.0033	0.0046	0.0040
R-Precision			
Exact	0.0800	0.1962	0.1381



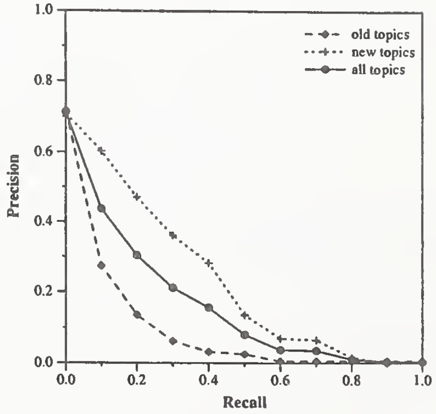


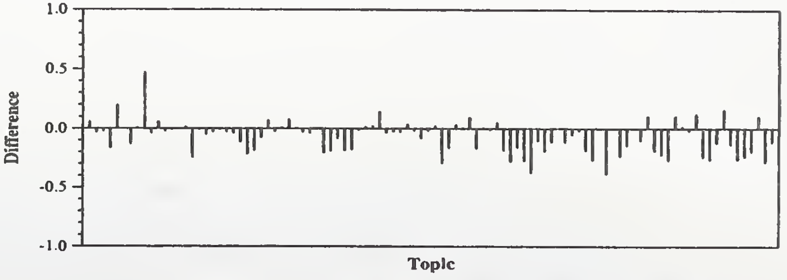
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	NLPR03vb25
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	1250	1252	2502
Total relevant	4416	1658	6074
Total relevant retrieved	343	419	762
Mean average precision	0.0784	0.2249	0.1516
% topics with no relevant in top 10	6.0	8.0	7.0
Integral of MAP(X) curve for worst 1/4 topics	0.0037	0.0081	0.0047

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3800	0.5160	0.4480
At 10 docs	0.3340	0.4600	0.3970
At 15 docs	0.3040	0.4133	0.3587
At 20 docs	0.2900	0.3740	0.3320
At 30 docs	0.2287	0.2793	0.2540
At 100 docs	0.0686	0.0838	0.0762
At 200 docs·	0.0343	0.0419	0.0381
At 500 docs	0.0137	0.0168	0.0152
At 1000 docs	0.0069	0.0084	0.0076
R-Precision			
Exact	0.1262	0.2842	0.2052



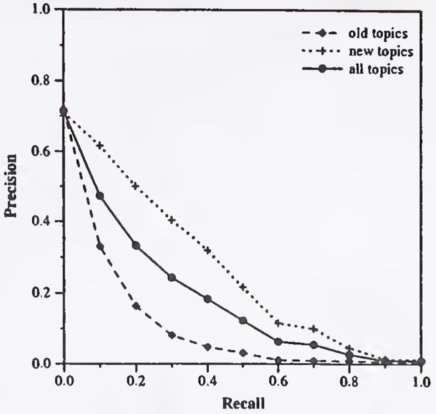


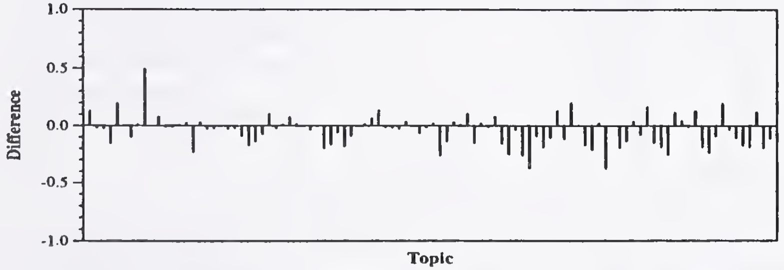
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	NLPR03vb50
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	2530	2500	5030
Total relevant	4416	1658	6074
Total relevant retrieved	522	571	1093
Mean average precision	0.0948	0.2592	0.1770
% topics with no relevant in top 10	6.0	8.0	7.0
Integral of MAP(X) curve for worst 1/4 topics	0.0054	0.0133	0.0075

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3800	0.5160	0.4480
At 10 docs	0.3340	0.4600	0.3970
At 15 docs	0.3040	0.4133	0.3587
At 20 docs	0.2900	0.3740	0.3320
At 30 docs	0.2533	0.3107	0.2820
At 100 docs	0.1044	0.1142	0.1093
At 200 docs	0.0522	0.0571	0.0546
At 500 docs	0.0209	0.0228	0.0219
At 1000 docs	0.0104	0.0114	0.0109
R-Precision			
Exact	0.1551	0.3212	0.2381



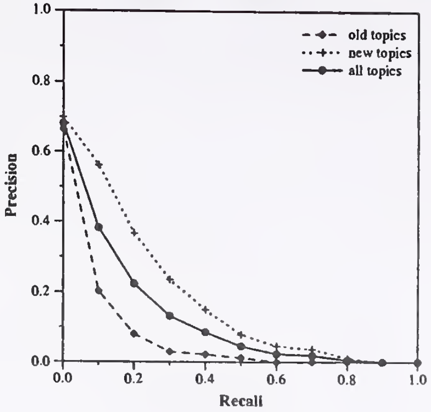


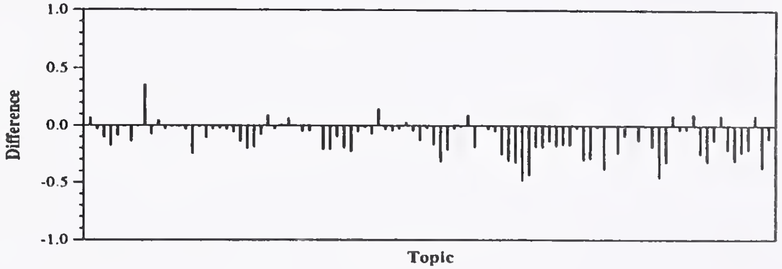
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	NLPR03w16
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	866	860	1726
Total relevant	4416	1658	6074
Total relevant retrieved	246	303	549
Mean average precision	0.0558	0.1747	0.1153
% topics with no relevant in top 10	10.0	10.0	10.0
Integral of MAP(X) curve for worst 1/4 topics	0.0017	0.0027	0.0021

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3280	0.4600	0.3940
At 10 docs	0.3040	0.4340	0.3690
At 15 docs	0.2960	0.3720	0.3340
At 20 docs	0.2400	0.2980	0.2690
At 30 docs	0.1640	0.2013	0.1827
At 100 docs	0.0492	0.0606	0.0549
At 200 docs	0.0246	0.0303	0.0274
At 500 docs	0.0098	0.0121	0.0110
At 1000 docs	0.0049	0.0061	0.0055
R-Precision			
Exact	0.1018	0.2294	0.1656



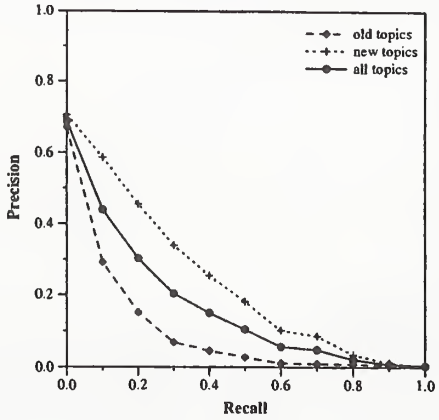


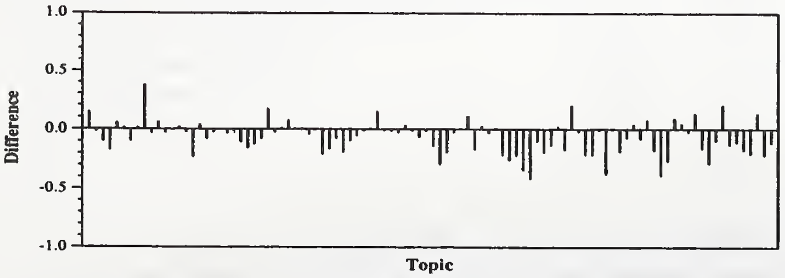
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	NLPR03w49
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	2470	2479	4949
Total relevant	4416	1658	6074
Total relevant retrieved	495	525	1020
Mean average precision	0.0828	0.2289	0.1559
% topics with no relevant in top 10	10.0	10.0	10.0
Integral of MAP(X) curve for worst 1/4 topics	0.0038	0.0092	0.0051

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3280	0.4600	0.3940
At 10 docs	0.3040	0.4340	0.3690
At 15 docs	0.2973	0.3720	0.3347
At 20 docs	0.2800	0.3310	0.3055
At 30 docs	0.2380	0.2833	0.2607
At 100 docs	0.0990	0.1050	0.1020
At 200 docs	0.0495	0.0525	0.0510
At 500 docs	0.0198	0.0210	0.0204
At 1000 docs	0.0099	0.0105	0.0102
R-Precision			
Exact	0.1476	0.2966	0.2221



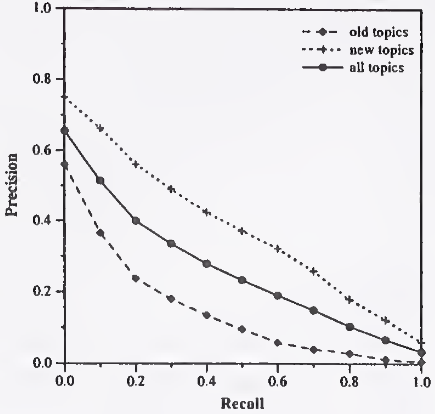


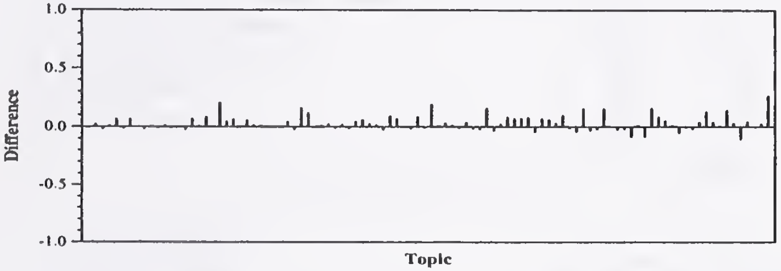
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	fub03IeOLKe3
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2053	1376	3429
Mean average precision	0.1315	0.3692	0.2503
% topics with no relevant in top 10	24.0	12.0	18.0
Integral of MAP(X) curve for worst 1/4 topics	0.0061	0.0117	0.0065

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3840	0.5480	0.4660
At 10 docs	0.3360	0.4780	0.4070
At 15 docs	0.2800	0.4227	0.3513
At 20 docs	0.2650	0.3890	0.3270
At 30 docs	0.2367	0.3160	0.2763
At 100 docs	0.1424	0.1598	0.1511
At 200 docs	0.1049	0.0997	0.1023
At 500 docs	0.0627	0.0495	0.0561
At 1000 docs	0.0411	0.0275	0.0343
R-Precision			
Exact	0.1805	0.3532	0.2669



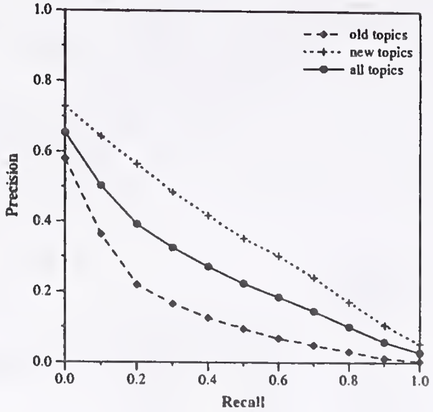


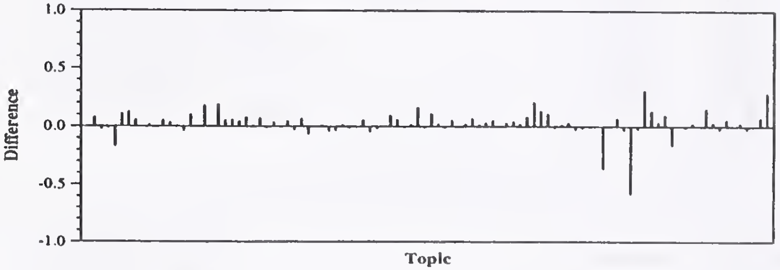
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	fub03InB2e3
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2157	1418	3575
Mean average precision	0.1317	0.3552	0.2434
% topics with no relevant in top 10	26.0	10.0	18.0
Integral of MAP(X) curve for worst 1/4 topics	0.0047	0.0192	0.0084

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3880	0.5520	0.4700
At 10 docs	0.3360	0.5000	0.4180
At 15 docs	0.2907	0.4400	0.3653
At 20 docs	0.2720	0.3980	0.3350
At 30 docs	0.2327	0.3280	0.2803
At 100 docs	0.1462	0.1622	0.1542
At 200 docs	0.1072	0.1035	0.1053
At 500 docs	0.0663	0.0511	0.0587
At 1000 docs	0.0431	0.0284	0.0357
R-Precision			
Exact	0.1712	0.3570	0.2641



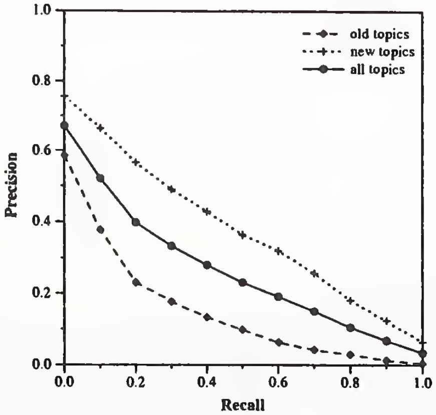


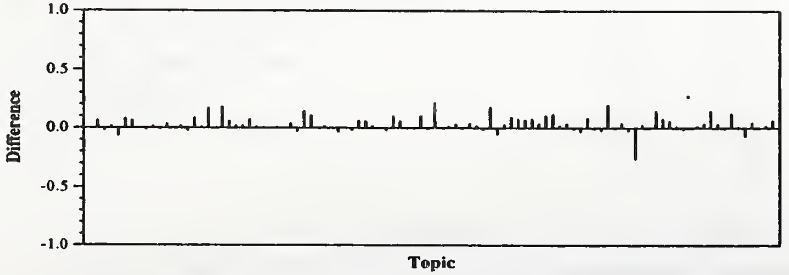
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	fub03InOLe3
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2124	1412	3536
Mean average precision	0.1340	0.3697	0.2519
% topics with no relevant in top 10	24.0	10.0	17.0
Integral of MAP(X) curve for worst 1/4 topics	0.0070	0.0152	0.0077

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3920	0.5600	0.4760
At 10 docs	0.3380	0.4880	0.4130
At 15 docs	0.2960	0.4240	0.3600
At 20 docs	0.2680	0.3900	0.3290
At 30 docs	0.2400	0.3187	0.2793
At 100 docs	0.1456	0.1636	0.1546
At 200 docs	0.1062	0.1018	0.1040
At 500 docs	0.0654	0.0515	0.0585
At 1000 docs	0.0425	0.0282	0.0354
R-Precision			
Exact	0.1779	0.3656	0.2717



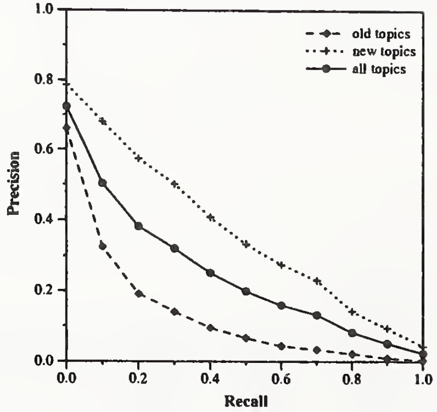


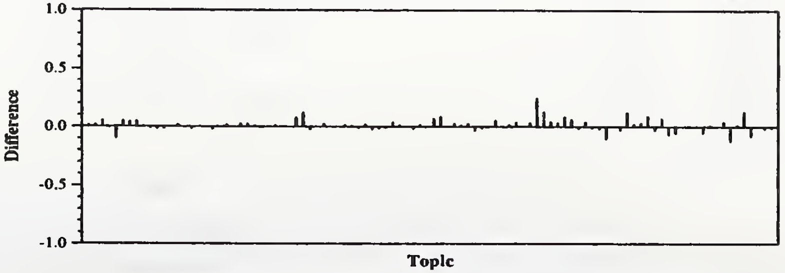
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	fub03IneOBu3
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1868	1372	3240
Mean average precision	0.1134	0.3524	0.2329
% topics with no relevant in top 10	14.0	8.0	11.0
Integral of MAP(X) curve for worst 1/4 topics	0.0052	0.0232	0.0096

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3480	0.5760	0.4620
At 10 docs	0.3080	0.4800	0.3940
At 15 docs	0.2787	0.4280	0.3533
At 20 docs	0.2560	0.3830	0.3195
At 30 docs	0.2180	0.3127	0.2653
At 100 docs	0.1280	0.1558	0.1419
At 200 docs	0.0928	0.0964	0.0946
At 500 docs	0.0578	0.0487	0.0533
At 1000 docs	0.0374	0.0274	0.0324
R-Precision			
Exact	0.1630	0.3599	0.2614



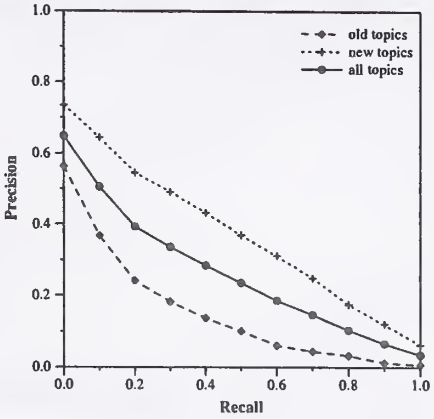


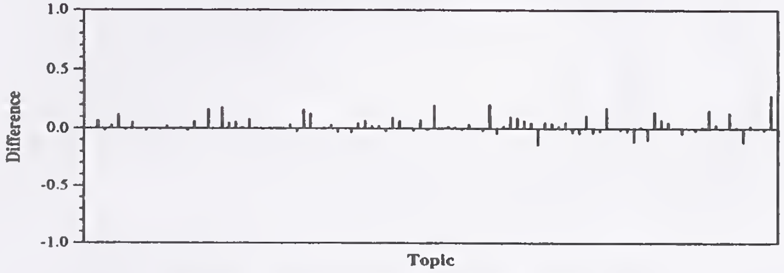
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	fub03IneOLe3
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2113	1367	3480
Mean average precision	0.1343	0.3614	0.2479
% topics with no relevant in top 10	24.0	16.0	20.0
Integral of MAP(X) curve for worst 1/4 topics	0.0057	0.0098	0.0058

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3760	0.5360	0.4560
At 10 docs	0.3300	0.4660	0.3980
At 15 docs	0.2853	0.4120	0.3487
At 20 docs	0.2610	0.3840	0.3225
At 30 docs	0.2313	0.3120	0.2717
At 100 docs	0.1444	0.1554	0.1499
At 200 docs	0.1068	0.0967	0.1017
At 500 docs	0.0642	0.0491	0.0566
At 1000 docs	0.0423	0.0273	0.0348
R-Precision			
Exact	0.1800	0.3552	0.2676



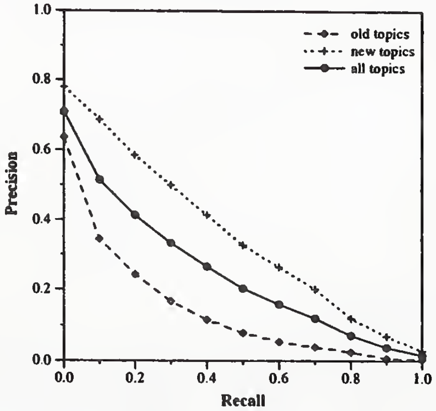


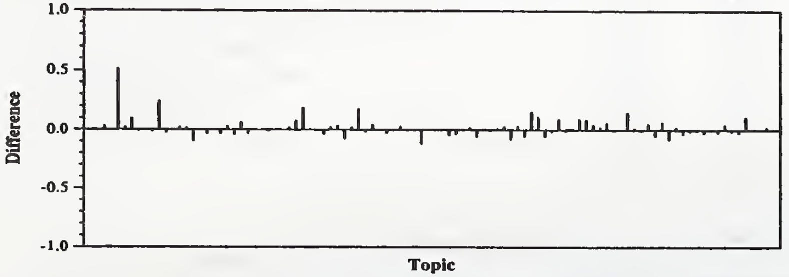
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	humR03d
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1686	1325	3011
Mean average precision	0.1271	0.3464	0.2367
% topics with no relevant in top 10	22.0	8.0	15.0
Integral of MAP(X) curve for worst 1/4 topics	0.0049	0.0258	0.0088

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3720	0.5760	0.4740
At 10 docs	0.2960	0.4760	0.3860
At 15 docs	0.2600	0.4267	0.3433
At 20 docs	0.2470	0.3860	0.3165
At 30 docs	0.2080	0.3140	0.2610
At 100 docs	0.1188	0.1506	0.1347
At 200 docs	0.0863	0.0932	0.0897
At 500 docs	0.0526	0.0468	0.0497
At 1000 docs	0.0337	0.0265	0.0301
R-Precision			
Exact	0.1743	0.3446	0.2594



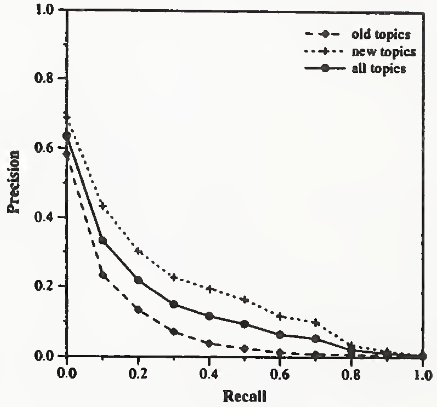


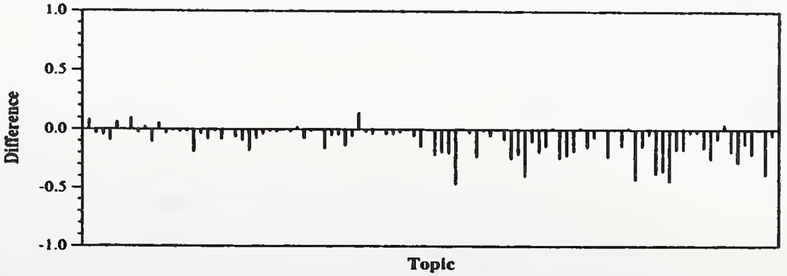
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	humR03dc
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	5000	5000	10000
Total relevant	4416	1658	6074
Total relevant retrieved	594	753	1347
Mean average precision	0.0713	0.1784	0.1248
% topics with no relevant in top 10	18.0	12.0	15.0
Integral of MAP(X) curve for worst 1/4 topics	0.0015	0.0129	0.0028

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.2600	0.3360	0.2980
At 10 docs	0.2060	0.2340	0.2200
At 15 docs	0.1973	0.2373	0.2173
At 20 docs	0.1730	0.2110	0.1920
At 30 docs	0.1533	0.2033	0.1783
At 100 docs	0.1188	0.1506	0.1347
At 200 docs	0.0594	0.0753	0.0674
At 500 docs	0.0238	0.0301	0.0269
At 1000 docs	0.0119	0.0151	0.0135
R-Precision			
Exact	0.1337	0.2083	0.1710



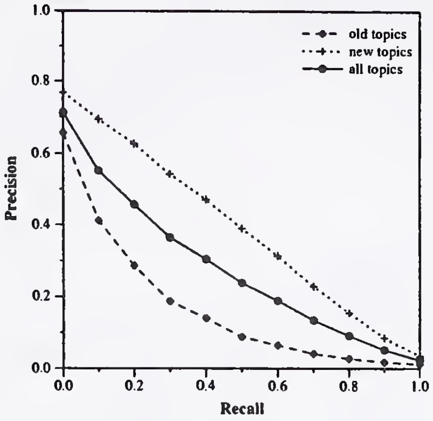


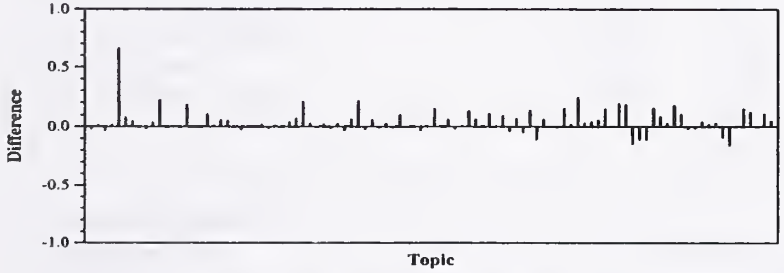
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	humR03de
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1709	1336	3045
Mean average precision	0.1481	0.3772	0.2627
% topics with no relevant in top 10	24.0	14.0	19.0
Integral of MAP(X) curve for worst 1/4 topics	0.0044	0.0226	0.0089

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3880	0.5720	0.4800
At 10 docs	0.3520	0.4840	0.4180
At 15 docs	0.3053	0.4347	0.3700
At 20 docs	0.2810	0.3990	0.3400
At 30 docs	0.2447	0.3273	0.2860
At 100 docs	0.1452	0.1588	0.1520
At 200 docs	0.1009	0.0988	0.0998
At 500 docs	0.0567	0.0480	0.0523
At 1000 docs	0.0342	0.0267	0.0304
R-Precision			
Exact	0.1956	0.3773	0.2864



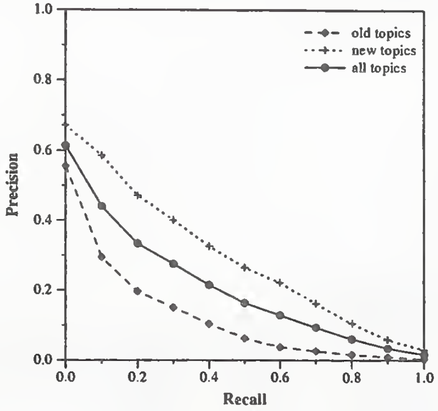


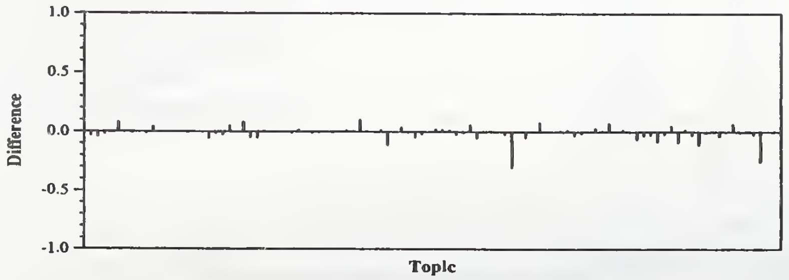
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	humR03t
Run Description	Automatic, title only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	49408	50000	99408
Total relevant	4416	1658	6074
Total relevant retrieved	1877	1383	3260
Mean average precision	0.1091	0.2799	0.1945
% topics with no relevant in top 10	16.0	4.0	10.0
Integral of MAP(X) curve for worst 1/4 topics	0.0077	0.0374	0.0145

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3040	0.4680	0.3860
At 10 docs	0.2880	0.4100	0.3490
At 15 docs	0.2560	0.3547	0.3053
At 20 docs	0.2390	0.3220	0.2805
At 30 docs	0.2167	0.2713	0.2440
At 100 docs	0.1326	0.1476	0.1401
At 200 docs	0.0962	0.0950	0.0956
At 500 docs	0.0596	0.0493	0.0544
At 1000 docs	0.0375	0.0277	0.0326
R-Precision			
Exact	0.1705	0.3061	0.2383



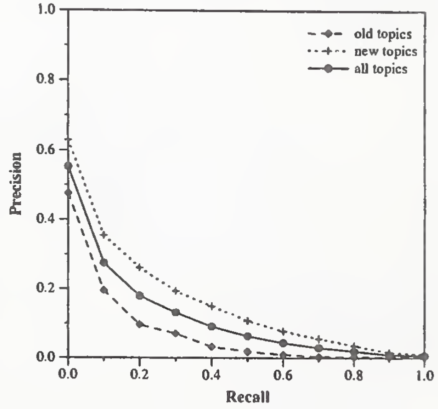


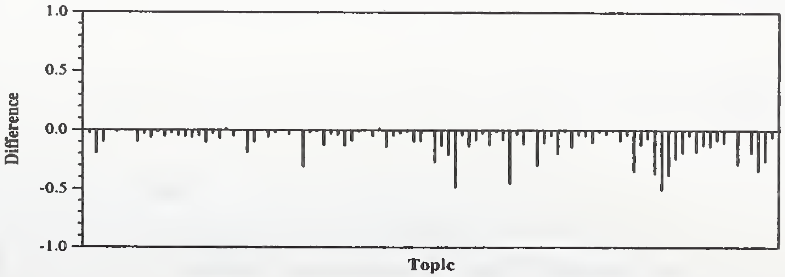
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	humR03tc
Run Description	Automatic, title only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	5000	5000	10000
Total relevant	4416	1658	6074
Total relevant retrieved	663	738	1401
Mean average precision	0.0570	0.1471	0.1020
% topics with no relevant in top 10	24.0	16.0	20.0
Integral of MAP(X) curve for worst 1/4 topics	0.0021	0.0153	0.0044

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.2080	0.2840	0.2460
At 10 docs	0.1780	0.2040	0.1910
At 15 docs	0.2040	0.2200	0.2120
At 20 docs	0.1810	0.1970	0.1890
At 30 docs	0.1713	0.1860	0.1787
At 100 docs	0.1326	0.1476	0.1401
At 200 docs	0.0663	0.0738	0.0701
At 500 docs	0.0265	0.0295	0.0280
At 1000 docs	0.0133	0.0148	0.0140
R-Precision			
Exact	0.1257	0.1891	0.1574



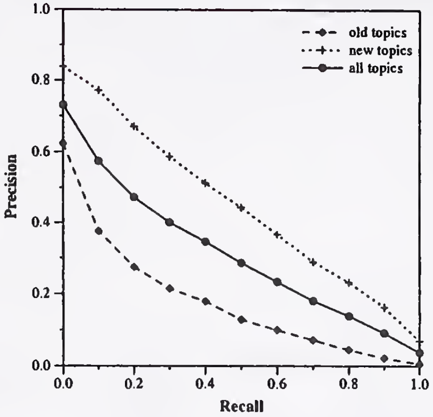


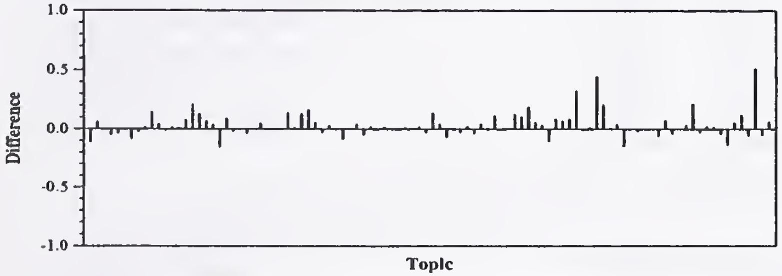
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	aplrob03a
Run Description	Automatic, title+desc+narr
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2679	1488	4167
Mean average precision	0.1628	0.4368	0.2998
% topics with no relevant in top 10	14.0	8.0	11.0
Integral of MAP(X) curve for worst 1/4 topics	0.0163	0.0733	0.0238

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3960	0.6320	0.5140
At 10 docs	0.3500	0.5520	0.4510
At 15 docs	0.3173	0.4867	0.4020
At 20 docs	0.2900	0.4380	0.3640
At 30 docs	0.2560	0.3747	0.3153
At 100 docs	0.1838	0.1890	0.1864
At 200 docs	0.1347	0.1180	0.1264
At 500 docs	0.0811	0.0543	0.0677
At 1000 docs	0.0536	0.0298	0.0417
R-Precision			
Exact	0.2030	0.4146	0.3088



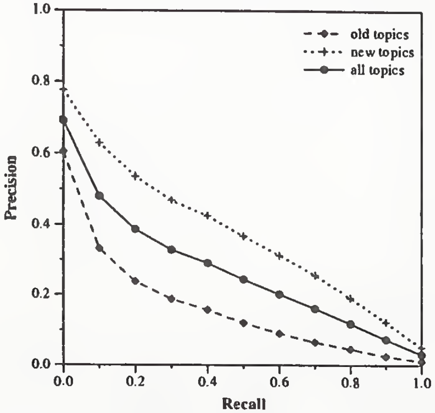


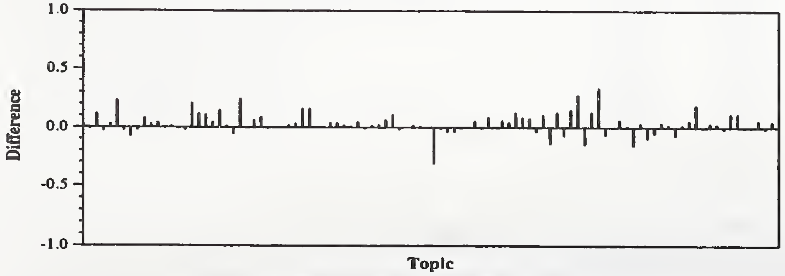
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	aplrob03b
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2276	1475	3751
Mean average precision	0.1466	0.3578	0.2522
% topics with no relevant in top 10	18.0	12.0	15.0
Integral of MAP(X) curve for worst 1/4 topics	0.0068	0.0401	0.0113

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3040	0.4960	0.4000
At 10 docs	0.2720	0.4480	0.3600
At 15 docs	0.2453	0.3840	0.3147
At 20 docs	0.2340	0.3740	0.3040
At 30 docs	0.2207	0.3447	0.2827
At 100 docs	0.1506	0.1736	0.1621
At 200 docs	0.1112	0.1106	0.1109
At 500 docs	0.0698	0.0520	0.0609
At 1000 docs	0.0455	0.0295	0.0375
R-Precision			
Exact	0.1979	0.3651	0.2815



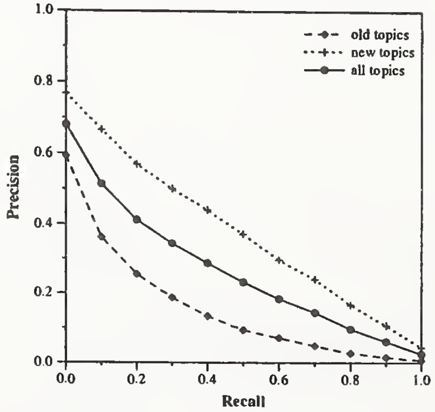


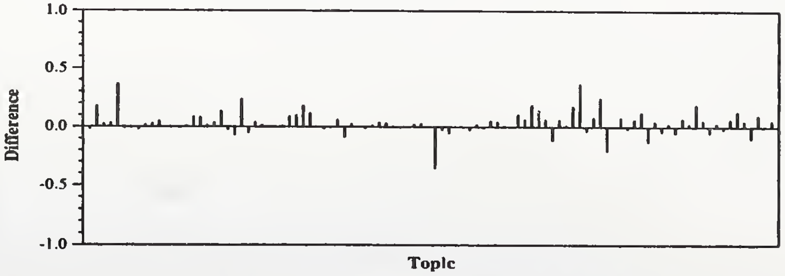
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	aplrob03c
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2115	1451	3566
Mean average precision	0.1400	0.3641	0.2521
% topics with no relevant in top 10	24.0	12.0	18.0
Integral of MAP(X) curve for worst 1/4 topics	0.0081	0.0408	0.0120

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3320	0.5600	0.4460
At 10 docs	0.2800	0.4900	0.3850
At 15 docs	0.2547	0.4373	0.3460
At 20 docs	0.2400	0.3960	0.3180
At 30 docs	0.2160	0.3293	0.2727
At 100 docs	0.1424	0.1678	0.1551
At 200 docs	0.1023	0.1057	0.1040
At 500 docs	0.0643	0.0523	0.0583
At 1000 docs	0.0423	0.0290	0.0357
R-Precision			
Exact	0.1867	0.3704	0.2786



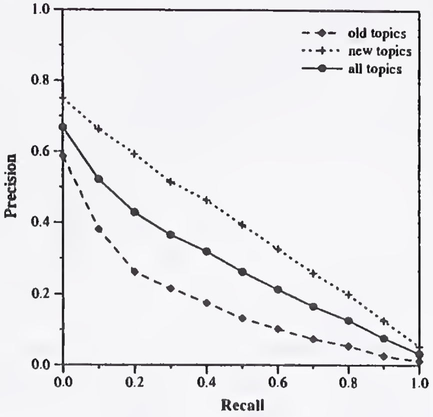


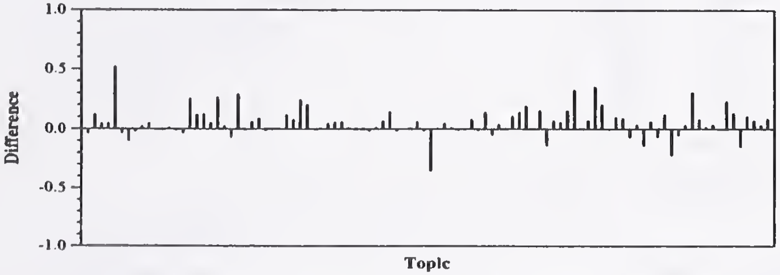
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	aplrob03d
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2276	1475	3751
Mean average precision	0.1619	0.3832	0.2726
% topics with no relevant in top 10	28.0	16.0	22.0
Integral of MAP(X) curve for worst 1/4 topics	0.0055	0.0269	0.0080

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3520	0.5600	0.4560
At 10 docs	0.2900	0.4960	0.3930
At 15 docs	0.2507	0.4427	0.3467
At 20 docs	0.2530	0.4130	0.3330
At 30 docs	0.2280	0.3527	0.2903
At 100 docs	0.1510	0.1732	0.1621
At 200 docs	0.1118	0.1101	0.1110
At 500 docs	0.0700	0.0520	0.0610
At 1000 docs	0.0455	0.0295	0.0375
R-Precision			
Exact	0.1993	0.3737	0.2865



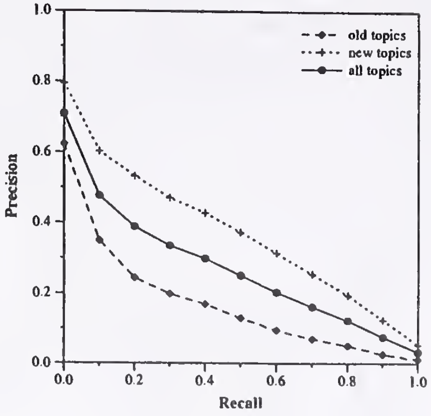


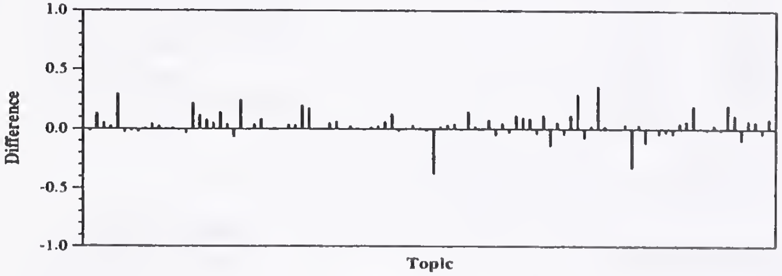
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	aplrob03e
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2276	1475	3751
Mean average precision	0.1497	0.3573	0.2535
% topics with no relevant in top 10	6.0	14.0	10.0
Integral of MAP(X) curve for worst 1/4 topics	0.0067	0.0329	0.0096

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3400	0.4840	0.412C
At 10 docs	0.2800	0.4360	0.3580
At 15 docs	0.2507	0.4187	0.3347
At 20 docs	0.2360	0.3970	0.3165
At 30 docs	0.2253	0.3533	0.2893
At 100 docs	0.1510	0.1740	0.1625
At 200 docs	0.1117	0.1104	0.1111
At 500 docs	0.0699	0.0520	0.0609
At 1000 docs	0.0455	0.0295	0.0375
R-Precision			
Exact	0.1982	0.3693	0.2837



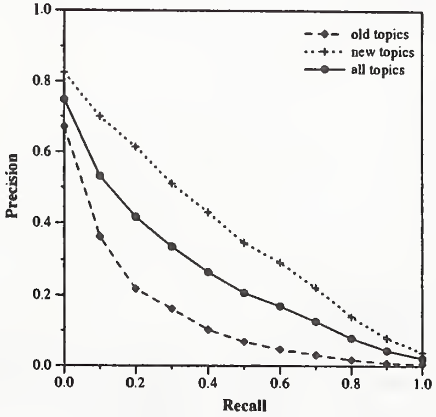


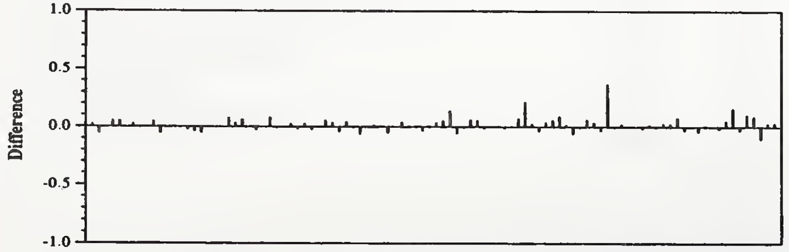
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	oce03Xbm
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2005	1419	3424
Mean average precision	0.1245	0.3646	0.2446
% topics with no relevant in top 10	12.0	10.0	11.0
Integral of MAP(X) curve for worst 1/4 topics	0.0117	0.0352	0.0163

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3800	0.5920	0.4860
At 10 docs	0.3260	0.4960	0.4110
At 15 docs	0.2853	0.4480	0.3667
At 20 docs	0.2630	0.3970	0.3300
At 30 docs	0.2267	0.3193	0.2730
At 100 docs	0.1426	0.1616	0.1521
At 200 docs	0.1025	0.1012	0.1018
At 500 docs	0.0626	0.0518	0.0572
At 1000 docs	0.0401	0.0284	0.0342
R-Precision			
Exact	0.1763	0.3567	0.2665



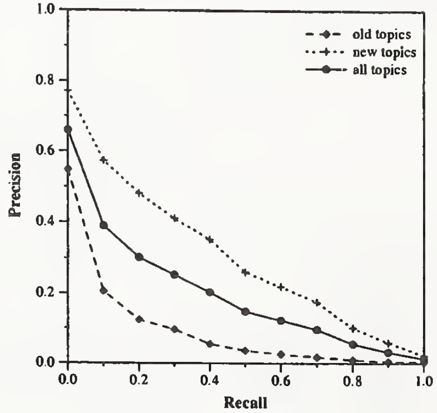


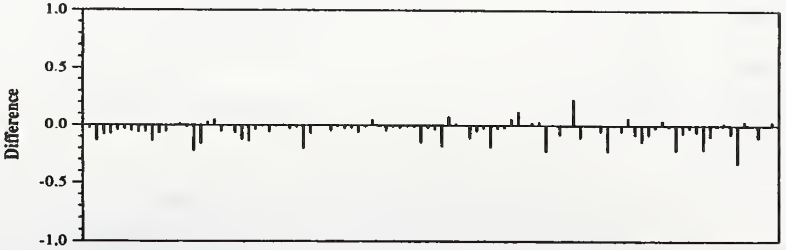
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	oce03Xpr
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1425	1241	2666
Mean average precision	0.0749	0.2921	0.1835
% topics with no relevant in top 10	20.0	12.0	16.0
Integral of MAP(X) curve for worst 1/4 topics	0.0041	0.0145	0.0063

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.2640	0.5040	0.3840
At 10 docs	0.2140	0.3940	0.3040
At 15 docs	0.2000	0.3400	0.2700
At 20 docs	0.1790	0.3110	0.2450
At 30 docs	0.1680	0.2653	0.2167
At 100 docs	0.1048	0.1322	0.1185
At 200 docs	0.0735	0.0852	0.0793
At 500 docs	0.0440	0.0429	0.0435
At 1000 docs	0.0285	0.0248	0.0267
R-Precision			
Exact	0.1312	0.3066	0.2189



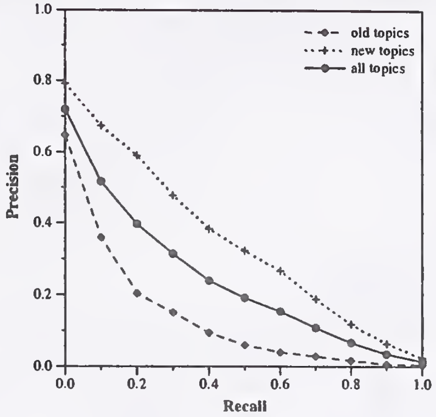


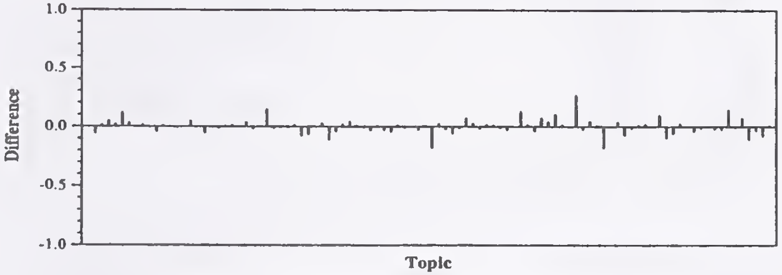
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	oce03noXbm
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1903	1428	3331
Mean average precision	0.1205	0.3379	0.2292
% topics with no relevant in top 10	14.0	6.0	10.0
Integral of MAP(X) curve for worst 1/4 topics	0.0101	0.0406	0.0168

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3760	0.5960	0.4860
At 10 docs	0.2920	0.4780	0.3850
At 15 docs	0.2733	0.4240	0.3487
At 20 docs	0.2580	0.3900	0.3240
At 30 docs	0.2253	0.3087	0.2670
At 100 docs	0.1402	0.1594	0.1498
At 200 docs	0.0993	0.1018	0.1006
At 500 docs	0.0599	0.0515	0.0557
At 1000 docs	0.0381	0.0286	0.0333
R-Precision			
Exact	0.1714	0.3423	0.2568



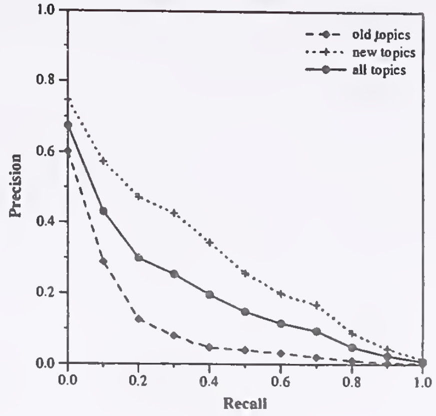


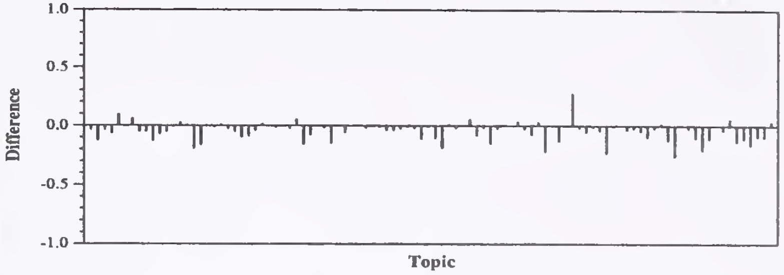
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	oce03noXpr
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1418	1255	2673
Mean average precision	0.0859	0.2846	0.1852
% topics with no relevant in top 10	20.0	10.0	15.0
Integral of MAP(X) curve for worst 1/4 topics	0.0038	0.0180	0.0066

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.2880	0.4920	0.3900
At 10 docs	0.2440	0.4100	0.3270
At 15 docs	0.2147	0.3627	0.2887
At 20 docs	0.1890	0.3210	0.2550
At 30 docs	0.1640	0.2700	0.2170
At 100 docs	0.1034	0.1356	0.1195
At 200 docs	0.0729	0.0844	0.0786
At 500 docs	0.0437	0.0432	0.0435
At 1000 docs	0.0284	0.0251	0.0267
R-Precision			
Exact	0.1363	0.3167	0.2265



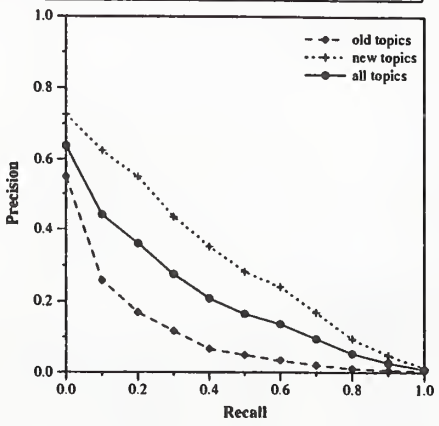


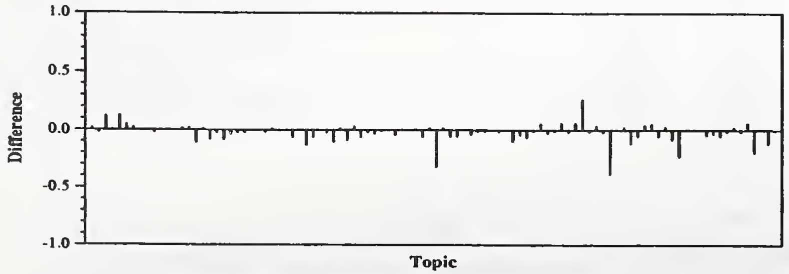
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	oce03noXbmD
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1570	1318	2888
Mean average precision	0.0923	0.3049	0.1986
% topics with no relevant in top 10	24.0	16.0	20.0
Integral of MAP(X) curve for worst 1/4 topics	0.0027	0.0134	0.0055

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.2760	0.5480	0.4120
At 10 docs	0.2400	0.4460	0.3430
At 15 docs	0.2227	0.3947	0.3087
At 20 docs	0.2130	0.3540	0.2835
At 30 docs	0.1833	0.2847	0.2340
At 100 docs	0.1110	0.1442	0.1276
At 200 docs	0.0803	0.0913	0.0858
At 500 docs	0.0478	0.0457	0.0467
At 1000 docs	0.0314	0.0264	0.0289
R-Precision			
Exact	0.1470	0.3159	0.2315



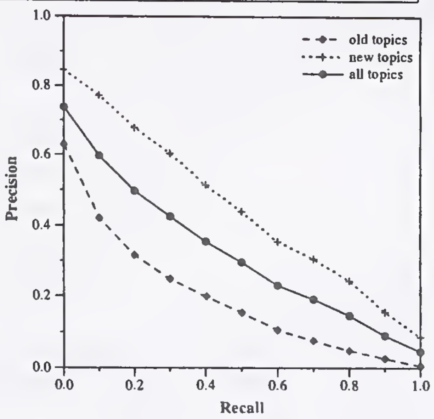


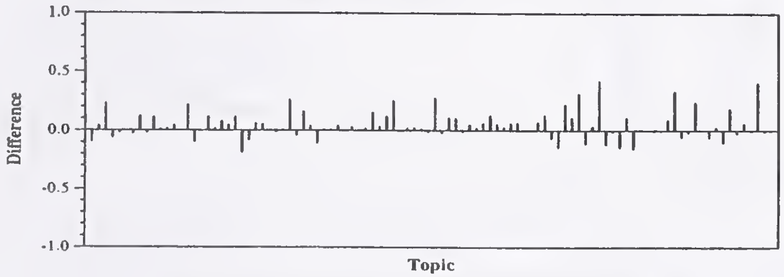
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	pircRBa1
Run Description	Automatic, title+desc+narr
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2562	1494	4056
Mean average precision	0.1796	0.4405	0.3101
% topics with no relevant in top 10	12.0	6.0	9.0
Integral of MAP(X) curve for worst 1/4 topics	0.0136	0.0716	0.0203

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3880	0.6520	0.5200
At 10 docs	0.3640	0.5440	0.4540
At 15 docs	0.3480	0.5013	0.4247
At 20 docs	0.3230	0.4550	0.3890
At 30 docs	0.2867	0.3800	0.3333
At 100 docs	0.1888	0.1922	0.1905
At 200 docs	0.1328	0.1187	0.1258
At 500 docs	0.0790	0.0559	0.0674
At 1000 docs	0.0512	0.0299	0.0406
R-Precision			
Exact	0.2282	0.4150	0.3216



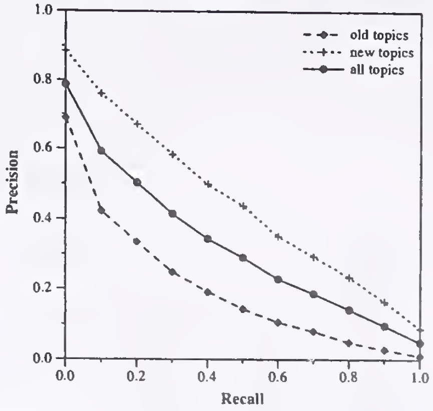


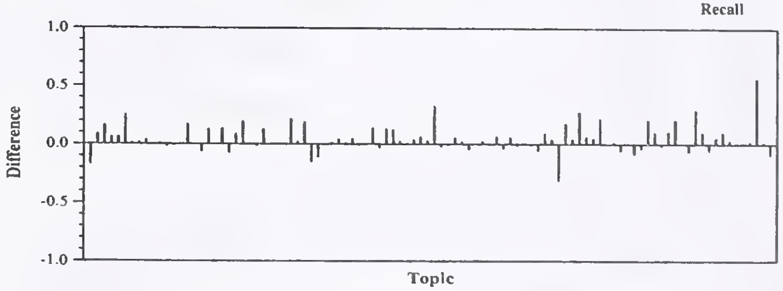
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	pircRBa2
Run Description	Automatic, title+desc+narr
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2641	1576	4217
Mean average precision	0.1854	0.4369	0.3111
% topics with no relevant in top 10	10.0	2.0	6.0
Integral of MAP(X) curve for worst 1/4 topics	0.0135	0.1062	0.0290

