NISTIR 5542

Binary Decision Clustering for Neural Network Based Optical Character Recognition

C. L. Wilson
P. J. Grother
C. S. Barnes

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards and Technology
Computer Systems Laboratory
Advanced Systems Division
Gaithersburg, MD 20899

NIST

NISTIR 5542

Binary Decision Clustering for Neural Network Based Optical Character Recognition

C. L. Wilson
P. J. Grother
C. S. Barnes

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards and Technology
Computer Systems Laboratory
Advanced Systems Division
Gaithersburg, MD 20899

December 1994

Binary Decision Clustering for Neural Network Based Optical Character Recognition

C. L. Wilson
P. J. Grother
and
C. S. Barnes

National Institute of Standards and Technology
Gaithersburg, MD 20899

Abstract

A multiple neural network system for handprinted character recognition is presented. It consists of a set of input networks which discriminate between all two class pairs, for example "1" from "7", and an output network which takes the signals from the input networks and yields a digit recognition decision. For a ten digit classification problem this requires forty five binary decision machines in the input network. The output stage is typically a single trained network. The neural network paradigms adopted in these input and output networks are the multi-layer perceptron, the radial basis function network, and the probabilistic neural network. A simple majority vote rule was also tested in place of the output network. The various resulting digit classifiers were trained on 7480 isolated images and tested on disjoint set of size 23140. The Karhunen-Loève transforms of the images of each pair of two classes formed the training set for each BDM. Several different combinations of neural network input and output structures gave similar classification performance. The minimum error rate achieved was 2.5% with no rejection obtained by combining a PNN input array with an RBF output stage. This combined network had an error rate of 0.7% with 10% rejection.

keywords: OCR, neural networks, data clustering, pattern recognition, K-L transform, dynamic systyems.

1 Introduction

In a previous study, the accuracies of statistical and neural network OCR methods were compared [1]. In that study, techniques which used clustering to generate an initial state for learning or statistical analysis such as radial basis functions (RBF) or Quadratic Minimum Distance (QMD) had consistently better performance than multilayer perceptron (MLP) based methods. The best results were obtained from the Probabalistic neural network (PNN) and the ubiquitous k nearest neighbor classifier (KNN). Both operate locally in the feature space and use no learning. This paper outlines a clustering method based on the construction of binary decision machines (BDM), allowing MLP based architectures to become as accurate as the best methods of the previous study. Clustering has been presented as an essential component in many biologically based methods: ART [2, 3, 4] aggregates data using a leader clustering method [5] prior to learning; DYSTAL [6] clusters data into patches during the learning process; cluster formation is a critical element of the learning discussed in [7, 8]; and FAUST [9] uses clustering of data to selectively control the learning process. In MLP based character recognition, the weight sharing methods [10, 11, 12, 13] provide a method that effectively clusters the input feature space. In neighbor based methods, the local intrinisic dimensionality [14] has been recognized as a critical factor in the pattern recognition capabilities of these methods.

In this paper we present an alternate method for clustering the feature data for OCR. It is more readily adapted to vector based feature sets than weight sharing and it can give high accuracy classification with only a simple winner-take-all voting method as an output process. When a more complex combination of input and output networks is used the error reject performance of BDM-based OCR is comparable to the best systems presented at the First Census OCR Systems Conference [15] and the improved PNN system in the NIST form-based OCR system [16]. In [1] and [15] many different OCR systems were presented which achieved 5% error rates and several others gave 2%-3% error rates. By reducing the problem to a series of BDMs we show that several different neural network methods can achieve 3% error rates. These systems also can exhibit error reject behavior comparable to the best presented in the original study. Analysis of the digit by digit performance with respect to feature set size and network type shows that much of this improvement is associated with local rank reduction in the feature set. The following section describes the network architectures and the functional forms of the various input and output networks. Section 3 describes the training and testing data. Section 4 details recognition accuracy, and error versus reject performance. Section 5 discusses the results and gives some conclusions.