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.4080	0.6480	0.5280
At 10 docs	0.4000	0.5760	0.4880
At 15 docs	0.3613	0.4973	0.4293
At 20 docs	0.3340	0.4520	0.3930
At 30 docs	0.2907	0.3760	0.3333
At 100 docs	0.1818	0.1954	0.1886
At 200 docs	0.1297	0.1211	0.1254
At 500 docs	0.0792	0.0575	0.0684
At 1000 docs	0.0528	0.0315	0.0422
R-Precision			
Exact	0.2234	0.4159	0.3197



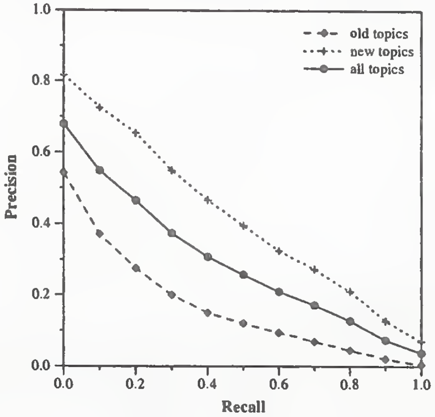


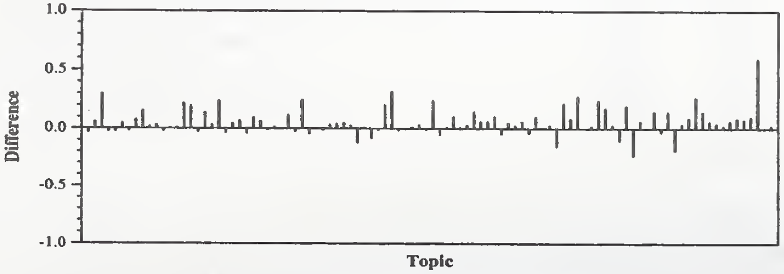
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	pircRBd1
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2216	1534	3750
Mean average precision	0.1526	0.4022	0.2774
% topics with no relevant in top 10	28.0	4.0	16.0
Integral of MAP(X) curve for worst 1/4 topics	0.0045	0.0804	0.0122

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3360	0.6120	0.4740
At 10 docs	0.3220	0.5200	0.4210
At 15 docs	0.3080	0.4640	0.3860
At 20 docs	0.2810	0.4230	0.3520
At 30 docs	0.2393	0.3500	0.2947
At 100 docs	0.1550	0.1836	0.1693
At 200 docs	0.1122	0.1160	0.1141
At 500 docs	0.0675	0.0564	0.0619
At 1000 docs	0.0443	0.0307	0.0375
R-Precision			
Exact	0.1887	0.3963	0.2925



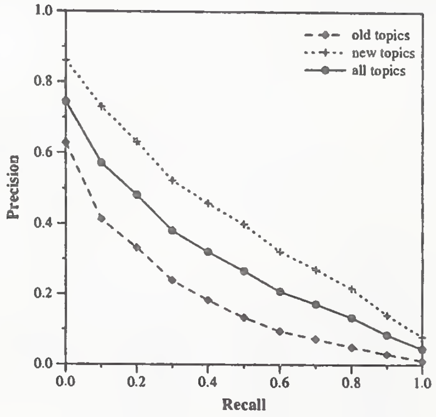


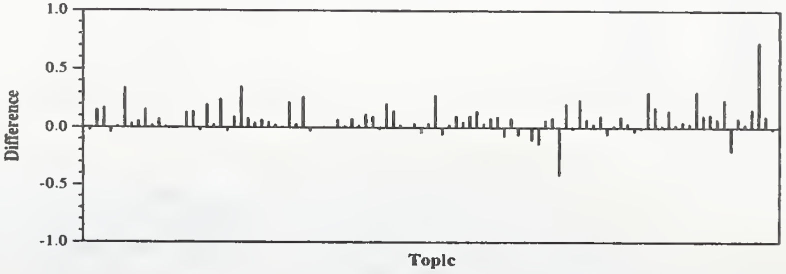
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	pircRBd2
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2377	1565	3942
Mean average precision	0.1772	0.4029	0.2900
% topics with no relevant in top 10	12.0	4.0	8.0
Integral of MAP(X) curve for worst 1/4 topics	0.0091	0.0819	0.0219

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.4200	0.6320	0.5260
At 10 docs	0.3820	0.5320	0.4570
At 15 docs	0.3493	0.4587	0.4040
At 20 docs	0.3240	0.4170	0.3705
At 30 docs	0.2787	0.3467	0.3127
At 100 docs	0.1714	0.1830	0.1772
At 200 docs	0.1204	0.1147	0.1175
At 500 docs	0.0735	0.0567	0.0651
At 1000 docs	0.0475	0.0313	0.0394
R-Precision			
Exact	0.2148	0.3845	0.2996



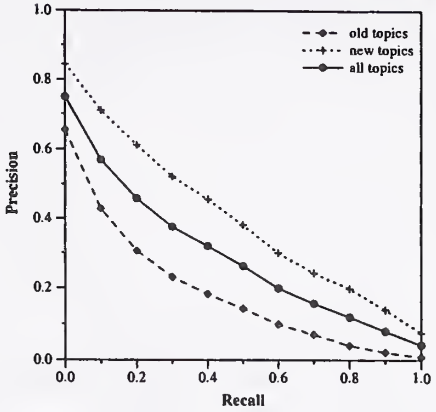


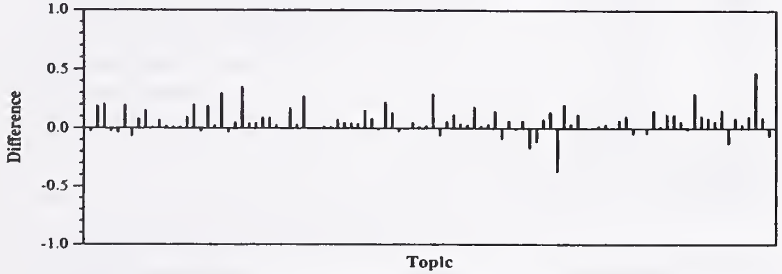
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	pircRBd3
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2287	1444	3731
Mean average precision	0.1754	0.3878	0.2816
% topics with no relevant in top 10	14.0	4.0	9.0
Integral of MAP(X) curve for worst 1/4 topics	0.0065	0.0540	0.0165

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.4240	0.6200	0.5220
At 10 docs	0.3760	0.5220	0.4490
At 15 docs	0.3533	0.4507	0.4020
At 20 docs	0.3250	0.4080	0.3665
At 30 docs	0.2727	0.3273	0.3000
At 100 docs	0.1666	0.1688	0.1677
At 200 docs	0.1208	0.1076	0.1142
At 500 docs	0.0715	0.0520	0.0618
At 1000 docs	0.0457	0.0289	0.0373
R-Precision			
Exact	0.2106	0.3781	0.2944



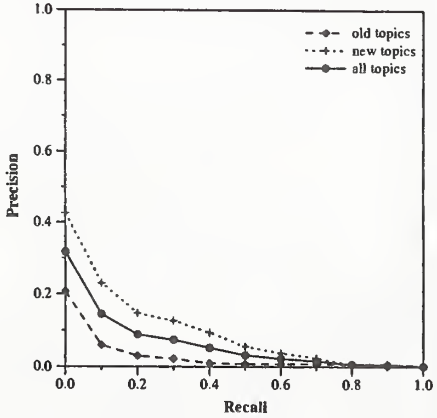


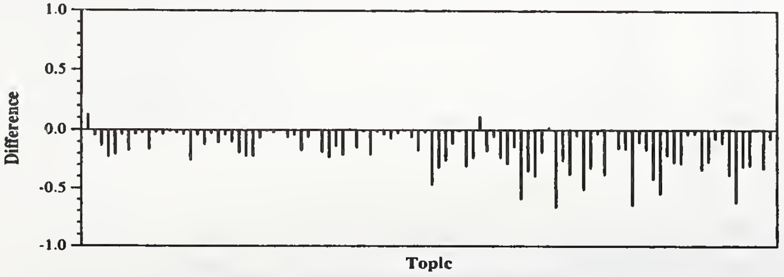
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	rutcor030
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	528	625	1153
Mean average precision	0.0196	0.0858	0.0527
% topics with no relevant in top 10	62.0	46.0	54.0
Integral of MAP(X) curve for worst 1/4 topics	0.0001	0.0002	0.0001

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.0680	0.2200	0.1440
At 10 docs	0.0600	0.1640	0.1120
At 15 docs	0.0533	0.1440	0.0987
At 20 docs	0.0530	0.1260	0.0895
At 30 docs	0.0480	0.1027	0.0753
At 100 docs	0.0330	0.0484	0.0407
At 200 docs	0.0262	0.0340	0.0301
At 500 docs	0.0157	0.0211	0.0184
At 1000 docs	0.0106	0.0125	0.0115
R-Precision			
Exact	0.0448	0.1156	0.0802



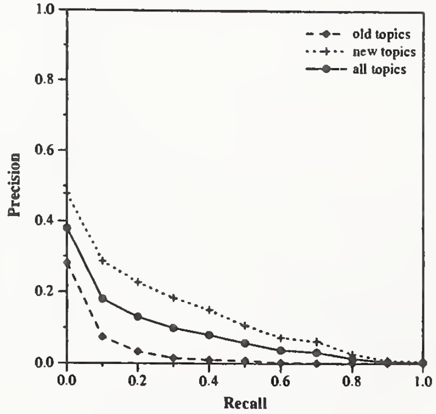


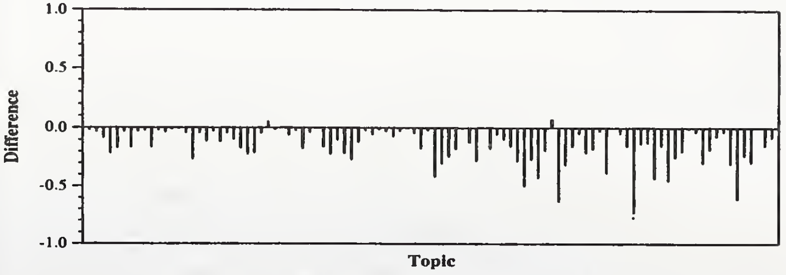
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	rutcor03100
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	734	873	1607
Mean average precision	0.0224	0.1250	0.0737
% topics with no relevant in top 10	40.0	34.0	37.0
Integral of MAP(X) curve for worst 1/4 topics	0.0002	0.0023	0.0005

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.1160	0.2640	0.1900
At 10 docs	0.1040	0.2120	0.1580
At 15 docs	0.0867	0.1973	0.1420
At 20 docs	0.0750	0.1750	0.1250
At 30 docs	0.0660	0.1413	0.1037
At 100 docs	0.0428	0.0774	0.0601
At 200 docs	0.0311	0.0492	0.0401
At 500 docs	0.0222	0.0287	0.0255
At 1000 docs	0.0147	0.0175	0.0161
R-Precision			
Exact	0.0537	0.1653	0.1095



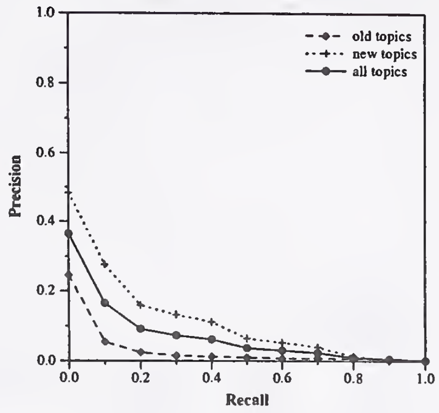


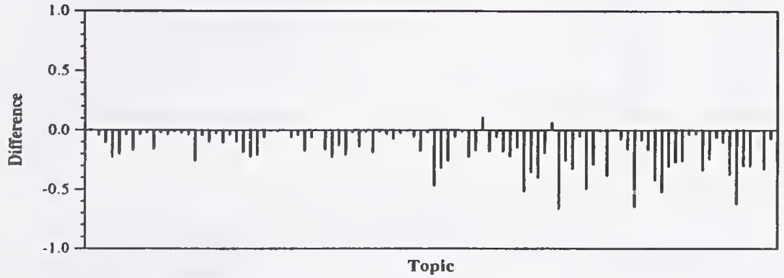
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	rutcor0325
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	639	699	1338
Mean average precision	0.0203	0.1007	0.0605
% topics with no relevant in top 10	52.0	38.0	45.0
Integral of MAP(X) curve for worst 1/4 topics	0.0002	0.0004	0.0002

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.1040	0.2520	0.1780
At 10 docs	0.0780	0.2020	0.1400
At 15 docs	0.0640	0.1627	0.1133
At 20 docs	0.0590	0.1380	0.0985
At 30 docs	0.0547	0.1120	0.0833
At 100 docs	0.0358	0.0548	0.0453
At 200 docs	0.0294	0.0402	0.0348
At 500 docs	0.0172	0.0236	0.0204
At 1000 docs	0.0128	0.0140	0.0134
R-Precision			
Exact	0.0457	0.1314	0.0886



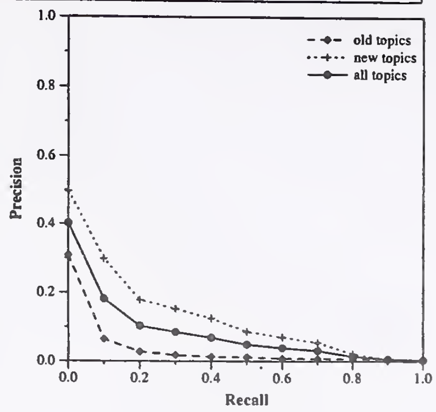


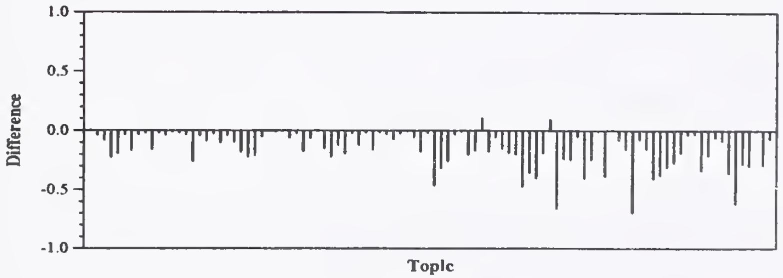
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	rutcor0350
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	692	781	1473
Mean average precision	0.0241	0.1156	0.0698
% topics with no relevant in top 10	46.0	38.0	42.0
Integral of MAP(X) curve for worst 1/4 topics	0.0001	0.0007	0.0003

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.1360	0.2600	0.1980
At 10 docs	0.0920	0.2060	0.1490
At 15 docs	0.0733	0.1720	0.1227
At 20 docs	0.0620	0.1500	0.1060
At 30 docs	0.0613	0.1280	0.0947
At 100 docs	0.0406	0.0664	0.0535
At 200 docs	0.0324	0.0470	0.0397
At 500 docs	0.0191	0.0258	0.0224
At 1000 docs	0.0138	0.0156	0.0147
R-Precision			
Exact	0.0483	0.1414	0.0948





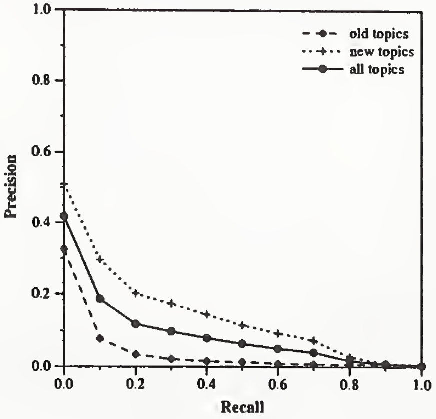
Difference from Median in Average Precision per Topic

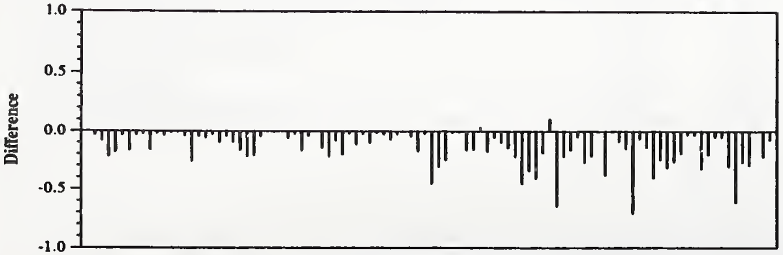
Summary Statistics	
Run ID:	rutcor0375
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	713	869	1582
Mean average precision	0.0277	0.1285	0.0781
% topics with no relevant in top 10	48.0	36.0	42.0
Integral of MAP(X) curve for worst 1/4 topics	0.0002	0.0010	0.0003

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.1400	0.2520	0.1960
At 10 docs	0.1040	0.2080	0.1560
At 15 docs	0.0787	0.1853	0.1320
At 20 docs	0.0710	0.1730	0.1220
At 30 docs	0.0700	0.1407	0.1053
At 100 docs	0.0432	0.0722	0.0577
At 200 docs	0.0338	0.0533	0.0435
At 500 docs	0.0213	0.0291	0.0252
At 1000 docs	0.0143	0.0174	0.0158

R-Precision			
Exact	0.0548	0.1547	0.1047



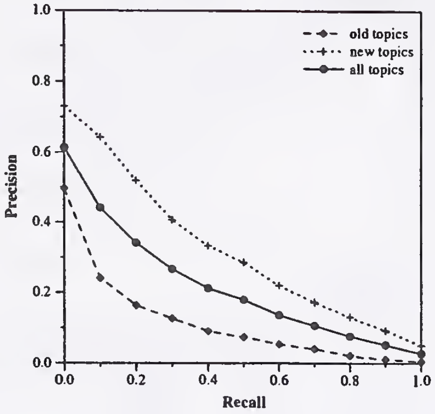


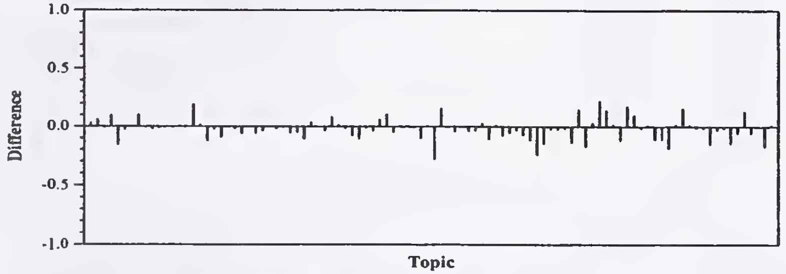
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	SABIR03BASE
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1964	1361	3325
Mean average precision	0.0971	0.3070	0.2021
% topics with no relevant in top 10	26.0	10.0	18.0
Integral of MAP(X) curve for worst 1/4 topics	0.0054	0.0268	0.0084

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.2360	0.4760	0.3560
At 10 docs	0.2240	0.4080	0.3160
At 15 docs	0.1933	0.3560	0.2747
At 20 docs	0.1830	0.3270	0.2550
At 30 docs	0.1720	0.2827	0.2273
At 100 docs	0.1278	0.1494	0.1386
At 200 docs	0.0964	0.0957	0.0960
At 500 docs	0.0600	0.0476	0.0538
At 1000 docs	0.0393	0.0272	0.0332
R-Precision			
Exact	0.1471	0.3120	0.2295



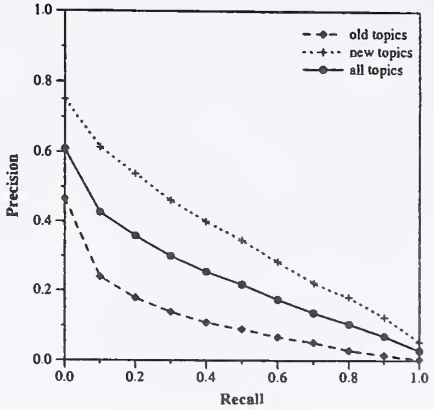


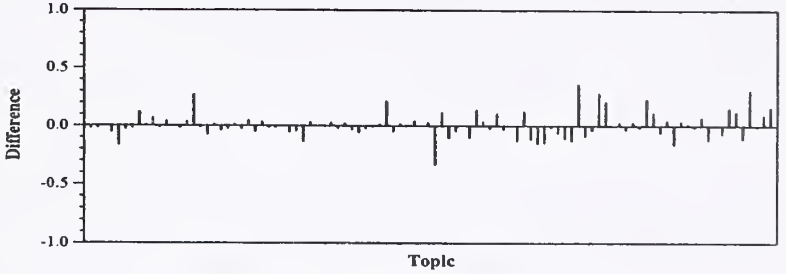
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	SABIR03BF
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2020	1406	3426
Mean average precision	0.1065	0.3461	0.2263
% topics with no relevant in top 10	34.0	12.0	23.0
Integral of MAP(X) curve for worst 1/4 topics	0.0030	0.0303	0.0062

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.2480	0.5160	0.3820
At 10 docs	0.2200	0.4640	0.3420
At 15 docs	0.2133	0.4133	0.3133
At 20 docs	0.2000	0.3770	0.2885
At 30 docs	0.1793	0.3193	0.2493
At 100 docs	0.1278	0.1634	0.1456
At 200 docs	0.1037	0.1041	0.1039
At 500 docs	0.0636	0.0514	0.0575
At 1000 docs	0.0404	0.0281	0.0343
R-Precision			
Exact	0.1437	0.3426	0.2431



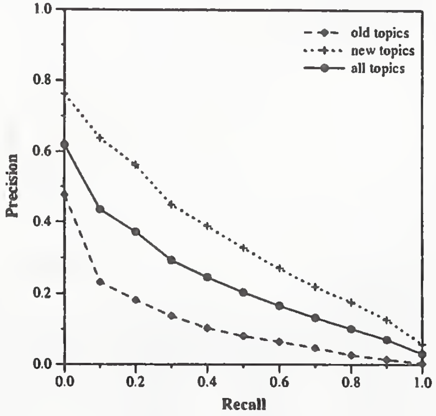


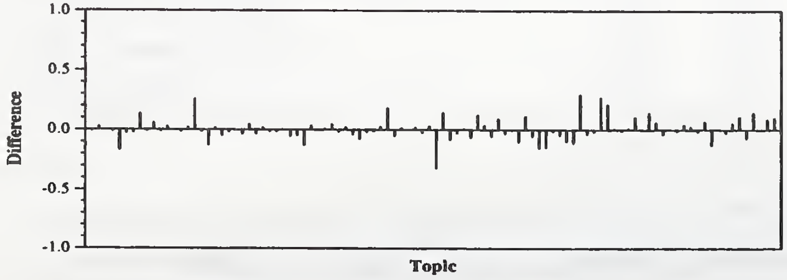
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	SABIR03MERGE
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1954	1403	3357
Mean average precision	0.1044	0.3464	0.2254
% topics with no relevant in top 10	36.0	10.0	23.0
Integral of MAP(X) curve for worst 1/4 topics	0.0037	0.0332	0.0069

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.2440	0.5280	0.3860
At 10 docs	0.2120	0.4500	0.3310
At 15 docs	0.2027	0.4000	0.3013
At 20 docs	0.1990	0.3700	0.2845
At 30 docs	0.1767	0.3187	0.2477
At 100 docs	0.1346	0.1638	0.1492
At 200 docs	0.1012	0.1035	0.1023
At 500 docs	0.0619	0.0507	0.0563
At 1000 docs	0.0391	0.0281	0.0336
R-Precision			
Exact	0.1481	0.3414	0.2447



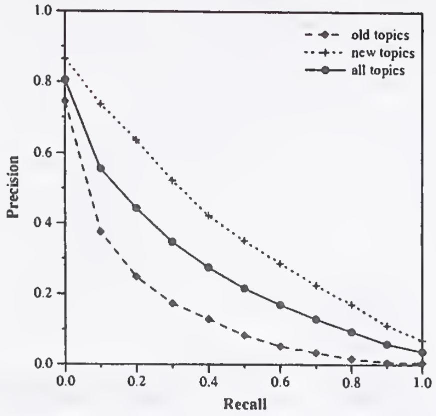


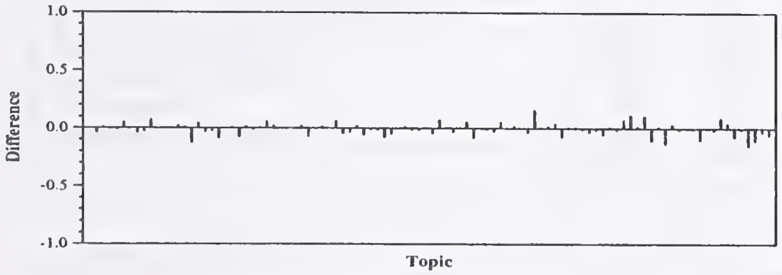
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	THUIRr0301
Run Description	Automatic, title+desc+narr
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2175	1446	3621
Mean average precision	0.1373	0.3821	0.2597
% topics with no relevant in top 10	4.0	4.0	4.0
Integral of MAP(X) curve for worst 1/4 topics	0.0203	0.0570	0.0289

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.4120	0.6360	0.5240
At 10 docs	0.3600	0.5320	0.4460
At 15 docs	0.3133	0.4680	0.3907
At 20 docs	0.2860	0.4170	0.3515
At 30 docs	0.2533	0.3407	0.2970
At 100 docs	0.1636	0.1658	0.1647
At 200 docs	0.1133	0.1027	0.1080
At 500 docs	0.0667	0.0527	0.0597
At 1000 docs	0.0435	0.0289	0.0362
R-Precision			
Exact	0.1947	0.3758	0.2852



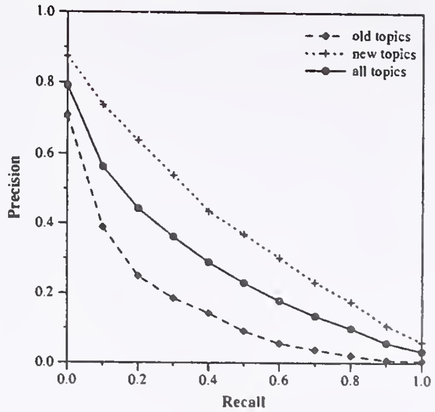


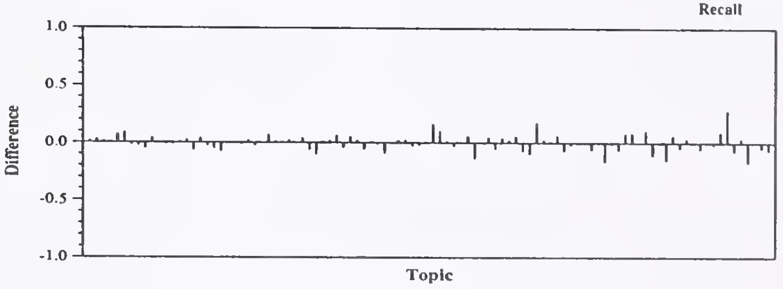
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	THUIRr0302
Run Description	Automatic, title+desc+narr
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2241	1449	3690
Mean average precision	0.1445	0.3888	0.2667
% topics with no relevant in top 10	8.0	4.0	6.0
Integral of MAP(X) curve for worst 1/4 topics	0.0161	0.0565	0.0235

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.4520	0.6440	0.5480
At 10 docs	0.3620	0.5420	0.4520
At 15 docs	0.3093	0.4867	0.3980
At 20 docs	0.2870	0.4230	0.3550
At 30 docs	0.2507	0.3453	0.2980
At 100 docs	0.1642	0.1670	0.1656
At 200 docs	0.1129	0.1067	0.1098
At 500 docs	0.0674	0.0538	0.0606
At 1000 docs	0.0448	0.0290	0.0369
R-Precision			
Exact	0.2034	0.3832	0.2933



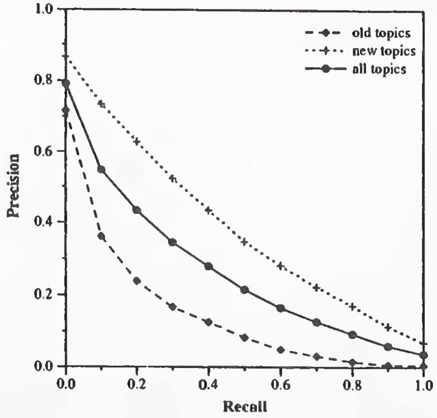


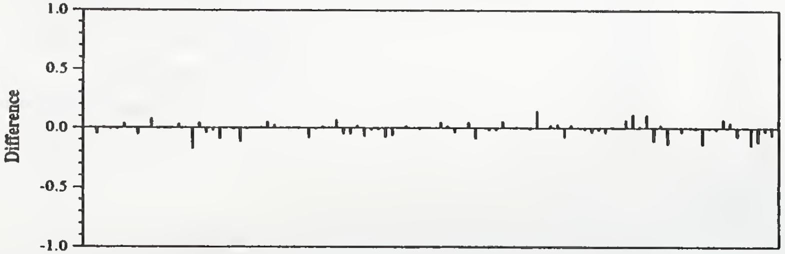
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	THUIRr0303
Run Description	Automatic, title+desc+narr
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2130	1439	3569
Mean average precision	0.1331	0.3810	0.2571
% topics with no relevant in top 10	6.0	6.0	6.0
Integral of MAP(X) curve for worst 1/4 topics	0.0199	0.0561	0.0284

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3880	0.6400	0.5140
At 10 docs	0.3500	0.5300	0.4400
At 15 docs	0.3093	0.4667	0.3880
At 20 docs	0.2770	0.4150	0.3460
At 30 docs	0.2467	0.3407	0.2937
At 100 docs	0.1610	0.1648	0.1629
At 200 docs	0.1119	0.1018	0.1069
At 500 docs	0.0653	0.0519	0.0586
At 1000 docs	0.0426	0.0288	0.0357
R-Precision			
Exact	0.1903	0.3750	0.2826



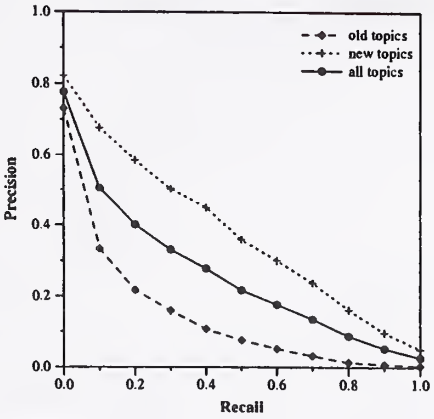


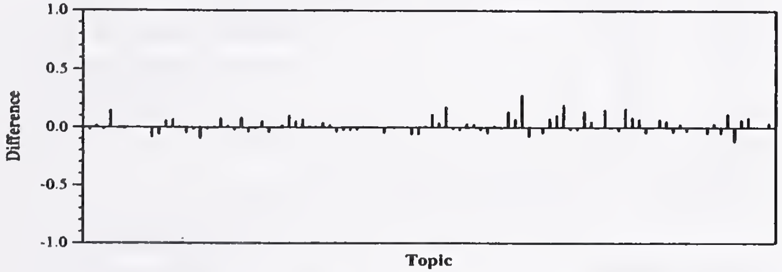
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	THUIRr0304
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	49414	50000	99414
Total relevant	4416	1658	6074
Total relevant retrieved	2025	1422	3447
Mean average precision	0.1237	0.3706	0.2472
% topics with no relevant in top 10	10.0	8.0	9.0
Integral of MAP(X) curve for worst 1/4 topics	0.0152	0.0441	0.0195

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3760	0.5800	0.4780
At 10 docs	0.3120	0.5040	0.4080
At 15 docs	0.2840	0.4453	0.3647
At 20 docs	0.2630	0.4030	0.3330
At 30 docs	0.2300	0.3333	0.2817
At 100 docs	0.1432	0.1636	0.1534
At 200 docs	0.1069	0.1040	0.1055
At 500 docs	0.0638	0.0504	0.0571
At 1000 docs	0.0405	0.0284	0.0345
R-Precision			
Exact	0.1804	0.3726	0.2765



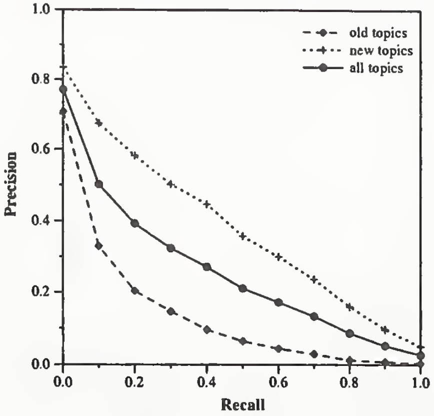


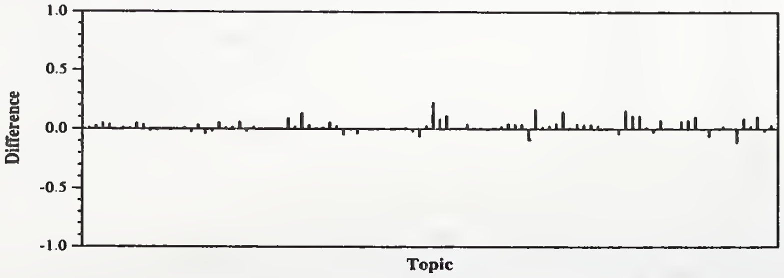
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	THUIRr0305
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1879	1416	3295
Mean average precision	0.1166	0.3702	0.2434
% topics with no relevant in top 10	16.0	8.0	12.0
Integral of MAP(X) curve for worst 1/4 topics	0.0094	0.0475	0.0148

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3720	0.5880	0.4800
At 10 docs	0.3120	0.5080	0.4100
At 15 docs	0.2840	0.4413	0.3627
At 20 docs	0.2580	0.3990	0.3285
At 30 docs	0.2207	0.3307	0.2757
At 100 docs	0.1356	0.1622	0.1489
At 200 docs	0.0979	0.1037	0.1008
At 500 docs	0.0568	0.0504	0.0536
At 1000 docs	0.0376	0.0283	0.0329
R-Precision			
Exact	0.1699	0.3720	0.2710



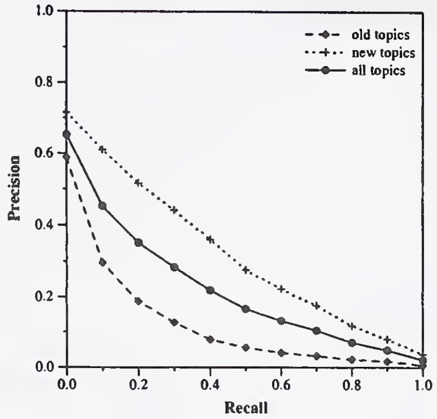


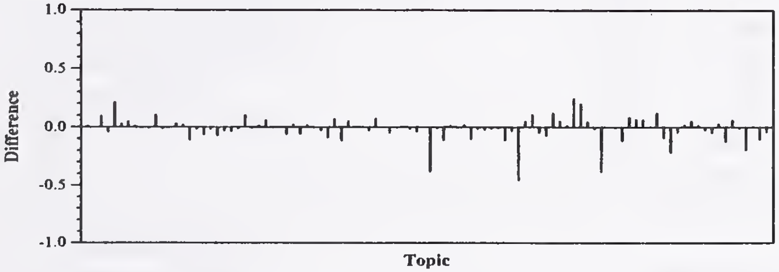
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UAmsT03RDesc
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1679	1286	2965
Mean average precision	0.1066	0.3064	0.2065
% topics with no relevant in top 10	14.0	16.0	15.0
Integral of MAP(X) curve for worst 1/4 topics	0.0064	0.0142	0.0076

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3040	0.5440	0.4240
At 10 docs	0.2640	0.4420	0.3530
At 15 docs	0.2373	0.3960	0.3167
At 20 docs	0.2200	0.3560	0.2880
At 30 docs	0.1940	0.2847	0.2393
At 100 docs	0.1208	0.1420	0.1314
At 200 docs	0.0842	0.0898	0.0870
At 500 docs	0.0512	0.0449	0.0481
At 1000 docs	0.0336	0.0257	0.0296
R-Precision			
Exact	0.1588	0.3203	0.2396



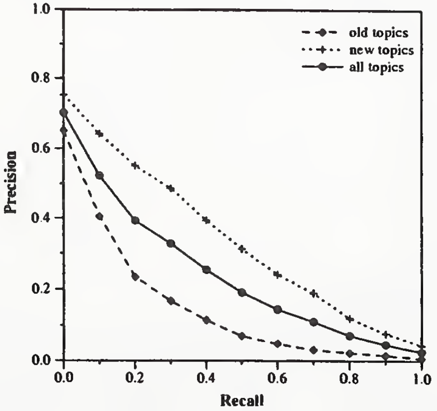


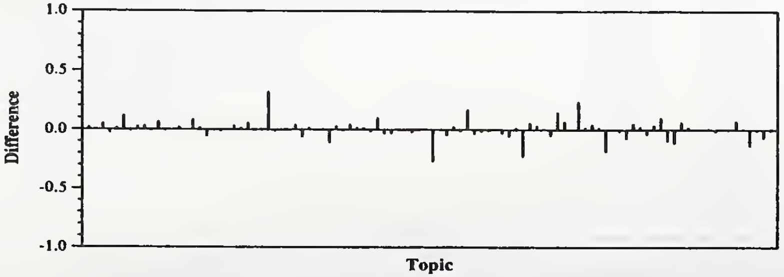
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UAmsT03R
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1994	1396	3390
Mean average precision	0.1349	0.3300	0.2324
% topics with no relevant in top 10	12.0	6.0	9.0
Integral of MAP(X) curve for worst 1/4 topics	0.0142	0.0433	0.0216

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3960	0.5560	0.4760
At 10 docs	0.3180	0.4920	0.4050
At 15 docs	0.2907	0.4280	0.3593
At 20 docs	0.2750	0.3780	0.3265
At 30 docs	0.2347	0.3080	0.2713
At 100 docs	0.1504	0.1538	0.1521
At 200 docs	0.1046	0.0987	0.1017
At 500 docs	0.0631	0.0505	0.0568
At 1000 docs	0.0399	0.0279	0.0339
R-Precision			
Exact	0.1902	0.3453	0.2678



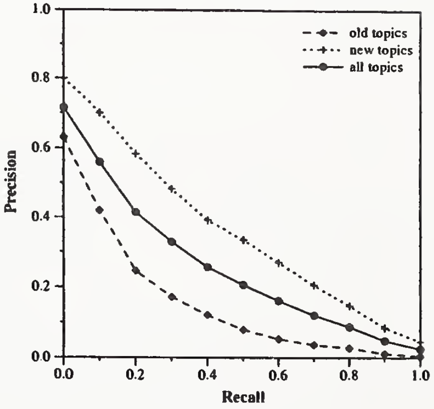


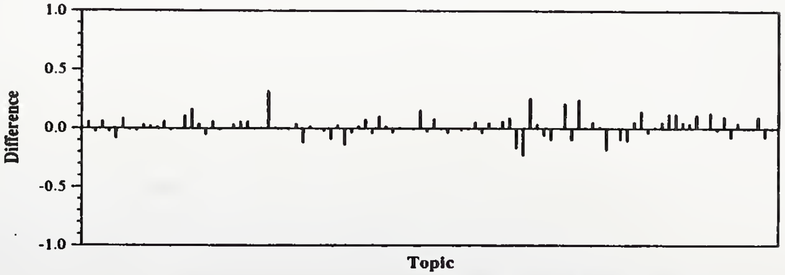
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UAmsT03RFb
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2121	1418	3539
Mean average precision	0.1377	0.3528	0.2452
% topics with no relevant in top 10	16.0	10.0	13.0
Integral of MAP(X) curve for worst 1/4 topics	0.0143	0.0368	0.0210

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3640	0.6080	0.4860
At 10 docs	0.3200	0.5020	0.4110
At 15 docs	0.3027	0.4440	0.3733
At 20 docs	0.2710	0.3860	0.3285
At 30 docs	0.2427	0.3100	0.2763
At 100 docs	0.1554	0.1544	0.1549
At 200 docs	0.1068	0.0993	0.1031
At 500 docs	0.0665	0.0510	0.0588
At 1000 docs	0.0424	0.0284	0.0354
R-Precision			
Exact	0.1846	0.3548	0.2697



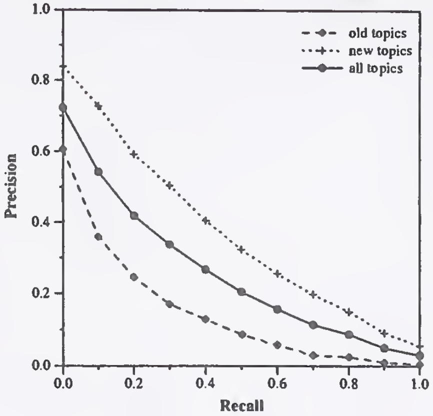


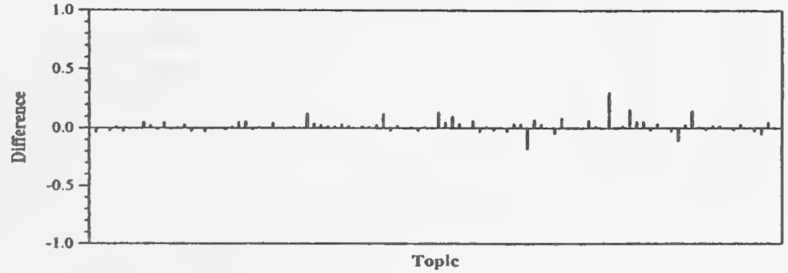
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UAmsT03RSt
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2141	1411	3552
Mean average precision	0.1327	0.3572	0.2450
% topics with no relevant in top 10	6.0	6.0	6.0
Integral of MAP(X) curve for worst 1/4 topics	0.0185	0.0551	0.0256

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3720	0.6120	0.4920
At 10 docs	0.3300	0.5000	0.4150
At 15 docs	0.2947	0.4480	0.3713
At 20 docs	0.2630	0.3920	0.3275
At 30 docs	0.2380	0.3167	0.2773
At 100 docs	0.1576	0.1590	0.1583
At 200 docs	0.1119	0.1030	0.1075
At 500 docs	0.0672	0.0512	0.0592
At 1000 docs	0.0428	0.0282	0.0355
R-Precision			
Exact	0.1900	0.3607	0.2754



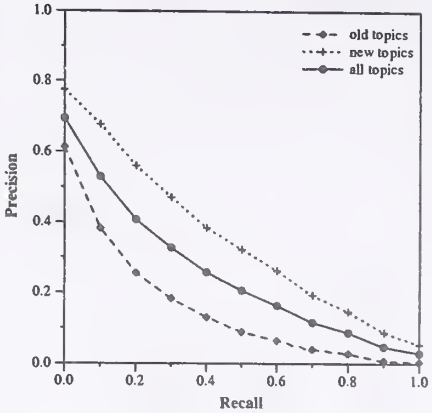


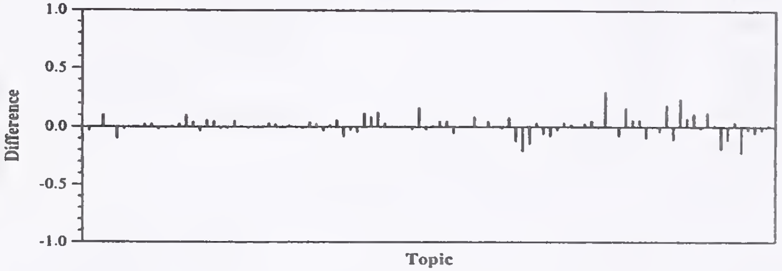
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UAmsT03RStFb
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2267	1407	3674
Mean average precision	0.1361	0.3386	0.2374
% topics with no relevant in top 10	16.0	12.0	14.0
Integral of MAP(X) curve for worst 1/4 topics	0.0204	0.0478	0.0273

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3920	0.5560	0.4740
At 10 docs	0.3300	0.4780	0.4040
At 15 docs	0.2987	0.4053	0.3520
At 20 docs	0.2630	0.3590	0.3110
At 30 docs	0.2427	0.2953	0.2690
At 100 docs	0.1600	0.1582	0.1591
At 200 docs	0.1175	0.0990	0.1082
At 500 docs	0.0715	0.0512	0.0613
At 1000 docs	0.0453	0.0281	0.0367
R-Precision			
Exact	0.1882	0.3516	0.2699



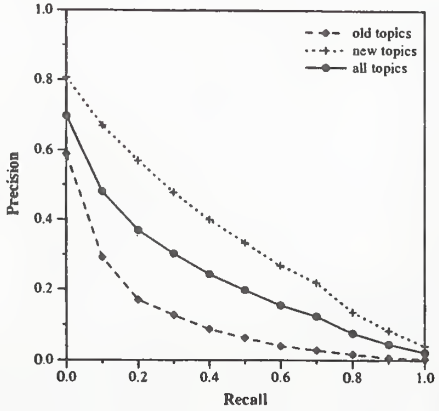


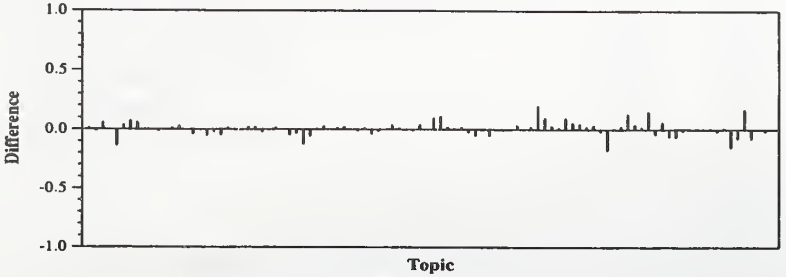
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	InexpC2
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1831	1367	3198
Mean average precision	0.1019	0.3478	0.2249
% topics with no relevant in top 10	20.0	8.0	14.0
Integral of MAP(X) curve for worst 1/4 topics	0.0056	0.0246	0.0094

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3120	0.5680	0.4400
At 10 docs	0.2700	0.4700	0.3700
At 15 docs	0.2640	0.4280	0.3460
At 20 docs	0.2370	0.3830	0.3100
At 30 docs	0.2007	0.3120	0.2563
At 100 docs	0.1292	0.1564	0.1428
At 200 docs	0.0927	0.0975	0.0951
At 500 docs	0.0562	0.0486	0.0524
At 1000 docs	0.0366	0.0273	0.0320
R-Precision			
Exact	0.1536	0.3469	0.2503



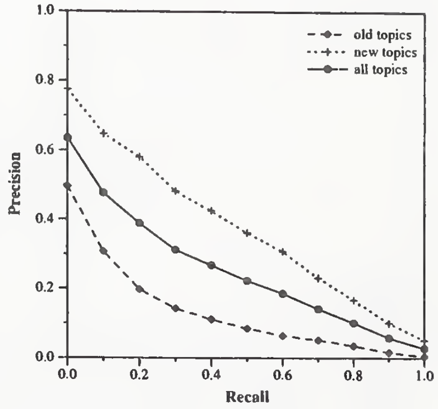


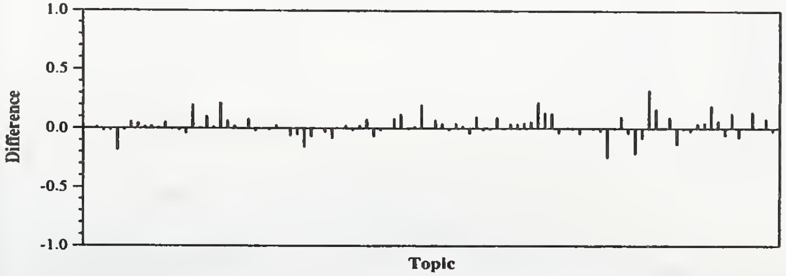
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	InexpC2QE
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2038	1415	3453
Mean average precision	0.1169	0.3600	0.2384
% topics with no relevant in top 10	34.0	14.0	24.0
Integral of MAP(X) curve for worst 1/4 topics	0.0011	0.0191	0.0039

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3280	0.5280	0.4280
At 10 docs	0.2860	0.4780	0.3820
At 15 docs	0.2480	0.4347	0.3413
At 20 docs	0.2240	0.4040	0.3140
At 30 docs	0.2000	0.3300	0.2650
At 100 docs	0.1362	0.1570	0.1466
At 200 docs	0.1020	0.1002	0.1011
At 500 docs	0.0633	0.0498	0.0565
At 1000 docs	0.0408	0.0283	0.0345
R-Precision			
Exact	0.1511	0.3558	0.2534



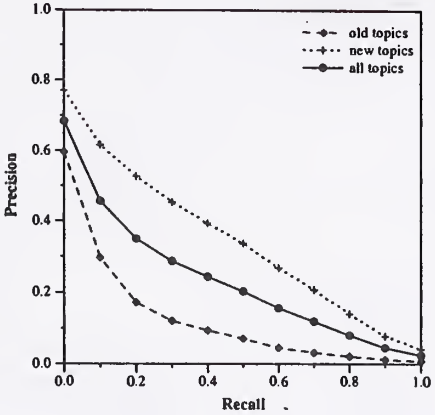


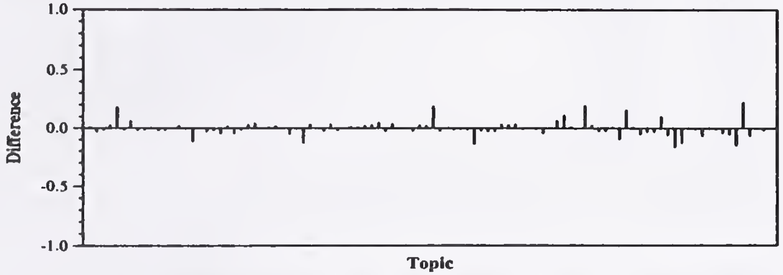
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	Sel50
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1677	1300	2977
Mean average precision	0.1054	0.3327	0.2190
% topics with no relevant in top 10	14.0	8.0	11.0
Integral of MAP(X) curve for worst 1/4 topics	0.0055	0.0162	0.0071

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3320	0.5200	0.4260
At 10 docs	0.2840	0.4440	0.3640
At 15 docs	0.2653	0.3893	0.3273
At 20 docs	0.2370	0.3480	0.2925
At 30 docs	0.2007	0.2933	0.2470
At 100 docs	0.1270	0.1470	0.1370
At 200 docs	0.0890	0.0912	0.0901
At 500 docs	0.0514	0.0450	0.0482
At 1000 docs	0.0335	0.0260	0.0298
R-Precision			
Exact	0.1617	0.3459	0.2538



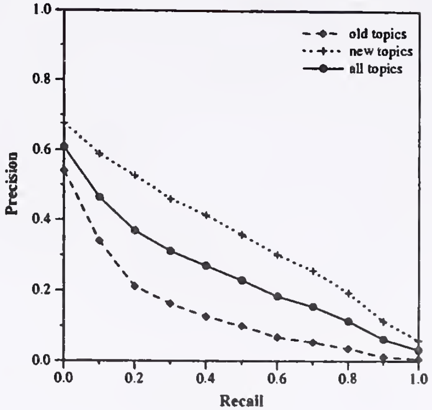


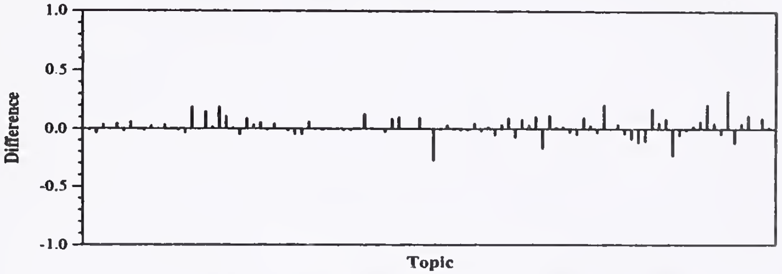
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	Sel50QE
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2100	1314	3414
Mean average precision	0.1295	0.3480	0.2387
% topics with no relevant in top 10	26.0	16.0	21.0
Integral of MAP(X) curve for worst 1/4 topics	0.0045	0.0053	0.0037

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3240	0.4960	0.4100
At 10 docs	0.2920	0.4480	0.3700
At 15 docs	0.2733	0.4067	0.3400
At 20 docs	0.2500	0.3790	0.3145
At 30 docs	0.2247	0.3047	0.2647
At 100 docs	0.1448	0.1482	0.1465
At 200 docs	0.1075	0.0913	0.0994
At 500 docs	0.0647	0.0468	0.0558
At 1000 docs	0.0420	0.0263	0.0341
R-Precision			
Exact	0.1757	0.3408	0.2582



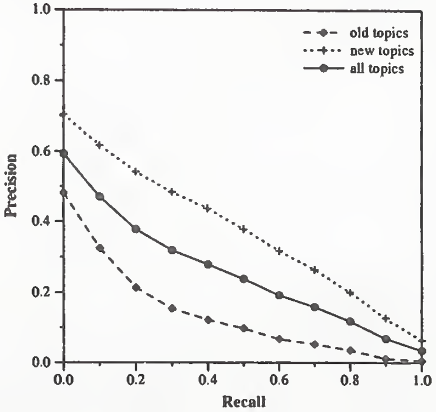


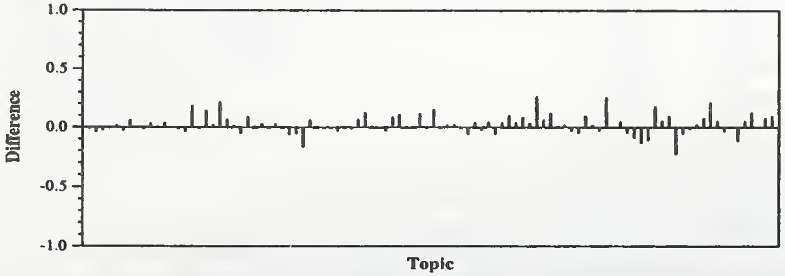
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	Sel78QE
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2082	1352	3434
Mean average precision	0.1238	0.3626	0.2432
% topics with no relevant in top 10	32.0	18.0	25.0
Integral of MAP(X) curve for worst 1/4 topics	0.0032	0.0071	0.0030

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3160	0.5040	0.4100
At 10 docs	0.2920	0.4520	0.3720
At 15 docs	0.2667	0.4027	0.3347
At 20 docs	0.2370	0.3810	0.3090
At 30 docs	0.2120	0.3113	0.2617
At 100 docs	0.1410	0.1542	0.1476
At 200 docs	0.1037	0.0952	0.0994
At 500 docs	0.0637	0.0486	0.0561
At 1000 docs	0.0416	0.0270	0.0343
R-Precision			
Exact	0.1655	0.3572	0.2613



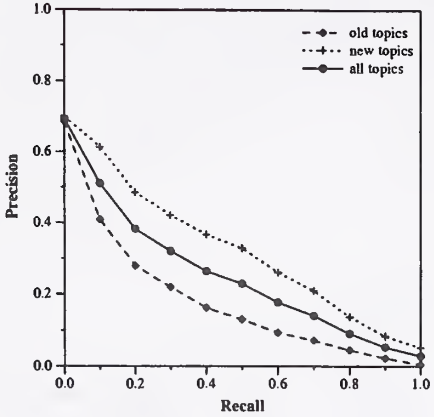


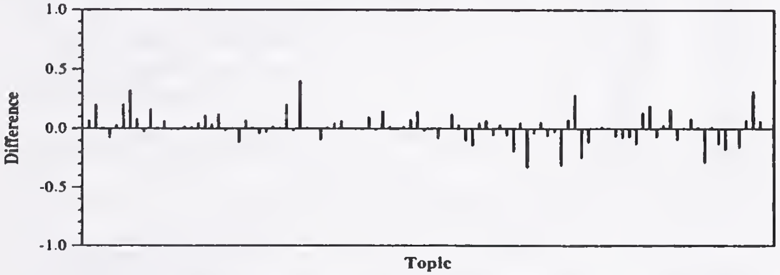
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	uic0301
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	49981	49988	99969
Total relevant	4416	1658	6074
Total relevant retrieved	2283	1424	3707
Mean average precision	0.1674	0.3133	0.2404
% topics with no relevant in top 10	16.0	10.0	13.0
Integral of MAP(X) curve for worst 1/4 topics	0.0076	0.0409	0.0141

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.4280	0.4920	0.4600
At 10 docs	0.3420	0.4380	0.3900
At 15 docs	0.3013	0.3933	0.3473
At 20 docs	0.2770	0.3540	0.3155
At 30 docs	0.2527	0.3060	0.2793
At 100 docs	0.1672	0.1614	0.1643
At 200 docs	0.1205	0.1016	0.1111
At 500 docs	0.0715	0.0517	0.0616
At 1000 docs	0.0457	0.0285	0.0371
R-Precision			
Exact	0.2017	0.3332	0.2675



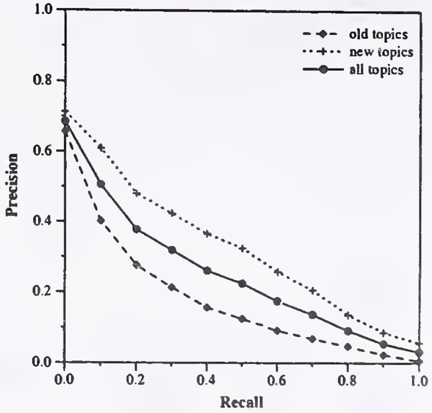


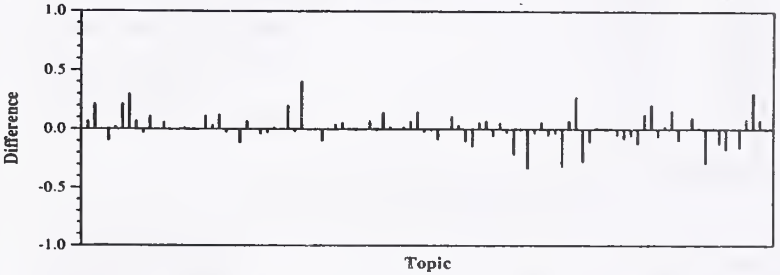
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	uic0303
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	49982	49989	99971
Total relevant	4416	1658	6074
Total relevant retrieved	2245	1425	3670
Mean average precision	0.1622	0.3125	0.2373
% topics with no relevant in top 10	20.0	12.0	16.0
Integral of MAP(X) curve for worst 1/4 topics	0.0065	0.0402	0.0126

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.4080	0.4880	0.4480
At 10 docs	0.3280	0.4340	0.3810
At 15 docs	0.2973	0.3853	0.3413
At 20 docs	0.2750	0.3580	0.3165
At 30 docs	0.2513	0.3047	0.2780
At 100 docs	0.1614	0.1618	0.1616
At 200 docs	0.1168	0.1013	0.1091
At 500 docs	0.0702	0.0520	0.0611
At 1000 docs	0.0449	0.0285	0.0367
R-Precision			
Exact	0.1968	0.3262	0.2615



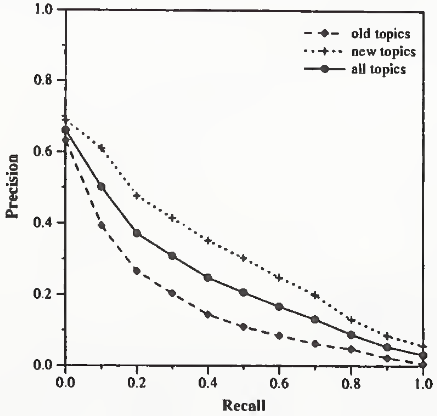


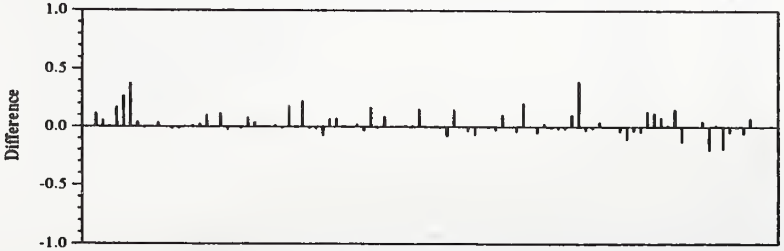
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	uic0302
Run Description	Automatic, title only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	49382	49988	99370
Total relevant	4416	1658	6074
Total relevant retrieved	2136	1416	3552
Mean average precision	0.1548	0.3037	0.2293
% topics with no relevant in top 10	20.0	12.0	16.0
Integral of MAP(X) curve for worst 1/4 topics	0.0052	0.0354	0.0114

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3760	0.4760	0.4260
At 10 docs	0.3120	0.4180	0.3650
At 15 docs	0.2907	0.3760	0.3333
At 20 docs	0.2640	0.3400	0.3020
At 30 docs	0.2420	0.2987	0.2703
At 100 docs	0.1570	0.1566	0.1568
At 200 docs	0.1115	0.1006	0.1061
At 500 docs	0.0671	0.0510	0.0590
At 1000 docs	0.0427	0.0283	0.0355
R-Precision			
Exact	0.1915	0.3202	0.2559



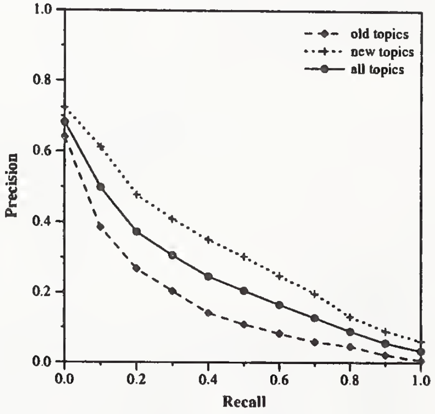


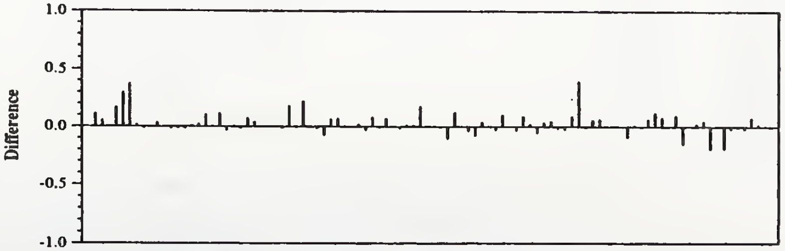
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	uic0304
Run Description	Automatic, title only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	49383	49988	99371
Total relevant	4416	1658	6074
Total relevant retrieved	2081	1415	3496
Mean average precision	0.1527	0.3065	0.2296
% topics with no relevant in top 10	22.0	12.0	17.0
Integral of MAP(X) curve for worst 1/4 topics	0.0050	0.0387	0.0107

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3600	0.5000	0.4300
At 10 docs	0.3080	0.4280	0.3680
At 15 docs	0.2867	0.3813	0.3340
At 20 docs	0.2610	0.3530	0.3070
At 30 docs	0.2360	0.3013	0.2687
At 100 docs	0.1574	0.1560	0.1567
At 200 docs	0.1111	0.1016	0.1064
At 500 docs	0.0661	0.0521	0.0591
At 1000 docs	0.0416	0.0283	0.0350
R-Precision			
Exact	0.1912	0.3214	0.2563



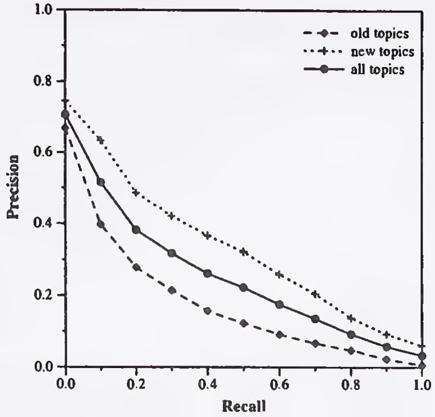


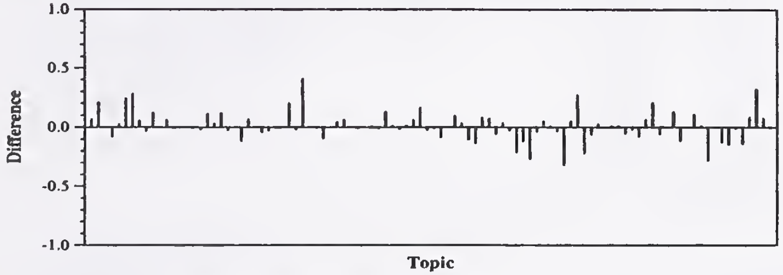
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	uic0305
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	49981	49990	99971
Total relevant	4416	1658	6074
Total relevant retrieved	2196	1430	3626
Mean average precision	0.1608	0.3172	0.2390
% topics with no relevant in top 10	20.0	10.0	15.0
Integral of MAP(X) curve for worst 1/4 topics	0.0061	0.0452	0.0124

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3880	0.5080	0.4480
At 10 docs	0.3300	0.4520	0.3910
At 15 docs	0.2907	0.3933	0.3420
At 20 docs	0.2670	0.3650	0.3160
At 30 docs	0.2507	0.3040	0.2773
At 100 docs	0.1610	0.1620	0.1615
At 200 docs	0.1169	0.1030	0.1099
At 500 docs	0.0687	0.0531	0.0609
At 1000 docs	0.0439	0.0286	0.0363
R-Precision			
Exact	0.1965	0.3351	0.2658



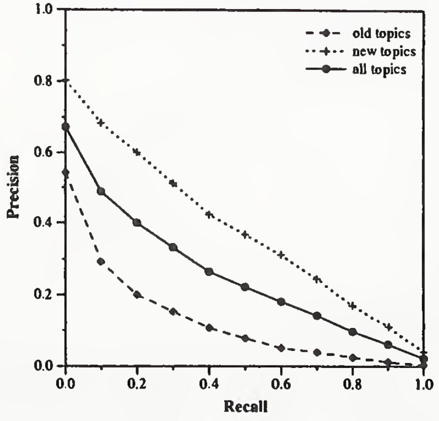


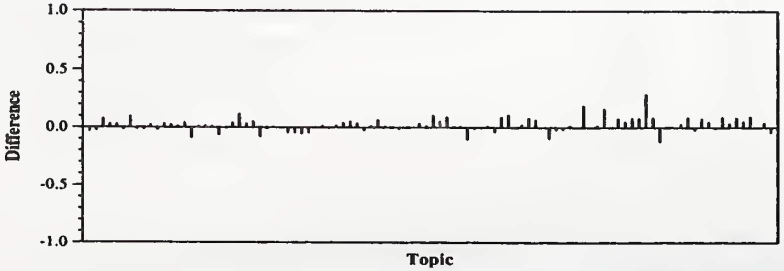
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UIUC03Rd1
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1827	1404	3231
Mean average precision	0.1132	0.3716	0.2424
% topics with no relevant in top 10	26.0	12.0	19.0
Integral of MAP(X) curve for worst 1/4 topics	0.0031	0.0207	0.0065

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.2800	0.5640	0.4220
At 10 docs	0.2660	0.4940	0.3800
At 15 docs	0.2467	0.4320	0.3393
At 20 docs	0.2350	0.3980	0.3165
At 30 docs	0.2053	0.3260	0.2657
At 100 docs	0.1242	0.1680	0.1461
At 200 docs	0.0918	0.1030	0.0974
At 500 docs	0.0562	0.0509	0.0536
At 1000 docs	0.0365	0.0281	0.0323
R-Precision			
Exact	0.1626	0.3614	0.2620



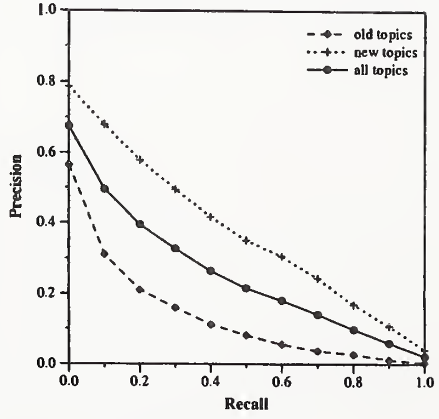


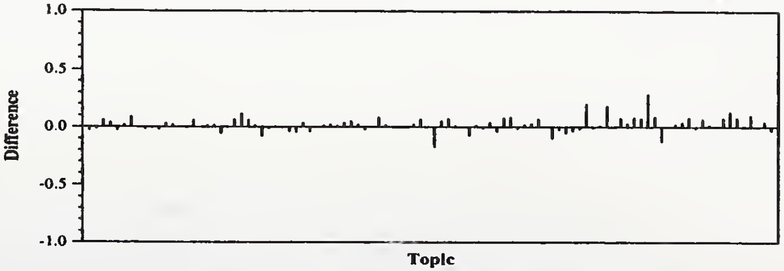
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UIUC03Rd2
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1892	1402	3294
Mean average precision	0.1178	0.3639	0.2408
% topics with no relevant in top 10	24.0	16.0	20.0
Integral of MAP(X) curve for worst 1/4 topics	0.0036	0.0193	0.0069

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3160	0.5760	0.4460
At 10 docs	0.2800	0.4860	0.3830
At 15 docs	0.2573	0.4307	0.3440
At 20 docs	0.2350	0.3910	0.3130
At 30 docs	0.2100	0.3253	0.2677
At 100 docs	0.1286	0.1678	0.1482
At 200 docs	0.0952	0.1030	0.0991
At 500 docs	0.0579	0.0510	0.0545
At 1000 docs	0.0378	0.0280	0.0329
R-Precision			
Exact	0.1681	0.3598	0.2639



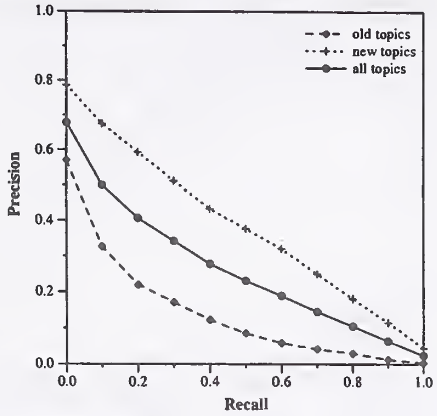


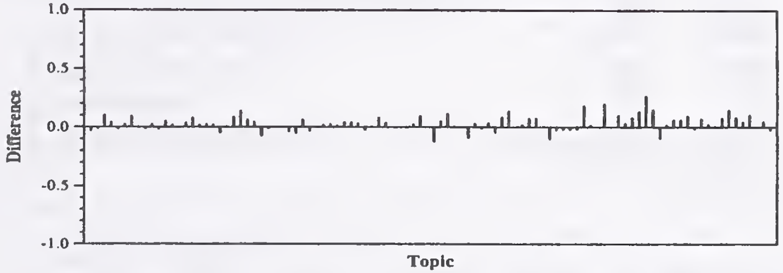
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UIUC03Rd3
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1933	1409	3342
Mean average precision	0.1250	0.3753	0.2502
% topics with no relevant in top 10	26.0	16.0	21.0
Integral of MAP(X) curve for worst 1/4 topics	0.0034	0.0189	0.0065

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3240	0.5720	0.4480
At 10 docs	0.2820	0.4980	0.3900
At 15 docs	0.2707	0.4400	0.3553
At 20 docs	0.2460	0.3980	0.3220
At 30 docs	0.2207	0.3320	0.2763
At 100 docs	0.1356	0.1708	0.1532
At 200 docs	0.0985	0.1050	0.1017
At 500 docs	0.0600	0.0512	0.0556
At 1000 docs	0.0387	0.0282	0.0334
R-Precision			
Exact	0.1751	0.3651	0.2701



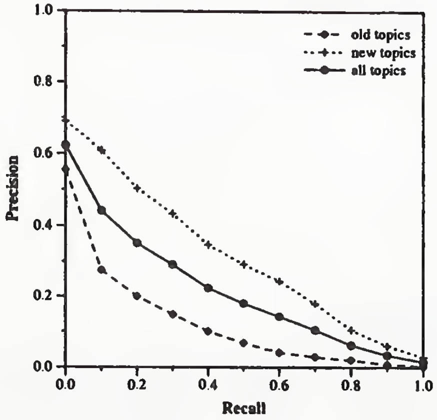


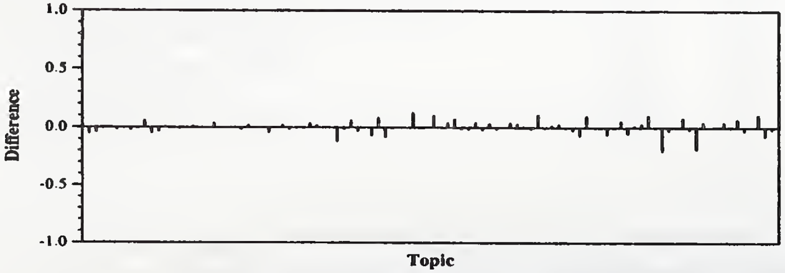
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UIUC03Rt1
Run Description	Automatic, title only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2034	1392	3426
Mean average precision	0.1079	0.3025	0.2052
% topics with no relevant in top 10	18.0	6.0	12.0
Integral of MAP(X) curve for worst 1/4 topics	0.0057	0.0240	0.0113

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3080	0.5000	0.4040
At 10 docs	0.2680	0.4300	0.3490
At 15 docs	0.2427	0.3787	0.3107
At 20 docs	0.2280	0.3390	0.2835
At 30 docs	0.2060	0.2827	0.2443
At 100 docs	0.1386	0.1526	0.1456
At 200 docs	0.1024	0.0989	0.1006
At 500 docs	0.0631	0.0502	0.0566
At 1000 docs	0.0407	0.0278	0.0343
R-Precision			
Exact	0.1649	0.3139	0.2394



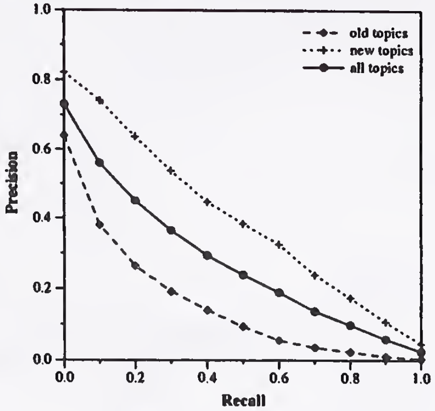


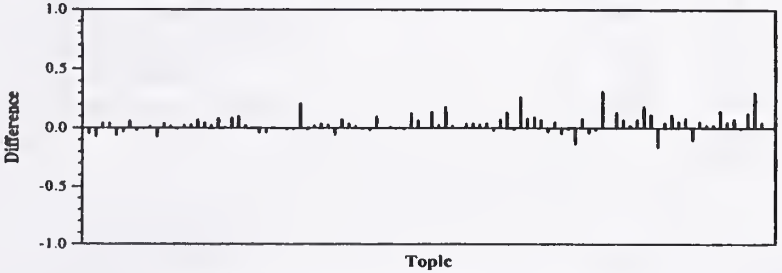
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	UIUC03Rtd1
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2178	1476	3654
Mean average precision	0.1398	0.3922	0.2660
% topics with no relevant in top 10	12.0	10.0	11.0
Integral of MAP(X) curve for worst 1/4 topics	0.0080	0.0466	0.0158

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3800	0.6360	0.5080
At 10 docs	0.3260	0.5280	0.4270
At 15 docs	0.3040	0.4640	0.3840
At 20 docs	0.2830	0.4150	0.3490
At 30 docs	0.2487	0.3533	0.3010
At 100 docs	0.1552	0.1794	0.1673
At 200 docs	0.1127	0.1120	0.1124
At 500 docs	0.0681	0.0547	0.0614
At 1000 docs	0.0436	0.0295	0.0365
R-Precision			
Exact	0.1924	0.3823	0.2873



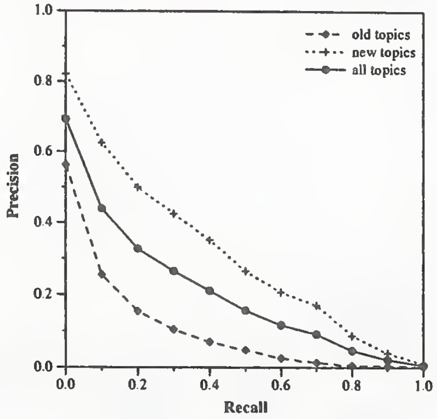


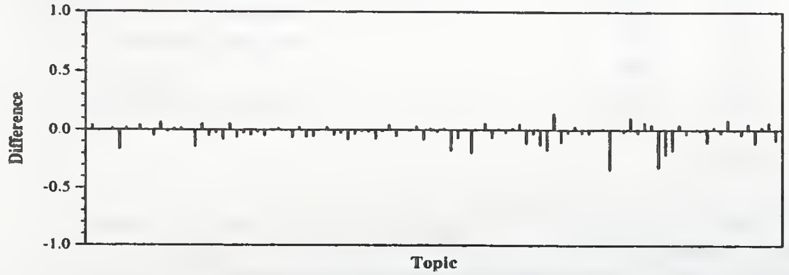
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	MU03rob01
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1639	1248	2887
Mean average precision	0.0886	0.2972	0.1929
% topics with no relevant in top 10	18.0	10.0	14.0
Integral of MAP(X) curve for worst 1/4 topics	0.0041	0.0205	0.0090

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.2880	0.5600	0.4240
At 10 docs	0.2680	0.4480	0.3580
At 15 docs	0.2333	0.3813	0.3073
At 20 docs	0.2150	0.3320	0.2735
At 30 docs	0.1887	0.2713	0.2300
At 100 docs	0.1146	0.1352	0.1249
At 200 docs	0.0817	0.0851	0.0834
At 500 docs	0.0498	0.0435	0.0466
At 1000 docs	0.0328	0.0250	0.0289
R-Precision			
Exact	0.1418	0.3214	0.2316



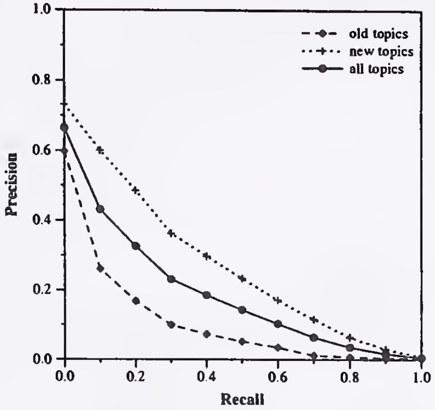


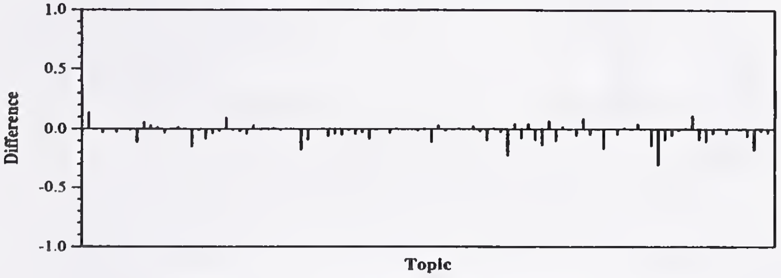
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	MU03rob03
Run Description	Automatic, title only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	49401	50000	99401
Total relevant	4416	1658	6074
Total relevant retrieved	1591	1256	2847
Mean average precision	0.0946	0.2594	0.1770
% topics with no relevant in top 10	16.0	8.0	12.0
Integral of MAP(X) curve for worst 1/4 topics	0.0029	0.0365	0.0087

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3160	0.4760	0.3960
At 10 docs	0.2800	0.4200	0.3500
At 15 docs	0.2533	0.3707	0.3120
At 20 docs	0.2200	0.3250	0.2725
At 30 docs	0.1940	0.2740	0.2340
At 100 docs	0.1208	0.1358	0.1283
At 200 docs	0.0849	0.0870	0.0859
At 500 docs	0.0505	0.0436	0.0470
At 1000 docs	0.0318	0.0251	0.0285
R-Precision			
Exact	0.1448	0.3051	0.2250





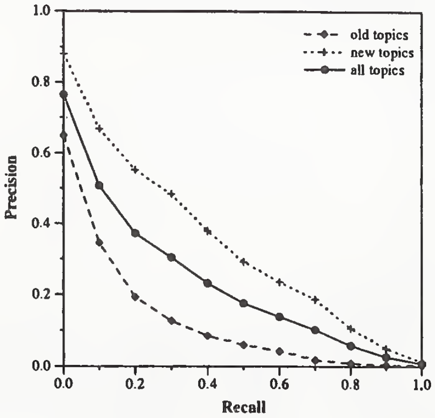
Difference from Median in Average Precision per Topic

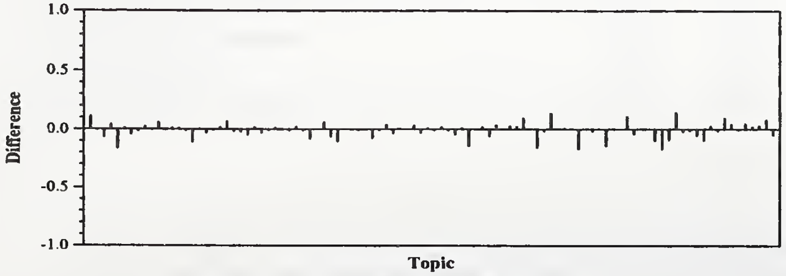
Summary Statistics	
Run ID:	MU03rob02
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1885	1347	3232
Mean average precision	0.1110	0.3267	0.2189
% topics with no relevant in top 10	14.0	2.0	8.0
Integral of MAP(X) curve for worst 1/4 topics	0.0093	0.0456	0.0175

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.4160	0.5880	0.5020
At 10 docs	0.3340	0.5060	0.4200
At 15 docs	0.2827	0.4293	0.3560
At 20 docs	0.2530	0.3690	0.3110
At 30 docs	0.2307	0.3033	0.2670
At 100 docs	0.1348	0.1512	0.1430
At 200 docs	0.0939	0.0936	0.0938
At 500 docs	0.0567	0.0482	0.0524
At 1000 docs	0.0377	0.0269	0.0323

R-Precision			
Exact	0.1633	0.3403	0.2518





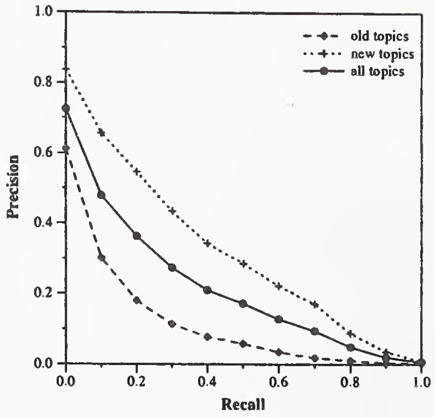
Difference from Median in Average Precision per Topic

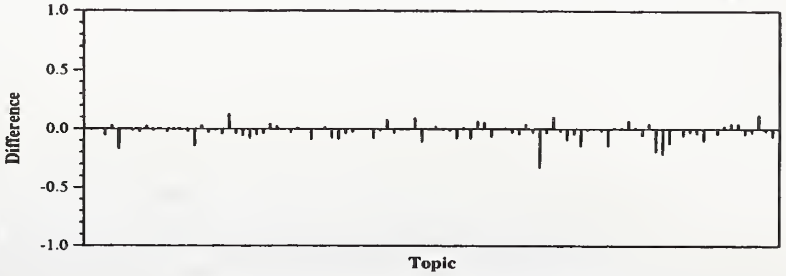
Summary Statistics	
Run ID:	MU03rob05
Run Description	Automatic, title+desc
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1846	1330	3176
Mean average precision	0.1025	0.3057	0.2041
% topics with no relevant in top 10	12.0	4.0	8.0
Integral of MAP(X) curve for worst 1/4 topics	0.0058	0.0448	0.0135

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3280	0.5720	0.4500
At 10 docs	0.2820	0.4640	0.3730
At 15 docs	0.2560	0.4013	0.3287
At 20 docs	0.2340	0.3470	0.2905
At 30 docs	0.2153	0.2953	0.2553
At 100 docs	0.1320	0.1462	0.1391
At 200 docs	0.0927	0.0911	0.0919
At 500 docs	0.0556	0.0471	0.0514
At 1000 docs	0.0369	0.0266	0.0318

R-Precision			
Exact	0.1658	0.3337	0.2497



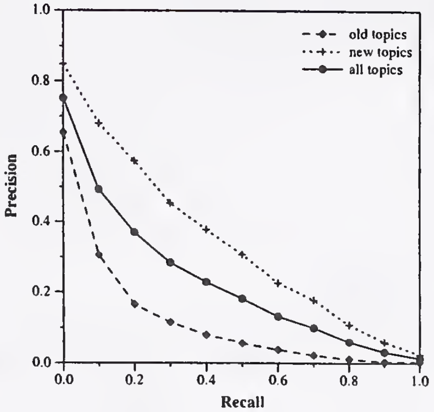


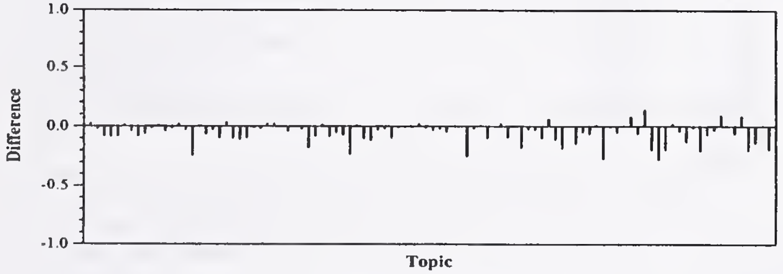
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	MU03rob04
Run Description	Automatic, title+desc+narr
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1834	1329	3163
Mean average precision	0.1017	0.3284	0.2151
% topics with no relevant in top 10	16.0	8.0	12.0
Integral of MAP(X) curve for worst 1/4 topics	0.0079	0.0383	0.0155

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3560	0.5960	0.4760
At 10 docs	0.2980	0.4880	0.3930
At 15 docs	0.2760	0.4200	0.3480
At 20 docs	0.2510	0.3710	0.3110
At 30 docs	0.2060	0.2953	0.2507
At 100 docs	0.1300	0.1454	0.1377
At 200 docs	0.0932	0.0904	0.0918
At 500 docs	0.0563	0.0461	0.0512
At 1000 docs	0.0367	0.0266	0.0316
R-Precision			
Exact	0.1505	0.3493	0.2499



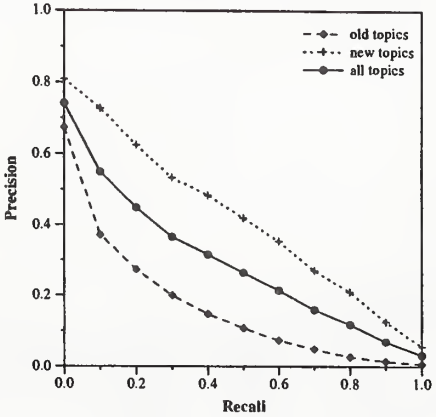


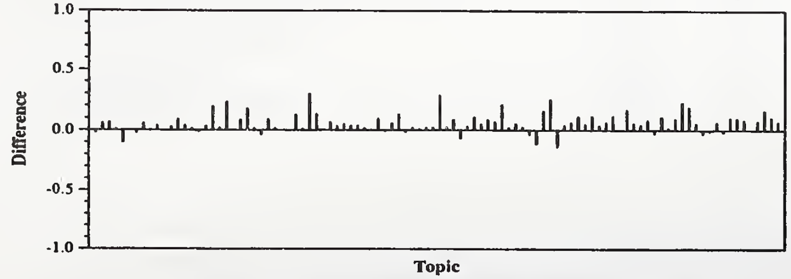
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	uwmtCR0
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2212	1480	3692
Mean average precision	0.1502	0.4025	0.2763
% topics with no relevant in top 10	14.0	8.0	11.0
Integral of MAP(X) curve for worst 1/4 topics	0.0113	0.0523	0.0179

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3920	0.6080	0.5000
At 10 docs	0.3700	0.5360	0.4530
At 15 docs	0.3213	0.4707	0.3960
At 20 docs	0.2900	0.4150	0.3525
At 30 docs	0.2573	0.3440	0.3007
At 100 docs	0.1666	0.1784	0.1725
At 200 docs	0.1177	0.1103	0.1140
At 500 docs	0.0695	0.0538	0.0616
At 1000 docs	0.0442	0.0296	0.0369
R-Precision			
Exact	0.1890	0.3980	0.2935



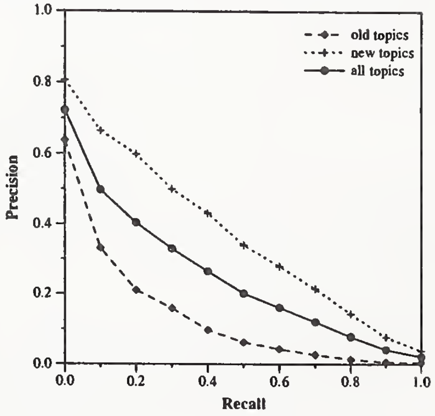


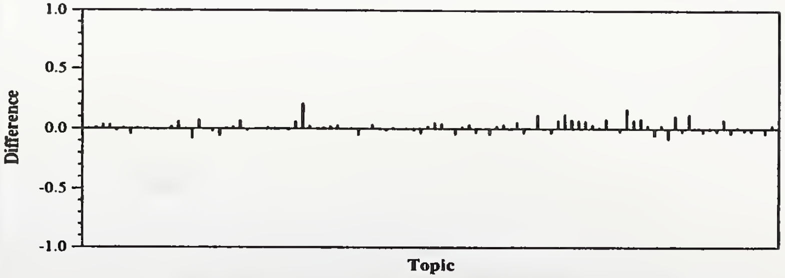
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	uwmtCR1
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	1900	1375	3275
Mean average precision	0.1139	0.3549	0.2344
% topics with no relevant in top 10	18.0	6.0	12.0
Integral of MAP(X) curve for worst 1/4 topics	0.0090	0.0348	0.0132

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3440	0.5880	0.4660
At 10 docs	0.3040	0.4780	0.3910
At 15 docs	0.2773	0.4307	0.3540
At 20 docs	0.2530	0.3940	0.3235
At 30 docs	0.2193	0.3227	0.2710
At 100 docs	0.1352	0.1622	0.1487
At 200 docs	0.0970	0.1001	0.0985
At 500 docs	0.0581	0.0486	0.0533
At 1000 docs	0.0380	0.0275	0.0327
R-Precision			
Exact	0.1726	0.3581	0.2654



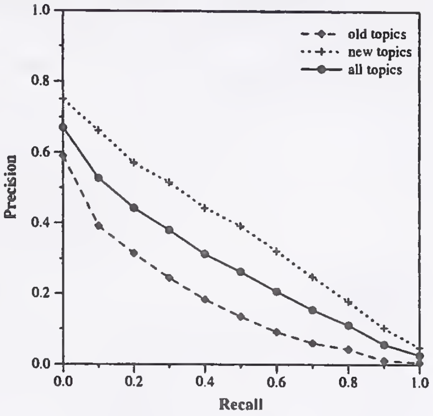


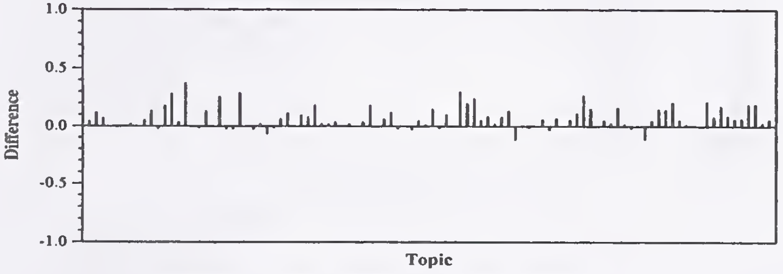
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	uwmtCR2
Run Description	Automatic, title only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2482	1522	4004
Mean average precision	0.1681	0.3704	0.2692
% topics with no relevant in top 10	22.0	10.0	16.0
Integral of MAP(X) curve for worst 1/4 topics	0.0059	0.0528	0.0150

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3840	0.5600	0.4720
At 10 docs	0.3640	0.4780	0.4210
At 15 docs	0.3267	0.4200	0.3733
At 20 docs	0.3110	0.3800	0.3455
At 30 docs	0.2767	0.3220	0.2993
At 100 docs	0.1872	0.1768	0.1820
At 200 docs	0.1347	0.1141	0.1244
At 500 docs	0.0796	0.0564	0.0680
At 1000 docs	0.0496	0.0304	0.0400
R-Precision			
Exact	0.2116	0.3644	0.2880



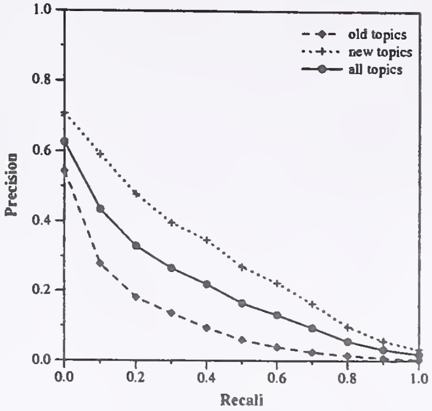


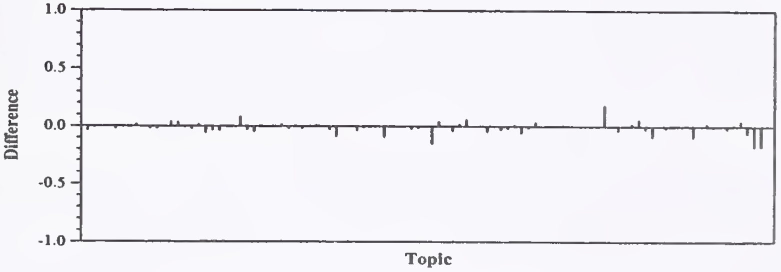
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	uwmtCR3
Run Description	Automatic, title only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	49414	50000	99414
Total relevant	4416	1658	6074
Total relevant retrieved	1959	1390	3349
Mean average precision	0.1024	0.2850	0.1937
% topics with no relevant in top 10	16.0	8.0	12.0
Integral of MAP(X) curve for worst 1/4 topics	0.0067	0.0415	0.0134

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.2800	0.5080	0.3940
At 10 docs	0.2820	0.4180	0.3500
At 15 docs	0.2547	0.3573	0.3060
At 20 docs	0.2370	0.3260	0.2815
At 30 docs	0.2080	0.2700	0.2390
At 100 docs	0.1382	0.1494	0.1438
At 200 docs	0.1033	0.0965	0.0999
At 500 docs	0.0605	0.0495	0.0550
At 1000 docs	0.0392	0.0278	0.0335
R-Precision			
Exact	0.1612	0.3155	0.2384



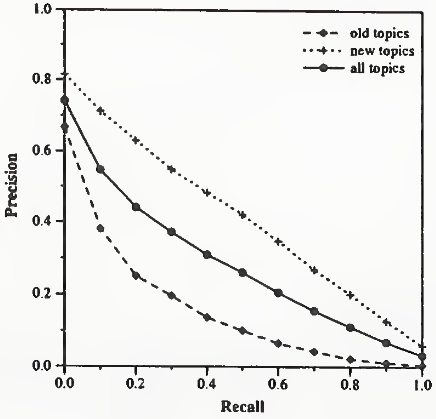


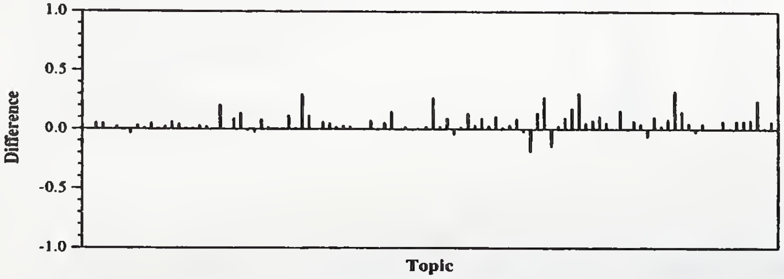
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	uwmtCR4
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2183	1488	3671
Mean average precision	0.1437	0.4037	0.2737
% topics with no relevant in top 10	20.0	8.0	14.0
Integral of MAP(X) curve for worst 1/4 topics	0.0135	0.0543	0.0186

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3920	0.6120	0.5020
At 10 docs	0.3400	0.5200	0.4300
At 15 docs	0.3120	0.4640	0.3880
At 20 docs	0.2750	0.4190	0.3470
At 30 docs	0.2393	0.3507	0.2950
At 100 docs	0.1566	0.1796	0.1681
At 200 docs	0.1179	0.1116	0.1147
At 500 docs	0.0684	0.0544	0.0614
At 1000 docs	0.0437	0.0298	0.0367
R-Precision			
Exact	0.1893	0.3932	0.2913



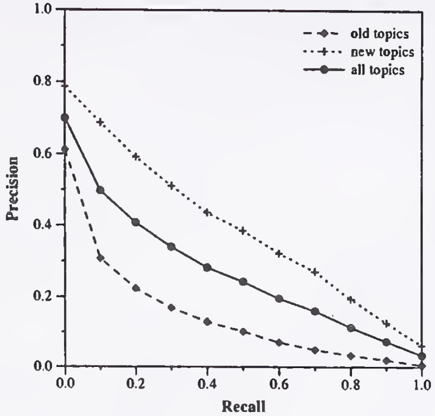


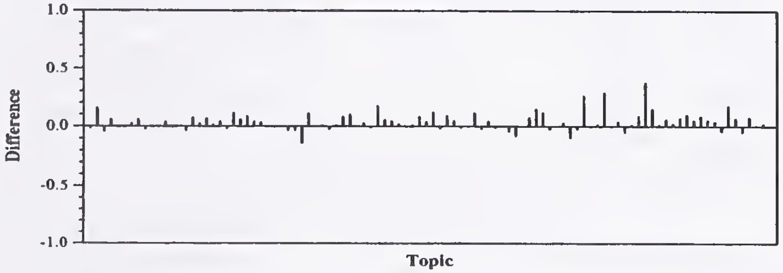
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	VTDokrcgp5
Run Description	Automatic, description only
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2254	1378	3632
Mean average precision	0.1304	0.3822	0.2563
% topics with no relevant in top 10	18.0	12.0	15.0
Integral of MAP(X) curve for worst 1/4 topics	0.0052	0.0227	0.0077

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.3480	0.5560	0.4520
At 10 docs	0.3140	0.5020	0.4080
At 15 docs	0.2760	0.4387	0.3573
At 20 docs	0.2650	0.4040	0.3345
At 30 docs	0.2300	0.3280	0.2790
At 100 docs	0.1430	0.1710	0.1570
At 200 docs	0.1047	0.1051	0.1049
At 500 docs	0.0683	0.0505	0.0594
At 1000 docs	0.0451	0.0276	0.0363
R-Precision			
Exact	0.1752	0.3780	0.2766



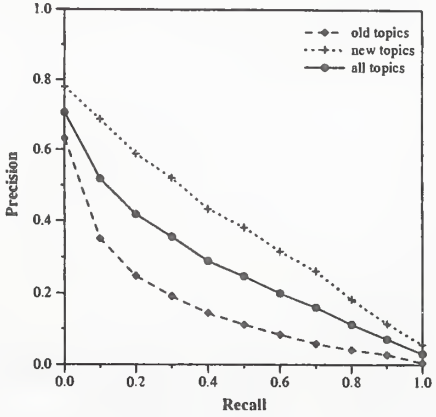


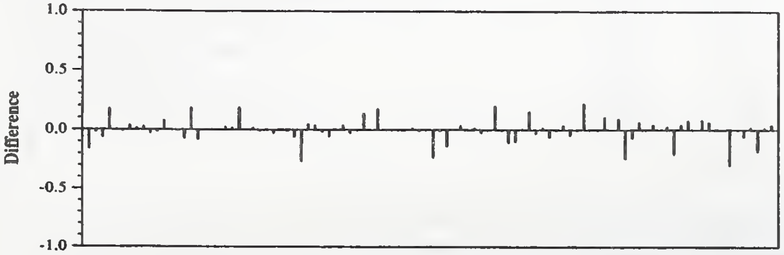
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	VTcdhgp1
Run Description	Automatic, title+desc+narr
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2344	1386	3730
Mean average precision	0.1511	0.3788	0.2649
% topics with no relevant in top 10	18.0	6.0	12.0
Integral of MAP(X) curve for worst 1/4 topics	0.0056	0.0286	0.0115

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.4000	0.6000	0.5000
At 10 docs	0.3520	0.5120	0.4320
At 15 docs	0.3200	0.4507	0.3853
At 20 docs	0.2900	0.4100	0.3500
At 30 docs	0.2520	0.3280	0.2900
At 100 docs	0.1708	0.1630	0.1669
At 200 docs	0.1228	0.1042	0.1135
At 500 docs	0.0725	0.0510	0.0618
At 1000 docs	0.0469	0.0277	0.0373
R-Precision			
Exact	0.1949	0.3774	0.2861



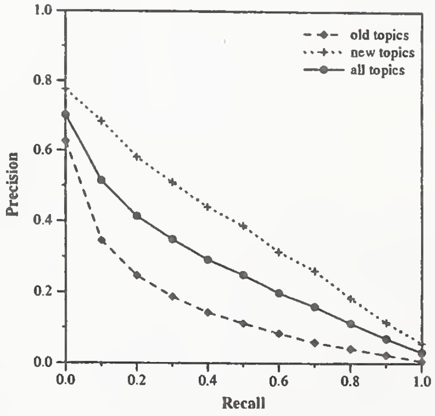


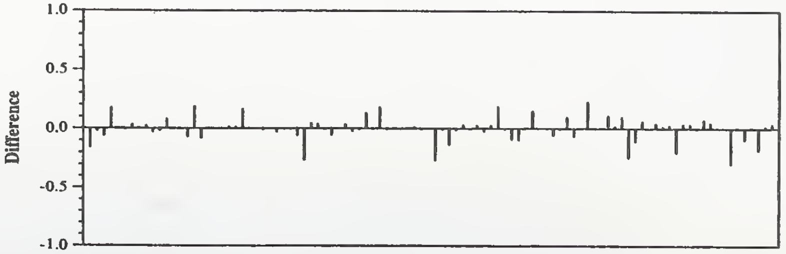
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	VTcdhgp3
Run Description	Automatic, title+desc+narr
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2338	1388	3726
Mean average precision	0.1500	0.3774	0.2637
% topics with no relevant in top 10	18.0	6.0	12.0
Integral of MAP(X) curve for worst 1/4 topics	0.0055	0.0299	0.0117

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.4000	0.5880	0.4940
At 10 docs	0.3500	0.5140	0.4320
At 15 docs	0.3173	0.4507	0.3840
At 20 docs	0.2820	0.4040	0.3430
At 30 docs	0.2513	0.3287	0.2900
At 100 docs	0.1708	0.1632	0.1670
At 200 docs	0.1218	0.1040	0.1129
At 500 docs	0.0723	0.0508	0.0616
At 1000 docs	0.0468	0.0278	0.0373
R-Precision			
Exact	0.1904	0.3704	0.2804



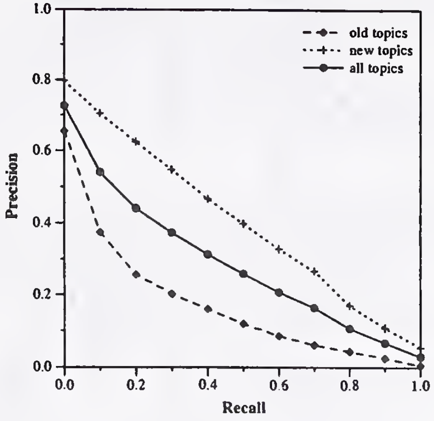


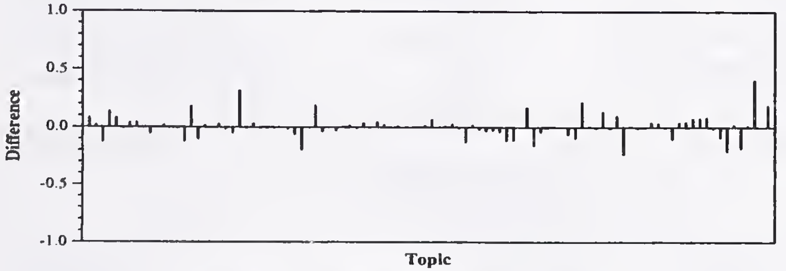
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	VTgpdhgp2
Run Description	Automatic, title+desc+narr
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2408	1394	3802
Mean average precision	0.1555	0.3907	0.2731
% topics with no relevant in top 10	12.0	6.0	9.0
Integral of MAP(X) curve for worst 1/4 topics	0.0084	0.0402	0.0152

Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.4040	0.6160	0.5100
At 10 docs	0.3680	0.5300	0.4490
At 15 docs	0.3253	0.4640	0.3947
At 20 docs	0.2940	0.4180	0.3560
At 30 docs	0.2593	0.3407	0.3000
At 100 docs	0.1694	0.1662	0.1678
At 200 docs	0.1217	0.1041	0.1129
At 500 docs	0.0742	0.0510	0.0626
At 1000 docs	0.0482	0.0279	0.0380
R-Precision			
Exact	0.1998	0.3843	0.2921



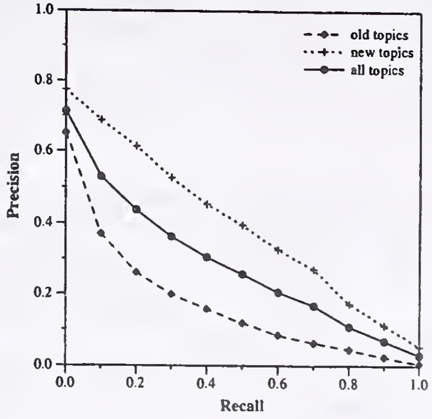


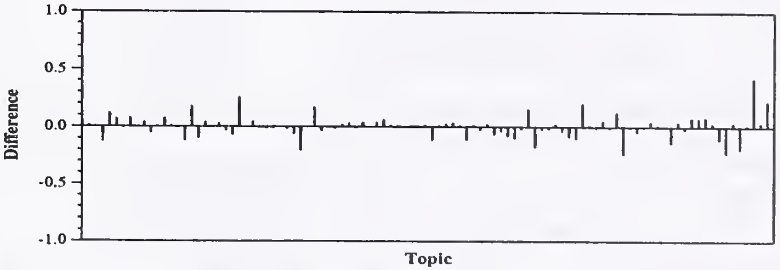
Difference from Median in Average Precision per Topic

Summary Statistics	
Run ID:	VTgpdhgp4
Run Description	Automatic, title+desc+narr
Number of Topics:	100