2 Network Structures

2.1 K-L Feature Extraction

The Karhunen Loève expansion of digit images is used as reduced dimensionality, optimally compact representation for input to the BDMs. The use of such features in OCR has been described in, for example, [17] [18] [1]. The handwritten binary characters are size and orientation normalized and represented as the ± 1 elements of a vector by some consistent ordering of the square image. The mean vector of P such images is subtracted from each and an ensemble matrix, $\mathbf{U}$ is formed with these P vectors as its columns. The symmetric covariance matrix, $\mathbf{R}$, gives the mean of of all the interpixel correlations.

$$\mathbf{R} = \frac{1}{P}\mathbf{U}\mathbf{U}^T \tag{1}$$

The covariance matrix $\mathbf{R}$ has eigenvectors as the columns of Ψ defined as:

$$\mathbf{R}\Psi = \Psi\Lambda \tag{2}$$

where the only non zero elements of Λ are the eigenvalues on its diagonal. The eigenvectors are the directions of maximum variance in the image space and form a complete orthonormal basis. They are the principal axes of a hyperellipse in that space. The eigenvalues define the statistical "length" of these axes; thus the first column of Ψ corresponding to the largest eigenvalue is the major axis. The eigensolution of the covariance matrix provides an ordered variance expansion of the image ensemble. The latter eigenvectors, describing very little variance in the images, are discarded thus affording reduced dimensionality. The Karhunen Loève transforms, $\mathbf{V}$, are just the projection of the zero mean images onto the principal axes:

$$\mathbf{V} = \Psi^T\mathbf{U} \tag{3}$$

For each BDM the covariance matrix, its eigenvectors, and the KL expansions are found independently for the appropriate class pair. The expense of doing this for 45 BDMs is considerable, although because it and any necessary training is performed off-line the resulting classifer is not compromised with regard to efficiency. The K-L transform forms the initial stage of classification. The 45 eigenvector sets perform the feature extraction for an unknown image ahead of the BDM input layer, so that the matrix multiplications conceptually define a linear first layer of the complete classifier.

2.2 Input and Output Networks

The architecture is divided into the input and output networks. The input consists of 45 BDMs each specializing on one digit pair. Each machine is trained only on the samples from its two classes such that the final assembly of all machines uses each class in combination with the other nine classes with no symmetric repetition. The training and testing of the input machines was done using up to 32 K-L components. The decision machines are either type 1 or 2 MLP's, type 1 or 2 RBF's, or PNN's. Type 1 MLP1 possesses two outputs, one for each class while MLP2 produces one output, a high or low value indicating the class. The two RBF variants are described in the next section. The outputs of the MLP and RBF classifiers are normalized by dividing by their sum, producing an input vector to the second stage whose elements sum to unity. For a PNN input layer the logarithm of the signals is taken before being handed to the output network.

The output network either implements the baseline voting rule or one of the neural network paradigms. The RBF output stages are parameterized by the number of clusters per class, and results for several values are reported.

The MLP1, MLP2, PNN, RBF1 and RBF2 classifiers were used as the 45 Input BDMs. Each type of network was trained on upto 32 K-L coefficients. The output signals of each machine are the input signals for the output stage. The individual training set for each digit is run on all 45 BDMs in order to create the training feature set for the Output Network. The output signals are combined in a set order to produce the feature set for the input to the Output Network.

The voting rule for the MLP1 BDMs is a winner-takes-all approach. A BDM votes for a class based on whether its output is high or low based on a threshold value of 0.5. The 45 votes are totalled by class, the maximum determining the hypothesis. In the case of a tie, the pattern is rejected. The maximum vote is 9. The other types of binary machine, namely MLP2, PNN, and the two RBF variants, each have two outputs. The voting rule accepts the class corresponding to the greater signal from each BDM. Again the maximum vote determines the classification, and ties imply rejection.

For the neural network output stages the five classifiers are trained on the 45 or 90 outputs from the binary decision machines. Training is achieved using the methods given in the following classifier description.

2.3 Classifiers

The MLP, RBF and PNN classifier variants are described below. All are putatively neural networks though PNN is closely analogous to the non-parametric Parzen type classifier.