	Topic set		
	Old 50	New 50	All 100
Total number retrieved	50000	50000	100000
Total relevant	4416	1658	6074
Total relevant retrieved	2401	1423	3824
Mean average precision	0.1554	0.3838	0.2696
% topics with no relevant in top 10	10.0	8.0	9.0
Integral of MAP(X) curve for worst 1/4 topics	0.0082	0.0369	0.0151

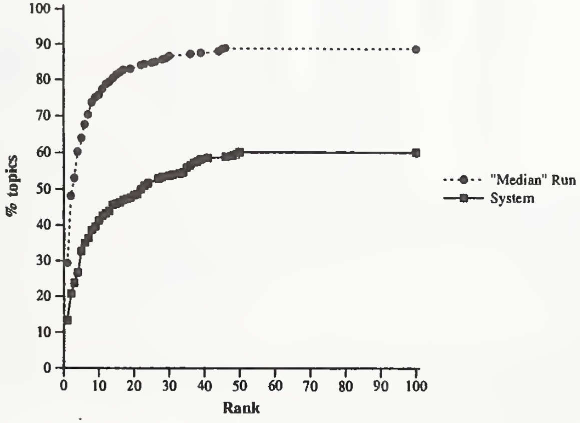
Document Level Averages			
	Precision		
	Old	New	All
At 5 docs	0.4080	0.6080	0.5080
At 10 docs	0.3720	0.5240	0.4480
At 15 docs	0.3227	0.4640	0.3933
At 20 docs	0.2960	0.4120	0.3540
At 30 docs	0.2587	0.3347	0.2967
At 100 docs	0.1682	0.1696	0.1689
At 200 docs	0.1244	0.1041	0.1143
At 500 docs	0.0742	0.0517	0.0630
At 1000 docs	0.0480	0.0285	0.0382
R-Precision			
Exact	0.2034	0.3810	0.2922





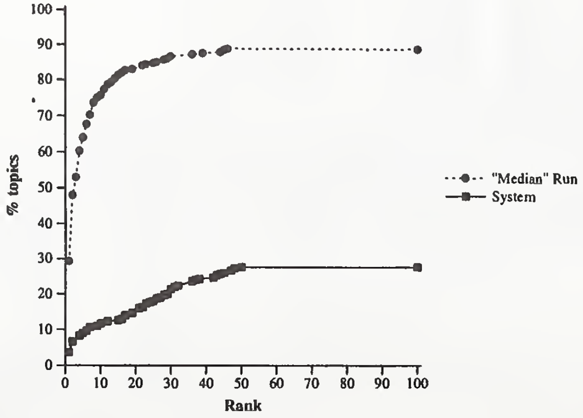
Difference from Median in Average Precision per Topic

Summary Statistics		
Run ID	ajouai0306	
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.220	
Num found at rank 1	40 (13.3%)	
Found in top 10	124 (41.3%)	
Not found in top 50	119 (39.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.311	0.129
Num found at rank 1	34 (22.7%)	6 (4.0%)
Found in top 10	73 (48.7%)	51 (34.0%)
Not found in top 50	45 (30.0%)	74 (49.3%)



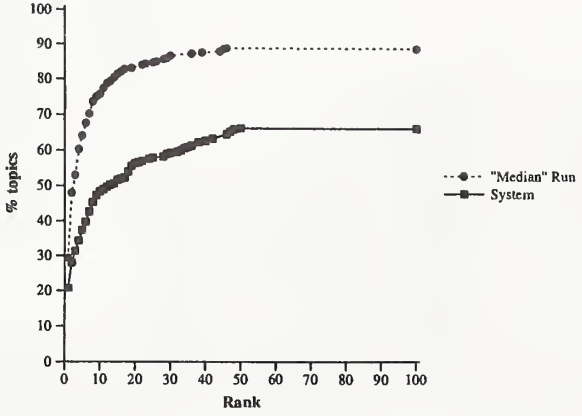
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	ajouai0308	
Run Description	DOCSTRUCT, NOANCHORTEXT, LINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.067	
Num found at rank 1	11 (3.7%)	
Found in top 10	35 (11.7%)	
Not found in top 50	217 (72.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.094	0.040
Num found at rank 1	8 (5.3%)	3 (2.0%)
Found in top 10	25 (16.7%)	10 (6.7%)
Not found in top 50	95 (63.3%)	122 (81.3%)



Cumulative % of topics that retrieve target page by given rank

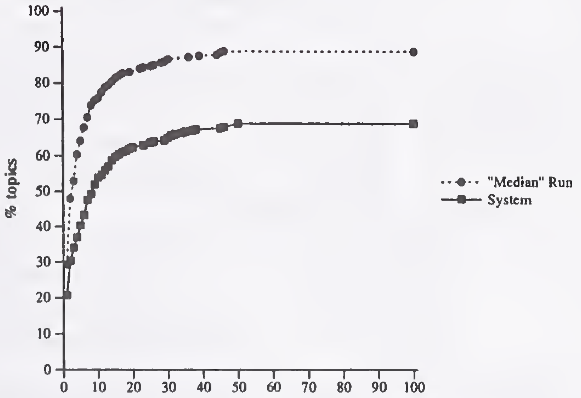
Summary Statistics		
Run ID	ajouai0309	
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.291	
Num found at rank 1	62 (20.7%)	
Found in top 10	145 (48.3%)	
Not found in top 50	101 (33.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.396	0.185
Num found at rank 1	44 (29.3%)	18 (12.0%)
Found in top 10	93 (62.0%)	52 (34.7%)
Not found in top 50	31 (20.7%)	70 (46.7%)



Cumulative % of topics that retrieve target page by given rank

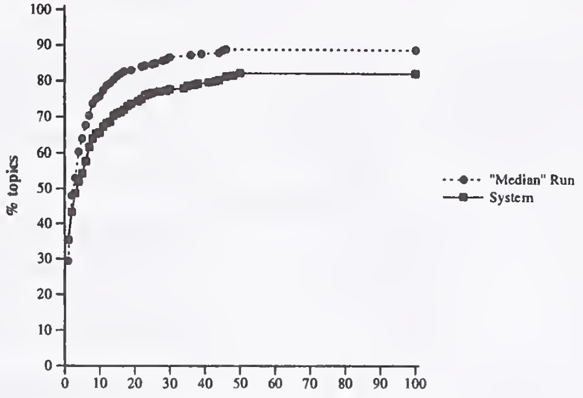
Web track, named/home page finding task results — Chinese Academy of Sciences (CAS-ICT)

Summary Statistics		
Run ID	ICTWebKI12A	
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.308	
Num found at rank 1	62 (20.7%)	
Found in top 10	162 (54.0%)	
Not found in top 50	93 (31.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.455	0.160
Num found at rank 1	51 (34.0%)	11 (7.3%)
Found in top 10	108 (72.0%)	54 (36.0%)
Not found in top 50	20 (13.3%)	73 (48.7%)



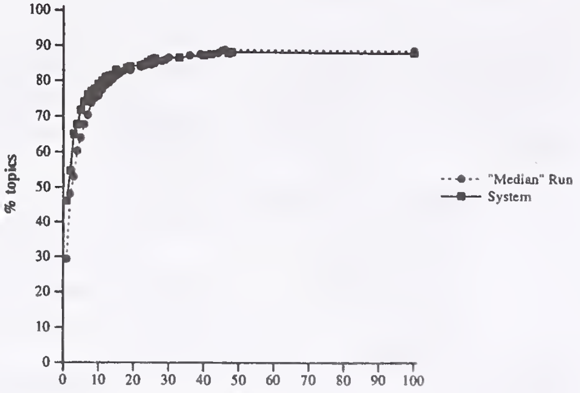
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	ICTWebKI12B	
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.449	
Num found at rank 1	106 (35.3%)	
Found in top 10	197 (65.7%)	
Not found in top 50	53 (17.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.612	0.285
Num found at rank 1	78 (52.0%)	28 (18.7%)
Found in top 10	120 (80.0%)	77 (51.3%)
Not found in top 50	9 (6.0%)	44 (29.3%)



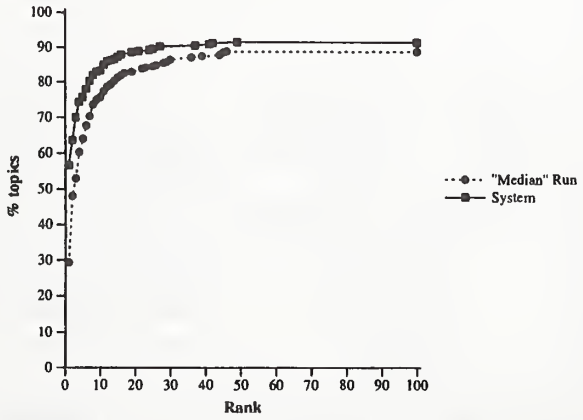
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	ICTWebKI12C	
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.568	
Num found at rank 1	138 (46.0%)	
Found in top 10	237 (79.0%)	
Not found in top 50	35 (11.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.611	0.524
Num found at rank 1	77 (51.3%)	61 (40.7%)
Found in top 10	122 (81.3%)	115 (76.7%)
Not found in top 50	8 (5.3%)	27 (18.0%)



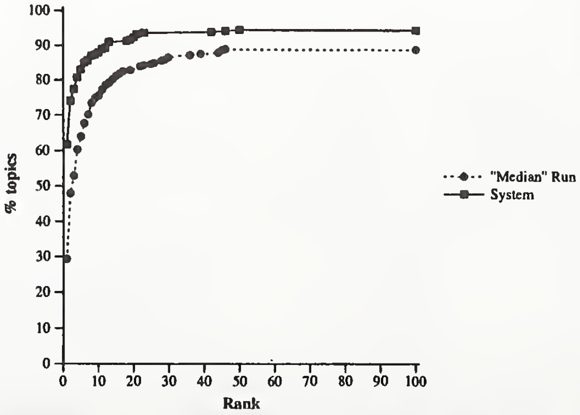
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		LmrEq
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.652	
Num found at rank 1	170 (56.7%)	
Found in top 10	250 (83.3%)	
Not found in top 50	25 (8.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.688	0.616
Num found at rank 1	91 (60.7%)	79 (52.7%)
Found in top 10	131 (87.3%)	119 (79.3%)
Not found in top 50	9 (6.0%)	16 (10.7%)



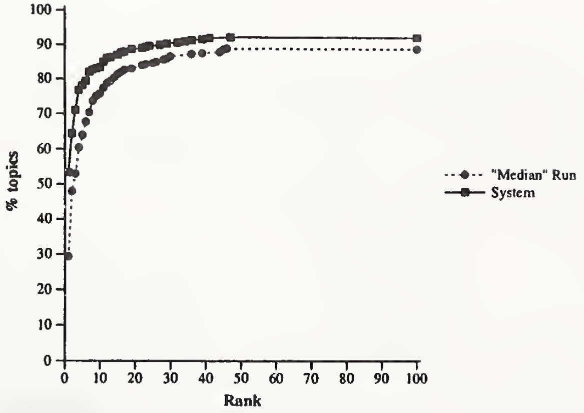
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		LmrEqUrl
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.713	
Num found at rank 1	185 (61.7%)	
Found in top 10	264 (88.0%)	
Not found in top 50	16 (5.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.633	0.794
Num found at rank 1	78 (52.0%)	107 (71.3%)
Found in top 10	124 (82.7%)	140 (93.3%)
Not found in top 50	11 (7.3%)	5 (3.3%)



Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		LmrEst
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.640	
Num found at rank 1	160 (53.3%)	
Found in top 10	250 (83.3%)	
Not found in top 50	23 (7.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.678	0.602
Num found at rank 1	86 (57.3%)	74 (49.3%)
Found in top 10	133 (88.7%)	117 (78.0%)
Not found in top 50	8 (5.3%)	15 (10.0%)

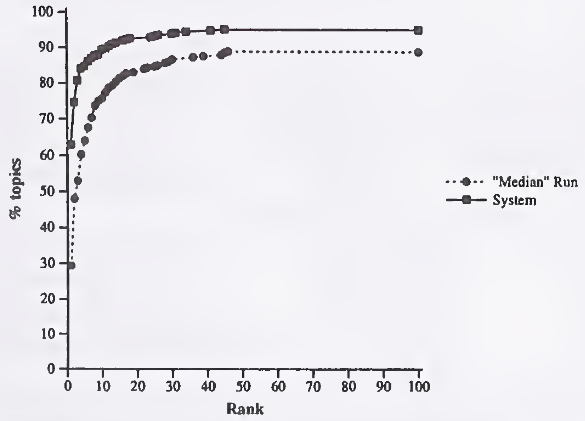


Cumulative % of topics that retrieve target page by given rank

Summary Statistics	
Run ID	LmrEstUrl
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics	300

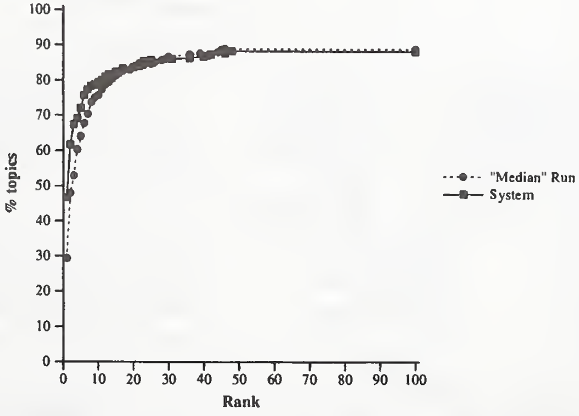
	overall
Mean reciprocal rank	0.727
Num found at rank 1	189 (63.0%)
Found in top 10	268 (89.3%)
Not found in top 50	14 (4.7%)

	named pgs.	home pgs.
Mean reciprocal rank	0.648	0.807
Num found at rank 1	79 (52.7%)	110 (73.3%)
Found in top 10	130 (86.7%)	138 (92.0%)
Not found in top 50	8 (5.3%)	6 (4.0%)



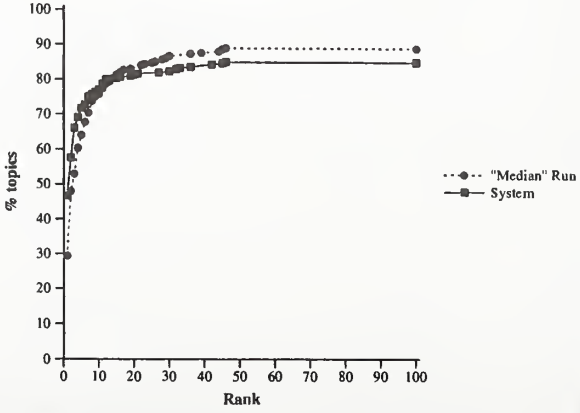
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		copNpRun1
Run Description		DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.586	
Num found at rank 1	140 (46.7%)	
Found in top 10	238 (79.3%)	
Not found in top 50	35 (11.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.553	0.619
Num found at rank 1	63 (42.0%)	77 (51.3%)
Found in top 10	120 (80.0%)	118 (78.7%)
Not found in top 50	19 (12.7%)	16 (10.7%)



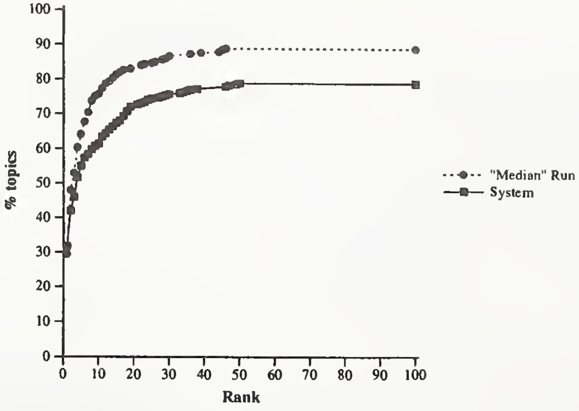
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		copNpRun2
Run Description		DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.574	
Num found at rank 1	140 (46.7%)	
Found in top 10	231 (77.0%)	
Not found in top 50	45 (15.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.519	0.629
Num found at rank 1	60 (40.0%)	80 (53.3%)
Found in top 10	111 (74.0%)	120 (80.0%)
Not found in top 50	27 (18.0%)	18 (12.0%)



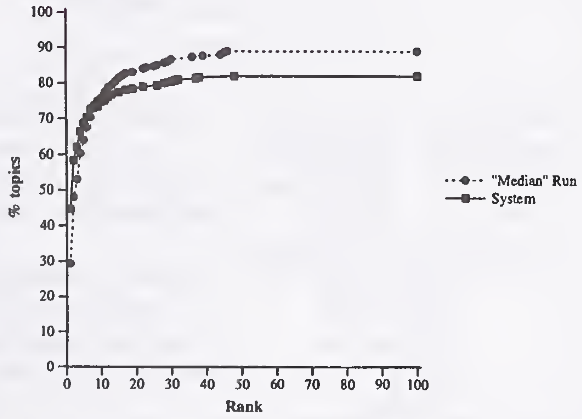
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		copNpRun3
Run Description		DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.421	
Num found at rank 1	95 (31.7%)	
Found in top 10	184 (61.3%)	
Not found in top 50	63 (21.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.549	0.293
Num found at rank 1	65 (43.3%)	30 (20.0%)
Found in top 10	113 (75.3%)	71 (47.3%)
Not found in top 50	18 (12.0%)	45 (30.0%)



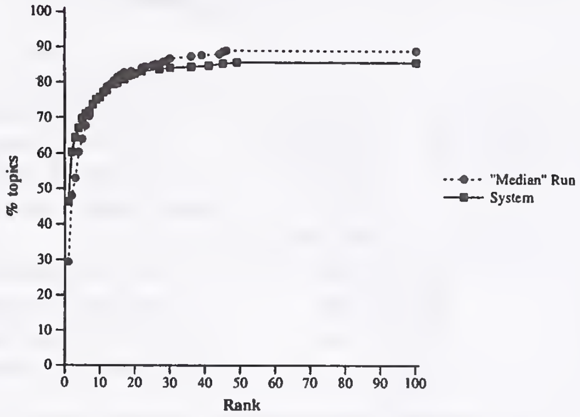
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	copNpRun4	
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.555	
Num found at rank 1	134 (44.7%)	
Found in top 10	224 (74.7%)	
Not found in top 50	54 (18.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.544	0.565
Num found at rank 1	65 (43.3%)	69 (46.0%)
Found in top 10	112 (74.7%)	112 (74.7%)
Not found in top 50	25 (16.7%)	29 (19.3%)



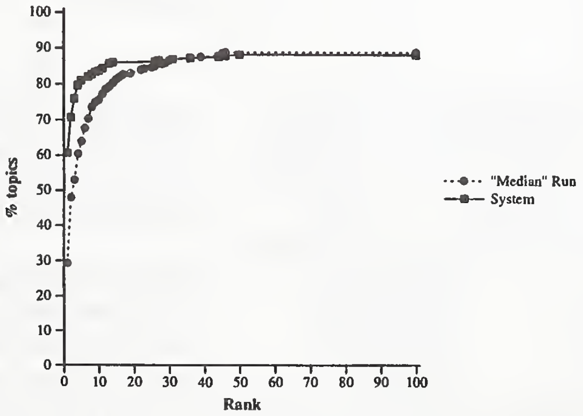
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	copNpRun5	
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.572	
Num found at rank 1	139 (46.3%)	
Found in top 10	227 (75.7%)	
Not found in top 50	43 (14.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.578	0.567
Num found at rank 1	67 (44.7%)	72 (48.0%)
Found in top 10	117 (78.0%)	110 (73.3%)
Not found in top 50	17 (11.3%)	26 (17.3%)



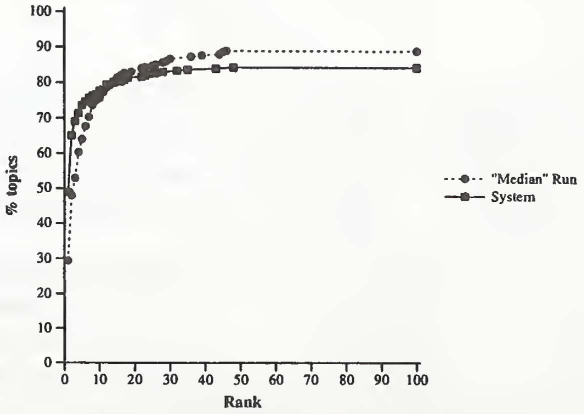
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		csiro03ki01
Run Description		DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.692	
Num found at rank 1	182 (60.7%)	
Found in top 10	251 (83.7%)	
Not found in top 50	35 (11.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.569	0.815
Num found at rank 1	66 (44.0%)	116 (77.3%)
Found in top 10	117 (78.0%)	134 (89.3%)
Not found in top 50	24 (16.0%)	11 (7.3%)



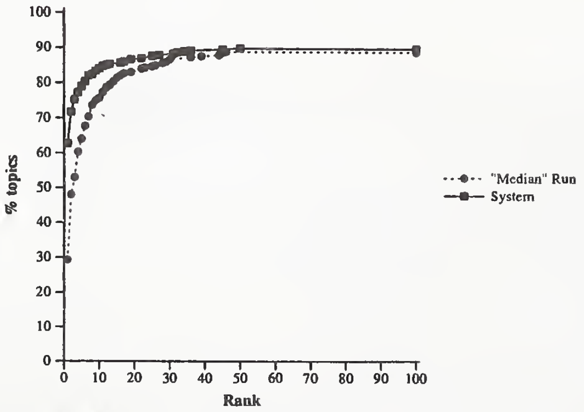
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		csiro03ki02
Run Description		DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.603	
Num found at rank 1	147 (49.0%)	
Found in top 10	233 (77.7%)	
Not found in top 50	47 (15.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.432	0.774
Num found at rank 1	42 (28.0%)	105 (70.0%)
Found in top 10	102 (68.0%)	131 (87.3%)
Not found in top 50	32 (21.3%)	15 (10.0%)



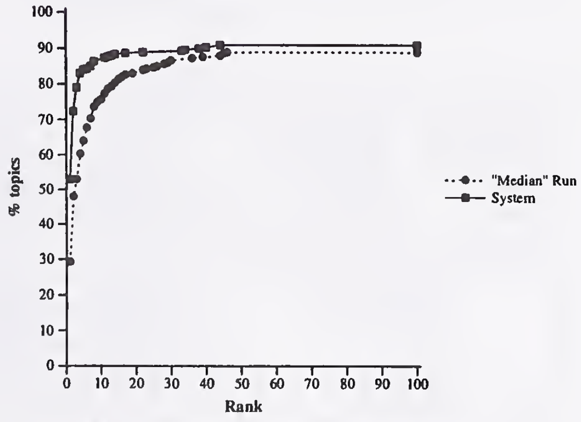
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		csiro03ki03
Run Description		DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.702	
Num found at rank 1	188 (62.7%)	
Found in top 10	252 (84.0%)	
Not found in top 50	30 (10.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.649	0.755
Num found at rank 1	84 (56.0%)	104 (69.3%)
Found in top 10	122 (81.3%)	130 (86.7%)
Not found in top 50	18 (12.0%)	12 (8.0%)



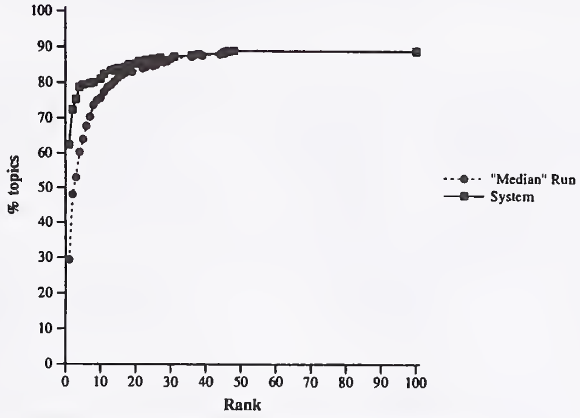
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	csiro03ki04	
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.667	
Num found at rank 1	159 (53.0%)	
Found in top 10	259 (86.3%)	
Not found in top 50	27 (9.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.532	0.801
Num found at rank 1	50 (33.3%)	109 (72.7%)
Found in top 10	123 (82.0%)	136 (90.7%)
Not found in top 50	16 (10.7%)	11 (7.3%)



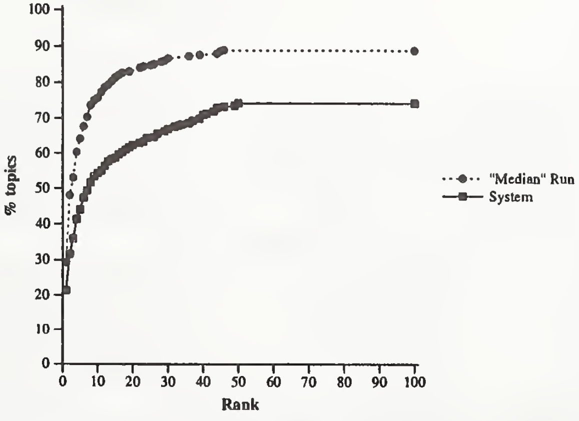
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	csiro03ki05	
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.699	
Num found at rank 1	187 (62.3%)	
Found in top 10	243 (81.0%)	
Not found in top 50	33 (11.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.587	0.812
Num found at rank 1	72 (48.0%)	115 (76.7%)
Found in top 10	111 (74.0%)	132 (88.0%)
Not found in top 50	21 (14.0%)	12 (8.0%)



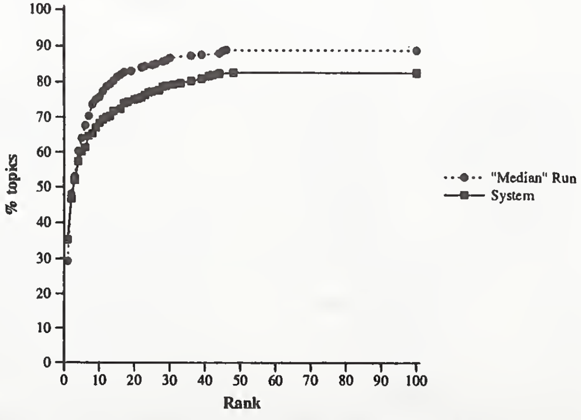
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	humNP03l	
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.321	
Num found at rank 1	64 (21.3%)	
Found in top 10	163 (54.3%)	
Not found in top 50	77 (25.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.420	0.223
Num found at rank 1	45 (30.0%)	19 (12.7%)
Found in top 10	101 (67.3%)	62 (41.3%)
Not found in top 50	16 (10.7%)	61 (40.7%)



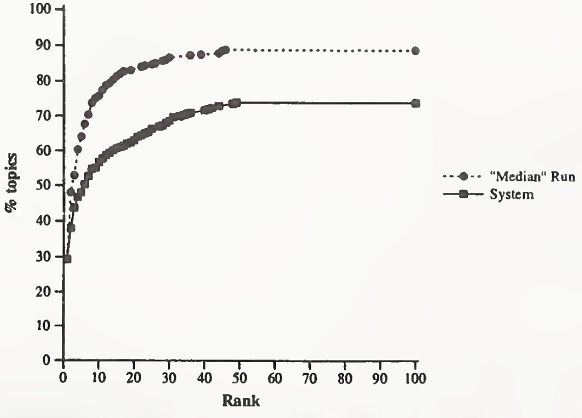
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	humNP03pl	
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.465	
Num found at rank 1	106 (35.3%)	
Found in top 10	205 (68.3%)	
Not found in top 50	52 (17.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.568	0.361
Num found at rank 1	68 (45.3%)	38 (25.3%)
Found in top 10	120 (80.0%)	85 (56.7%)
Not found in top 50	11 (7.3%)	41 (27.3%)



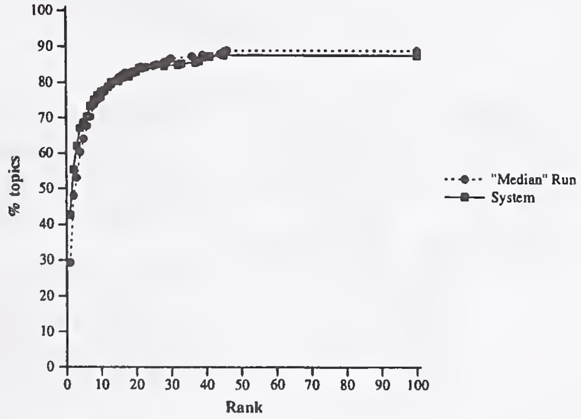
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	humNP03uhpl	
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.386	
Num found at rank 1	88 (29.3%)	
Found in top 10	170 (56.7%)	
Not found in top 50	78 (26.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.271	0.500
Num found at rank 1	29 (19.3%)	59 (39.3%)
Found in top 10	64 (42.7%)	106 (70.7%)
Not found in top 50	53 (35.3%)	25 (16.7%)



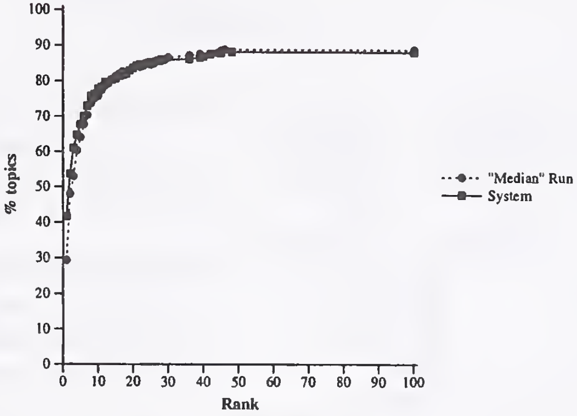
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		humNP03up
Run Description		DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.545	
Num found at rank 1	128 (42.7%)	
Found in top 10	232 (77.3%)	
Not found in top 50	37 (12.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.506	0.584
Num found at rank 1	59 (39.3%)	69 (46.0%)
Found in top 10	109 (72.7%)	123 (82.0%)
Not found in top 50	24 (16.0%)	13 (8.7%)



Cumulative % of topics that retrieve target page by given rank

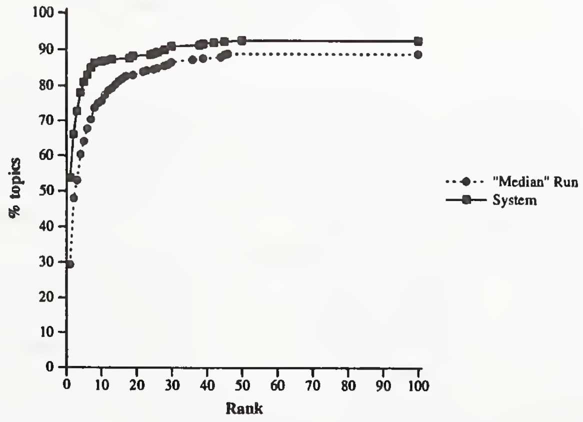
Summary Statistics		
Run ID		humNP03upl
Run Description		DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.535	
Num found at rank 1	125 (41.7%)	
Found in top 10	233 (77.7%)	
Not found in top 50	35 (11.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.480	0.591
Num found at rank 1	54 (36.0%)	71 (47.3%)
Found in top 10	109 (72.7%)	124 (82.7%)
Not found in top 50	23 (15.3%)	12 (8.0%)



Cumulative % of topics that retrieve target page by given rank

Summary Statistics

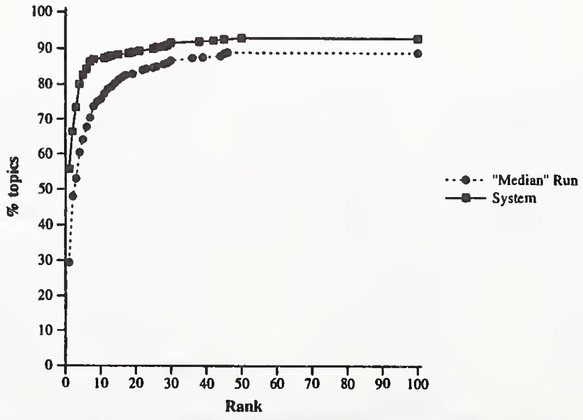
Run ID		iit03sa
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.651	
Num found at rank 1	161 (53.7%)	
Found in top 10	260 (86.7%)	
Not found in top 50	22 (7.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.568	0.733
Num found at rank 1	64 (42.7%)	97 (64.7%)
Found in top 10	125 (83.3%)	135 (90.0%)
Not found in top 50	13 (8.7%)	9 (6.0%)



Cumulative % of topics that retrieve target page by given rank

Summary Statistics

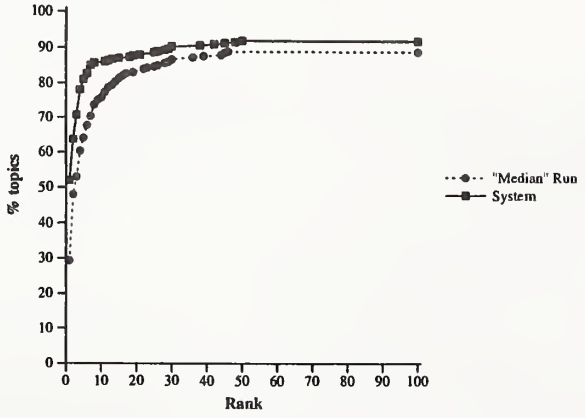
Run ID		iit03sau
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.665	
Num found at rank 1	167 (55.7%)	
Found in top 10	261 (87.0%)	
Not found in top 50	21 (7.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.572	0.757
Num found at rank 1	65 (43.3%)	102 (68.0%)
Found in top 10	125 (83.3%)	136 (90.7%)
Not found in top 50	13 (8.7%)	8 (5.3%)



Cumulative % of topics that retrieve target page by given rank

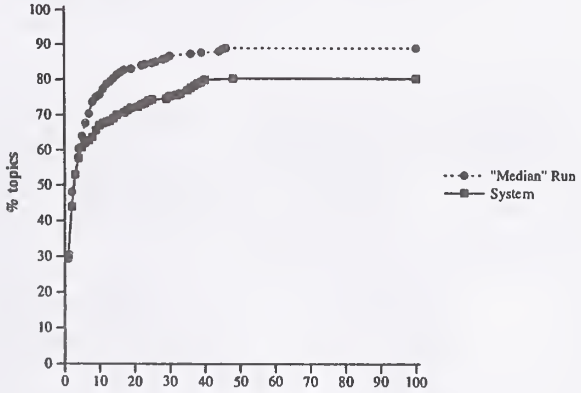
Summary Statistics

Run ID		iit03su
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.636	
Num found at rank 1	156 (52.0%)	
Found in top 10	257 (85.7%)	
Not found in top 50	24 (8.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.572	0.699
Num found at rank 1	65 (43.3%)	91 (60.7%)
Found in top 10	125 (83.3%)	132 (88.0%)
Not found in top 50	13 (8.7%)	11 (7.3%)



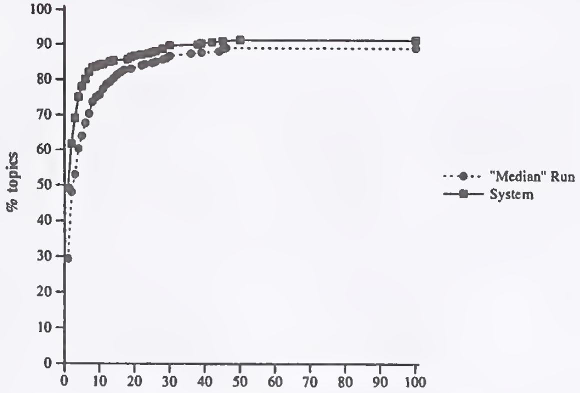
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		iit03wp75
Run Description		NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.433	
Num found at rank 1	91 (30.3%)	
Found in top 10	201 (67.0%)	
Not found in top 50	59 (19.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.514	0.353
Num found at rank 1	55 (36.7%)	36 (24.0%)
Found in top 10	116 (77.3%)	85 (56.7%)
Not found in top 50	16 (10.7%)	43 (28.7%)



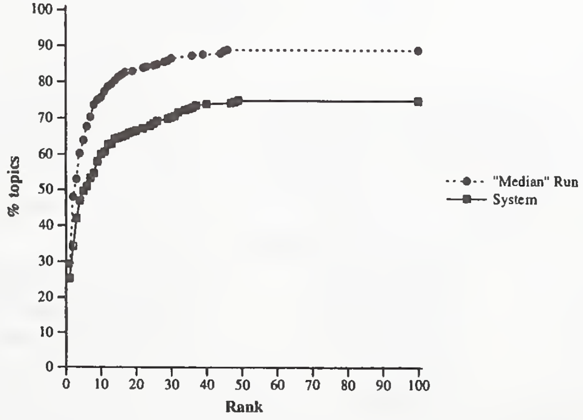
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		iit03wtaez
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.611	
Num found at rank 1	147 (49.0%)	
Found in top 10	252 (84.0%)	
Not found in top 50	26 (8.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.568	0.653
Num found at rank 1	64 (42.7%)	83 (55.3%)
Found in top 10	125 (83.3%)	127 (84.7%)
Not found in top 50	13 (8.7%)	13 (8.7%)



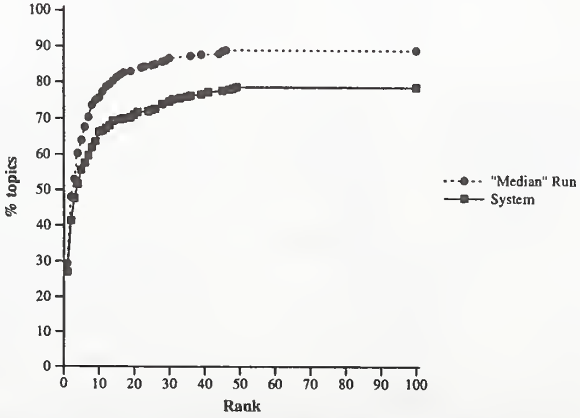
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	widitpfb1	
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.362	
Num found at rank 1	76 (25.3%)	
Found in top 10	180 (60.0%)	
Not found in top 50	75 (25.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.426	0.299
Num found at rank 1	46 (30.7%)	30 (20.0%)
Found in top 10	103 (68.7%)	77 (51.3%)
Not found in top 50	28 (18.7%)	47 (31.3%)



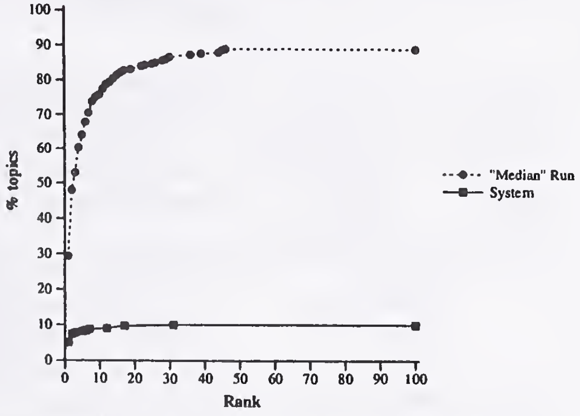
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	widitpff1	
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.400	
Num found at rank 1	81 (27.0%)	
Found in top 10	199 (66.3%)	
Not found in top 50	64 (21.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.482	0.318
Num found at rank 1	52 (34.7%)	29 (19.3%)
Found in top 10	112 (74.7%)	87 (58.0%)
Not found in top 50	24 (16.0%)	40 (26.7%)



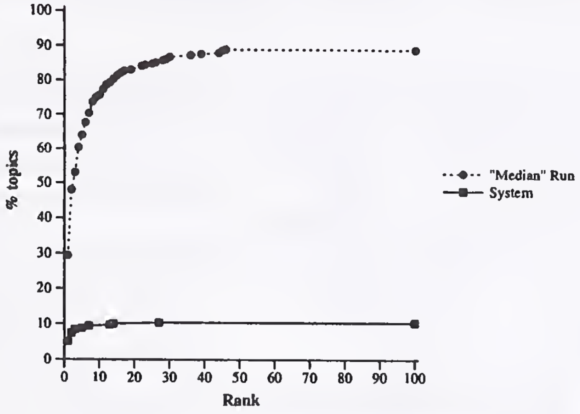
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		03wume296
Run Description		NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.065	
Num found at rank 1	15 (5.0%)	
Found in top 10	26 (8.7%)	
Not found in top 50	270 (90.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.044	0.087
Num found at rank 1	3 (2.0%)	12 (8.0%)
Found in top 10	12 (8.0%)	14 (9.3%)
Not found in top 50	134 (89.3%)	136 (90.7%)



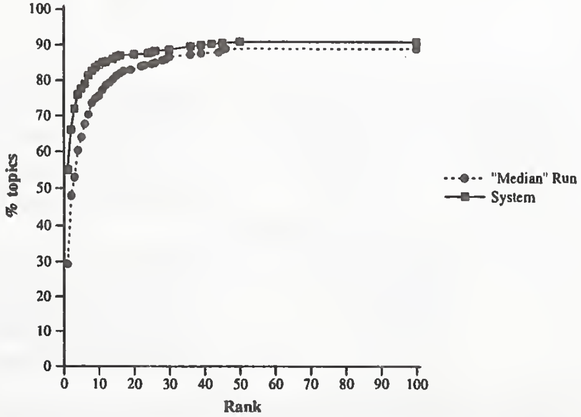
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		03wume298
Run Description		NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.067	
Num found at rank 1	15 (5.0%)	
Found in top 10	28 (9.3%)	
Not found in top 50	269 (89.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.044	0.090
Num found at rank 1	3 (2.0%)	12 (8.0%)
Found in top 10	13 (8.7%)	15 (10.0%)
Not found in top 50	134 (89.3%)	135 (90.0%)



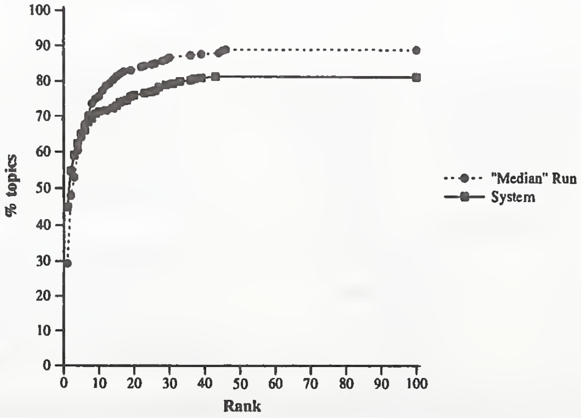
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		MSRANP1
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.651	
Num found at rank 1	165 (55.0%)	
Found in top 10	253 (84.3%)	
Not found in top 50	27 (9.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.646	0.655
Num found at rank 1	80 (53.3%)	85 (56.7%)
Found in top 10	132 (88.0%)	121 (80.7%)
Not found in top 50	7 (4.7%)	20 (13.3%)



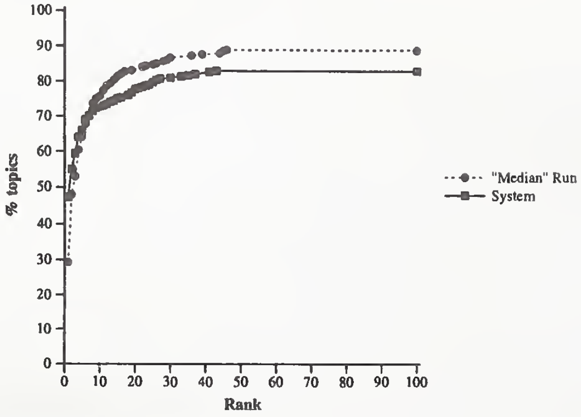
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		MSRANP2
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.540	
Num found at rank 1	135 (45.0%)	
Found in top 10	214 (71.3%)	
Not found in top 50	56 (18.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.470	0.609
Num found at rank 1	56 (37.3%)	79 (52.7%)
Found in top 10	99 (66.0%)	115 (76.7%)
Not found in top 50	28 (18.7%)	28 (18.7%)



Cumulative % of topics that retrieve target page by given rank

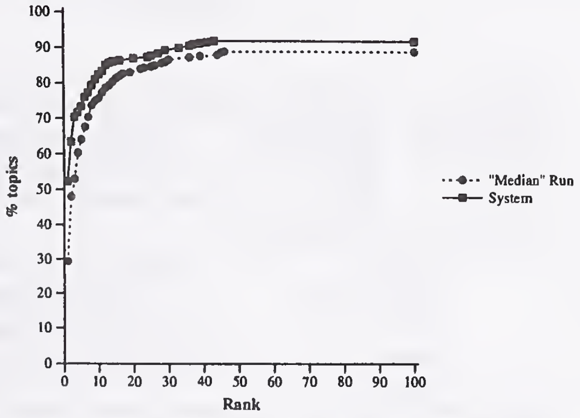
Summary Statistics		
Run ID		MSRANP3
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.556	
Num found at rank 1	142 (47.3%)	
Found in top 10	218 (72.7%)	
Not found in top 50	51 (17.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.483	0.630
Num found at rank 1	59 (39.3%)	83 (55.3%)
Found in top 10	102 (68.0%)	116 (77.3%)
Not found in top 50	32 (21.3%)	19 (12.7%)



Cumulative % of topics that retrieve target page by given rank

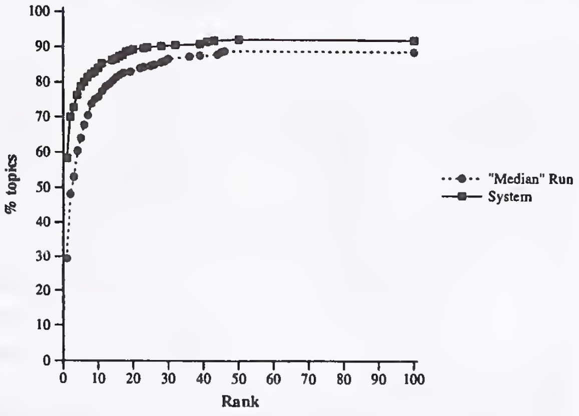
Web track, named/home page finding task results — Universite de Neuchatel

Summary Statistics		
Run ID	UniNEnp1	
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.626	
Num found at rank 1	157 (52.3%)	
Found in top 10	247 (82.3%)	
Not found in top 50	24 (8.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.649	0.602
Num found at rank 1	82 (54.7%)	75 (50.0%)
Found in top 10	130 (86.7%)	117 (78.0%)
Not found in top 50	10 (6.7%)	14 (9.3%)



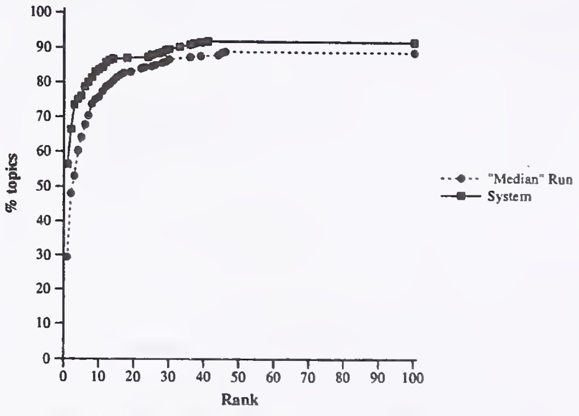
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	UniNEnp2	
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.676	
Num found at rank 1	175 (58.3%)	
Found in top 10	252 (84.0%)	
Not found in top 50	23 (7.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.665	0.687
Num found at rank 1	84 (56.0%)	91 (60.7%)
Found in top 10	128 (85.3%)	124 (82.7%)
Not found in top 50	9 (6.0%)	14 (9.3%)



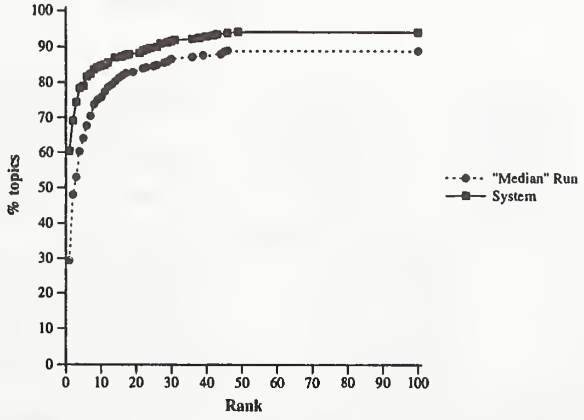
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	UniNEnp3	
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.658	
Num found at rank 1	169 (56.3%)	
Found in top 10	251 (83.7%)	
Not found in top 50	24 (8.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.653	0.663
Num found at rank 1	83 (55.3%)	86 (57.3%)
Found in top 10	130 (86.7%)	121 (80.7%)
Not found in top 50	10 (6.7%)	14 (9.3%)



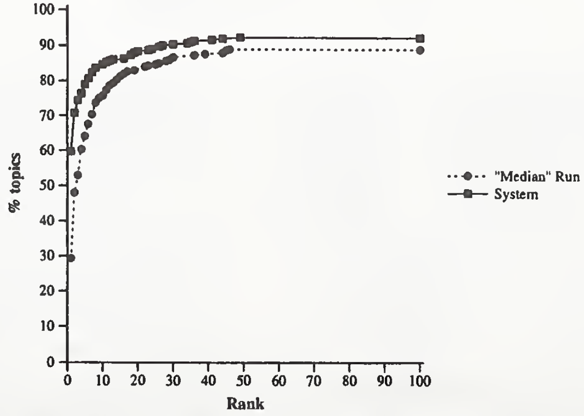
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		UniNEnp4
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.688	
Num found at rank 1	181 (60.3%)	
Found in top 10	254 (84.7%)	
Not found in top 50	17 (5.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.646	0.731
Num found at rank 1	81 (54.0%)	100 (66.7%)
Found in top 10	125 (83.3%)	129 (86.0%)
Not found in top 50	11 (7.3%)	6 (4.0%)



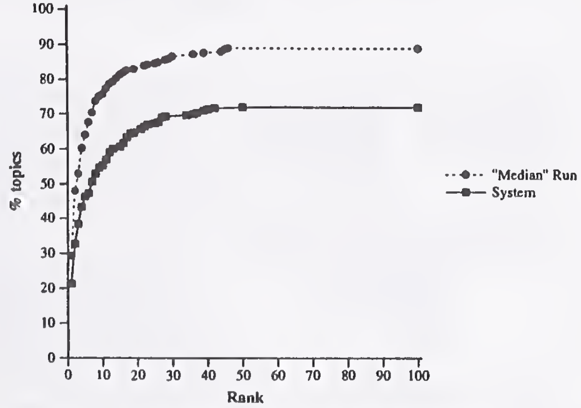
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		UniNEnp5
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.686	
Num found at rank 1	179 (59.7%)	
Found in top 10	254 (84.7%)	
Not found in top 50	23 (7.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.666	0.706
Num found at rank 1	83 (55.3%)	96 (64.0%)
Found in top 10	129 (86.0%)	125 (83.3%)
Not found in top 50	10 (6.7%)	13 (8.7%)



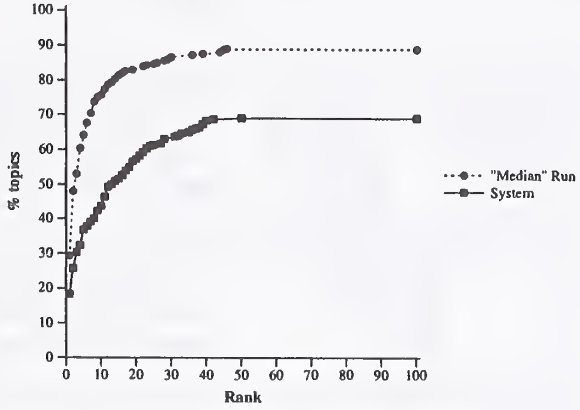
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	RMITSEG1	
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.329	
Num found at rank 1	64 (21.3%)	
Found in top 10	166 (55.3%)	
Not found in top 50	84 (28.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.270	0.387
Num found at rank 1	19 (12.7%)	45 (30.0%)
Found in top 10	87 (58.0%)	79 (52.7%)
Not found in top 50	21 (14.0%)	63 (42.0%)



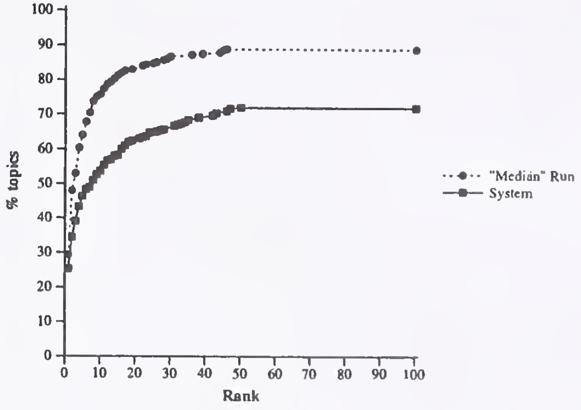
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	RMITSEG2	
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.272	
Num found at rank 1	55 (18.3%)	
Found in top 10	131 (43.7%)	
Not found in top 50	93 (31.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.199	0.344
Num found at rank 1	14 (9.3%)	41 (27.3%)
Found in top 10	62 (41.3%)	69 (46.0%)
Not found in top 50	24 (16.0%)	69 (46.0%)



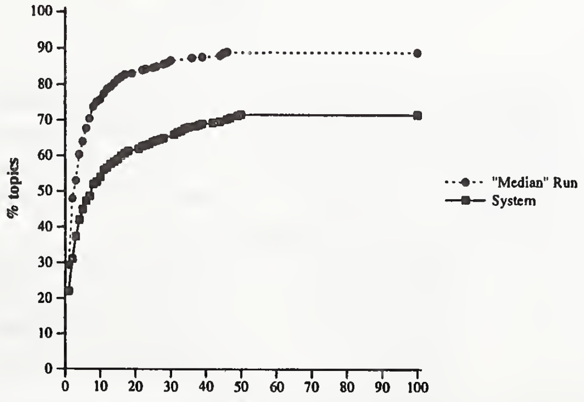
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	RMITSEG3	
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.350	
Num found at rank 1	76 (25.3%)	
Found in top 10	161 (53.7%)	
Not found in top 50	84 (28.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.478	0.222
Num found at rank 1	54 (36.0%)	22 (14.7%)
Found in top 10	109 (72.7%)	52 (34.7%)
Not found in top 50	21 (14.0%)	63 (42.0%)



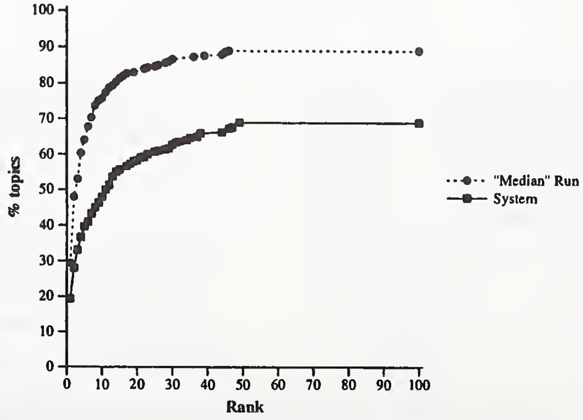
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		RMITSEG4
Run Description		DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.325	
Num found at rank 1	66 (22.0%)	
Found in top 10	162 (54.0%)	
Not found in top 50	85 (28.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.447	0.202
Num found at rank 1	48 (32.0%)	18 (12.0%)
Found in top 10	108 (72.0%)	54 (36.0%)
Not found in top 50	23 (15.3%)	62 (41.3%)



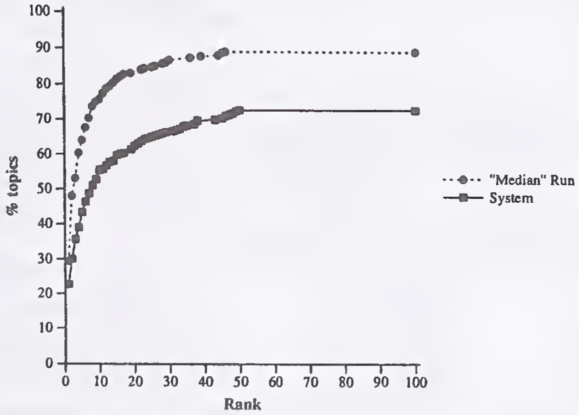
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		RMITSEG5
Run Description		DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.290	
Num found at rank 1	58 (19.3%)	
Found in top 10	144 (48.0%)	
Not found in top 50	93 (31.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.415	0.166
Num found at rank 1	44 (29.3%)	14 (9.3%)
Found in top 10	99 (66.0%)	45 (30.0%)
Not found in top 50	24 (16.0%)	69 (46.0%)



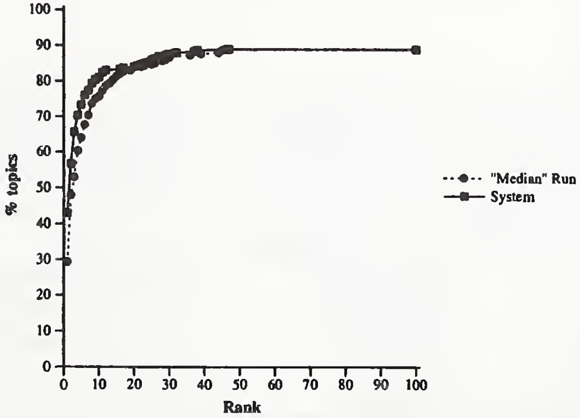
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	homepages0	
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.323	
Num found at rank 1	68 (22.7%)	
Found in top 10	166 (55.3%)	
Not found in top 50	82 (27.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.422	0.224
Num found at rank 1	45 (30.0%)	23 (15.3%)
Found in top 10	103 (68.7%)	63 (42.0%)
Not found in top 50	24 (16.0%)	58 (38.7%)



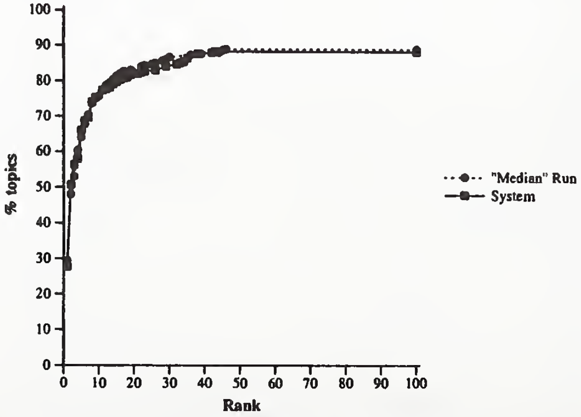
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	THUIRpf0301	
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.561	
Num found at rank 1	129 (43.0%)	
Found in top 10	243 (81.0%)	
Not found in top 50	33 (11.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.503	0.618
Num found at rank 1	52 (34.7%)	77 (51.3%)
Found in top 10	122 (81.3%)	121 (80.7%)
Not found in top 50	14 (9.3%)	19 (12.7%)



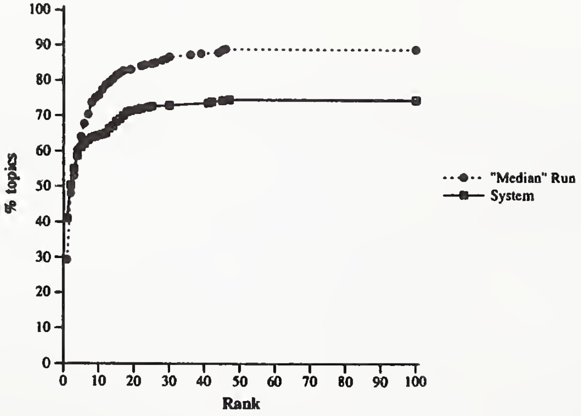
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	THUIRpf0302	
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.450	
Num found at rank 1	83 (27.7%)	
Found in top 10	226 (75.3%)	
Not found in top 50	35 (11.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.535	0.366
Num found at rank 1	58 (38.7%)	25 (16.7%)
Found in top 10	122 (81.3%)	104 (69.3%)
Not found in top 50	13 (8.7%)	22 (14.7%)



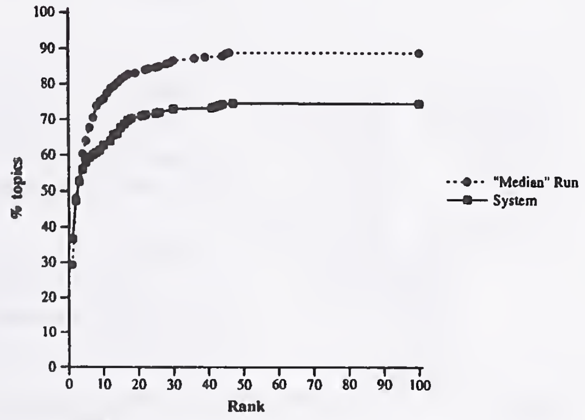
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	THUIRpf0303	
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.496	
Num found at rank 1	123 (41.0%)	
Found in top 10	193 (64.3%)	
Not found in top 50	76 (25.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.413	0.580
Num found at rank 1	49 (32.7%)	74 (49.3%)
Found in top 10	87 (58.0%)	106 (70.7%)
Not found in top 50	44 (29.3%)	32 (21.3%)



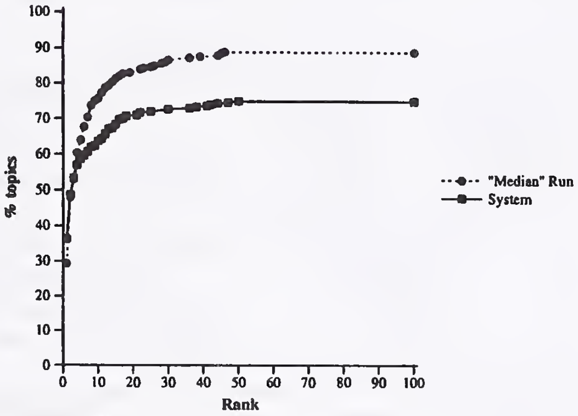
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		THUIRpf0304
Run Description		NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.463	
Num found at rank 1	110 (36.7%)	
Found in top 10	188 (62.7%)	
Not found in top 50	76 (25.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.429	0.497
Num found at rank 1	52 (34.7%)	58 (38.7%)
Found in top 10	87 (58.0%)	101 (67.3%)
Not found in top 50	44 (29.3%)	32 (21.3%)



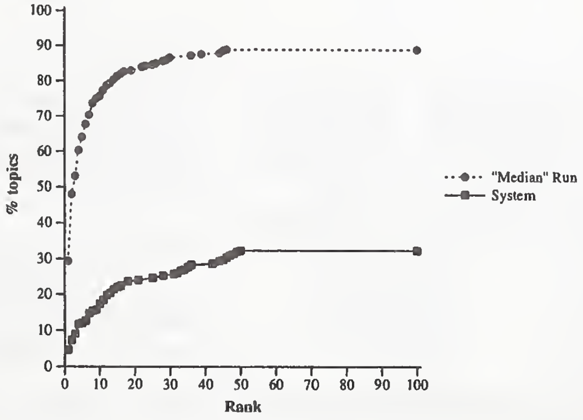
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		THUIRpf0305
Run Description		NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.466	
Num found at rank 1	109 (36.3%)	
Found in top 10	191 (63.7%)	
Not found in top 50	75 (25.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.444	0.488
Num found at rank 1	54 (36.0%)	55 (36.7%)
Found in top 10	91 (60.7%)	100 (66.7%)
Not found in top 50	43 (28.7%)	32 (21.3%)



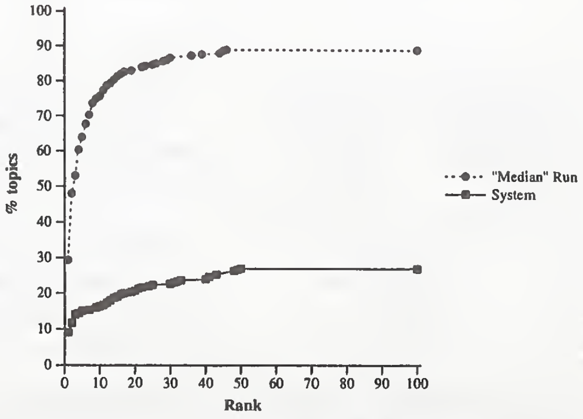
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		irtfgrep
Run Description		NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.087	
Num found at rank 1	14 (4.7%)	
Found in top 10	52 (17.3%)	
Not found in top 50	203 (67.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.087	0.086
Num found at rank 1	9 (6.0%)	5 (3.3%)
Found in top 10	22 (14.7%)	30 (20.0%)
Not found in top 50	112 (74.7%)	91 (60.7%)



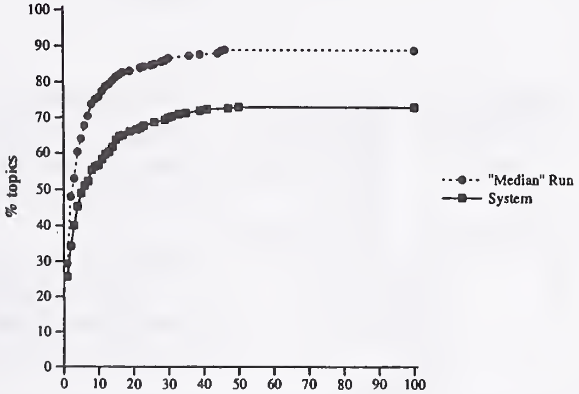
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		irttgrep
Run Description		NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.120	
Num found at rank 1	27 (9.0%)	
Found in top 10	49 (16.3%)	
Not found in top 50	219 (73.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.161	0.078
Num found at rank 1	20 (13.3%)	7 (4.7%)
Found in top 10	30 (20.0%)	19 (12.7%)
Not found in top 50	113 (75.3%)	106 (70.7%)



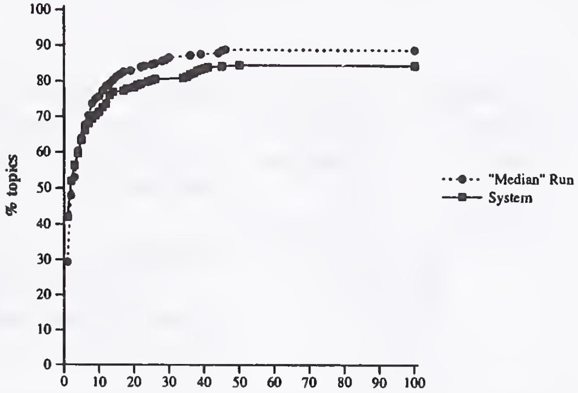
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	UAmsT03WnLM	
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.359	
Num found at rank 1	77 (25.7%)	
Found in top 10	170 (56.7%)	
Not found in top 50	81 (27.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.472	0.246
Num found at rank 1	52 (34.7%)	25 (16.7%)
Found in top 10	106 (70.7%)	64 (42.7%)
Not found in top 50	21 (14.0%)	60 (40.0%)



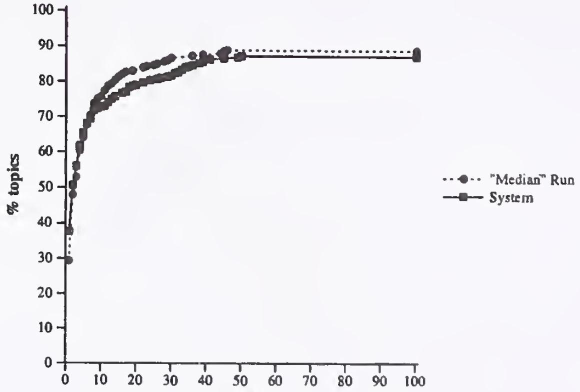
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	UAmsT03WnLM3	
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.519	
Num found at rank 1	126 (42.0%)	
Found in top 10	214 (71.3%)	
Not found in top 50	46 (15.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.597	0.440
Num found at rank 1	76 (50.7%)	50 (33.3%)
Found in top 10	113 (75.3%)	101 (67.3%)
Not found in top 50	13 (8.7%)	33 (22.0%)



Cumulative % of topics that retrieve target page by given rank

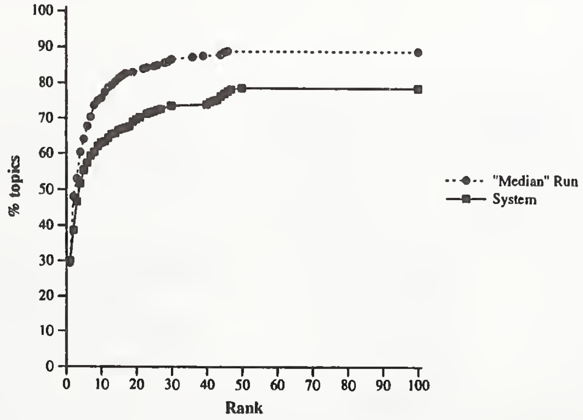
Summary Statistics		
Run ID	UAmsT03WnLn3	
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.498	
Num found at rank 1	113 (37.7%)	
Found in top 10	218 (72.7%)	
Not found in top 50	38 (12.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.586	0.410
Num found at rank 1	70 (46.7%)	43 (28.7%)
Found in top 10	121 (80.7%)	97 (64.7%)
Not found in top 50	12 (8.0%)	26 (17.3%)



Cumulative % of topics that retrieve target page by given rank

Summary Statistics	
Run ID	UAmsT03WnMSW
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Num topics	300

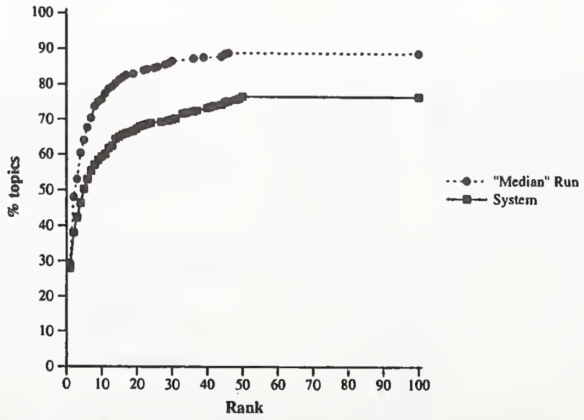
	overall	
Mean reciprocal rank	0.407	
Num found at rank 1	90 (30.0%)	
Found in top 10	189 (63.0%)	
Not found in top 50	64 (21.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.544	0.271
Num found at rank 1	63 (42.0%)	27 (18.0%)
Found in top 10	116 (77.3%)	73 (48.7%)
Not found in top 50	11 (7.3%)	53 (35.3%)



Cumulative % of topics that retrieve target page by given rank

Summary Statistics	
Run ID	UAmsT03WnOWS
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Num topics	300

	overall	
Mean reciprocal rank	0.383	
Num found at rank 1	84 (28.0%)	
Found in top 10	178 (59.3%)	
Not found in top 50	70 (23.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.510	0.257
Num found at rank 1	58 (38.7%)	26 (17.3%)
Found in top 10	111 (74.0%)	67 (44.7%)
Not found in top 50	15 (10.0%)	55 (36.7%)

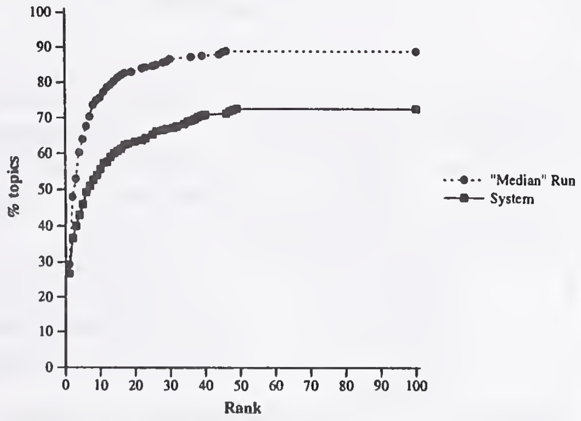


Cumulative % of topics that retrieve target page by given rank

Summary Statistics	
Run ID	uogki1c
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Num topics	300

	overall
Mean reciprocal rank	0.363
Num found at rank 1	80 (26.7%)
Found in top 10	167 (55.7%)
Not found in top 50	82 (27.3%)

	named pgs.	home pgs.
Mean reciprocal rank	0.520	0.206
Num found at rank 1	61 (40.7%)	19 (12.7%)
Found in top 10	110 (73.3%)	57 (38.0%)
Not found in top 50	13 (8.7%)	69 (46.0%)

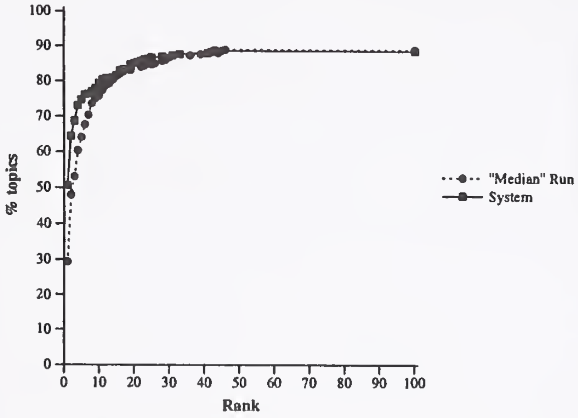


Cumulative % of topics that retrieve target page by given rank

Summary Statistics	
Run ID	uogki2ca
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics	300

	overall
Mean reciprocal rank	0.615
Num found at rank 1	152 (50.7%)
Found in top 10	238 (79.3%)
Not found in top 50	34 (11.3%)

	named pgs.	home pgs.
Mean reciprocal rank	0.613	0.617
Num found at rank 1	73 (48.7%)	79 (52.7%)
Found in top 10	123 (82.0%)	115 (76.7%)
Not found in top 50	12 (8.0%)	22 (14.7%)

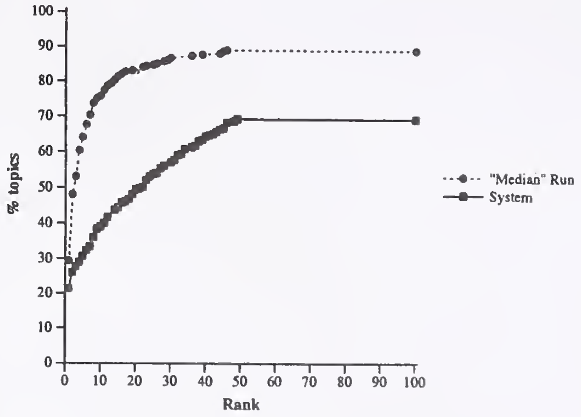


Cumulative % of topics that retrieve target page by given rank

Summary Statistics	
Run ID	uogki3cah
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics	300

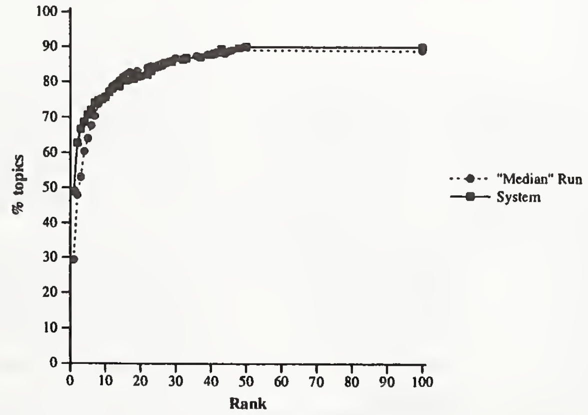
	overall
Mean reciprocal rank	0.273
Num found at rank 1	64 (21.3%)
Found in top 10	117 (39.0%)
Not found in top 50	92 (30.7%)

	named pgs.	home pgs.
Mean reciprocal rank	0.059	0.487
Num found at rank 1	2 (1.3%)	62 (41.3%)
Found in top 10	19 (12.7%)	98 (65.3%)
Not found in top 50	65 (43.3%)	27 (18.0%)



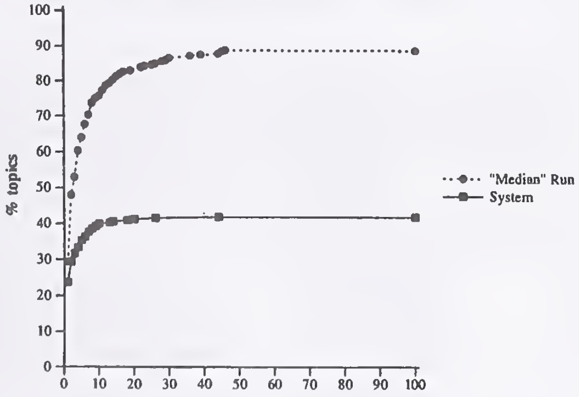
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	uogki4cahs	
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.595	
Num found at rank 1	147 (49.0%)	
Found in top 10	227 (75.7%)	
Not found in top 50	30 (10.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.554	0.636
Num found at rank 1	66 (44.0%)	81 (54.0%)
Found in top 10	109 (72.7%)	118 (78.7%)
Not found in top 50	14 (9.3%)	16 (10.7%)



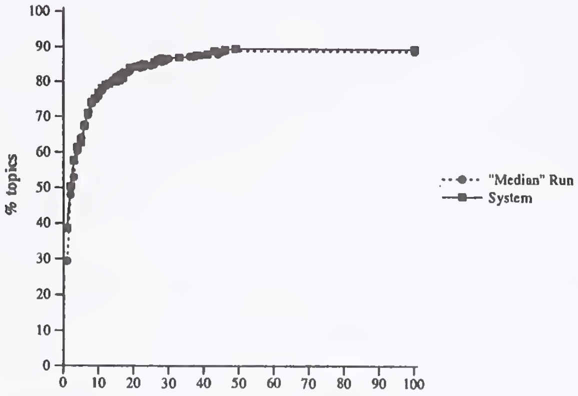
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		MU03np1
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.288	
Num found at rank 1	71 (23.7%)	
Found in top 10	120 (40.0%)	
Not found in top 50	174 (58.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.083	0.493
Num found at rank 1	11 (7.3%)	60 (40.0%)
Found in top 10	16 (10.7%)	104 (69.3%)
Not found in top 50	134 (89.3%)	40 (26.7%)



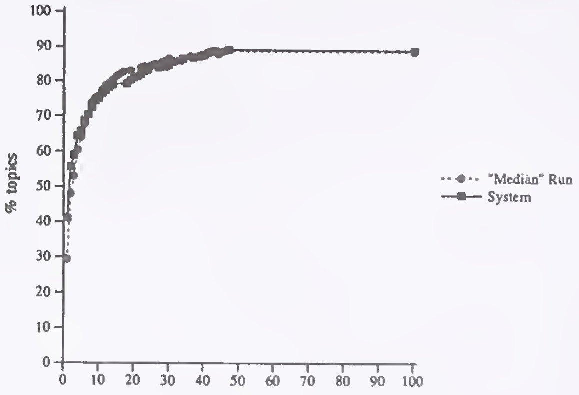
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		MU03np3
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.508	
Num found at rank 1	116 (38.7%)	
Found in top 10	230 (76.7%)	
Not found in top 50	31 (10.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.588	0.428
Num found at rank 1	72 (48.0%)	44 (29.3%)
Found in top 10	122 (81.3%)	108 (72.0%)
Not found in top 50	11 (7.3%)	20 (13.3%)



Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		MU03np4
Run Description		DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.530	
Num found at rank 1	123 (41.0%)	
Found in top 10	225 (75.0%)	
Not found in top 50	32 (10.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.597	0.464
Num found at rank 1	75 (50.0%)	48 (32.0%)
Found in top 10	117 (78.0%)	108 (72.0%)
Not found in top 50	16 (10.7%)	16 (10.7%)

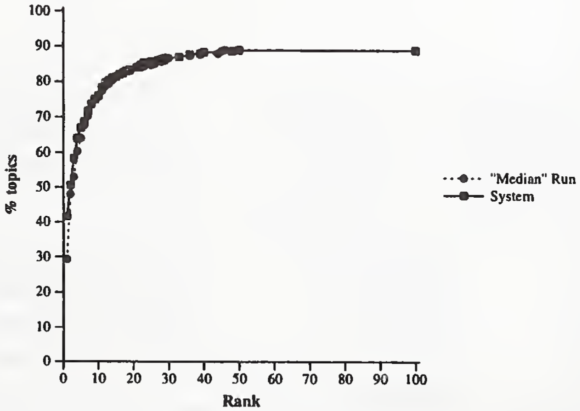


Cumulative % of topics that retrieve target page by given rank

Summary Statistics	
Run ID	MU03np5
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics	300

	overall
Mean reciprocal rank	0.527
Num found at rank 1	125 (41.7%)
Found in top 10	228 (76.0%)
Not found in top 50	33 (11.0%)

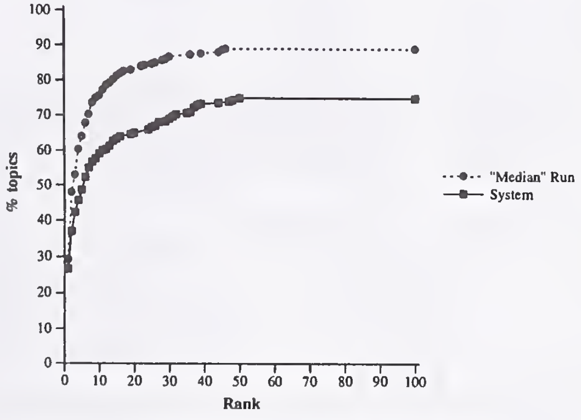
	named pgs.	home pgs.
Mean reciprocal rank	0.568	0.486
Num found at rank 1	69 (46.0%)	56 (37.3%)
Found in top 10	117 (78.0%)	111 (74.0%)
Not found in top 50	14 (9.3%)	19 (12.7%)



Cumulative % of topics that retrieve target page by given rank

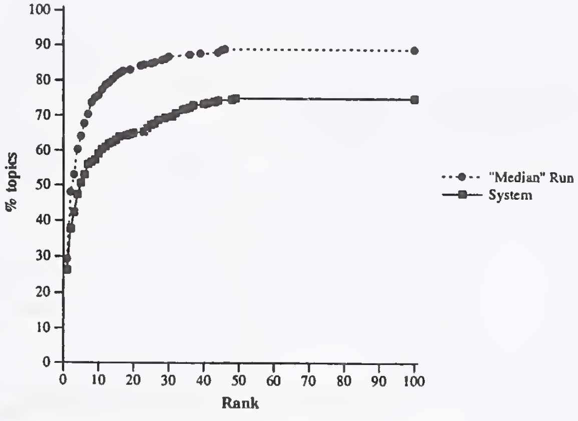
Web track, named/home page finding task results — Virginia Tech

Summary Statistics		
Run ID		VTnhpgp33
Run Description		NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.372	
Num found at rank 1	80 (26.7%)	
Found in top 10	177 (59.0%)	
Not found in top 50	75 (25.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.513	0.232
Num found at rank 1	58 (38.7%)	22 (14.7%)
Found in top 10	109 (72.7%)	68 (45.3%)
Not found in top 50	16 (10.7%)	59 (39.3%)



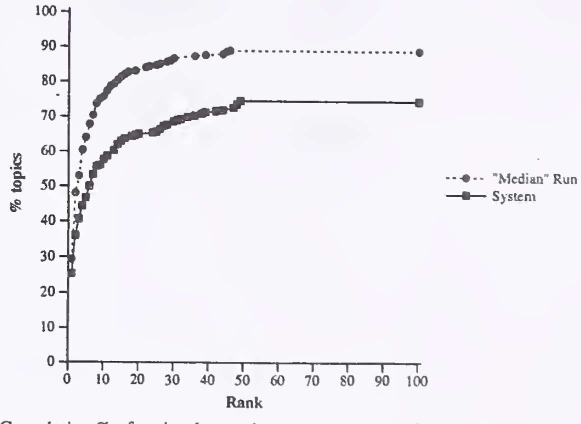
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		VTnhpgp42
Run Description		NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.374	
Num found at rank 1	79 (26.3%)	
Found in top 10	177 (59.0%)	
Not found in top 50	75 (25.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.508	0.239
Num found at rank 1	56 (37.3%)	23 (15.3%)
Found in top 10	108 (72.0%)	69 (46.0%)
Not found in top 50	17 (11.3%)	58 (38.7%)



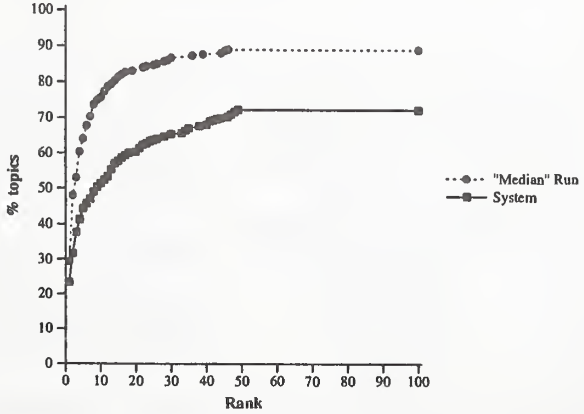
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID		VTnhpgp55
Run Description		NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Num topics		300
	overall	
Mean reciprocal rank	0.359	
Num found at rank 1	76 (25.3%)	
Found in top 10	173 (57.7%)	
Not found in top 50	76 (25.3%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.497	0.222
Num found at rank 1	56 (37.3%)	20 (13.3%)
Found in top 10	107 (71.3%)	66 (44.0%)
Not found in top 50	18 (12.0%)	58 (38.7%)



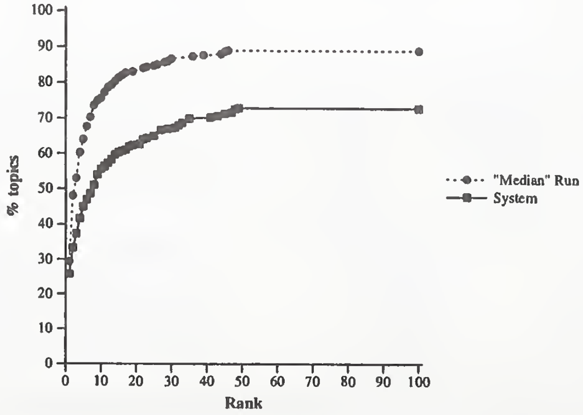
Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	VTnhpgpd4	
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.330	
Num found at rank 1	70 (23.3%)	
Found in top 10	154 (51.3%)	
Not found in top 50	83 (27.7%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.492	0.167
Num found at rank 1	58 (38.7%)	12 (8.0%)
Found in top 10	102 (68.0%)	52 (34.7%)
Not found in top 50	18 (12.0%)	65 (43.3%)



Cumulative % of topics that retrieve target page by given rank

Summary Statistics		
Run ID	VTnhpok1	
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT	
Num topics	300	
	overall	
Mean reciprocal rank	0.348	
Num found at rank 1	77 (25.7%)	
Found in top 10	167 (55.7%)	
Not found in top 50	81 (27.0%)	
	named pgs.	home pgs.
Mean reciprocal rank	0.493	0.202
Num found at rank 1	57 (38.0%)	20 (13.3%)
Found in top 10	105 (70.0%)	62 (41.3%)
Not found in top 50	20 (13.3%)	61 (40.7%)



Cumulative % of topics that retrieve target page by given rank

Summary Statistics	
Run ID:	ajouai0301
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

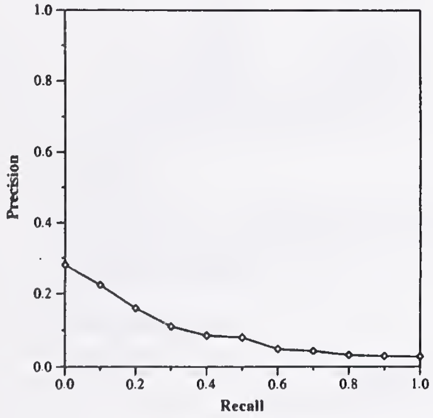
Total number of documents over all topics	
Retrieved:	49070
Relevant:	516
Rel-ret:	282

Recall Level Averages	
Recall	Precision
0.00	0.2802
0.10	0.2247
0.20	0.1604
0.30	0.1103
0.40	0.0864
0.50	0.0811
0.60	0.0497
0.70	0.0443
0.80	0.0329
0.90	0.0300
1.00	0.0285

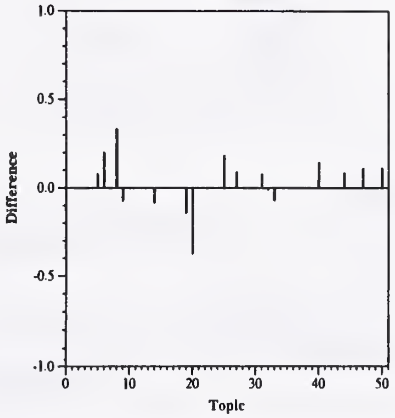
Mean average precision	
non-interpolated	0.0896

Document Level Averages	
	Precision
At 5 docs	0.1120
At 10 docs	0.0700
At 15 docs	0.0693
At 20 docs	0.0720
At 30 docs	0.0560
At 100 docs	0.0280
At 200 docs	0.0187
At 500 docs	0.0095
At 1000 docs	0.0056

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0699



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	ajouai0302
Run Description	DOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

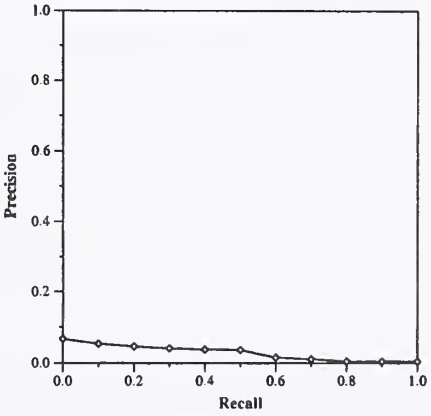
Total number of documents over all topics	
Retrieved:	46709
Relevant:	516
Rel-ret:	159

Recall Level Averages	
Recall	Precision
0.00	0.0666
0.10	0.0530
0.20	0.0457
0.30	0.0404
0.40	0.0372
0.50	0.0361
0.60	0.0151
0.70	0.0108
0.80	0.0049
0.90	0.0045
1.00	0.0042

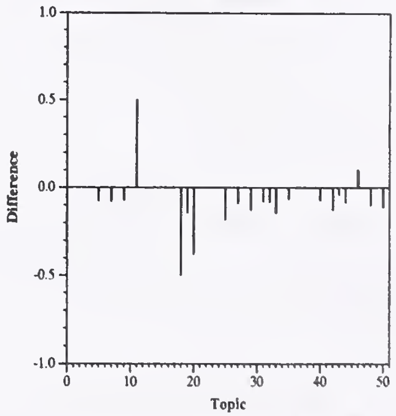
Mean average precision	
non-interpolated	0.0250

Document Level Averages	
	Precision
At 5 docs	0.0160
At 10 docs	0.0160
At 15 docs	0.0187
At 20 docs	0.0160
At 30 docs	0.0113
At 100 docs	0.0112
At 200 docs	0.0125
At 500 docs	0.0058
At 1000 docs	0.0032

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0181



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	ajouai0305
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

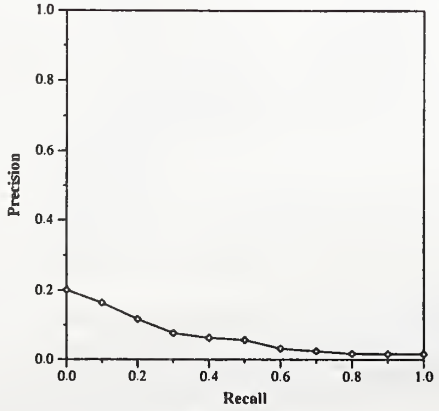
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	159

Recall Level Averages	
Recall	Precision
0.00	0.2012
0.10	0.1647
0.20	0.1173
0.30	0.0769
0.40	0.0634
0.50	0.0571
0.60	0.0323
0.70	0.0250
0.80	0.0171
0.90	0.0164
1.00	0.0162

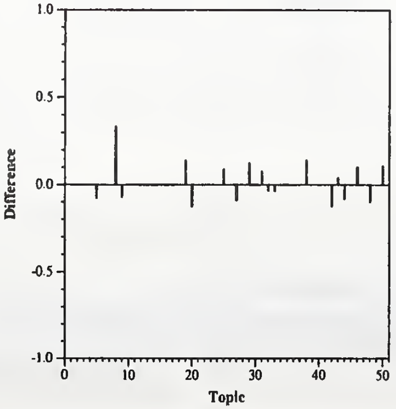
Mean average precision	
non-interpolated	0.0648

Document Level Averages	
	Precision
At 5 docs	0.0680
At 10 docs	0.0580
At 15 docs	0.0480
At 20 docs	0.0390
At 30 docs	0.0367
At 100 docs	0.0194
At 200 docs	0.0125
At 500 docs	0.0058
At 1000 docs	0.0032

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0652



Recall-Precision Curve



Difference from Median in R-Precision per Topic

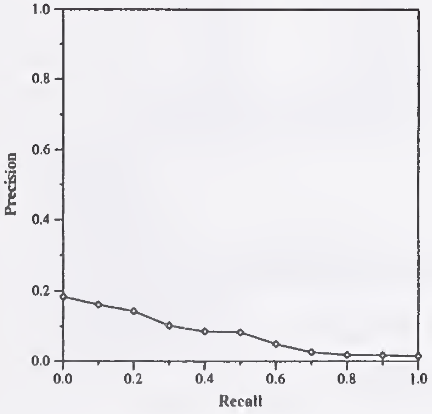
Web track, topic distillation task results — Chinese Academy of Sciences (CAS-ICT)

Summary Statistics	
Run ID:	ICTWebTD12A
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

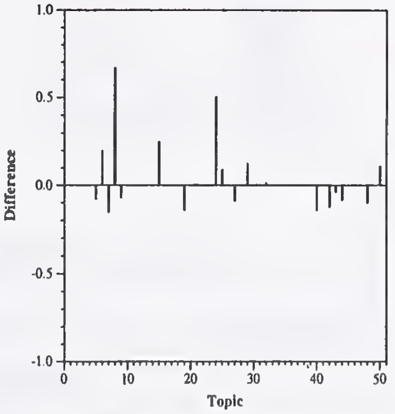
Total number of documents over all topics	
Retrieved:	49474
Relevant:	516
Rel-ret:	278

Recall Level Averages	
Recall	Precision
0.00	0.1838
0.10	0.1615
0.20	0.1434
0.30	0.1025
0.40	0.0857
0.50	0.0833
0.60	0.0496
0.70	0.0261
0.80	0.0182
0.90	0.0170
1.00	0.0146
Mean average precision	
non-interpolated	0.0728

Document Level Averages	
	Precision
At 5 docs	0.0600
At 10 docs	0.0520
At 15 docs	0.0387
At 20 docs	0.0440
At 30 docs	0.0360
At 100 docs	0.0256
At 200 docs	0.0167
At 500 docs	0.0092
At 1000 docs	0.0056
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0754



Recall-Precision Curve



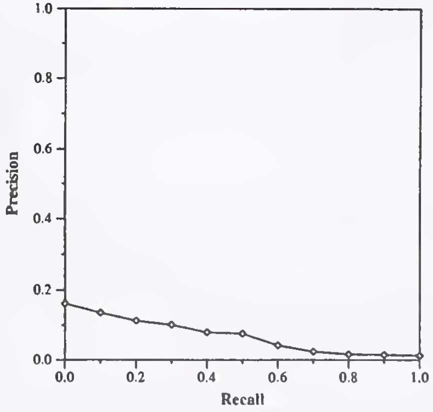
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	ICTWebTD12B
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

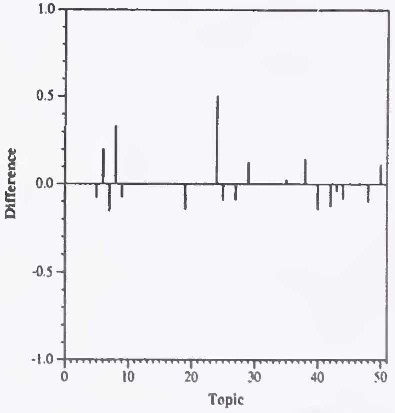
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	265

Recall Level Averages	
Recall	Precision
0.00	0.1615
0.10	0.1356
0.20	0.1128
0.30	0.1010
0.40	0.0794
0.50	0.0759
0.60	0.0423
0.70	0.0242
0.80	0.0166
0.90	0.0149
1.00	0.0127
Mean average precision	
non-interpolated	0.0639

Document Level Averages	
	Precision
At 5 docs	0.0520
At 10 docs	0.0380
At 15 docs	0.0373
At 20 docs	0.0390
At 30 docs	0.0360
At 100 docs	0.0252
At 200 docs	0.0157
At 500 docs	0.0089
At 1000 docs	0.0053
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0632



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	ICTWebTD12C
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

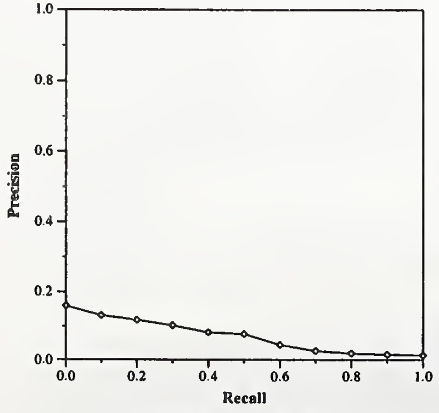
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	292

Recall Level Averages	
Recall	Precision
0.00	0.1600
0.10	0.1318
0.20	0.1182
0.30	0.1025
0.40	0.0822
0.50	0.0772
0.60	0.0451
0.70	0.0268
0.80	0.0191
0.90	0.0159
1.00	0.0136

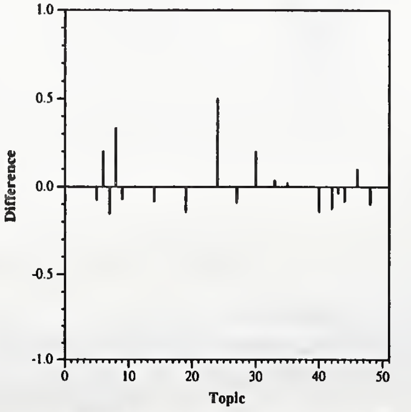
Mean average precision	
non-interpolated	0.0647

Document Level Averages	
	Precision
At 5 docs	0.0520
At 10 docs	0.0400
At 15 docs	0.0453
At 20 docs	0.0430
At 30 docs	0.0413
At 100 docs	0.0254
At 200 docs	0.0163
At 500 docs	0.0097
At 1000 docs	0.0058

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0625



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	copTdRun1
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

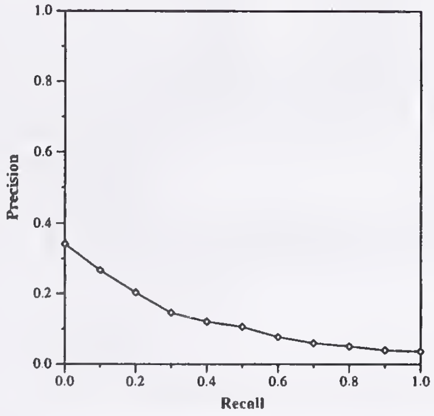
Total number of documents over all topics	
Retrieved:	49391
Relevant:	516
Rel-ret:	319

Recall Level Averages	
Recall	Precision
0.00	0.3421
0.10	0.2667
0.20	0.2032
0.30	0.1456
0.40	0.1206
0.50	0.1060
0.60	0.0770
0.70	0.0595
0.80	0.0505
0.90	0.0399
1.00	0.0362

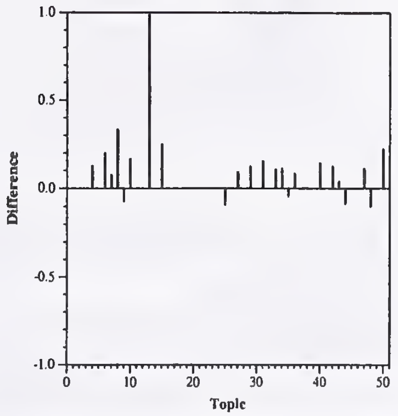
Mean average precision	
non-interpolated	0.1180

Document Level Averages	
	Precision
At 5 docs	0.1120
At 10 docs	0.0960
At 15 docs	0.0827
At 20 docs	0.0800
At 30 docs	0.0653
At 100 docs	0.0372
At 200 docs	0.0233
At 500 docs	0.0116
At 1000 docs	0.0064

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1183



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	copTdRun2
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

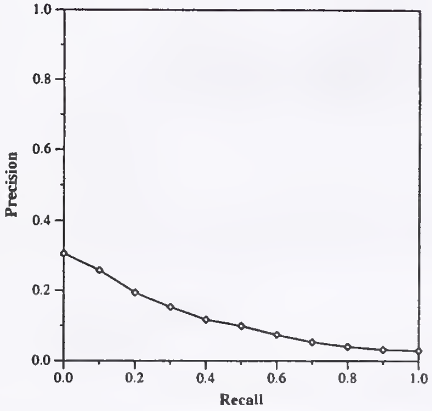
Total number of documents over all topics	
Retrieved:	49391
Relevant:	516
Rel-ret:	298

Recall Level Averages	
Recall	Precision
0.00	0.3067
0.10	0.2582
0.20	0.1939
0.30	0.1531
0.40	0.1172
0.50	0.0992
0.60	0.0741
0.70	0.0533
0.80	0.0405
0.90	0.0321
1.00	0.0292

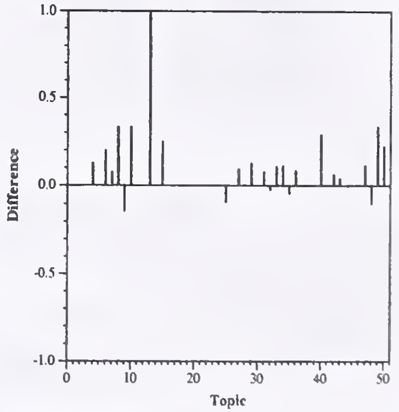
Mean average precision	
non-interpolated	0.1107

Document Level Averages	
	Precision
At 5 docs	0.1280
At 10 docs	0.0960
At 15 docs	0.0800
At 20 docs	0.0730
At 30 docs	0.0633
At 100 docs	0.0366
At 200 docs	0.0223
At 500 docs	0.0107
At 1000 docs	0.0060

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1282



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	copTdRun3
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

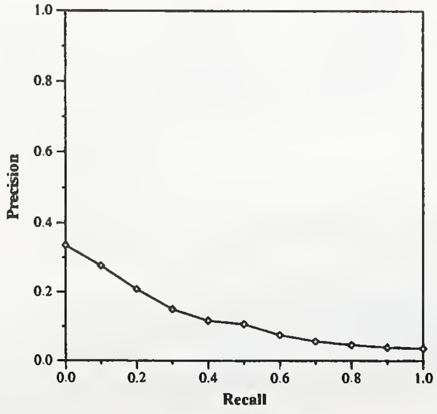
Total number of documents over all topics	
Retrieved:	49391
Relevant:	516
Rel-ret:	306

Recall Level Averages	
Recall	Precision
0.00	0.3351
0.10	0.2764
0.20	0.2083
0.30	0.1497
0.40	0.1165
0.50	0.1067
0.60	0.0750
0.70	0.0561
0.80	0.0456
0.90	0.0386
1.00	0.0351

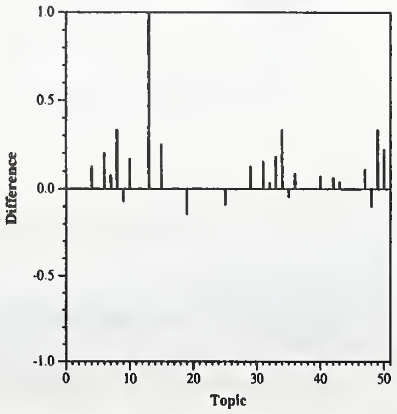
Mean average precision	
non-interpolated	0.1187

Document Level Averages	
	Precision
At 5 docs	0.1200
At 10 docs	0.0980
At 15 docs	0.0867
At 20 docs	0.0750
At 30 docs	0.0653
At 100 docs	0.0360
At 200 docs	0.0226
At 500 docs	0.0112
At 1000 docs	0.0061

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1259



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	copTdRun4
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

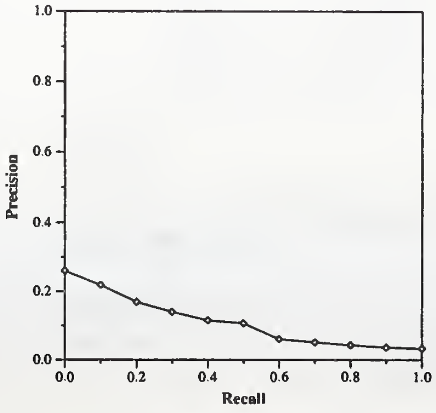
Total number of documents over all topics	
Retrieved:	49391
Relevant:	516
Rel-ret:	293

Recall Level Averages	
Recall	Precision
0.00	0.2598
0.10	0.2187
0.20	0.1689
0.30	0.1404
0.40	0.1153
0.50	0.1068
0.60	0.0603
0.70	0.0508
0.80	0.0427
0.90	0.0358
1.00	0.0327

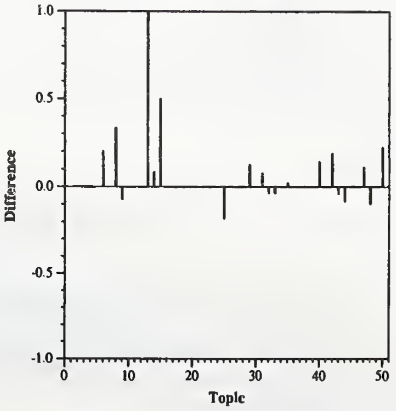
Mean average precision	
non-interpolated	0.0993

Document Level Averages	
	Precision
At 5 docs	0.0840
At 10 docs	0.0660
At 15 docs	0.0693
At 20 docs	0.0650
At 30 docs	0.0560
At 100 docs	0.0312
At 200 docs	0.0203
At 500 docs	0.0099
At 1000 docs	0.0059

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1060



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	copTdRun5
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

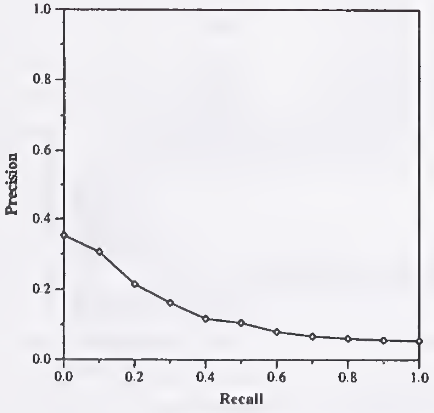
Total number of documents over all topics	
Retrieved:	49391
Relevant:	516
Rel-ret:	287

Recall Level Averages	
Recall	Precision
0.00	0.3525
0.10	0.3061
0.20	0.2158
0.30	0.1625
0.40	0.1176
0.50	0.1059
0.60	0.0805
0.70	0.0675
0.80	0.0609
0.90	0.0567
1.00	0.0546

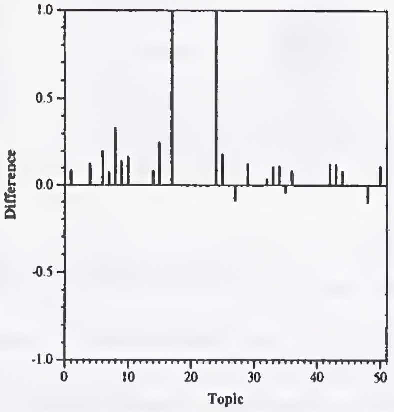
Mean average precision	
non-interpolated	0.1325

Document Level Averages	
	Precision
At 5 docs	0.1280
At 10 docs	0.0980
At 15 docs	0.0853
At 20 docs	0.0730
At 30 docs	0.0613
At 100 docs	0.0320
At 200 docs	0.0212
At 500 docs	0.0100
At 1000 docs	0.0057

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1430



Recall-Precision Curve



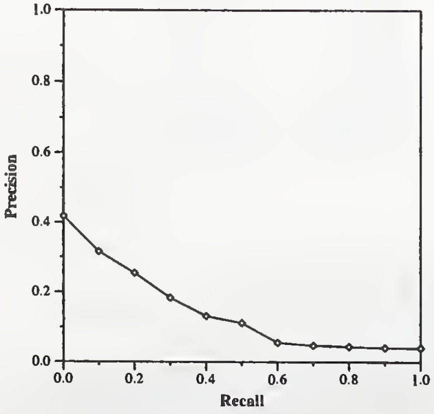
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	csiro03td01
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

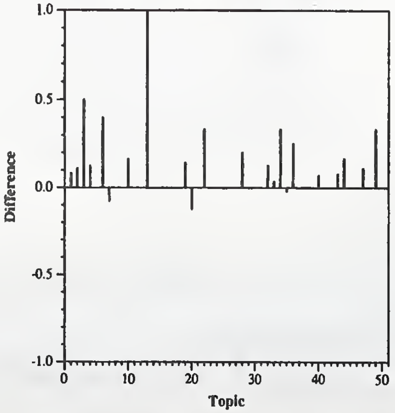
Total number of documents over all topics	
Retrieved:	48503
Relevant:	516
Rel-ret:	269

Recall Level Averages	
Recall	Precision
0.00	0.4179
0.10	0.3152
0.20	0.2546
0.30	0.1837
0.40	0.1324
0.50	0.1126
0.60	0.0560
0.70	0.0474
0.80	0.0440
0.90	0.0409
1.00	0.0402
Mean average precision	
non-interpolated	0.1354

Document Level Averages	
	Precision
At 5 docs	0.1640
At 10 docs	0.1120
At 15 docs	0.0947
At 20 docs	0.0810
At 30 docs	0.0680
At 100 docs	0.0320
At 200 docs	0.0197
At 500 docs	0.0092
At 1000 docs	0.0054
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1438



Recall-Precision Curve



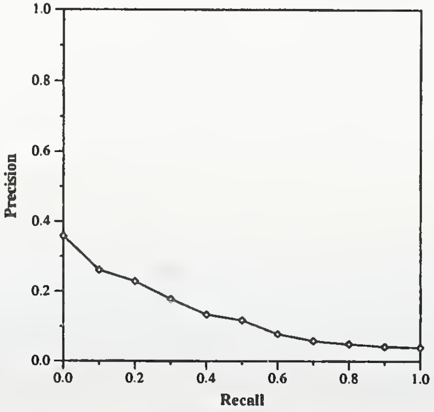
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	csiro03td02
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

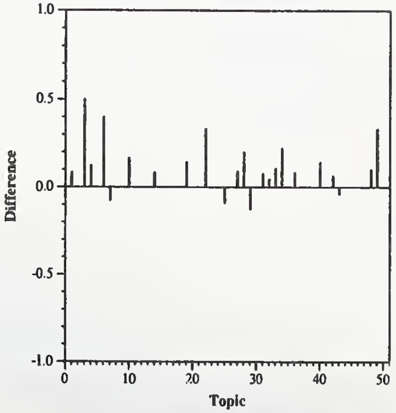
Total number of documents over all topics	
Retrieved:	48503
Relevant:	516
Rel-ret:	294

Recall Level Averages	
Recall	Precision
0.00	0.3578
0.10	0.2608
0.20	0.2289
0.30	0.1782
0.40	0.1343
0.50	0.1179
0.60	0.0779
0.70	0.0576
0.80	0.0487
0.90	0.0417
1.00	0.0391
Mean average precision	
non-interpolated	0.1272

Document Level Averages	
	Precision
At 5 docs	0.1200
At 10 docs	0.1060
At 15 docs	0.0973
At 20 docs	0.0870
At 30 docs	0.0647
At 100 docs	0.0352
At 200 docs	0.0228
At 500 docs	0.0108
At 1000 docs	0.0059
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1162



Recall-Precision Curve



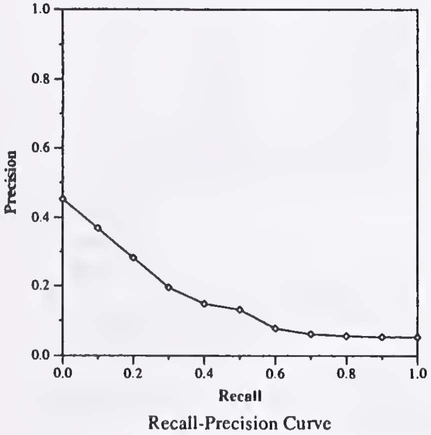
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	csiro03td03
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

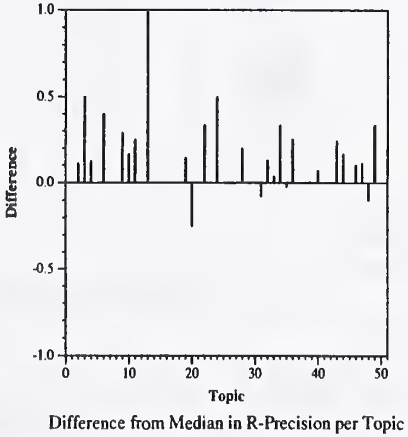
Total number of documents over all topics	
Retrieved:	49387
Relevant:	516
Rel-ret:	280

Recall Level Averages	
Recall	Precision
0.00	0.4531
0.10	0.3695
0.20	0.2827
0.30	0.1956
0.40	0.1482
0.50	0.1310
0.60	0.0775
0.70	0.0614
0.80	0.0558
0.90	0.0535
1.00	0.0526
Mean average precision	
non-interpolated	0.1543

Document Level Averages	
	Precision
At 5 docs	0.1880
At 10 docs	0.1240
At 15 docs	0.1027
At 20 docs	0.0920
At 30 docs	0.0747
At 100 docs	0.0346
At 200 docs	0.0209
At 500 docs	0.0098
At 1000 docs	0.0056
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1636



Recall-Precision Curve



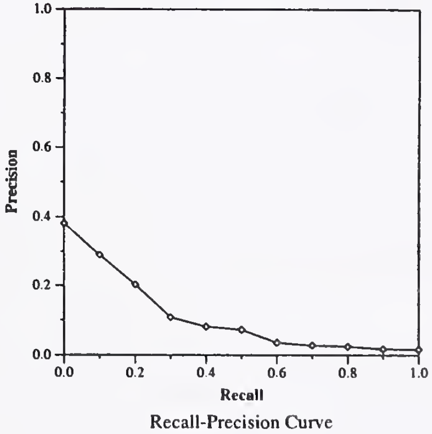
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	csiro03td04
Run Description	DOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

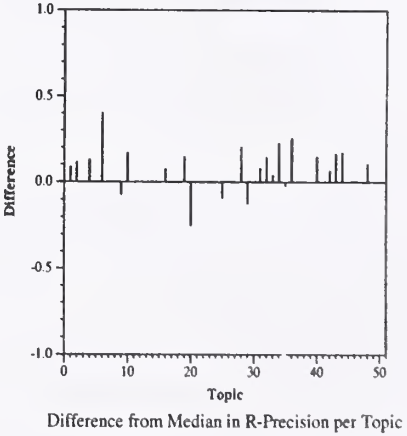
Total number of documents over all topics	
Retrieved:	48503
Relevant:	516
Rel-ret:	273

Recall Level Averages	
Recall	Precision
0.00	0.3817
0.10	0.2897
0.20	0.2025
0.30	0.1079
0.40	0.0811
0.50	0.0722
0.60	0.0355
0.70	0.0273
0.80	0.0244
0.90	0.0172
1.00	0.0161
Mean average precision	
non-interpolated	0.1004

Document Level Averages	
	Precision
At 5 docs	0.1600
At 10 docs	0.1200
At 15 docs	0.0907
At 20 docs	0.0760
At 30 docs	0.0600
At 100 docs	0.0304
At 200 docs	0.0185
At 500 docs	0.0092
At 1000 docs	0.0055
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0988



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	csiro03td05
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

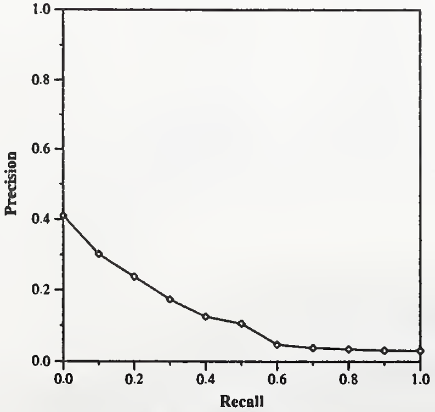
Total number of documents over all topics	
Retrieved:	48503
Relevant:	516
Rel-ret:	270

Recall Level Averages	
Recall	Precision
0.00	0.4103
0.10	0.3028
0.20	0.2385
0.30	0.1736
0.40	0.1250
0.50	0.1056
0.60	0.0473
0.70	0.0380
0.80	0.0344
0.90	0.0310
1.00	0.0303

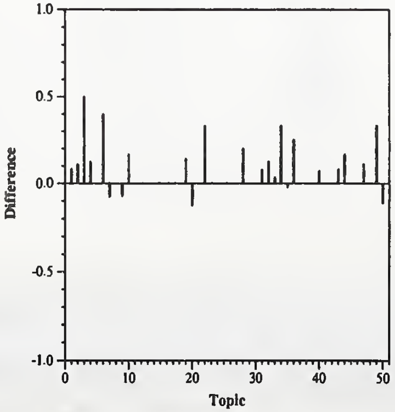
Mean average precision	
non-interpolated	0.1258

Document Level Averages	
	Precision
At 5 docs	0.1600
At 10 docs	0.1080
At 15 docs	0.0920
At 20 docs	0.0820
At 30 docs	0.0673
At 100 docs	0.0322
At 200 docs	0.0197
At 500 docs	0.0093
At 1000 docs	0.0054

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1217



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Web track, topic distillation task results — Fondazione Ugo Bordoni

Summary Statistics	
Run ID:	fub03InBMt
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

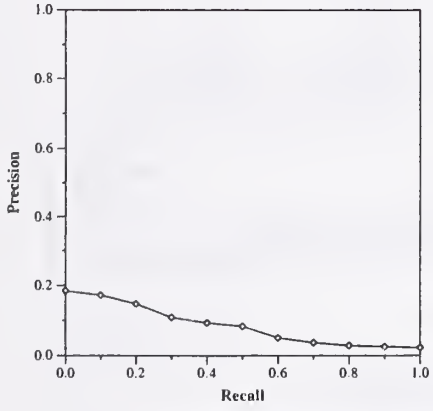
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	352

Recall Level Averages	
Recall	Precision
0.00	0.1848
0.10	0.1727
0.20	0.1473
0.30	0.1085
0.40	0.0929
0.50	0.0832
0.60	0.0502
0.70	0.0366
0.80	0.0282
0.90	0.0251
1.00	0.0226

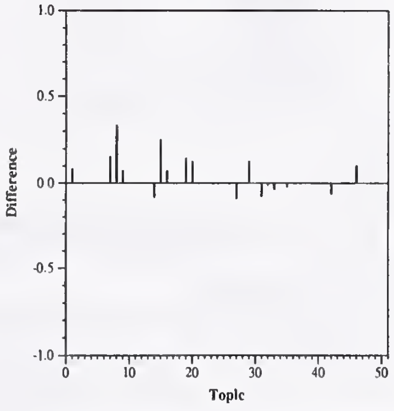
Mean average precision	
non-interpolated	0.0775

Document Level Averages	
	Precision
At 5 docs	0.0680
At 10 docs	0.0640
At 15 docs	0.0640
At 20 docs	0.0580
At 30 docs	0.0467
At 100 docs	0.0270
At 200 docs	0.0197
At 500 docs	0.0117
At 1000 docs	0.0070

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0783



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	fub03InLBo1t
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

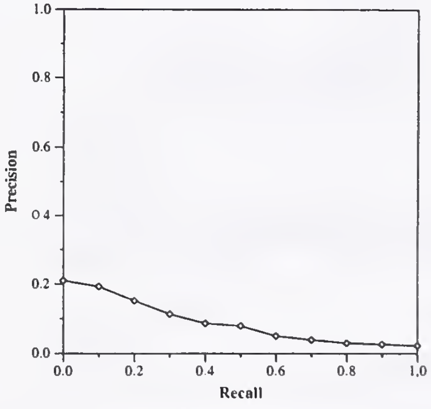
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	359

Recall Level Averages	
Recall	Precision
0.00	0.2096
0.10	0.1927
0.20	0.1519
0.30	0.1137
0.40	0.0866
0.50	0.0795
0.60	0.0505
0.70	0.0390
0.80	0.0298
0.90	0.0262
1.00	0.0222

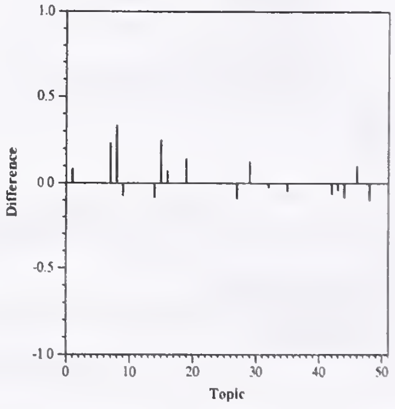
Mean average precision	
non-interpolated	0.0810

Document Level Averages	
	Precision
At 5 docs	0.0760
At 10 docs	0.0580
At 15 docs	0.0587
At 20 docs	0.0570
At 30 docs	0.0480
At 100 docs	0.0272
At 200 docs	0.0201
At 500 docs	0.0121
At 1000 docs	0.0072

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0716



Recall-Precision Curve



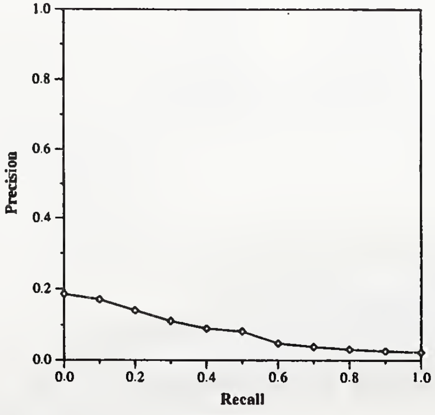
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	fub03InLBt
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

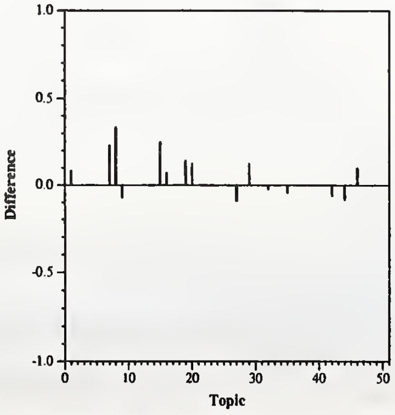
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	367

Recall Level Averages	
Recall	Precision
0.00	0.1862
0.10	0.1713
0.20	0.1412
0.30	0.1118
0.40	0.0909
0.50	0.0827
0.60	0.0486
0.70	0.0388
0.80	0.0312
0.90	0.0268
1.00	0.0230
Mean average precision	
non-interpolated	0.0778

Document Level Averages	
	Precision
At 5 docs	0.0640
At 10 docs	0.0620
At 15 docs	0.0600
At 20 docs	0.0580
At 30 docs	0.0473
At 100 docs	0.0274
At 200 docs	0.0205
At 500 docs	0.0124
At 1000 docs	0.0073
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0786



Recall-Precision Curve



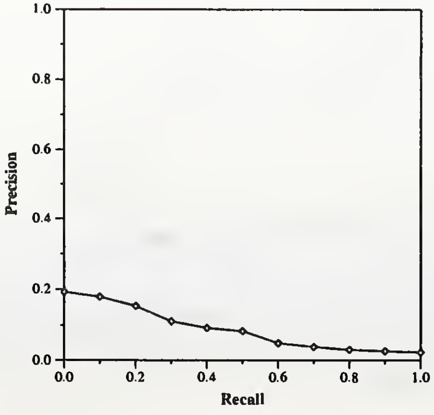
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	fub03IneBBt
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

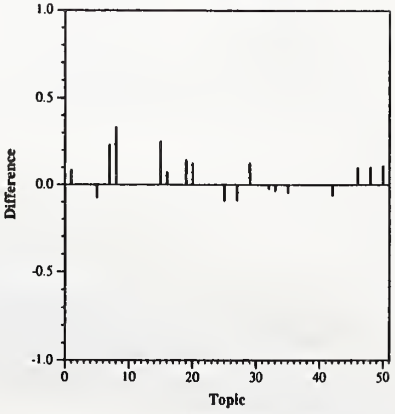
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	366

Recall Level Averages	
Recall	Precision
0.00	0.1930
0.10	0.1794
0.20	0.1539
0.30	0.1113
0.40	0.0926
0.50	0.0838
0.60	0.0494
0.70	0.0386
0.80	0.0305
0.90	0.0268
1.00	0.0234
Mean average precision	
non-interpolated	0.0799

Document Level Averages	
	Precision
At 5 docs	0.0680
At 10 docs	0.0640
At 15 docs	0.0667
At 20 docs	0.0590
At 30 docs	0.0460
At 100 docs	0.0272
At 200 docs	0.0203
At 500 docs	0.0122
At 1000 docs	0.0073
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0818



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	fub03IneBMt
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

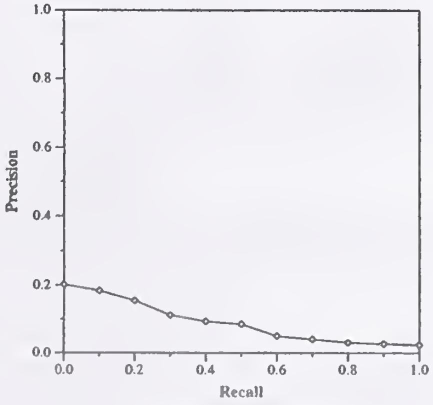
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	367

Recall Level Averages	
Recall	Precision
0.00	0.2016
0.10	0.1841
0.20	0.1549
0.30	0.1121
0.40	0.0933
0.50	0.0851
0.60	0.0496
0.70	0.0401
0.80	0.0304
0.90	0.0266
1.00	0.0232

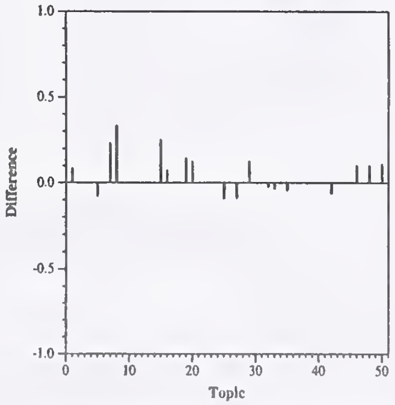
Mean average precision	
non-interpolated	0.0818

Document Level Averages	
	Precision
At 5 docs	0.0680
At 10 docs	0.0620
At 15 docs	0.0680
At 20 docs	0.0590
At 30 docs	0.0467
At 100 docs	0.0276
At 200 docs	0.0201
At 500 docs	0.0124
At 1000 docs	0.0073

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0818



Recall-Precision Curve



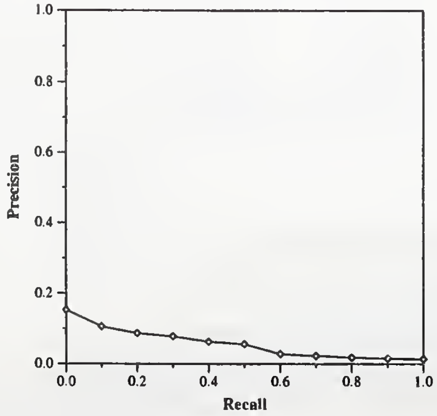
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	humTD03l
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

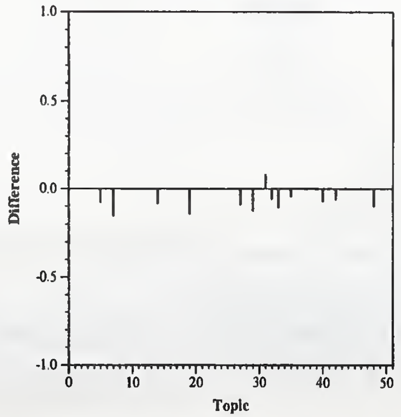
Total number of documents over all topics	
Retrieved:	49380
Relevant:	516
Rel-ret:	309

Recall Level Averages	
Recall	Precision
0.00	0.1518
0.10	0.1050
0.20	0.0861
0.30	0.0771
0.40	0.0620
0.50	0.0550
0.60	0.0275
0.70	0.0225
0.80	0.0178
0.90	0.0153
1.00	0.0137
Mean average precision	
non-interpolated	0.0512

Document Level Averages	
	Precision
At 5 docs	0.0480
At 10 docs	0.0440
At 15 docs	0.0320
At 20 docs	0.0310
At 30 docs	0.0253
At 100 docs	0.0198
At 200 docs	0.0158
At 500 docs	0.0093
At 1000 docs	0.0062
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0361



Recall-Precision Curve



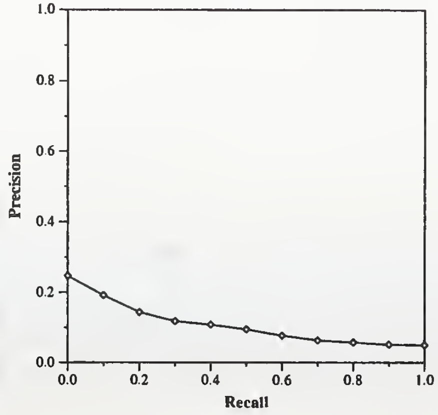
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	humTD03pl
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

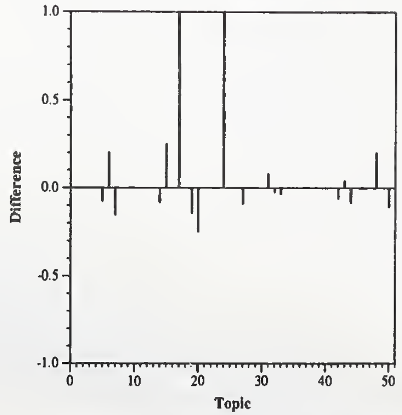
Total number of documents over all topics	
Retrieved:	49380
Relevant:	516
Rel-ret:	326

Recall Level Averages	
Recall	Precision
0.00	0.2473
0.10	0.1918
0.20	0.1435
0.30	0.1184
0.40	0.1081
0.50	0.0952
0.60	0.0769
0.70	0.0634
0.80	0.0577
0.90	0.0514
1.00	0.0499
Mean average precision	
non-interpolated	0.1003

Document Level Averages	
	Precision
At 5 docs	0.0680
At 10 docs	0.0560
At 15 docs	0.0560
At 20 docs	0.0520
At 30 docs	0.0480
At 100 docs	0.0262
At 200 docs	0.0181
At 500 docs	0.0104
At 1000 docs	0.0065
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0899



Recall-Precision Curve



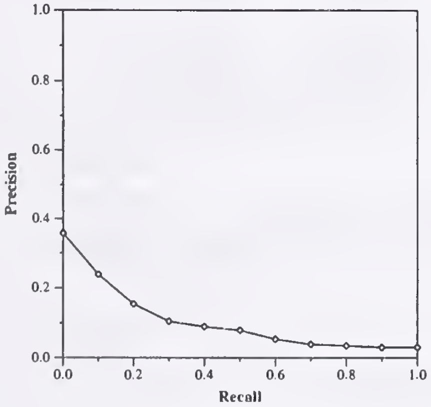
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	humTD03uhpl
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

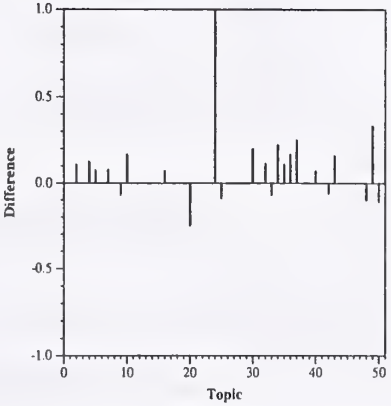
Total number of documents over all topics	
Retrieved:	49380
Relevant:	516
Rel-ret:	287

Recall Level Averages	
Recall	Precision
0.00	0.3570
0.10	0.2388
0.20	0.1546
0.30	0.1049
0.40	0.0897
0.50	0.0798
0.60	0.0541
0.70	0.0390
0.80	0.0349
0.90	0.0303
1.00	0.0303
Mean average precision	
non-interpolated	0.0978

Document Level Averages	
	Precision
At 5 docs	0.1320
At 10 docs	0.1020
At 15 docs	0.0800
At 20 docs	0.0690
At 30 docs	0.0560
At 100 docs	0.0292
At 200 docs	0.0179
At 500 docs	0.0098
At 1000 docs	0.0057
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1069



Rccall-Precision Curve



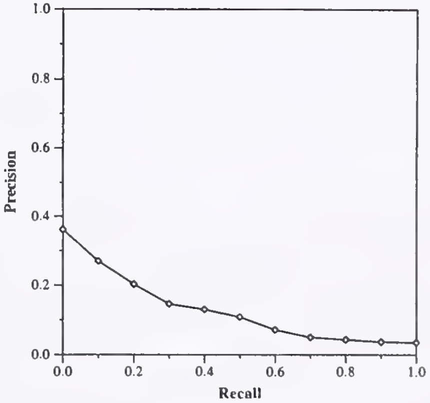
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	humTD03up
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

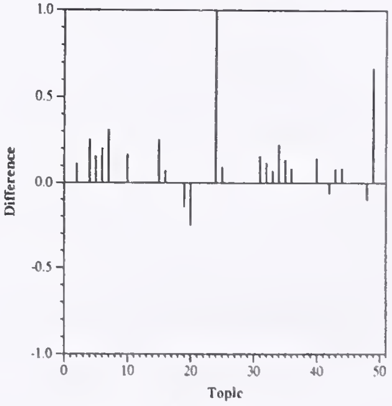
Total number of documents over all topics	
Retrieved:	48511
Relevant:	516
Rel-ret:	358

Recall Level Averages	
Recall	Precision
0.00	0.3616
0.10	0.2698
0.20	0.2033
0.30	0.1466
0.40	0.1304
0.50	0.1088
0.60	0.0721
0.70	0.0506
0.80	0.0438
0.90	0.0372
1.00	0.0354
Mean average precision	
non-interpolated	0.1198

Document Level Averages	
	Precision
At 5 docs	0.1480
At 10 docs	0.1240
At 15 docs	0.1013
At 20 docs	0.0890
At 30 docs	0.0693
At 100 docs	0.0370
At 200 docs	0.0222
At 500 docs	0.0123
At 1000 docs	0.0072
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1328



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	humTD03upl
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

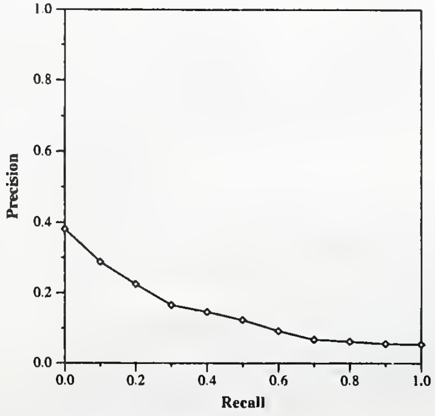
Total number of documents over all topics	
Retrieved:	49380
Relevant:	516
Rel-ret:	375

Recall Level Averages	
Recall	Precision
0.00	0.3816
0.10	0.2885
0.20	0.2248
0.30	0.1663
0.40	0.1467
0.50	0.1236
0.60	0.0922
0.70	0.0675
0.80	0.0618
0.90	0.0556
1.00	0.0536

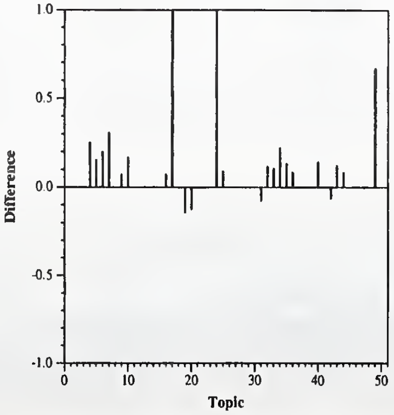
Mean average precision	
non-interpolated	0.1387

Document Level Averages	
	Precision
At 5 docs	0.1440
At 10 docs	0.1280
At 15 docs	0.1000
At 20 docs	0.0920
At 30 docs	0.0727
At 100 docs	0.0370
At 200 docs	0.0236
At 500 docs	0.0129
At 1000 docs	0.0075

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1485



Recall-Precision Curve



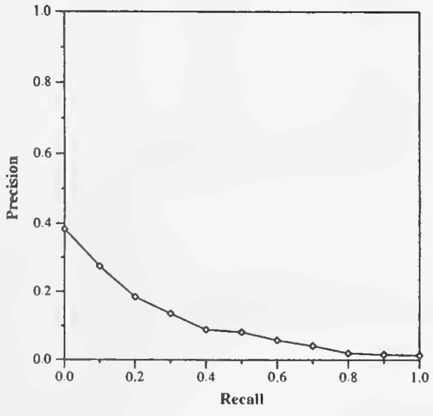
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	JuruFull
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

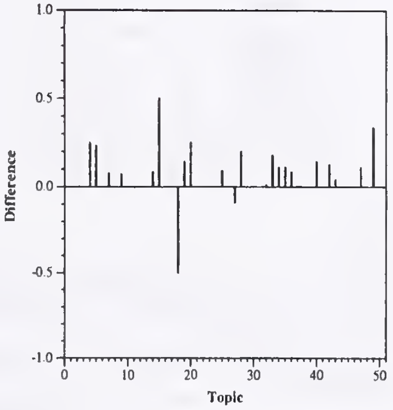
Total number of documents over all topics	
Retrieved:	7500
Relevant:	516
Rel-ret:	217

Recall Level Averages	
Recall	Precision
0.00	0.3833
0.10	0.2743
0.20	0.1846
0.30	0.1359
0.40	0.0889
0.50	0.0815
0.60	0.0575
0.70	0.0410
0.80	0.0196
0.90	0.0155
1.00	0.0134
Mean average precision	
non-interpolated	0.1008

Document Level Averages	
	Precision
At 5 docs	0.1240
At 10 docs	0.1220
At 15 docs	0.1000
At 20 docs	0.0830
At 30 docs	0.0727
At 100 docs	0.0382
At 200 docs	0.0217
At 500 docs	0.0087
At 1000 docs	0.0043
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1079



Recall-Precision Curve



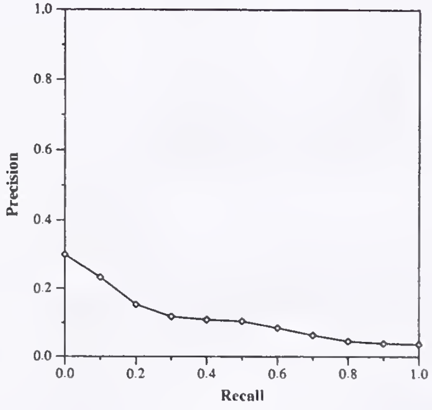
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	JuruNoAnchor
Run Description	DOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

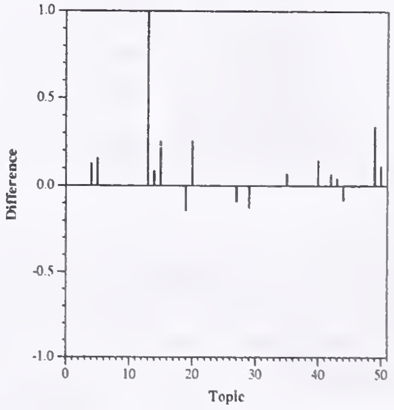
Total number of documents over all topics	
Retrieved:	7500
Relevant:	516
Rel-ret:	194

Recall Level Averages	
Recall	Precision
0.00	0.2989
0.10	0.2330
0.20	0.1530
0.30	0.1177
0.40	0.1087
0.50	0.1041
0.60	0.0845
0.70	0.0626
0.80	0.0452
0.90	0.0386
1.00	0.0360
Mean average precision	
non-interpolated	0.1041

Document Level Averages	
	Precision
At 5 docs	0.1040
At 10 docs	0.0880
At 15 docs	0.0707
At 20 docs	0.0640
At 30 docs	0.0587
At 100 docs	0.0322
At 200 docs	0.0194
At 500 docs	0.0078
At 1000 docs	0.0039
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1004



Recall-Precision Curve



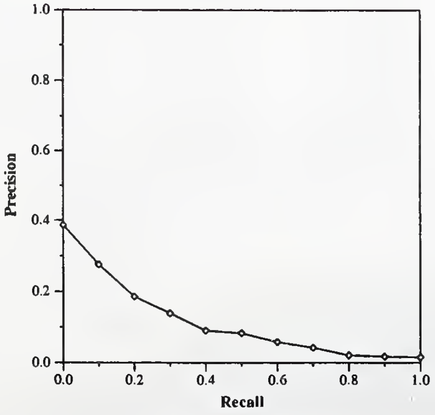
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	JuruNoCohes
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

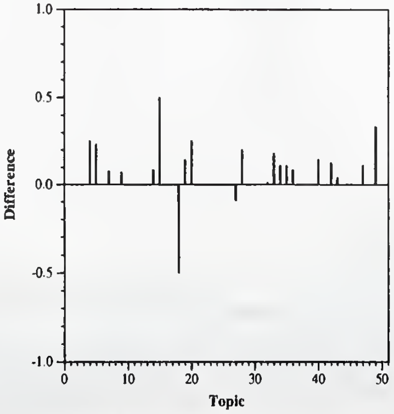
Total number of documents over all topics	
Retrieved:	7500
Relevant:	516
Rel-ret:	217

Recall Level Averages	
Recall	Precision
0.00	0.3859
0.10	0.2768
0.20	0.1861
0.30	0.1387
0.40	0.0908
0.50	0.0835
0.60	0.0583
0.70	0.0421
0.80	0.0207
0.90	0.0172
1.00	0.0151
Mean average precision	
non-interpolated	0.1022

Document Level Averages	
	Precision
At 5 docs	0.1240
At 10 docs	0.1220
At 15 docs	0.0987
At 20 docs	0.0820
At 30 docs	0.0727
At 100 docs	0.0378
At 200 docs	0.0217
At 500 docs	0.0087
At 1000 docs	0.0043
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1061



Recall-Precision Curve



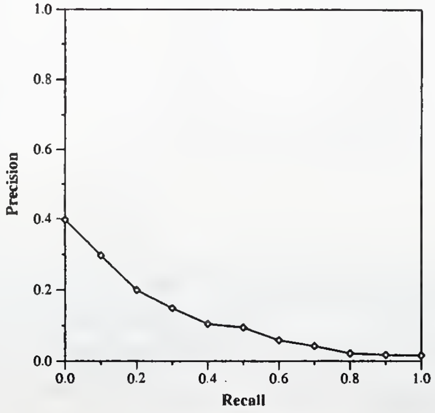
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	JuruNoQDiff
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

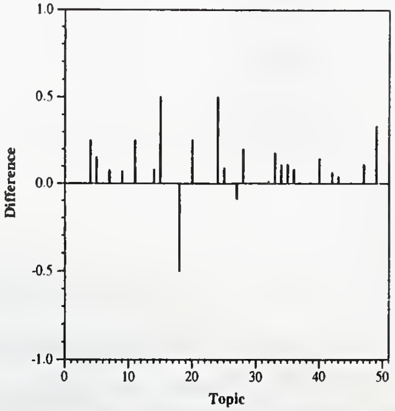
Total number of documents over all topics	
Retrieved:	7500
Relevant:	516
Rel-ret:	216

Recall Level Averages	
Recall	Precision
0.00	0.3971
0.10	0.2972
0.20	0.1997
0.30	0.1491
0.40	0.1045
0.50	0.0950
0.60	0.0588
0.70	0.0423
0.80	0.0216
0.90	0.0176
1.00	0.0157
Mean average precision	
non-interpolated	0.1091

Document Level Averages	
	Precision
At 5 docs	0.1360
At 10 docs	0.1220
At 15 docs	0.0947
At 20 docs	0.0780
At 30 docs	0.0707
At 100 docs	0.0378
At 200 docs	0.0216
At 500 docs	0.0086
At 1000 docs	0.0043
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1173



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	JuruNoSS
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

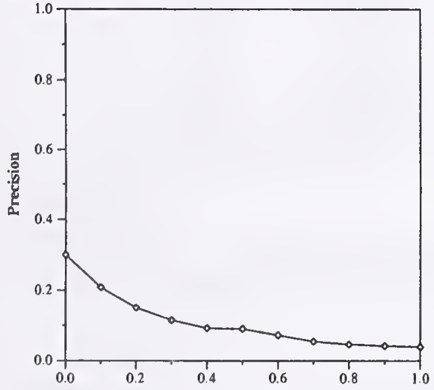
Total number of documents over all topics	
Retrieved:	7500
Relevant:	516
Rel-ret:	217

Recall Level Averages	
Recall	Precision
0.00	0.3010
0.10	0.2084
0.20	0.1502
0.30	0.1147
0.40	0.0927
0.50	0.0906
0.60	0.0729
0.70	0.0552
0.80	0.0464
0.90	0.0426
1.00	0.0396

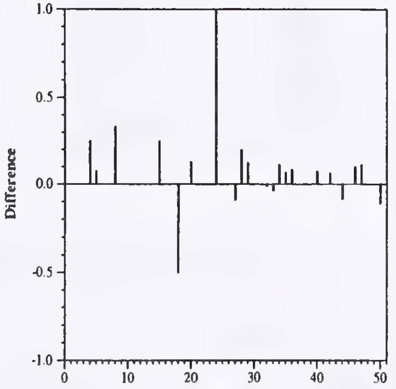
Mean average precision	
non-interpolated	0.0982

Document Level Averages	
	Precision
At 5 docs	0.1160
At 10 docs	0.0860
At 15 docs	0.0720
At 20 docs	0.0620
At 30 docs	0.0553
At 100 docs	0.0346
At 200 docs	0.0217
At 500 docs	0.0087
At 1000 docs	0.0043

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0995



Recall-Precision Curve



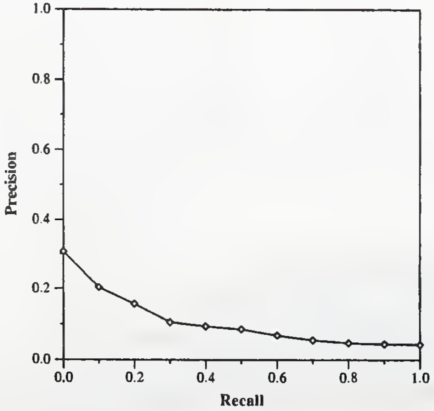
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	widittdb1
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

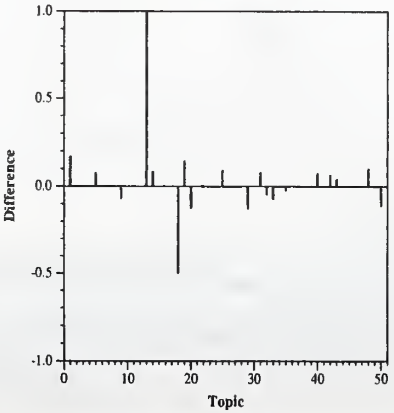
Total number of documents over all topics	
Retrieved:	49281
Relevant:	516
Rel-ret:	338

Recall Level Averages	
Recall	Precision
0.00	0.3075
0.10	0.2041
0.20	0.1568
0.30	0.1053
0.40	0.0941
0.50	0.0856
0.60	0.0683
0.70	0.0552
0.80	0.0470
0.90	0.0444
1.00	0.0427
Mean average precision	
non-interpolated	0.1016

Document Level Averages	
	Precision
At 5 docs	0.1000
At 10 docs	0.0760
At 15 docs	0.0693
At 20 docs	0.0610
At 30 docs	0.0473
At 100 docs	0.0304
At 200 docs	0.0207
At 500 docs	0.0114
At 1000 docs	0.0068
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0736



Recall-Precision Curve



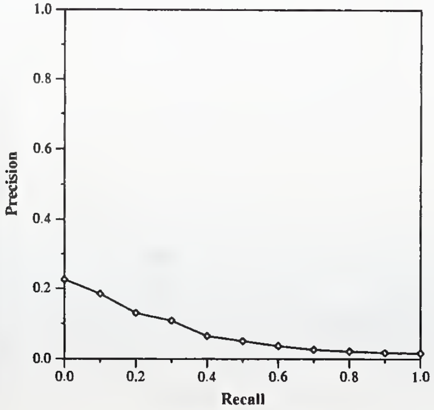
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	widittdb1r1
Run Description	NODOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

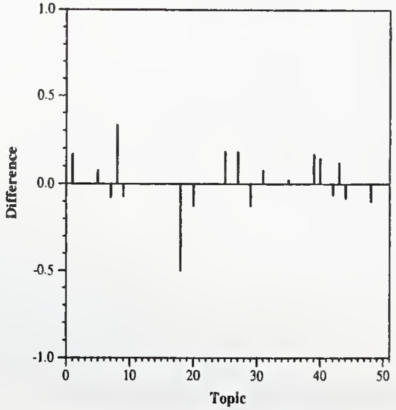
Total number of documents over all topics	
Retrieved:	10000
Relevant:	516
Rel-ret:	207

Recall Level Averages	
Recall	Precision
0.00	0.2255
0.10	0.1845
0.20	0.1296
0.30	0.1082
0.40	0.0646
0.50	0.0504
0.60	0.0368
0.70	0.0264
0.80	0.0216
0.90	0.0178
1.00	0.0164
Mean average precision	
non-interpolated	0.0687

Document Level Averages	
	Precision
At 5 docs	0.0880
At 10 docs	0.0880
At 15 docs	0.0707
At 20 docs	0.0640
At 30 docs	0.0527
At 100 docs	0.0302
At 200 docs	0.0207
At 500 docs	0.0083
At 1000 docs	0.0041
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0634



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Web track, topic distillation task results — Indiana University, Bloomington

Summary Statistics	
Run ID:	widittdf1r1
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

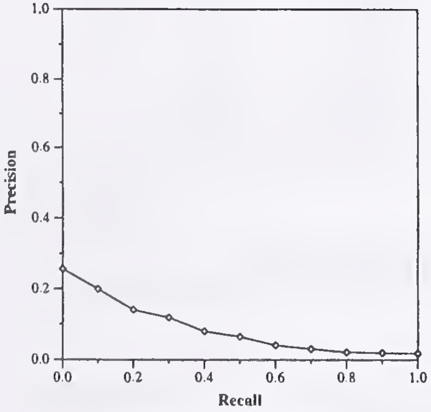
Total number of documents over all topics	
Retrieved:	10232
Relevant:	516
Rel-ret:	215

Recall Level Averages	
Recall	Precision
0.00	0.2565
0.10	0.1997
0.20	0.1406
0.30	0.1189
0.40	0.0805
0.50	0.0654
0.60	0.0411
0.70	0.0312
0.80	0.0216
0.90	0.0198
1.00	0.0182

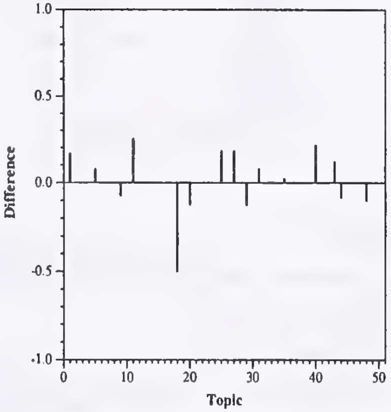
Mean average precision	
non-interpolated	0.0787

Document Level Averages	
	Precision
At 5 docs	0.1040
At 10 docs	0.0980
At 15 docs	0.0867
At 20 docs	0.0750
At 30 docs	0.0607
At 100 docs	0.0334
At 200 docs	0.0214
At 500 docs	0.0086
At 1000 docs	0.0043

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0626



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	widittdf1r2
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

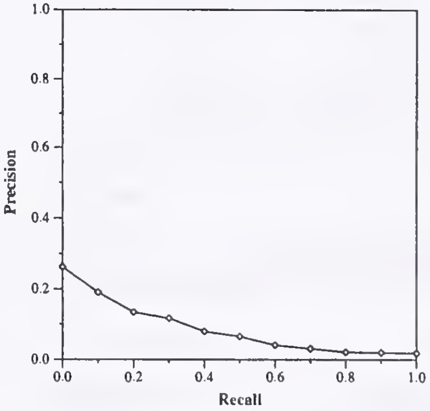
Total number of documents over all topics	
Retrieved:	10232
Relevant:	516
Rel-ret:	215

Recall Level Averages	
Recall	Precision
0.00	0.2626
0.10	0.1908
0.20	0.1342
0.30	0.1169
0.40	0.0795
0.50	0.0654
0.60	0.0408
0.70	0.0309
0.80	0.0213
0.90	0.0198
1.00	0.0182

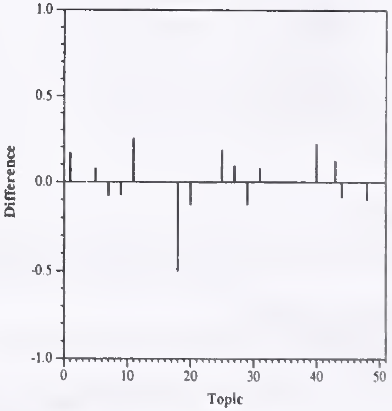
Mean average precision	
non-interpolated	0.0773

Document Level Averages	
	Precision
At 5 docs	0.0920
At 10 docs	0.0880
At 15 docs	0.0800
At 20 docs	0.0720
At 30 docs	0.0627
At 100 docs	0.0332
At 200 docs	0.0214
At 500 docs	0.0086
At 1000 docs	0.0043

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0588



Recall-Precision Curve



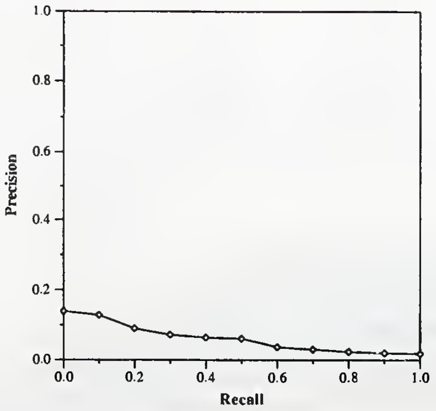
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	Merc1td
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

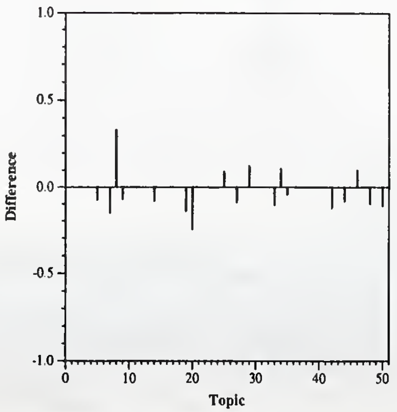
Total number of documents over all topics	
Retrieved:	49465
Relevant:	516
Rel-ret:	281

Recall Level Averages	
Recall	Precision
0.00	0.1389
0.10	0.1281
0.20	0.0907
0.30	0.0727
0.40	0.0647
0.50	0.0617
0.60	0.0372
0.70	0.0303
0.80	0.0237
0.90	0.0202
1.00	0.0178
Mean average precision	
non-interpolated	0.0555

Document Level Averages	
	Precision
At 5 docs	0.0400
At 10 docs	0.0400
At 15 docs	0.0360
At 20 docs	0.0330
At 30 docs	0.0313
At 100 docs	0.0216
At 200 docs	0.0145
At 500 docs	0.0088
At 1000 docs	0.0056
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0433



Recall-Precision Curve



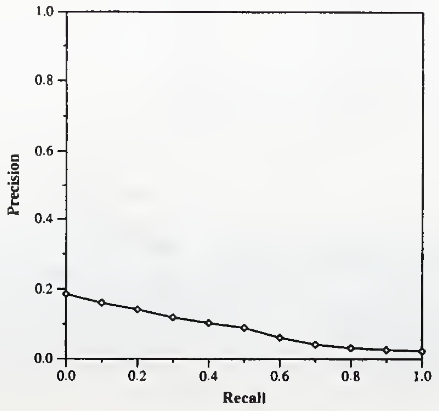
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	Merc1ti
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

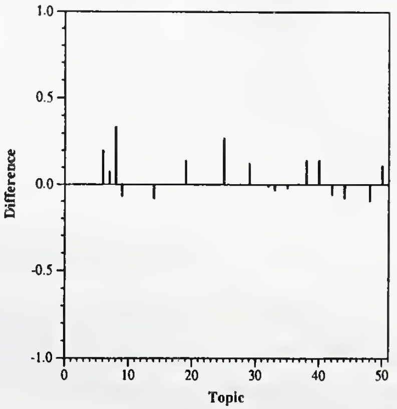
Total number of documents over all topics	
Retrieved:	48971
Relevant:	516
Rel-ret:	352

Recall Level Averages	
Recall	Precision
0.00	0.1858
0.10	0.1604
0.20	0.1417
0.30	0.1187
0.40	0.1027
0.50	0.0896
0.60	0.0615
0.70	0.0416
0.80	0.0317
0.90	0.0265
1.00	0.0229
Mean average precision	
non-interpolated	0.0818

Document Level Averages	
	Precision
At 5 docs	0.0720
At 10 docs	0.0720
At 15 docs	0.0653
At 20 docs	0.0560
At 30 docs	0.0473
At 100 docs	0.0290
At 200 docs	0.0205
At 500 docs	0.0110
At 1000 docs	0.0070
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0784



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	Merc2tm
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

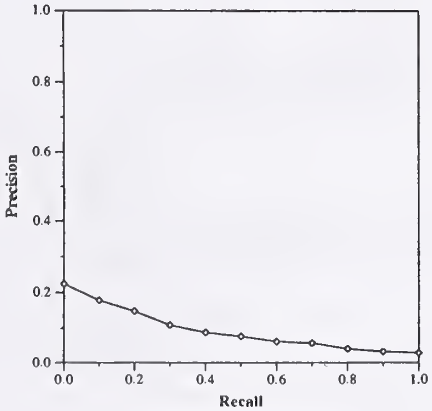
Total number of documents over all topics	
Retrieved:	48967
Relevant:	516
Rel-ret:	385

Recall Level Averages	
Recall	Precision
0.00	0.2241
0.10	0.1783
0.20	0.1478
0.30	0.1092
0.40	0.0883
0.50	0.0766
0.60	0.0615
0.70	0.0572
0.80	0.0405
0.90	0.0329
1.00	0.0286

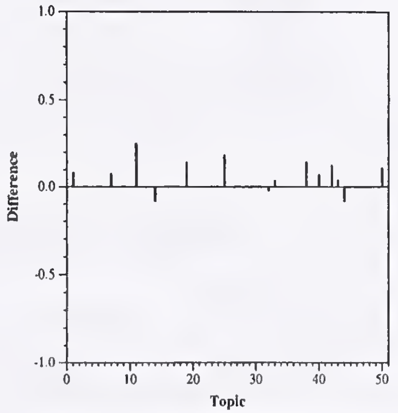
Mean average precision	
non-interpolated	0.0870

Document Level Averages	
	Precision
At 5 docs	0.0960
At 10 docs	0.0760
At 15 docs	0.0667
At 20 docs	0.0660
At 30 docs	0.0593
At 100 docs	0.0312
At 200 docs	0.0225
At 500 docs	0.0130
At 1000 docs	0.0077

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0783



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	Merc2tp
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

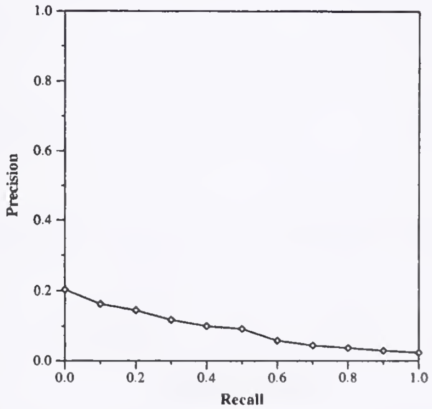
Total number of documents over all topics	
Retrieved:	48918
Relevant:	516
Rel-ret:	344

Recall Level Averages	
Recall	Precision
0.00	0.2037
0.10	0.1627
0.20	0.1449
0.30	0.1180
0.40	0.0997
0.50	0.0922
0.60	0.0584
0.70	0.0445
0.80	0.0373
0.90	0.0296
1.00	0.0236

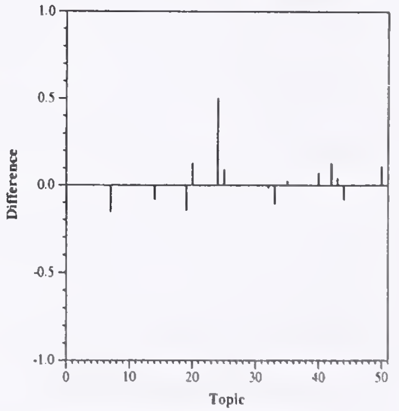
Mean average precision	
non-interpolated	0.0845

Document Level Averages	
	Precision
At 5 docs	0.0800
At 10 docs	0.0680
At 15 docs	0.0573
At 20 docs	0.0520
At 30 docs	0.0487
At 100 docs	0.0290
At 200 docs	0.0205
At 500 docs	0.0108
At 1000 docs	0.0069

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0669



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	KUCONTENT
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

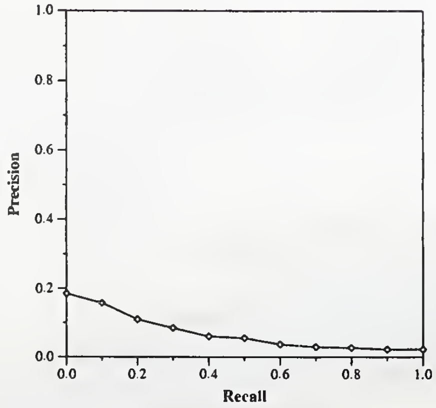
Total number of documents over all topics	
Retrieved:	49994
Relevant:	516
Rel-ret:	244

Recall Level Averages	
Recall	Precision
0.00	0.1853
0.10	0.1580
0.20	0.1098
0.30	0.0846
0.40	0.0605
0.50	0.0556
0.60	0.0371
0.70	0.0298
0.80	0.0269
0.90	0.0225
1.00	0.0225

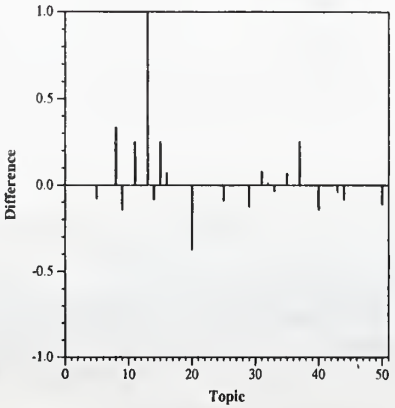
Mean average precision	
non-interpolated	0.0660

Document Level Averages	
	Precision
At 5 docs	0.0640
At 10 docs	0.0440
At 15 docs	0.0400
At 20 docs	0.0390
At 30 docs	0.0360
At 100 docs	0.0238
At 200 docs	0.0162
At 500 docs	0.0083
At 1000 docs	0.0049

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0769



Recall-Precision Curve



Difference from Median in R-Precision per Topic

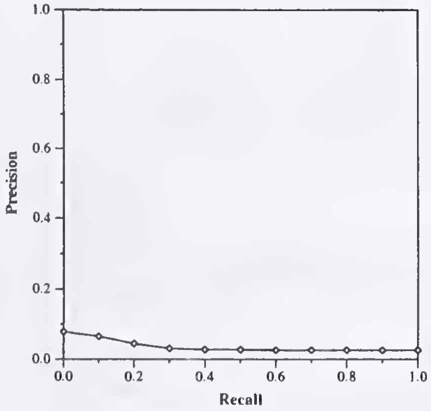
Web track, topic distillation task results — Lehigh University

Summary Statistics	
Run ID:	03wume206
Run Description	NODOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

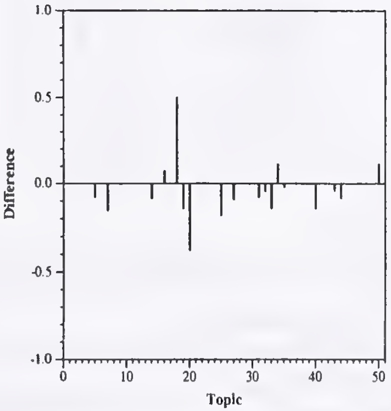
Total number of documents over all topics	
Retrieved:	19831
Relevant:	516
Rel-ret:	46

Recall Level Averages	
Recall	Precision
0.00	0.0784
0.10	0.0654
0.20	0.0447
0.30	0.0311
0.40	0.0276
0.50	0.0276
0.60	0.0260
0.70	0.0260
0.80	0.0260
0.90	0.0259
1.00	0.0257
Mean average precision	
non-interpolated	0.0343

Document Level Averages	
	Precision
At 5 docs	0.0240
At 10 docs	0.0280
At 15 docs	0.0253
At 20 docs	0.0220
At 30 docs	0.0173
At 100 docs	0.0076
At 200 docs	0.0044
At 500 docs	0.0018
At 1000 docs	0.0009
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0395



Recall-Precision Curve



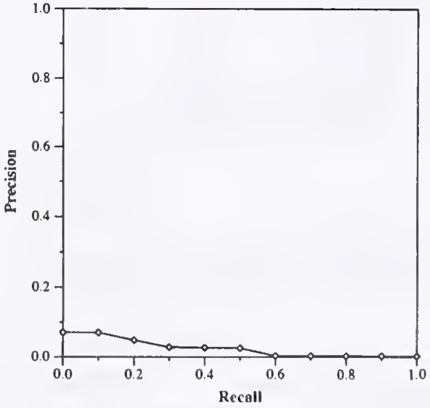
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	03wume359
Run Description	NODOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

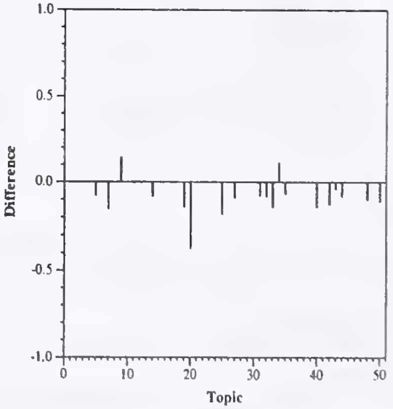
Total number of documents over all topics	
Retrieved:	45118
Relevant:	516
Rel-ret:	75

Recall Level Averages	
Recall	Precision
0.00	0.0708
0.10	0.0699
0.20	0.0481
0.30	0.0284
0.40	0.0268
0.50	0.0257
0.60	0.0039
0.70	0.0035
0.80	0.0031
0.90	0.0029
1.00	0.0028
Mean average precision	
non-interpolated	0.0225

Document Level Averages	
	Precision
At 5 docs	0.0280
At 10 docs	0.0180
At 15 docs	0.0133
At 20 docs	0.0110
At 30 docs	0.0093
At 100 docs	0.0066
At 200 docs	0.0048
At 500 docs	0.0028
At 1000 docs	0.0015
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0204



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	meijihilw1
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

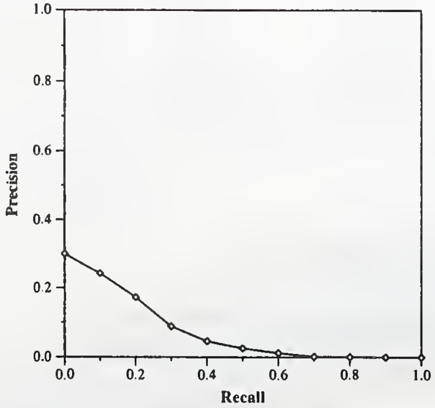
Total number of documents over all topics	
Retrieved:	48559
Relevant:	516
Rel-ret:	217

Recall Level Averages	
Recall	Precision
0.00	0.2996
0.10	0.2431
0.20	0.1726
0.30	0.0882
0.40	0.0457
0.50	0.0252
0.60	0.0117
0.70	0.0017
0.80	0.0006
0.90	0.0001
1.00	0.0001

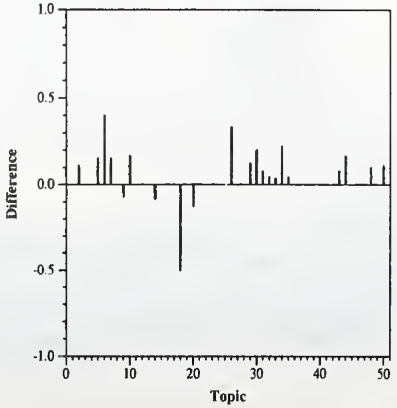
Mean average precision	
non-interpolated	0.0698

Document Level Averages	
	Precision
At 5 docs	0.1080
At 10 docs	0.0920
At 15 docs	0.0773
At 20 docs	0.0700
At 30 docs	0.0553
At 100 docs	0.0230
At 200 docs	0.0131
At 500 docs	0.0070
At 1000 docs	0.0043

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0918



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	meijihilw2
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

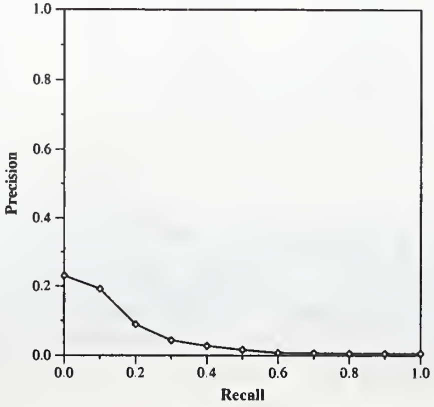
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	161

Recall Level Averages	
Recall	Precision
0.00	0.2309
0.10	0.1916
0.20	0.0890
0.30	0.0430
0.40	0.0269
0.50	0.0162
0.60	0.0075
0.70	0.0066
0.80	0.0052
0.90	0.0052
1.00	0.0052

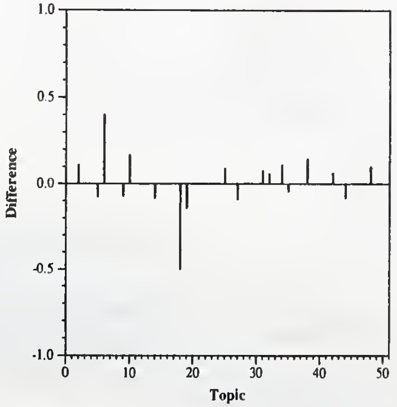
Mean average precision	
non-interpolated	0.0486

Document Level Averages	
	Precision
At 5 docs	0.0960
At 10 docs	0.0700
At 15 docs	0.0533
At 20 docs	0.0460
At 30 docs	0.0367
At 100 docs	0.0166
At 200 docs	0.0095
At 500 docs	0.0052
At 1000 docs	0.0032

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0614



Recall-Precision Curve



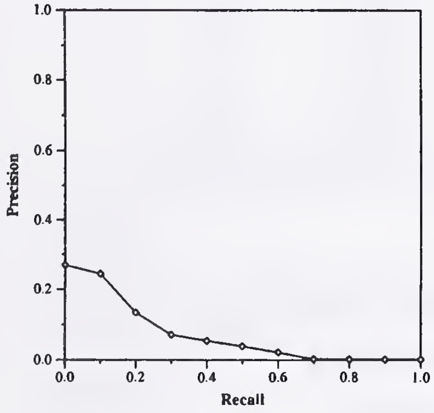
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	meijihilw3
Run Description	DOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

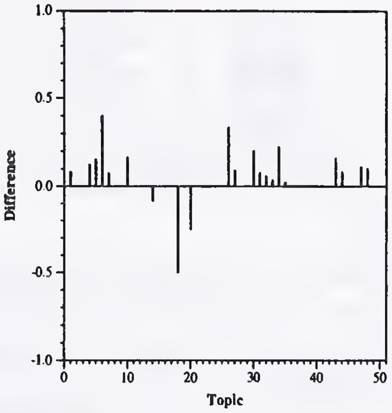
Total number of documents over all topics	
Retrieved:	47666
Relevant:	516
Rel-ret:	192

Recall Level Averages	
Recall	Precision
0.00	0.2695
0.10	0.2442
0.20	0.1343
0.30	0.0717
0.40	0.0547
0.50	0.0394
0.60	0.0218
0.70	0.0024
0.80	0.0016
0.90	0.0013
1.00	0.0013
Mean average precision	
non-interpolated	0.0652

Document Level Averages	
	Precision
At 5 docs	0.1160
At 10 docs	0.1060
At 15 docs	0.0827
At 20 docs	0.0680
At 30 docs	0.0513
At 100 docs	0.0236
At 200 docs	0.0138
At 500 docs	0.0064
At 1000 docs	0.0038
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0902



Recall-Precision Curve



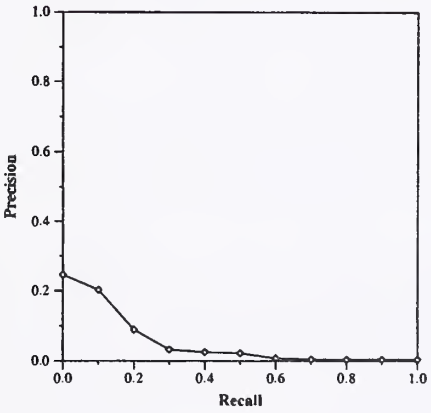
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	meijihilw4
Run Description	DOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

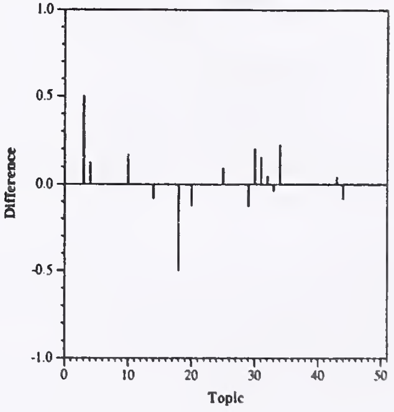
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	148

Recall Level Averages	
Recall	Precision
0.00	0.2458
0.10	0.2027
0.20	0.0895
0.30	0.0333
0.40	0.0257
0.50	0.0226
0.60	0.0080
0.70	0.0044
0.80	0.0040
0.90	0.0040
1.00	0.0040
Mean average precision	
non-interpolated	0.0486

Document Level Averages	
	Precision
At 5 docs	0.0920
At 10 docs	0.0700
At 15 docs	0.0587
At 20 docs	0.0530
At 30 docs	0.0407
At 100 docs	0.0168
At 200 docs	0.0095
At 500 docs	0.0048
At 1000 docs	0.0030
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0687



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	meijihilw5
Run Description	DOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

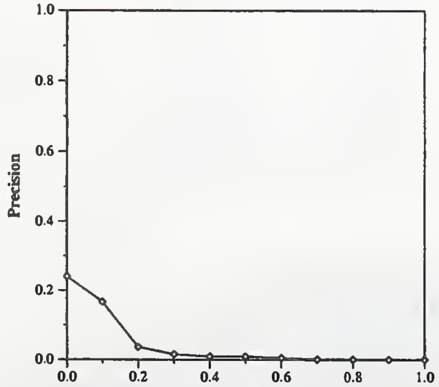
Total number of documents over all topics	
Retrieved:	18839
Relevant:	516
Rel-ret:	106

Recall Level Averages	
Recall	Precision
0.00	0.2410
0.10	0.1670
0.20	0.0364
0.30	0.0156
0.40	0.0097
0.50	0.0094
0.60	0.0053
0.70	0.0012
0.80	0.0007
0.90	0.0007
1.00	0.0007

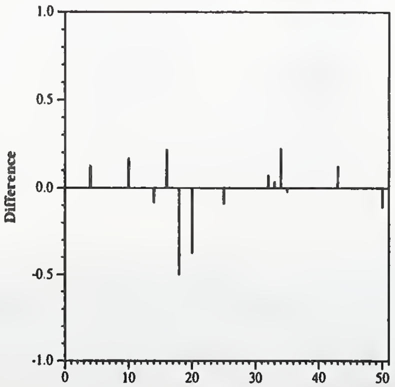
Mean average precision	
non-interpolated	0.0352

Document Level Averages	
	Precision
At 5 docs	0.0760
At 10 docs	0.0620
At 15 docs	0.0547
At 20 docs	0.0520
At 30 docs	0.0413
At 100 docs	0.0178
At 200 docs	0.0100
At 500 docs	0.0042
At 1000 docs	0.0021

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0523



Recall-Precision Curve



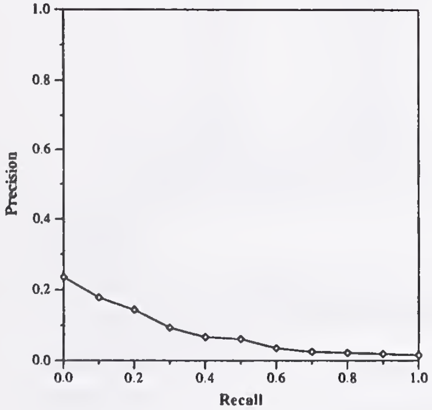
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	MSRA1001
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

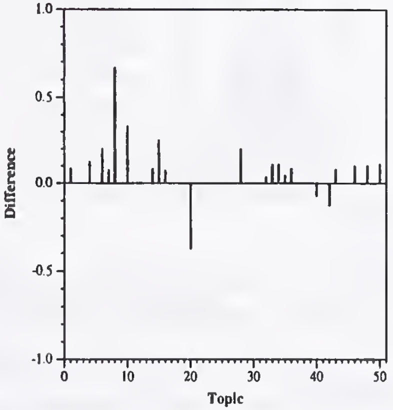
Total number of documents over all topics	
Retrieved:	49012
Relevant:	516
Rel-ret:	373

Recall Level Averages	
Recall	Precision
0.00	0.2361
0.10	0.1791
0.20	0.1450
0.30	0.0938
0.40	0.0672
0.50	0.0616
0.60	0.0351
0.70	0.0250
0.80	0.0219
0.90	0.0190
1.00	0.0154
Mean average precision	
non-interpolated	0.0699

Document Level Averages	
	Precision
At 5 docs	0.1280
At 10 docs	0.0960
At 15 docs	0.0827
At 20 docs	0.0730
At 30 docs	0.0560
At 100 docs	0.0256
At 200 docs	0.0166
At 500 docs	0.0127
At 1000 docs	0.0075
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1027



Recall-Precision Curve



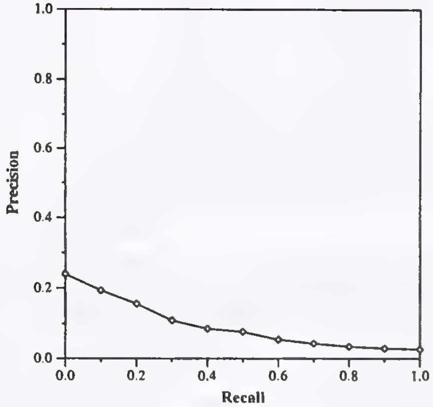
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	MSRA1002
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

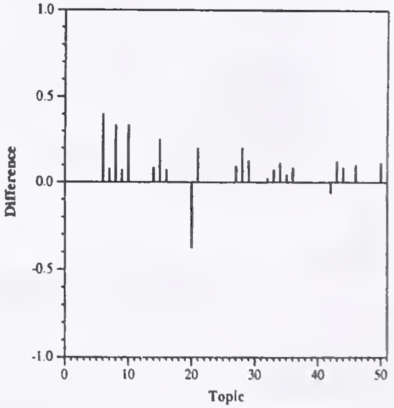
Total number of documents over all topics	
Retrieved:	49012
Relevant:	516
Rel-ret:	373

Recall Level Averages	
Recall	Precision
0.00	0.2394
0.10	0.1934
0.20	0.1561
0.30	0.1093
0.40	0.0852
0.50	0.0764
0.60	0.0541
0.70	0.0428
0.80	0.0341
0.90	0.0288
1.00	0.0260
Mean average precision	
non-interpolated	0.0824

Document Level Averages	
	Precision
At 5 docs	0.1160
At 10 docs	0.1100
At 15 docs	0.0880
At 20 docs	0.0780
At 30 docs	0.0653
At 100 docs	0.0316
At 200 docs	0.0192
At 500 docs	0.0124
At 1000 docs	0.0075
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1078



Recall-Precision Curve



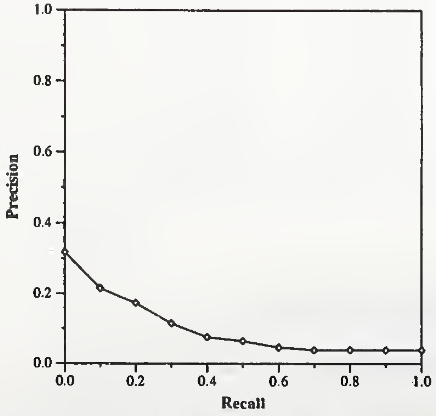
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	MSRA3
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

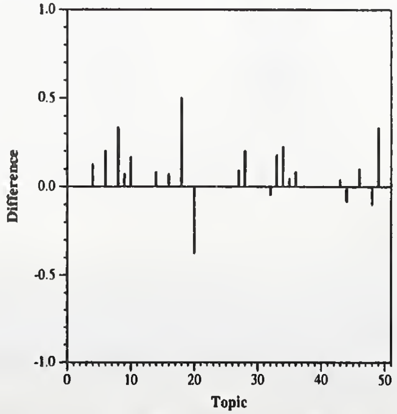
Total number of documents over all topics	
Retrieved:	2310
Relevant:	516
Rel-ret:	109

Recall Level Averages	
Recall	Precision
0.00	0.3175
0.10	0.2149
0.20	0.1731
0.30	0.1142
0.40	0.0750
0.50	0.0642
0.60	0.0458
0.70	0.0385
0.80	0.0385
0.90	0.0385
1.00	0.0385
Mean average precision	
non-interpolated	0.0933

Document Level Averages	
	Precision
At 5 docs	0.1320
At 10 docs	0.1040
At 15 docs	0.0813
At 20 docs	0.0760
At 30 docs	0.0627
At 100 docs	0.0218
At 200 docs	0.0109
At 500 docs	0.0044
At 1000 docs	0.0022
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1016



Recall-Precision Curve



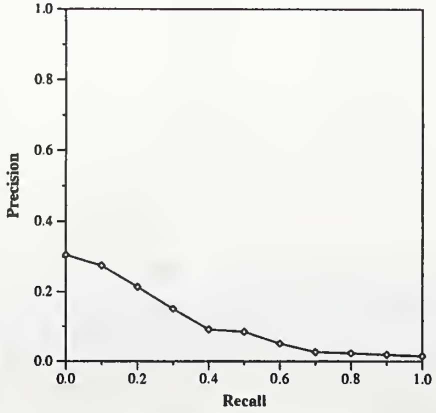
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	MSRA4002
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

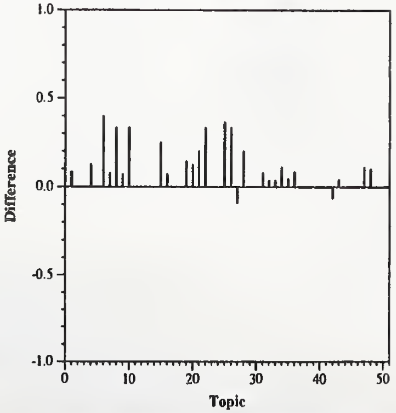
Total number of documents over all topics	
Retrieved:	49053
Relevant:	516
Rel-ret:	368

Recall Level Averages	
Recall	Precision
0.00	0.3056
0.10	0.2752
0.20	0.2143
0.30	0.1511
0.40	0.0912
0.50	0.0847
0.60	0.0507
0.70	0.0265
0.80	0.0230
0.90	0.0189
1.00	0.0148
Mean average precision	
non-interpolated	0.1027

Document Level Averages	
	Precision
At 5 docs	0.1360
At 10 docs	0.1160
At 15 docs	0.0947
At 20 docs	0.0840
At 30 docs	0.0647
At 100 docs	0.0318
At 200 docs	0.0193
At 500 docs	0.0125
At 1000 docs	0.0074
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1354



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	MSRA4003
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

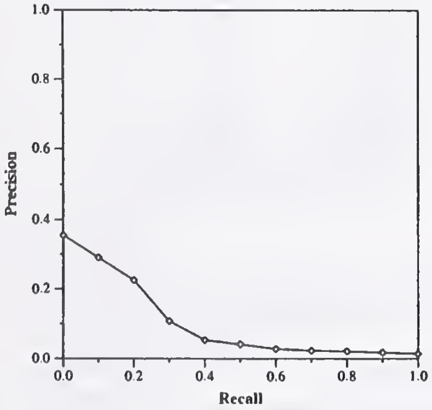
Total number of documents over all topics	
Retrieved:	49067
Relevant:	516
Rel-ret:	370

Recall Level Averages	
Recall	Precision
0.00	0.3539
0.10	0.2905
0.20	0.2253
0.30	0.1077
0.40	0.0526
0.50	0.0409
0.60	0.0280
0.70	0.0230
0.80	0.0210
0.90	0.0178
1.00	0.0140

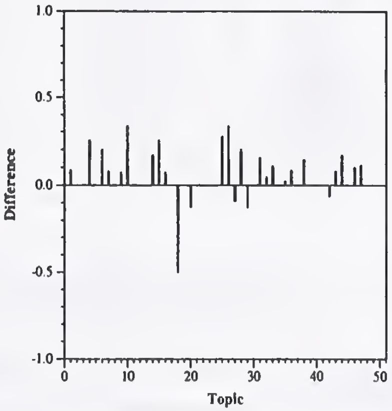
Mean average precision	
non-interpolated	0.0946

Document Level Averages	
	Precision
At 5 docs	0.1480
At 10 docs	0.1140
At 15 docs	0.0880
At 20 docs	0.0760
At 30 docs	0.0573
At 100 docs	0.0260
At 200 docs	0.0176
At 500 docs	0.0130
At 1000 docs	0.0074

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1052



Recall-Precision Curve

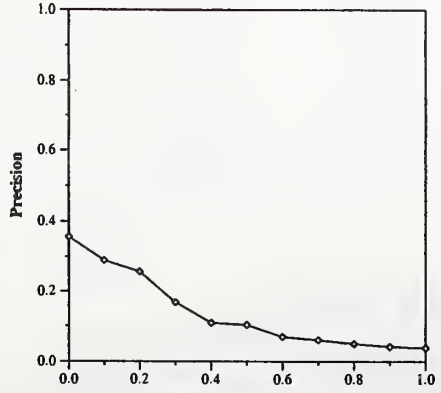


Difference from Median in R-Precision per Topic

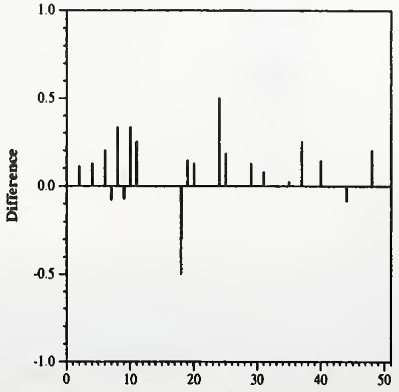
Summary Statistics	
Run ID:	UniNEtd1
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50
Total number of documents over all topics	
Retrieved:	48507
Relevant:	516
Rel-ret:	366

Recall Level Averages	
Recall	Precision
0.00	0.3559
0.10	0.2894
0.20	0.2562
0.30	0.1675
0.40	0.1088
0.50	0.1029
0.60	0.0696
0.70	0.0602
0.80	0.0495
0.90	0.0415
1.00	0.0377
Mean average precision	
non-interpolated	0.1285

Document Level Averages	
	Precision
At 5 docs	0.1440
At 10 docs	0.0980
At 15 docs	0.0840
At 20 docs	0.0760
At 30 docs	0.0627
At 100 docs	0.0338
At 200 docs	0.0222
At 500 docs	0.0124
At 1000 docs	0.0073
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1046



Recall-Precision Curve

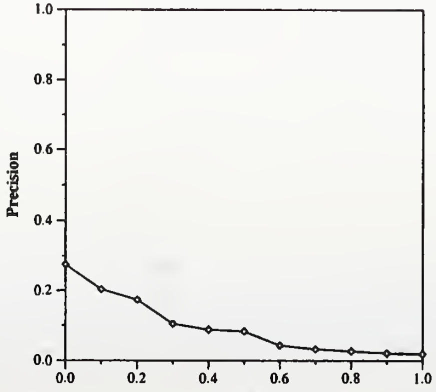


Difference from Median in R-Precision per Topic

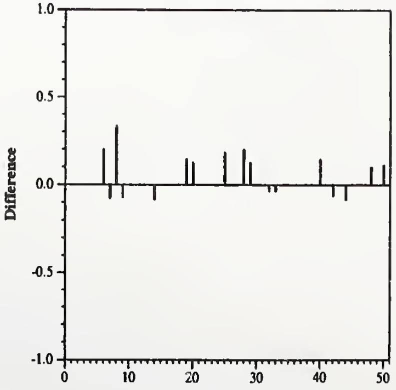
Summary Statistics	
Run ID:	UniNEtd2
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50
Total number of documents over all topics	
Retrieved:	48505
Relevant:	516
Rel-ret:	325

Recall Level Averages	
Recall	Precision
0.00	0.2748
0.10	0.2033
0.20	0.1725
0.30	0.1039
0.40	0.0875
0.50	0.0828
0.60	0.0427
0.70	0.0318
0.80	0.0264
0.90	0.0207
1.00	0.0186
Mean average precision	
non-interpolated	0.0864

Document Level Averages	
	Precision
At 5 docs	0.1160
At 10 docs	0.0760
At 15 docs	0.0640
At 20 docs	0.0540
At 30 docs	0.0473
At 100 docs	0.0270
At 200 docs	0.0182
At 500 docs	0.0105
At 1000 docs	0.0065
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0811



Recall-Precision Curve



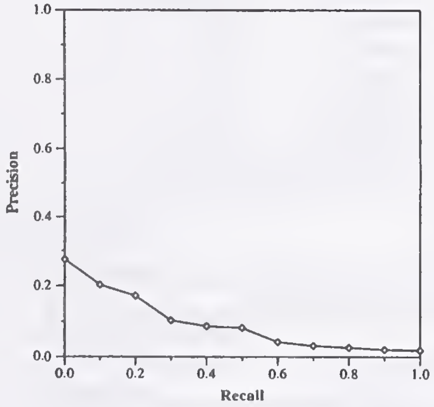
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	UniNEtd3
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

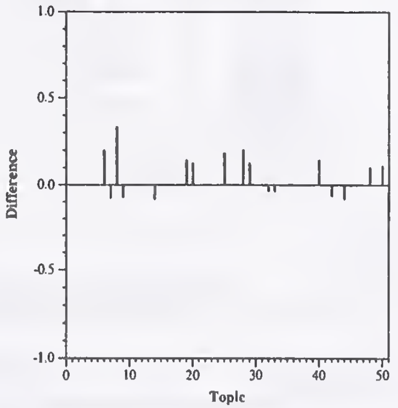
Total number of documents over all topics	
Retrieved:	48505
Relevant:	516
Rel-ret:	325

Recall Level Averages	
Recall	Precision
0.00	0.2748
0.10	0.2033
0.20	0.1725
0.30	0.1039
0.40	0.0875
0.50	0.0828
0.60	0.0427
0.70	0.0318
0.80	0.0264
0.90	0.0207
1.00	0.0186
Mean average precision	
non-interpolated	0.0864

Document Level Averages	
	Precision
At 5 docs	0.1160
At 10 docs	0.0760
At 15 docs	0.0640
At 20 docs	0.0540
At 30 docs	0.0473
At 100 docs	0.0270
At 200 docs	0.0182
At 500 docs	0.0105
At 1000 docs	0.0065
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0811



Recall-Precision Curve



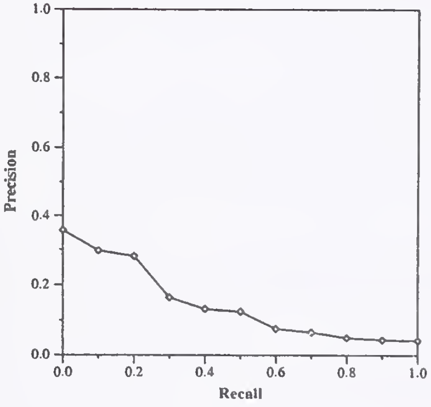
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	UniNEtd4
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

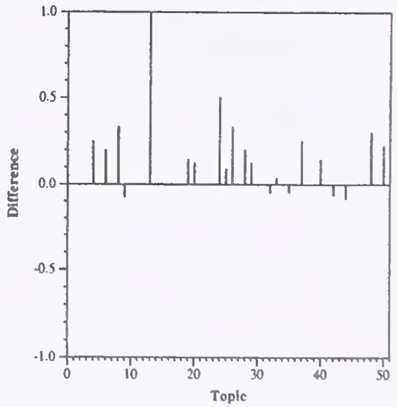
Total number of documents over all topics	
Retrieved:	48783
Relevant:	516
Rel-ret:	315

Recall Level Averages	
Recall	Precision
0.00	0.3568
0.10	0.2971
0.20	0.2811
0.30	0.1642
0.40	0.1314
0.50	0.1242
0.60	0.0761
0.70	0.0659
0.80	0.0501
0.90	0.0442
1.00	0.0422
Mean average precision	
non-interpolated	0.1371

Document Level Averages	
	Precision
At 5 docs	0.1560
At 10 docs	0.0880
At 15 docs	0.0760
At 20 docs	0.0680
At 30 docs	0.0527
At 100 docs	0.0296
At 200 docs	0.0183
At 500 docs	0.0104
At 1000 docs	0.0063
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1357



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Web track, topic distillation task results — Universite de Neuchatel

Summary Statistics	
Run ID:	UniNEtd5
Run Description	DOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

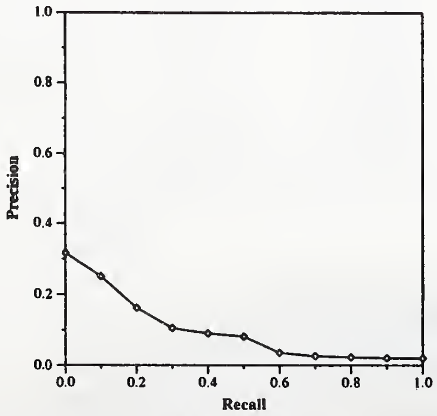
Total number of documents over all topics	
Retrieved:	47672
Relevant:	516
Rel-ret:	314

Recall Level Averages	
Recall	Precision
0.00	0.3168
0.10	0.2504
0.20	0.1626
0.30	0.1059
0.40	0.0911
0.50	0.0822
0.60	0.0360
0.70	0.0270
0.80	0.0233
0.90	0.0211
1.00	0.0205

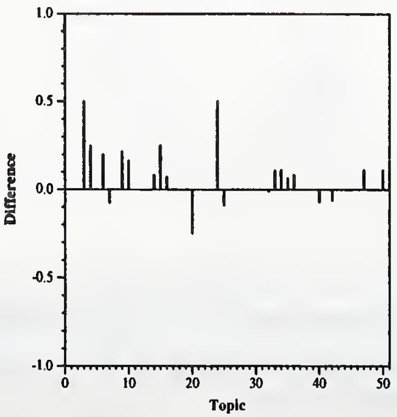
Mean average precision	
non-interpolated	0.0902

Document Level Averages	
	Precision
At 5 docs	0.0960
At 10 docs	0.0840
At 15 docs	0.0760
At 20 docs	0.0650
At 30 docs	0.0540
At 100 docs	0.0250
At 200 docs	0.0174
At 500 docs	0.0103
At 1000 docs	0.0063

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1021



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	topics0
Run Description	NODOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

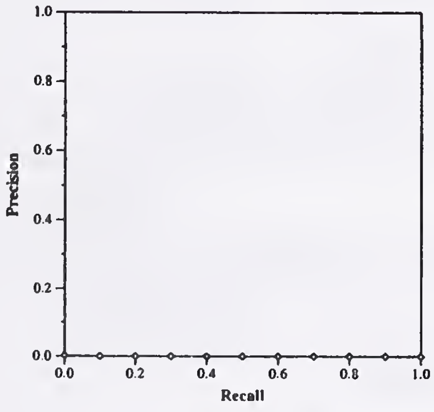
Total number of documents over all topics	
Retrieved:	6251
Relevant:	516
Rel-ret:	4

Recall Level Averages	
Recall	Precision
0.00	0.0021
0.10	0.0000
0.20	0.0000
0.30	0.0000
0.40	0.0000
0.50	0.0000
0.60	0.0000
0.70	0.0000
0.80	0.0000
0.90	0.0000
1.00	0.0000

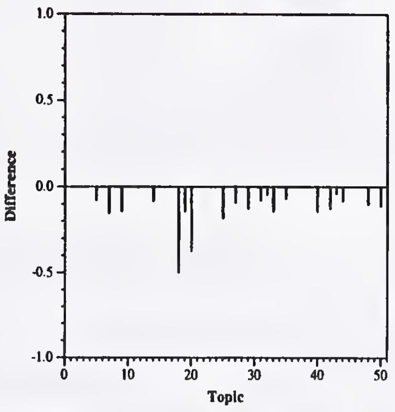
Mean average precision	
non-interpolated	0.0001

Document Level Averages	
	Precision
At 5 docs	0.0000
At 10 docs	0.0000
At 15 docs	0.0000
At 20 docs	0.0010
At 30 docs	0.0013
At 100 docs	0.0008
At 200 docs	0.0004
At 500 docs	0.0002
At 1000 docs	0.0001

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0007



Recall-Precision Curve



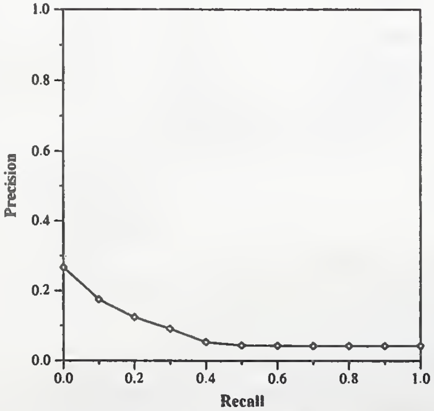
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	THUIRtd0301
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

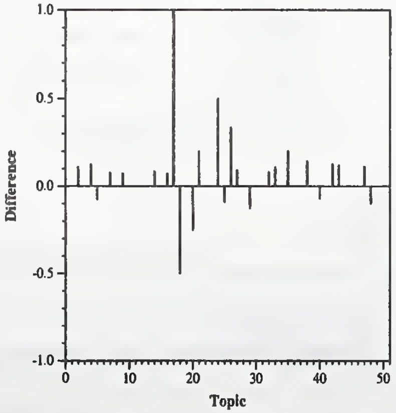
Total number of documents over all topics	
Retrieved:	41532
Relevant:	516
Rel-ret:	178

Recall Level Averages	
Recall	Precision
0.00	0.2669
0.10	0.1756
0.20	0.1244
0.30	0.0912
0.40	0.0533
0.50	0.0440
0.60	0.0431
0.70	0.0424
0.80	0.0422
0.90	0.0422
1.00	0.0422
Mean average precision	
non-interpolated	0.0764

Document Level Averages	
	Precision
At 5 docs	0.0960
At 10 docs	0.0800
At 15 docs	0.0653
At 20 docs	0.0590
At 30 docs	0.0493
At 100 docs	0.0250
At 200 docs	0.0141
At 500 docs	0.0067
At 1000 docs	0.0036
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1036



Recall-Precision Curve



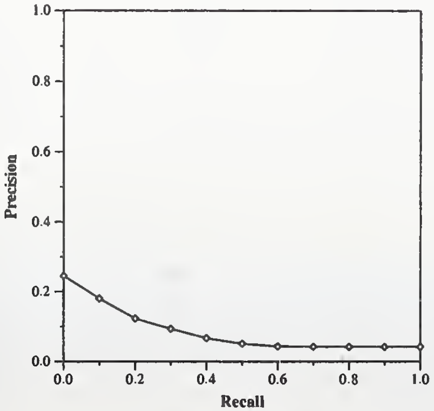
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	THUIRtd0302
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

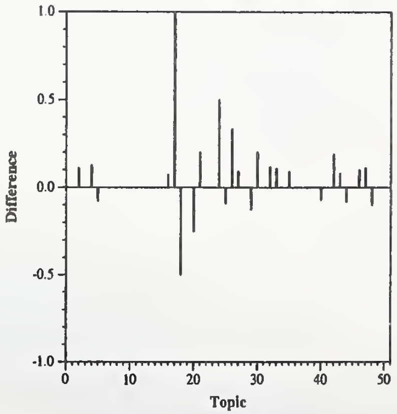
Total number of documents over all topics	
Retrieved:	42782
Relevant:	516
Rel-ret:	194

Recall Level Averages	
Recall	Precision
0.00	0.2454
0.10	0.1806
0.20	0.1235
0.30	0.0940
0.40	0.0671
0.50	0.0509
0.60	0.0431
0.70	0.0422
0.80	0.0420
0.90	0.0420
1.00	0.0420
Mean average precision	
non-interpolated	0.0763

Document Level Averages	
	Precision
At 5 docs	0.1040
At 10 docs	0.0840
At 15 docs	0.0653
At 20 docs	0.0520
At 30 docs	0.0460
At 100 docs	0.0258
At 200 docs	0.0151
At 500 docs	0.0071
At 1000 docs	0.0039
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0994



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Web track, topic distillation task results — Tsinghua University (Ma)

Summary Statistics	
Run ID:	THUIRtd0303
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

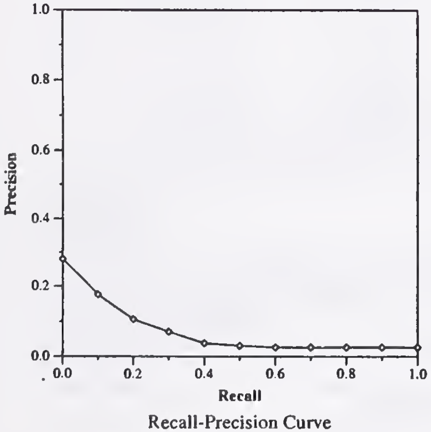
Total number of documents over all topics	
Retrieved:	42889
Relevant:	516
Rel-ret:	151

Recall Level Averages	
Recall	Precision
0.00	0.2801
0.10	0.1767
0.20	0.1062
0.30	0.0711
0.40	0.0383
0.50	0.0311
0.60	0.0263
0.70	0.0262
0.80	0.0262
0.90	0.0262
1.00	0.0262

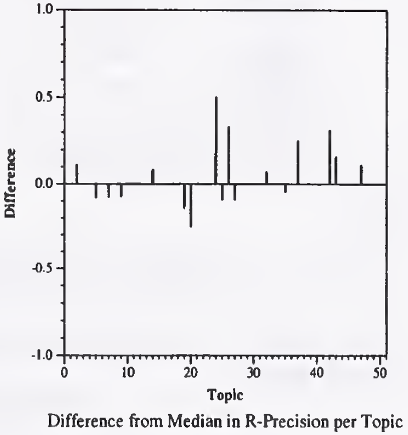
Mean average precision	
non-interpolated	0.0646

Document Level Averages	
	Precision
At 5 docs	0.0880
At 10 docs	0.0620
At 15 docs	0.0507
At 20 docs	0.0390
At 30 docs	0.0333
At 100 docs	0.0186
At 200 docs	0.0111
At 500 docs	0.0052
At 1000 docs	0.0030

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0786



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	THUIRtd0304
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

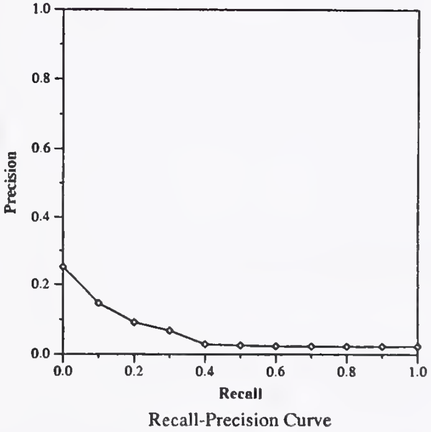
Total number of documents over all topics	
Retrieved:	38670
Relevant:	516
Rel-ret:	143

Recall Level Averages	
Recall	Precision
0.00	0.2514
0.10	0.1462
0.20	0.0920
0.30	0.0682
0.40	0.0286
0.50	0.0254
0.60	0.0231
0.70	0.0224
0.80	0.0224
0.90	0.0224
1.00	0.0224

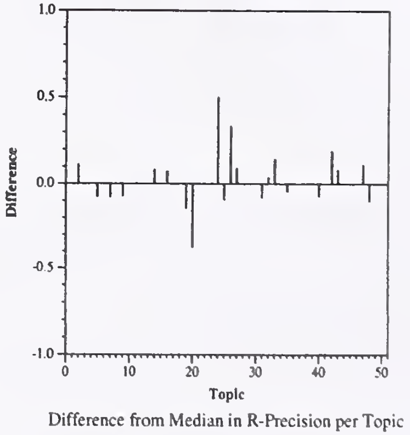
Mean average precision	
non-interpolated	0.0558

Document Level Averages	
	Precision
At 5 docs	0.0840
At 10 docs	0.0600
At 15 docs	0.0480
At 20 docs	0.0420
At 30 docs	0.0400
At 100 docs	0.0178
At 200 docs	0.0107
At 500 docs	0.0051
At 1000 docs	0.0029

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0692



Recall-Precision Curve



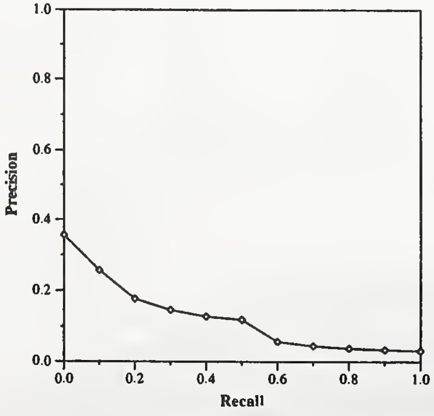
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	THUIRtd0305
Run Description	DOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

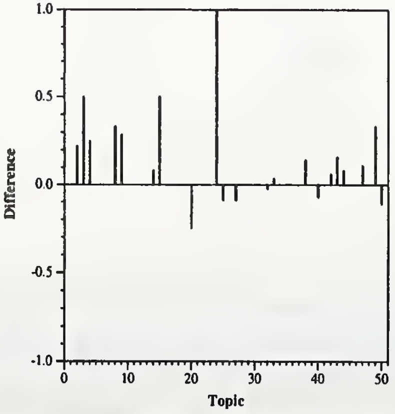
Total number of documents over all topics	
Retrieved:	49060
Relevant:	516
Rel-ret:	369

Recall Level Averages	
Recall	Precision
0.00	0.3552
0.10	0.2571
0.20	0.1778
0.30	0.1465
0.40	0.1287
0.50	0.1197
0.60	0.0573
0.70	0.0450
0.80	0.0383
0.90	0.0342
1.00	0.0309
Mean average precision	
non-interpolated	0.1131

Document Level Averages	
	Precision
At 5 docs	0.1080
At 10 docs	0.1060
At 15 docs	0.0853
At 20 docs	0.0730
At 30 docs	0.0553
At 100 docs	0.0338
At 200 docs	0.0228
At 500 docs	0.0128
At 1000 docs	0.0074
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1262



Recall-Precision Curve



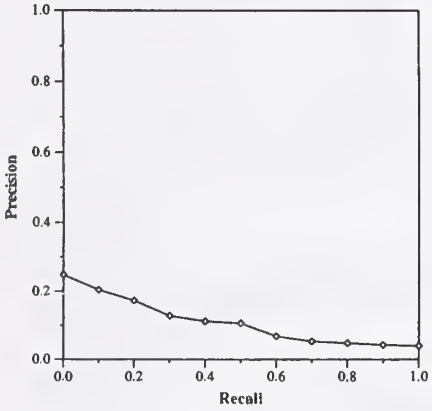
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	UAmsT03WtLM3
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

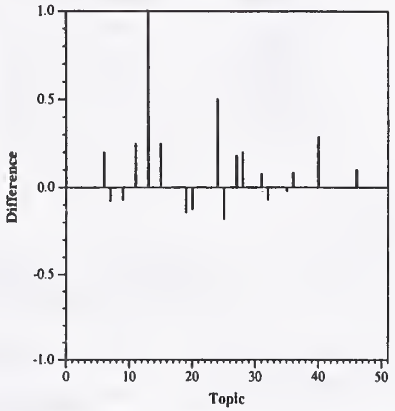
Total number of documents over all topics	
Retrieved:	49357
Relevant:	516
Rel-ret:	355

Recall Level Averages	
Recall	Precision
0.00	0.2481
0.10	0.2043
0.20	0.1725
0.30	0.1279
0.40	0.1121
0.50	0.1062
0.60	0.0683
0.70	0.0535
0.80	0.0479
0.90	0.0429
1.00	0.0401
Mean average precision	
non-interpolated	0.1019

Document Level Averages	
	Precision
At 5 docs	0.0840
At 10 docs	0.0840
At 15 docs	0.0720
At 20 docs	0.0630
At 30 docs	0.0533
At 100 docs	0.0308
At 200 docs	0.0214
At 500 docs	0.0116
At 1000 docs	0.0071
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1056



Recall-Precision Curve



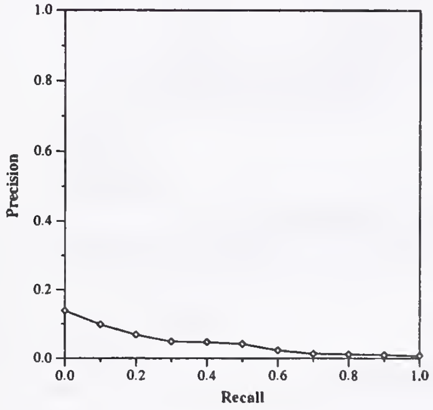
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	UAmsT03WtLMI
Run Description	NODOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

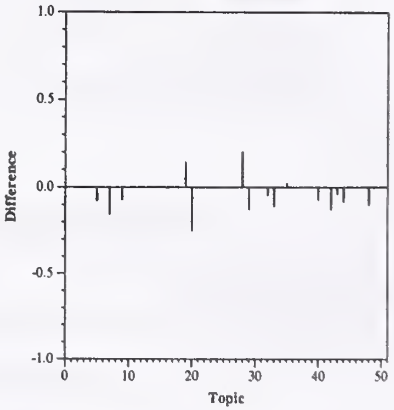
Total number of documents over all topics	
Retrieved:	49371
Relevant:	516
Rel-ret:	310

Recall Level Averages	
Recall	Precision
0.00	0.1387
0.10	0.0983
0.20	0.0683
0.30	0.0487
0.40	0.0470
0.50	0.0419
0.60	0.0235
0.70	0.0136
0.80	0.0119
0.90	0.0099
1.00	0.0076
Mean average precision	
non-interpolated	0.0412

Document Level Averages	
	Precision
At 5 docs	0.0280
At 10 docs	0.0280
At 15 docs	0.0293
At 20 docs	0.0260
At 30 docs	0.0267
At 100 docs	0.0216
At 200 docs	0.0152
At 500 docs	0.0097
At 1000 docs	0.0062
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0391



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	UAmsT03WtOk3
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

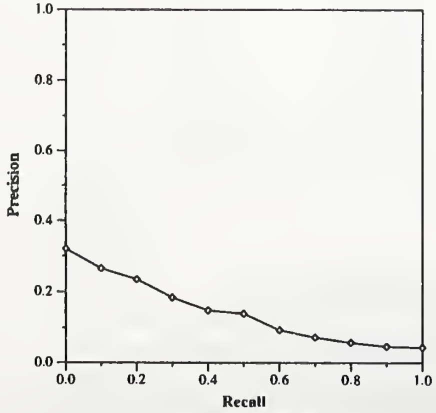
Total number of documents over all topics	
Retrieved:	49357
Relevant:	516
Rel-ret:	371

Recall Level Averages	
Recall	Precision
0.00	0.3201
0.10	0.2657
0.20	0.2354
0.30	0.1849
0.40	0.1494
0.50	0.1399
0.60	0.0943
0.70	0.0737
0.80	0.0585
0.90	0.0475
1.00	0.0444

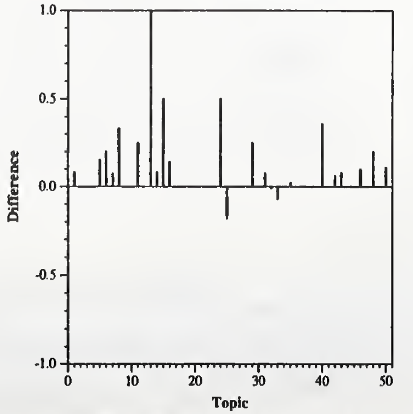
Mean average precision	
non-interpolated	0.1344

Document Level Averages	
	Precision
At 5 docs	0.1160
At 10 docs	0.0980
At 15 docs	0.0880
At 20 docs	0.0810
At 30 docs	0.0787
At 100 docs	0.0418
At 200 docs	0.0259
At 500 docs	0.0132
At 1000 docs	0.0074

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1432



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	UAmsT03WtOkC
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

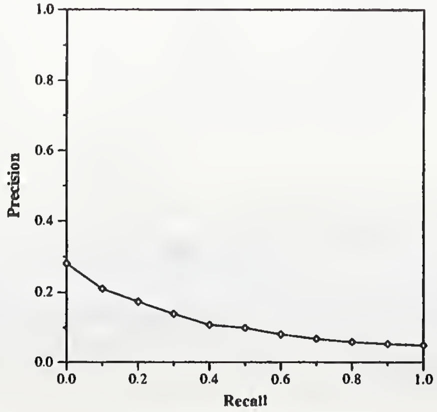
Total number of documents over all topics	
Retrieved:	49357
Relevant:	516
Rel-ret:	371

Recall Level Averages	
Recall	Precision
0.00	0.2811
0.10	0.2104
0.20	0.1744
0.30	0.1405
0.40	0.1087
0.50	0.0997
0.60	0.0811
0.70	0.0680
0.80	0.0591
0.90	0.0534
1.00	0.0496

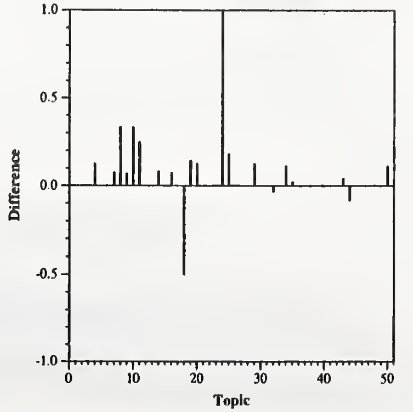
Mean average precision	
non-interpolated	0.1127

Document Level Averages	
	Precision
At 5 docs	0.1280
At 10 docs	0.0860
At 15 docs	0.0773
At 20 docs	0.0650
At 30 docs	0.0540
At 100 docs	0.0310
At 200 docs	0.0217
At 500 docs	0.0123
At 1000 docs	0.0074

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1086



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Web track, topic distillation task results — University of Amsterdam

Summary Statistics	
Run ID:	UAmsT03WtOkI
Run Description	NODOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

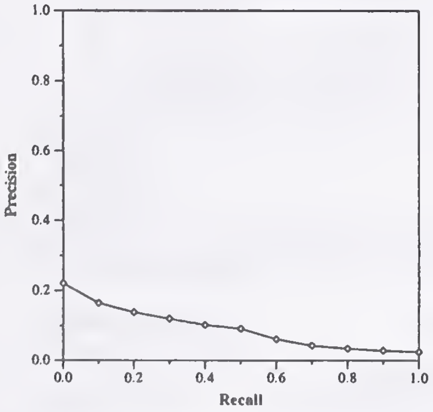
Total number of documents over all topics	
Retrieved:	49374
Relevant:	516
Rel-ret:	371

Recall Level Averages	
Recall	Precision
0.00	0.2208
0.10	0.1648
0.20	0.1388
0.30	0.1206
0.40	0.1028
0.50	0.0917
0.60	0.0612
0.70	0.0427
0.80	0.0339
0.90	0.0281
1.00	0.0245

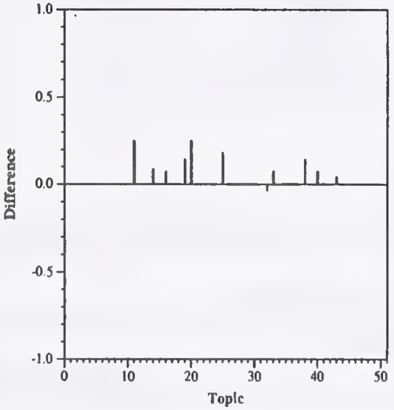
Mean average precision	
non-interpolated	0.0862

Document Level Averages	
	Precision
At 5 docs	0.1040
At 10 docs	0.0760
At 15 docs	0.0787
At 20 docs	0.0660
At 30 docs	0.0567
At 100 docs	0.0326
At 200 docs	0.0223
At 500 docs	0.0123
At 1000 docs	0.0074

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0823



Recall-Precision Curve



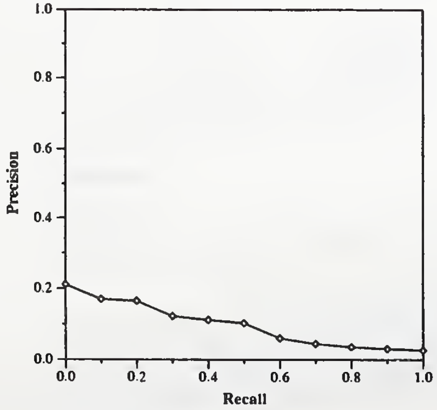
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	uogtd1c
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

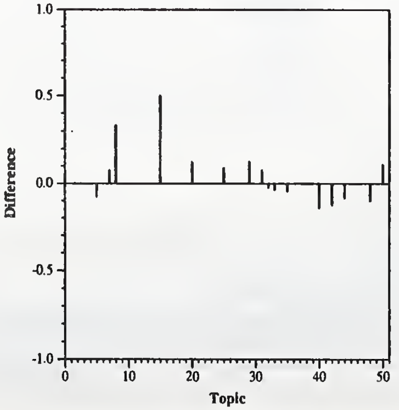
Total number of documents over all topics	
Retrieved:	49355
Relevant:	516
Rel-ret:	365

Recall Level Averages	
Recall	Precision
0.00	0.2108
0.10	0.1702
0.20	0.1654
0.30	0.1224
0.40	0.1120
0.50	0.1031
0.60	0.0606
0.70	0.0450
0.80	0.0359
0.90	0.0309
1.00	0.0269
Mean average precision	
non-interpolated	0.0886

Document Level Averages	
	Precision
At 5 docs	0.0760
At 10 docs	0.0680
At 15 docs	0.0587
At 20 docs	0.0510
At 30 docs	0.0487
At 100 docs	0.0306
At 200 docs	0.0222
At 500 docs	0.0123
At 1000 docs	0.0073
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0730



Recall-Precision Curve



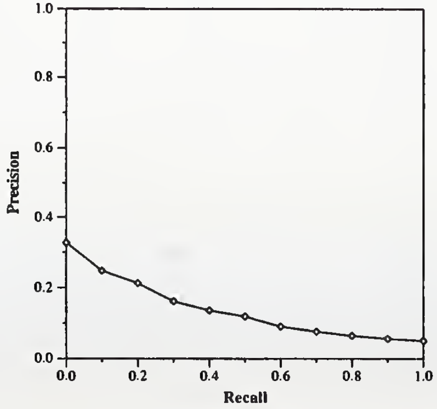
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	uogtd2ca
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

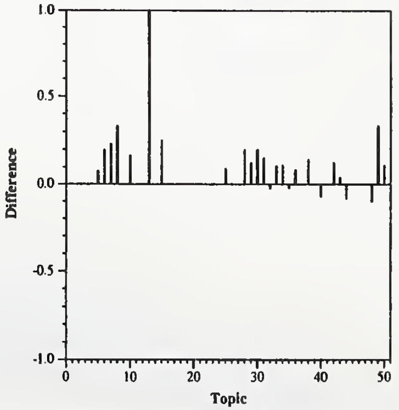
Total number of documents over all topics	
Retrieved:	49357
Relevant:	516
Rel-ret:	386

Recall Level Averages	
Recall	Precision
0.00	0.3271
0.10	0.2477
0.20	0.2128
0.30	0.1621
0.40	0.1361
0.50	0.1191
0.60	0.0908
0.70	0.0766
0.80	0.0649
0.90	0.0566
1.00	0.0506
Mean average precision	
non-interpolated	0.1273

Document Level Averages	
	Precision
At 5 docs	0.1240
At 10 docs	0.1020
At 15 docs	0.0947
At 20 docs	0.0830
At 30 docs	0.0700
At 100 docs	0.0378
At 200 docs	0.0261
At 500 docs	0.0133
At 1000 docs	0.0077
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1325



Recall-Precision Curve



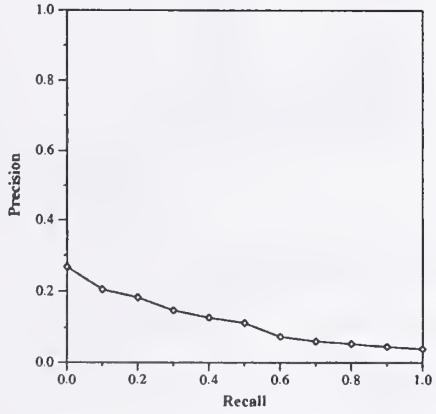
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	uogtd3cas
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

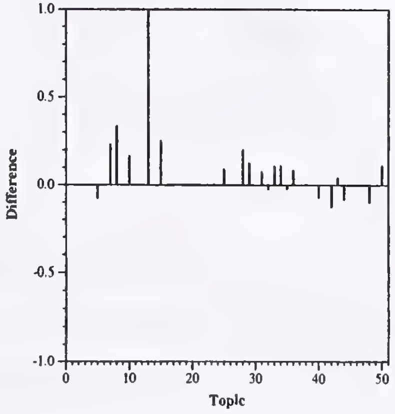
Total number of documents over all topics	
Retrieved:	49355
Relevant:	516
Rel-ret:	380

Recall Level Averages	
Recall	Precision
0.00	0.2674
0.10	0.2053
0.20	0.1834
0.30	0.1480
0.40	0.1278
0.50	0.1133
0.60	0.0747
0.70	0.0617
0.80	0.0544
0.90	0.0465
1.00	0.0398
Mean average precision	
non-interpolated	0.1099

Document Level Averages	
	Precision
At 5 docs	0.0960
At 10 docs	0.0820
At 15 docs	0.0747
At 20 docs	0.0670
At 30 docs	0.0620
At 100 docs	0.0354
At 200 docs	0.0249
At 500 docs	0.0130
At 1000 docs	0.0076
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1053



Recall-Precision Curve



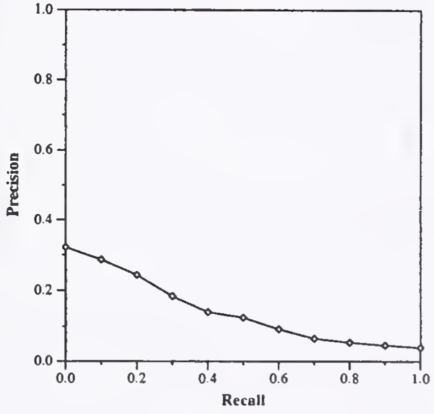
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	uogtd4cahs
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

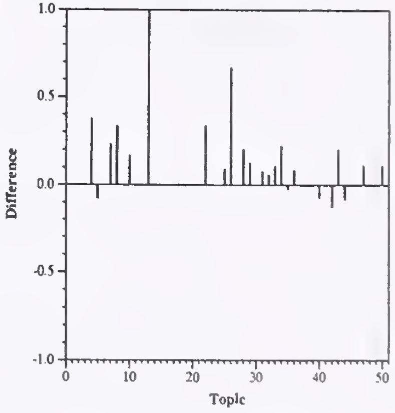
Total number of documents over all topics	
Retrieved:	49355
Relevant:	516
Rel-ret:	380

Recall Level Averages	
Recall	Precision
0.00	0.3213
0.10	0.2863
0.20	0.2433
0.30	0.1843
0.40	0.1401
0.50	0.1247
0.60	0.0922
0.70	0.0657
0.80	0.0548
0.90	0.0466
1.00	0.0401
Mean average precision	
non-interpolated	0.1336

Document Level Averages	
	Precision
At 5 docs	0.1360
At 10 docs	0.1140
At 15 docs	0.0947
At 20 docs	0.0790
At 30 docs	0.0713
At 100 docs	0.0380
At 200 docs	0.0268
At 500 docs	0.0137
At 1000 docs	0.0076
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1391



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	uogtd5cass
Run Description	NODOCSTRUCT, ANCHORTEXT, LINKSTRUCT
Number of Topics:	50

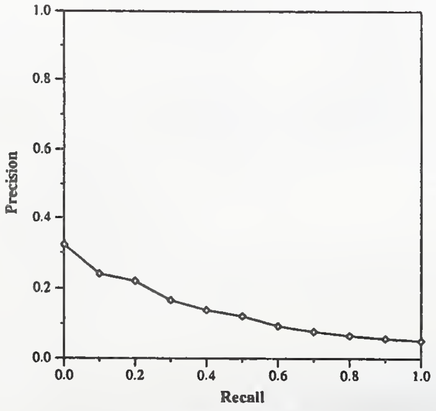
Total number of documents over all topics	
Retrieved:	49357
Relevant:	516
Rel-ret:	388

Recall Level Averages	
Recall	Precision
0.00	0.3218
0.10	0.2394
0.20	0.2194
0.30	0.1671
0.40	0.1390
0.50	0.1220
0.60	0.0934
0.70	0.0781
0.80	0.0660
0.90	0.0578
1.00	0.0511

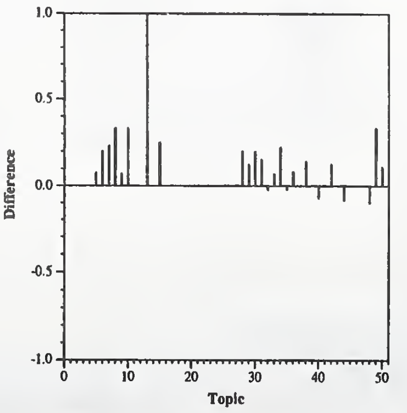
Mean average precision	
non-interpolated	0.1284

Document Level Averages	
	Precision
At 5 docs	0.1280
At 10 docs	0.1080
At 15 docs	0.0960
At 20 docs	0.0800
At 30 docs	0.0700
At 100 docs	0.0388
At 200 docs	0.0267
At 500 docs	0.0136
At 1000 docs	0.0078

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1361



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	UIUC03W2s
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

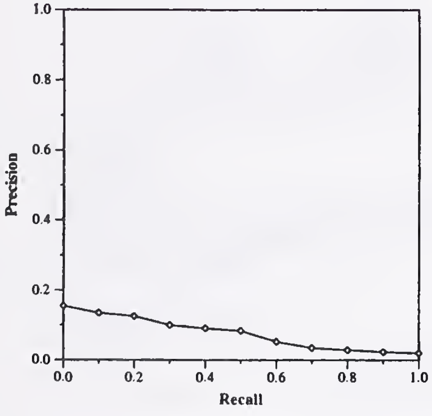
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	371

Recall Level Averages	
Recall	Precision
0.00	0.1546
0.10	0.1343
0.20	0.1251
0.30	0.1000
0.40	0.0905
0.50	0.0838
0.60	0.0525
0.70	0.0346
0.80	0.0284
0.90	0.0232
1.00	0.0199

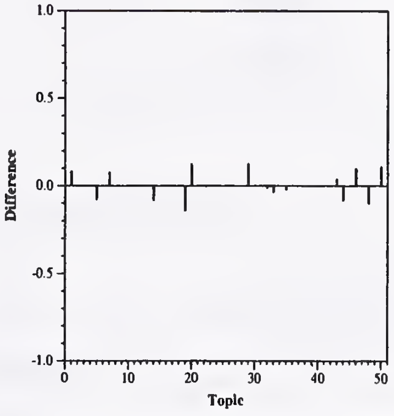
Mean average precision	
non-interpolated	0.0691

Document Level Averages	
	Precision
At 5 docs	0.0640
At 10 docs	0.0640
At 15 docs	0.0573
At 20 docs	0.0570
At 30 docs	0.0480
At 100 docs	0.0298
At 200 docs	0.0211
At 500 docs	0.0124
At 1000 docs	0.0074

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0590



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	UIUC03Wb
Run Description	NODOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

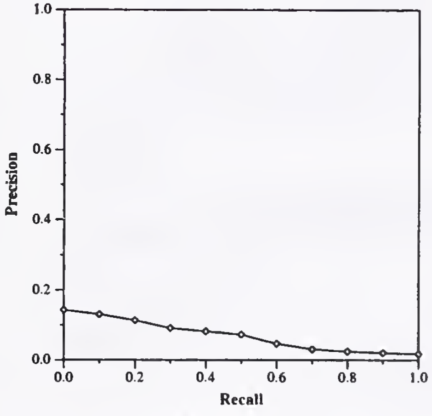
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	361

Recall Level Averages	
Recall	Precision
0.00	0.1425
0.10	0.1303
0.20	0.1129
0.30	0.0910
0.40	0.0819
0.50	0.0730
0.60	0.0466
0.70	0.0306
0.80	0.0244
0.90	0.0204
1.00	0.0173

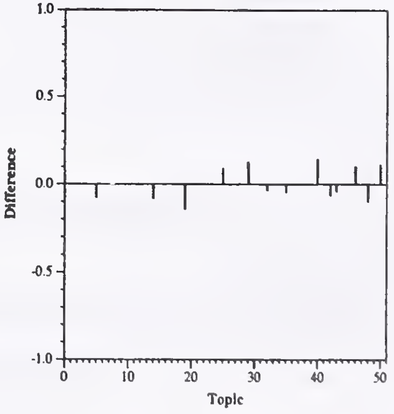
Mean average precision	
non-interpolated	0.0627

Document Level Averages	
	Precision
At 5 docs	0.0440
At 10 docs	0.0540
At 15 docs	0.0547
At 20 docs	0.0500
At 30 docs	0.0407
At 100 docs	0.0256
At 200 docs	0.0196
At 500 docs	0.0120
At 1000 docs	0.0072

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0566



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	UIUC03Wp
Run Description	NODOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

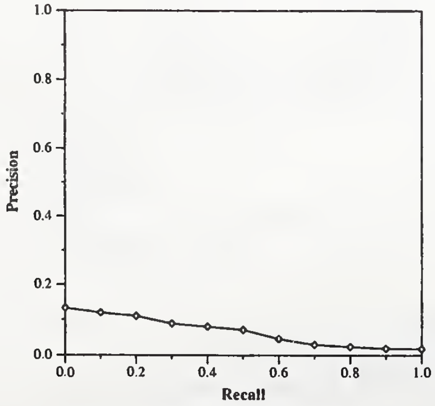
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	285

Recall Level Averages	
Recall	Precision
0.00	0.1325
0.10	0.1194
0.20	0.1101
0.30	0.0890
0.40	0.0804
0.50	0.0717
0.60	0.0465
0.70	0.0305
0.80	0.0244
0.90	0.0199
1.00	0.0182

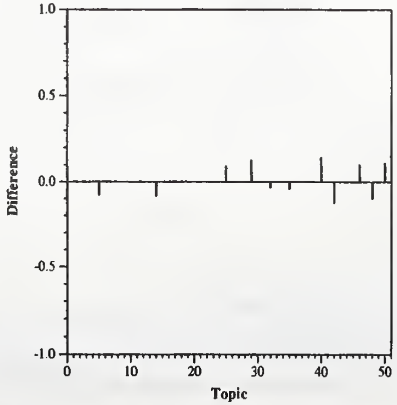
Mean average precision	
non-interpolated	0.0611

Document Level Averages	
	Precision
At 5 docs	0.0400
At 10 docs	0.0580
At 15 docs	0.0560
At 20 docs	0.0510
At 30 docs	0.0413
At 100 docs	0.0250
At 200 docs	0.0195
At 500 docs	0.0105
At 1000 docs	0.0057

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0590



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	UIUC03Wu1
Run Description	NODOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

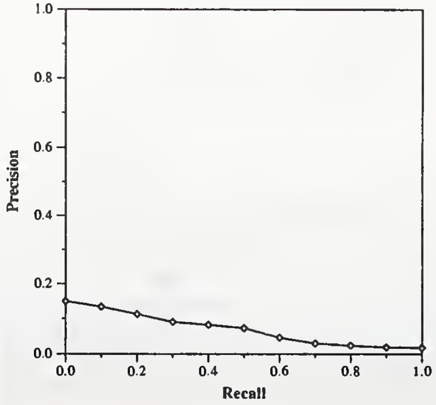
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	285

Recall Level Averages	
Recall	Precision
0.00	0.1492
0.10	0.1331
0.20	0.1122
0.30	0.0904
0.40	0.0820
0.50	0.0729
0.60	0.0463
0.70	0.0301
0.80	0.0240
0.90	0.0195
1.00	0.0177

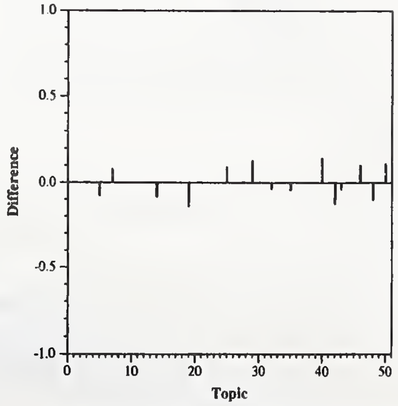
Mean average precision	
non-interpolated	0.0631

Document Level Averages	
	Precision
At 5 docs	0.0520
At 10 docs	0.0540
At 15 docs	0.0560
At 20 docs	0.0490
At 30 docs	0.0407
At 100 docs	0.0246
At 200 docs	0.0194
At 500 docs	0.0104
At 1000 docs	0.0057

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0568



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	UIUC03Wu2
Run Description	NODOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

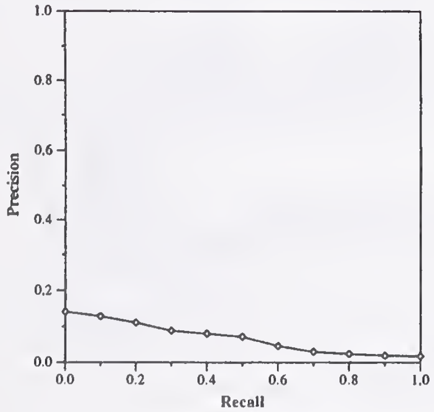
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	285

Recall Level Averages	
Recall	Precision
0.00	0.1411
0.10	0.1287
0.20	0.1113
0.30	0.0886
0.40	0.0803
0.50	0.0716
0.60	0.0458
0.70	0.0300
0.80	0.0239
0.90	0.0194
1.00	0.0177

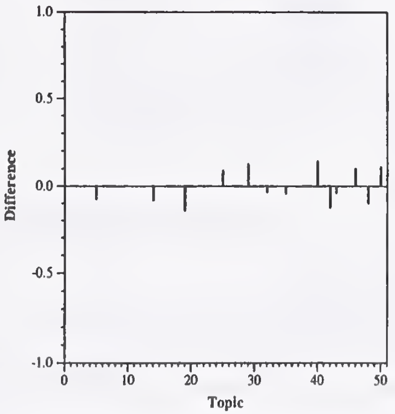
Mean average precision	
non-interpolated	0.0616

Document Level Averages	
	Precision
At 5 docs	0.0440
At 10 docs	0.0540
At 15 docs	0.0560
At 20 docs	0.0490
At 30 docs	0.0400
At 100 docs	0.0246
At 200 docs	0.0193
At 500 docs	0.0105
At 1000 docs	0.0057

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0553



Recall-Precision Curve



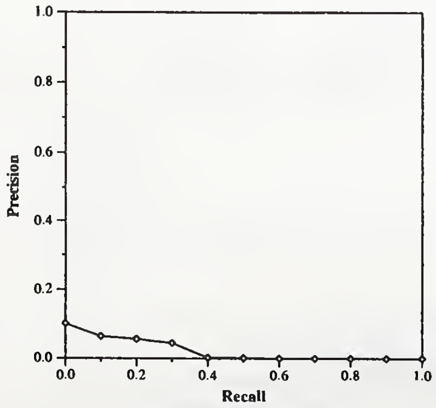
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	C2A
Run Description	NODOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

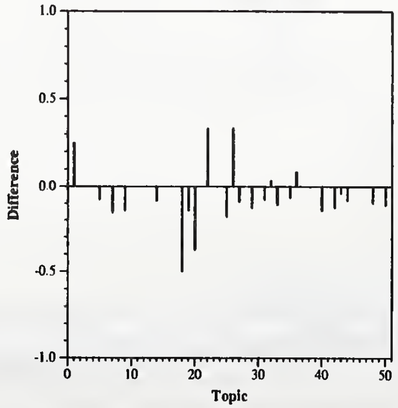
Total number of documents over all topics	
Retrieved:	48496
Relevant:	516
Rel-ret:	107

Recall Level Averages	
Recall	Precision
0.00	0.1015
0.10	0.0644
0.20	0.0562
0.30	0.0449
0.40	0.0027
0.50	0.0019
0.60	0.0006
0.70	0.0004
0.80	0.0004
0.90	0.0004
1.00	0.0004
Mean average precision	
non-interpolated	0.0222

Document Level Averages	
	Precision
At 5 docs	0.0280
At 10 docs	0.0200
At 15 docs	0.0147
At 20 docs	0.0130
At 30 docs	0.0113
At 100 docs	0.0074
At 200 docs	0.0055
At 500 docs	0.0035
At 1000 docs	0.0021
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0230



Recall-Precision Curve



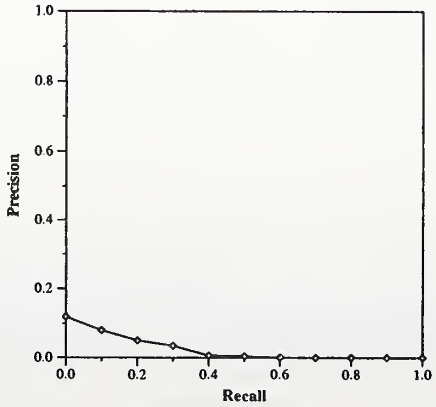
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	C2B
Run Description	NODOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

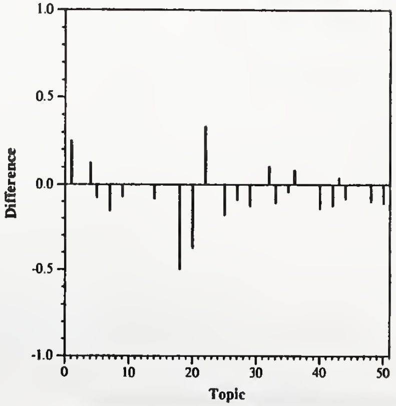
Total number of documents over all topics	
Retrieved:	31929
Relevant:	516
Rel-ret:	115

Recall Level Averages	
Recall	Precision
0.00	0.1196
0.10	0.0797
0.20	0.0503
0.30	0.0349
0.40	0.0067
0.50	0.0048
0.60	0.0012
0.70	0.0006
0.80	0.0006
0.90	0.0006
1.00	0.0006
Mean average precision	
non-interpolated	0.0226

Document Level Averages	
	Precision
At 5 docs	0.0320
At 10 docs	0.0320
At 15 docs	0.0253
At 20 docs	0.0240
At 30 docs	0.0193
At 100 docs	0.0118
At 200 docs	0.0072
At 500 docs	0.0041
At 1000 docs	0.0023
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0281



Recall-Precision Curve



Difference from Median in R-Precision per Topic

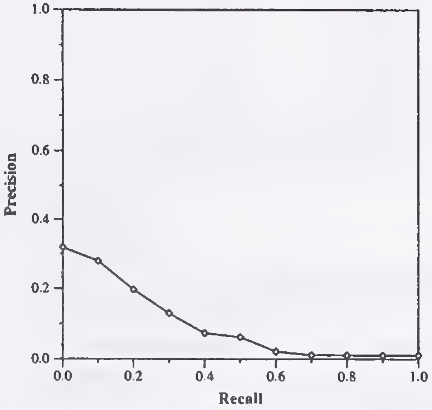
Web track, topic distillation task results — University of Melbourne

Summary Statistics	
Run ID:	MU03td01
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

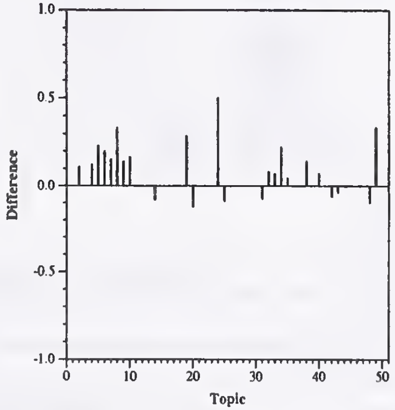
Total number of documents over all topics	
Retrieved:	42676
Relevant:	516
Rel-ret:	182

Recall Level Averages	
Recall	Precision
0.00	0.3183
0.10	0.2793
0.20	0.1974
0.30	0.1298
0.40	0.0739
0.50	0.0627
0.60	0.0217
0.70	0.0116
0.80	0.0112
0.90	0.0105
1.00	0.0105
Mean average precision	
non-interpolated	0.0897

Document Level Averages	
	Precision
At 5 docs	0.1280
At 10 docs	0.0920
At 15 docs	0.0787
At 20 docs	0.0730
At 30 docs	0.0573
At 100 docs	0.0232
At 200 docs	0.0148
At 500 docs	0.0070
At 1000 docs	0.0036
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1096



Recall-Precision Curve



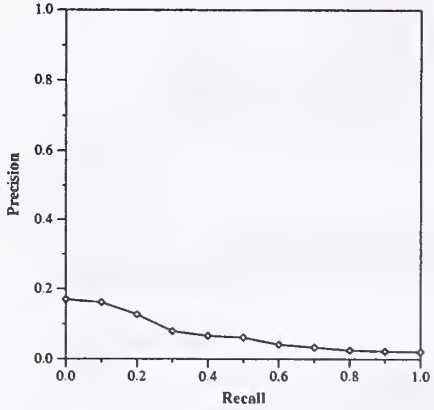
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	MU03td03
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

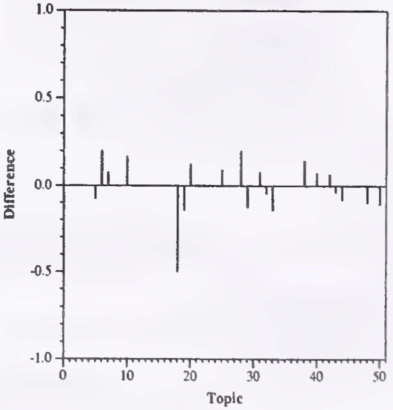
Total number of documents over all topics	
Retrieved:	49343
Relevant:	516
Rel-ret:	325

Recall Level Averages	
Recall	Precision
0.00	0.1698
0.10	0.1614
0.20	0.1269
0.30	0.0799
0.40	0.0666
0.50	0.0617
0.60	0.0416
0.70	0.0335
0.80	0.0254
0.90	0.0226
1.00	0.0208
Mean average precision	
non-interpolated	0.0650

Document Level Averages	
	Precision
At 5 docs	0.0800
At 10 docs	0.0660
At 15 docs	0.0560
At 20 docs	0.0460
At 30 docs	0.0373
At 100 docs	0.0264
At 200 docs	0.0188
At 500 docs	0.0110
At 1000 docs	0.0065
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0537



Recall-Precision Curve



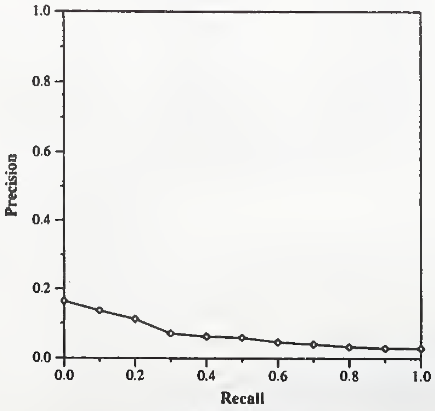
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	MU03td04
Run Description	DOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

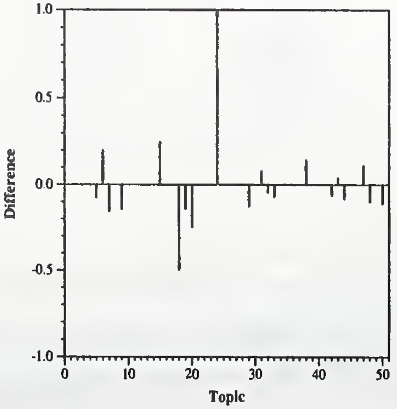
Total number of documents over all topics	
Retrieved:	49343
Relevant:	516
Rel-ret:	291

Recall Level Averages	
Recall	Precision
0.00	0.1640
0.10	0.1364
0.20	0.1125
0.30	0.0711
0.40	0.0626
0.50	0.0596
0.60	0.0465
0.70	0.0406
0.80	0.0334
0.90	0.0297
1.00	0.0292
Mean average precision	
non-interpolated	0.0636

Document Level Averages	
	Precision
At 5 docs	0.0480
At 10 docs	0.0400
At 15 docs	0.0400
At 20 docs	0.0400
At 30 docs	0.0360
At 100 docs	0.0254
At 200 docs	0.0182
At 500 docs	0.0091
At 1000 docs	0.0058
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0559



Recall-Precision Curve



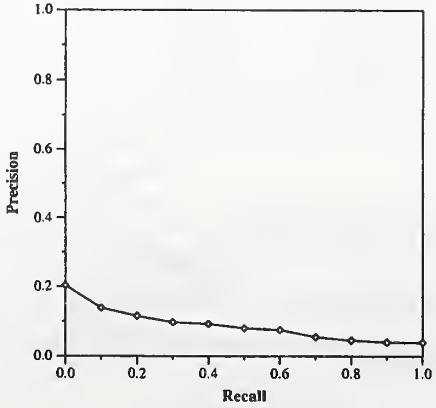
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	MU03td05
Run Description	DOCSTRUCT, NOANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

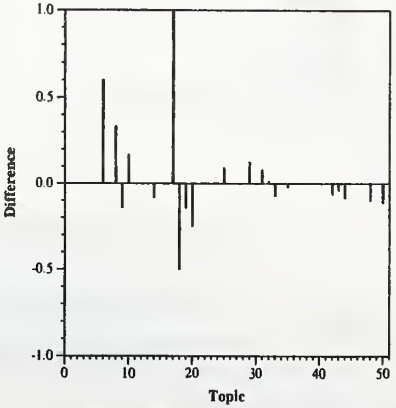
Total number of documents over all topics	
Retrieved:	49343
Relevant:	516
Rel-ret:	325

Recall Level Averages	
Recall	Precision
0.00	0.2029
0.10	0.1385
0.20	0.1152
0.30	0.0973
0.40	0.0924
0.50	0.0803
0.60	0.0754
0.70	0.0549
0.80	0.0457
0.90	0.0404
1.00	0.0394
Mean average precision	
non-interpolated	0.0806

Document Level Averages	
	Precision
At 5 docs	0.0640
At 10 docs	0.0580
At 15 docs	0.0507
At 20 docs	0.0450
At 30 docs	0.0407
At 100 docs	0.0280
At 200 docs	0.0197
At 500 docs	0.0107
At 1000 docs	0.0065
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0728



Recall-Precision Curve



Difference from Median in R-Precision per Topic

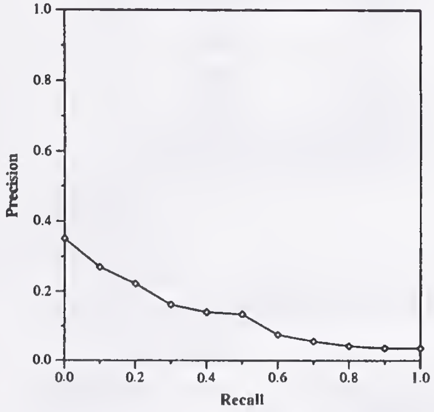
Web track, topic distillation task results — University of Sunderland

Summary Statistics	
Run ID:	SBBASE
Run Description	DOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

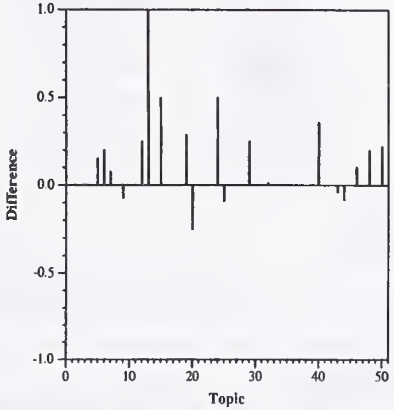
Total number of documents over all topics	
Retrieved:	48400
Relevant:	516
Rel-ret:	293

Recall Level Averages	
Recall	Precision
0.00	0.3510
0.10	0.2700
0.20	0.2222
0.30	0.1626
0.40	0.1411
0.50	0.1351
0.60	0.0749
0.70	0.0555
0.80	0.0418
0.90	0.0360
1.00	0.0354
Mean average precision	
non-interpolated	0.1259

Document Level Averages	
	Precision
At 5 docs	0.1280
At 10 docs	0.1020
At 15 docs	0.0853
At 20 docs	0.0680
At 30 docs	0.0560
At 100 docs	0.0304
At 200 docs	0.0195
At 500 docs	0.0099
At 1000 docs	0.0059
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1283



Recall-Precision Curve



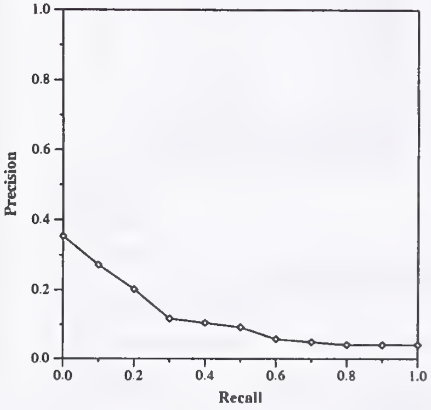
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	SBUNIQUE
Run Description	DOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

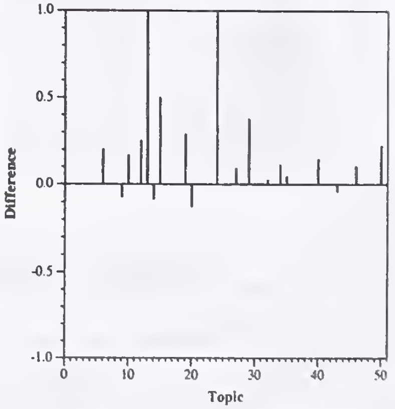
Total number of documents over all topics	
Retrieved:	43780
Relevant:	516
Rel-ret:	136

Recall Level Averages	
Recall	Precision
0.00	0.3537
0.10	0.2719
0.20	0.2019
0.30	0.1175
0.40	0.1046
0.50	0.0920
0.60	0.0573
0.70	0.0486
0.80	0.0406
0.90	0.0406
1.00	0.0406
Mean average precision	
non-interpolated	0.1114

Document Level Averages	
	Precision
At 5 docs	0.1320
At 10 docs	0.0940
At 15 docs	0.0733
At 20 docs	0.0620
At 30 docs	0.0480
At 100 docs	0.0194
At 200 docs	0.0116
At 500 docs	0.0052
At 1000 docs	0.0027
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1407



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	TBBASE
Run Description	DOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

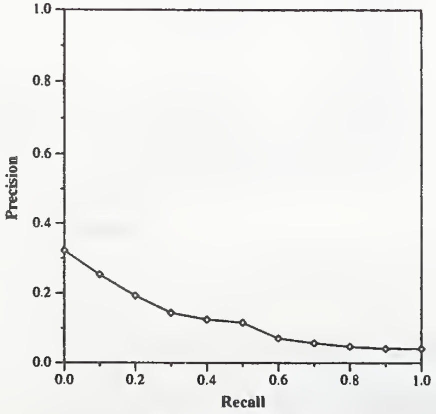
Total number of documents over all topics	
Retrieved:	48400
Relevant:	516
Rel-ret:	289

Recall Level Averages	
Recall	Precision
0.00	0.3222
0.10	0.2531
0.20	0.1929
0.30	0.1442
0.40	0.1253
0.50	0.1166
0.60	0.0712
0.70	0.0574
0.80	0.0470
0.90	0.0412
1.00	0.0407

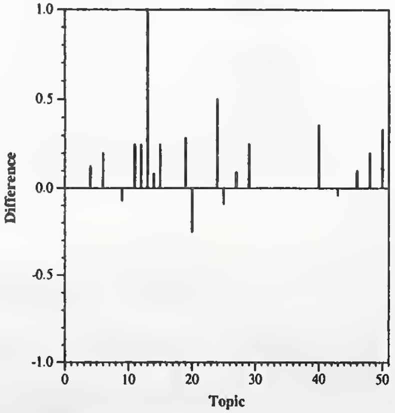
Mean average precision	
non-interpolated	0.1166

Document Level Averages	
	Precision
At 5 docs	0.1240
At 10 docs	0.1020
At 15 docs	0.0840
At 20 docs	0.0690
At 30 docs	0.0573
At 100 docs	0.0296
At 200 docs	0.0190
At 500 docs	0.0096
At 1000 docs	0.0058

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1333



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	TBUNIQUE
Run Description	DOCSTRUCT, NOANCHORTEXT, LINKSTRUCT
Number of Topics:	50

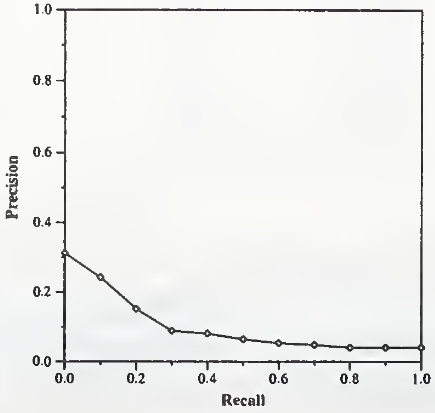
Total number of documents over all topics	
Retrieved:	43818
Relevant:	516
Rel-ret:	128

Recall Level Averages	
Recall	Precision
0.00	0.3122
0.10	0.2423
0.20	0.1519
0.30	0.0891
0.40	0.0813
0.50	0.0648
0.60	0.0534
0.70	0.0483
0.80	0.0406
0.90	0.0406
1.00	0.0406

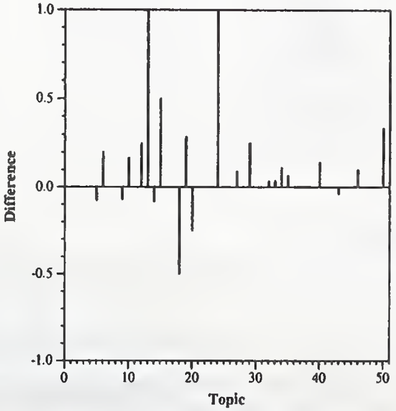
Mean average precision	
non-interpolated	0.0948

Document Level Averages	
	Precision
At 5 docs	0.1280
At 10 docs	0.0880
At 15 docs	0.0693
At 20 docs	0.0600
At 30 docs	0.0467
At 100 docs	0.0182
At 200 docs	0.0104
At 500 docs	0.0048
At 1000 docs	0.0026

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.1278



Recall-Precision Curve



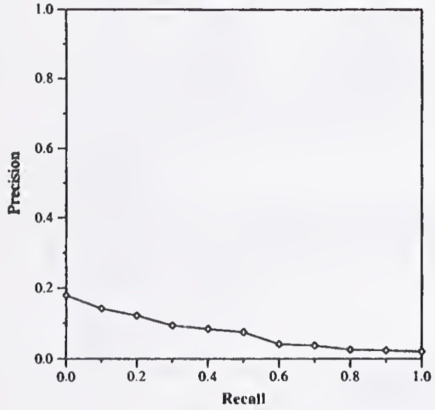
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	VTtdgp33
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

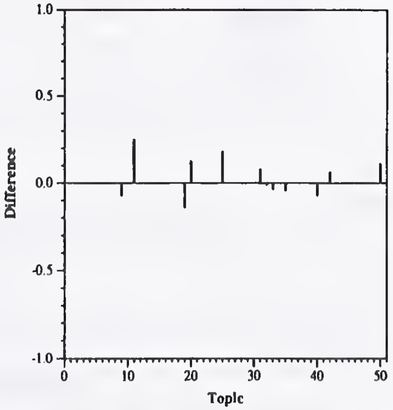
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	316

Recall Level Averages	
Recall	Precision
0.00	0.1796
0.10	0.1418
0.20	0.1213
0.30	0.0935
0.40	0.0838
0.50	0.0751
0.60	0.0413
0.70	0.0372
0.80	0.0259
0.90	0.0238
1.00	0.0201
Mean average precision	
non-interpolated	0.0699

Document Level Averages	
	Precision
At 5 docs	0.0640
At 10 docs	0.0540
At 15 docs	0.0560
At 20 docs	0.0560
At 30 docs	0.0460
At 100 docs	0.0280
At 200 docs	0.0195
At 500 docs	0.0107
At 1000 docs	0.0063
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0655



Recall-Precision Curve



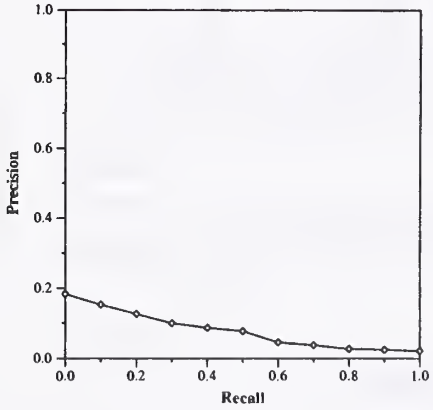
Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	VTtdgp41
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

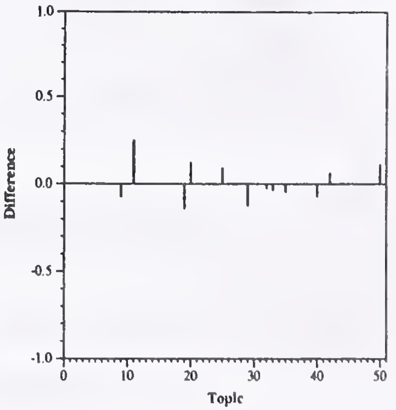
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	325

Recall Level Averages	
Recall	Precision
0.00	0.1838
0.10	0.1530
0.20	0.1265
0.30	0.0999
0.40	0.0866
0.50	0.0774
0.60	0.0458
0.70	0.0376
0.80	0.0273
0.90	0.0249
1.00	0.0213
Mean average precision	
non-interpolated	0.0733

Document Level Averages	
	Precision
At 5 docs	0.0640
At 10 docs	0.0620
At 15 docs	0.0600
At 20 docs	0.0550
At 30 docs	0.0473
At 100 docs	0.0284
At 200 docs	0.0197
At 500 docs	0.0108
At 1000 docs	0.0065
R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0594



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	VTtdgp5055
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

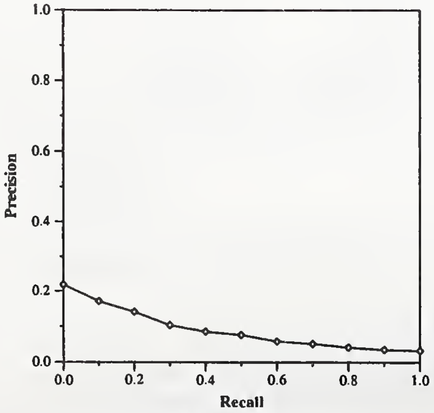
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	351

Recall Level Averages	
Recall	Precision
0.00	0.2181
0.10	0.1707
0.20	0.1412
0.30	0.1039
0.40	0.0858
0.50	0.0764
0.60	0.0584
0.70	0.0514
0.80	0.0415
0.90	0.0353
1.00	0.0322

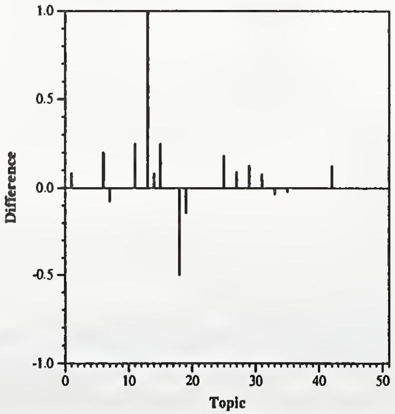
Mean average precision	
non-interpolated	0.0848

Document Level Averages	
	Precision
At 5 docs	0.0880
At 10 docs	0.0760
At 15 docs	0.0587
At 20 docs	0.0550
At 30 docs	0.0520
At 100 docs	0.0288
At 200 docs	0.0211
At 500 docs	0.0117
At 1000 docs	0.0070

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0906



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	VTtdgp52
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

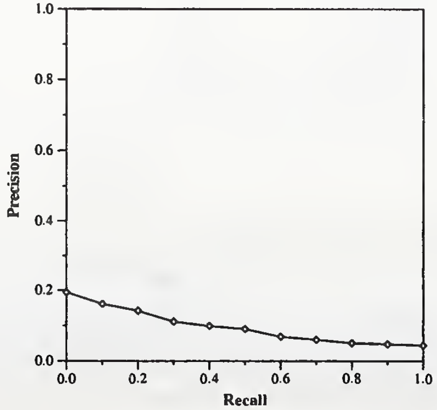
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	315

Recall Level Averages	
Recall	Precision
0.00	0.1935
0.10	0.1609
0.20	0.1413
0.30	0.1112
0.40	0.0988
0.50	0.0900
0.60	0.0682
0.70	0.0597
0.80	0.0501
0.90	0.0474
1.00	0.0437

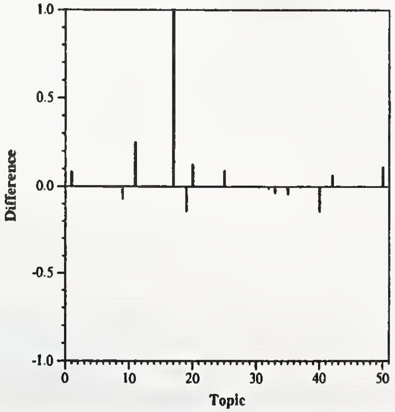
Mean average precision	
non-interpolated	0.0898

Document Level Averages	
	Precision
At 5 docs	0.0680
At 10 docs	0.0660
At 15 docs	0.0640
At 20 docs	0.0560
At 30 docs	0.0487
At 100 docs	0.0292
At 200 docs	0.0197
At 500 docs	0.0107
At 1000 docs	0.0063

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0823



Recall-Precision Curve



Difference from Median in R-Precision per Topic

Summary Statistics	
Run ID:	VTtdok4
Run Description	NODOCSTRUCT, ANCHORTEXT, NOLINKSTRUCT
Number of Topics:	50

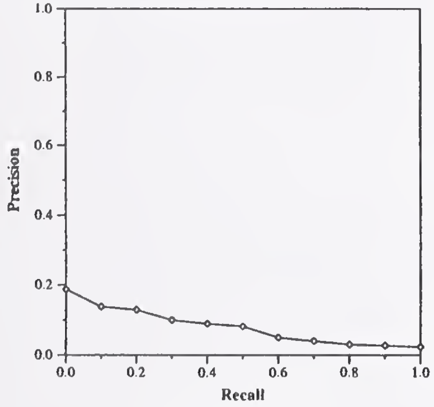
Total number of documents over all topics	
Retrieved:	50000
Relevant:	516
Rel-ret:	355

Recall Level Averages	
Recall	Precision
0.00	0.1873
0.10	0.1383
0.20	0.1293
0.30	0.1003
0.40	0.0898
0.50	0.0822
0.60	0.0498
0.70	0.0400
0.80	0.0301
0.90	0.0271
1.00	0.0231

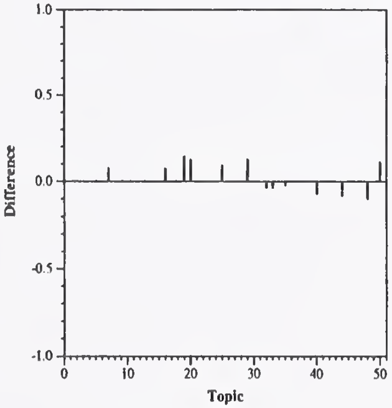
Mean average precision	
non-interpolated	0.0748

Document Level Averages	
	Precision
At 5 docs	0.0840
At 10 docs	0.0620
At 15 docs	0.0600
At 20 docs	0.0530
At 30 docs	0.0447
At 100 docs	0.0286
At 200 docs	0.0195
At 500 docs	0.0113
At 1000 docs	0.0071

R-Precision: precision after R (number relevant) documents retrieved	
Exact	0.0648



Recall-Precision Curve



Difference from Median in R-Precision per Topic

NIST Technical Publications

Periodical

Journal of Research of the National Institute of Standards and Technology—Reports NIST research and development in metrology and related fields of physical science, engineering, applied mathematics, statistics, biotechnology, and information technology. Papers cover a broad range of subjects, with major emphasis on measurement methodology and the basic technology underlying standardization. Also included from time to time are survey articles on topics closely related to the Institute's technical and scientific programs. Issued six times a year.

Nonperiodicals

Monographs—Major contributions to the technical literature on various subjects related to the Institute's scientific and technical activities.

Handbooks—Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications—Include proceedings of conferences sponsored by NIST, NIST annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, and bibliographies.

National Standard Reference Data Series—Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a worldwide program coordinated by NIST under the authority of the National Standard Data Act (Public Law 90-396). NOTE: The Journal of Physical and Chemical Reference Data (JPCRD) is published bimonthly for NIST by the American Institute of Physics (AIP). Subscription orders and renewals are available from AIP, P.O. Box 503284, St. Louis, MO 63150-3284.

Building Science Series—Disseminates technical information developed at the Institute on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.

Technical Notes—Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NIST under the sponsorship of other government agencies.

Voluntary Product Standards—Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The standards establish nationally recognized requirements for products, and provide all concerned interests with a basis for common understanding of the characteristics of the products. NIST administers this program in support of the efforts of private-sector standardizing organizations.

Order the **following** *NIST publications—FIPS and NISTIRs—from the National Technical Information Service, Springfield, VA 22161.*

Federal Information Processing Standards Publications (FIPS PUB)—Publications in this series collectively constitute the Federal Information Processing Standards Register. The Register serves as the official source of information in the Federal Government regarding standards issued by NIST pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations).

NIST Interagency or Internal Reports (NISTIR)—The series includes interim or final reports on work performed by NIST for outside sponsors (both government and nongovernment). In general, initial distribution is handled by the sponsor; public distribution is handled by sales through the National Technical Information Service, Springfield, VA 22161, in hard copy, electronic media, or microfiche form. NISTIR's may also report results of NIST projects of transitory or limited interest, including those that will be published subsequently in more comprehensive form.

U.S. Department of Commerce
National Institute of Standards
and Technology
Gaithersburg, MD 20899–0001

Official Business
Penalty for Private Use $300