2.3.1 Multi-Layer Perceptron

This classifier is also known as a feed forward neural net. We have used an MLP with three layers (counting the inputs as a layer). It will be convenient to define the following notation:

$$\begin{aligned}
N^{(i)} &= \text{number of nodes in } i^{\text{th}} \text{ layer } (i = 0,1,2),\ N^{(0)} = n,\ N^{(2)} = L \\
f(x) &= 1/(1+e^{-x}) = \text{sigmoid function} \\
b_i^{(k)} &= \text{bias of } i^{\text{th}} \text{ node of } k^{\text{th}} \text{ layer } (k = 1,2) \\
w_{ij}^{(k)} &= \text{weight connecting } i^{\text{th}} \text{ node of } k^{\text{th}} \text{ layer to } j^{\text{th}} \text{ node of} \\
&\quad\ (k-1)^{\text{th}} \text{ layer } (k = 1,2; 1 \le i \le N^{(k)}; 1 \le j \le N^{(k-1)})
\end{aligned}$$

The discriminant functions are then of the form

$$D_i(\mathbf{x}) = f\left(b_i^{(2)} + \sum_{j=1}^{N^{(1)}} w_{ij}^{(2)} f\left(b_j^{(1)} + \sum_{k=1}^{N^{(0)}} w_{jk}^{(1)} x_k\right)\right).$$

For the training of this network, we followed usual procedure of employing an optimization algorithm. Although backpropagation [19] is ubiquitous and readily available we favored a scaled conjugate gradient method [20, 21, 22, 23] affording order of magnitude training time gains. The standard mean squared error function between the actual and desired discriminant values is modified by the addition of a scalar "regularization" term [24], equal to a tunable constant, λ, multiplied by the mean square weight, $\overline{w_{ij}^2}$. This term prevents the large weights associated with the overfitting of the training data; although a higher training error is obtained the resulting networks have been shown to have an increased generalization ability [23].

2.3.2 Radial Basis Functions

Typically the RBF neural networks employ radially symmetric Gaussian kernels as their input layer. Our implementation of RBF nets is more general: the kernels are defined by an ellipsoid whose axes remain parallel to the coordinate axes and whose shape is parameterized by a vector of widths.

We have experimented with RBF networks of two types, which will be denoted RBF1 and RBF2. The following notation will be convenient:

$$\begin{aligned}
N^{(i)} &= \text{number of nodes in } i^{\text{th}} \text{ layer } (i = 0, 1, 2)\\
\mathbf{c}^{(j)} &= \text{center vector of } j^{\text{th}} \text{ hidden node } (1 \leq j \leq N^{(1)})\ (\mathbf{c}^{(j)} \in \mathbf{R}^n) = (c_1^{(j)}, \ldots, c_n^{(j)})^{\mathrm{T}}\\
\boldsymbol{\sigma}^{(j)} &= \text{width vector of } j^{\text{th}} \text{ hidden node } (1 \leq j \leq N^{(1)})\ (\boldsymbol{\sigma}^{(j)} \in \mathbf{R}^n) = (\sigma_1^{(j)}, \ldots, \sigma_n^{(j)})^{\mathrm{T}}\\
b_j^{(k)} &= \text{bias to the } j^{\text{th}} \text{ node of the } k^{\text{th}} \text{ layer}\\
f(x) &= 1/(1+e^{-x}) = \text{sigmoid function}\\
w_{ij} &= \text{weight connecting } i^{\text{th}} \text{ output node to } j^{\text{th}} \text{ hidden node } (1 \leq i \leq N^{(2)}; 1 \leq j \leq N^{(1)})
\end{aligned}$$

Each hidden node computes a radial basis function. For RBF1, these functions are unbiased exponentials

$$\phi_j(\mathbf{x}) = \exp\left(-r^2(\mathbf{x}, \mathbf{c}^{(j)}, \boldsymbol{\sigma}^{(j)})\right),$$

and for RBF2, they are of the biased sigmoidal form

$$\phi_j(\mathbf{x}) = f\left(-b_j^{(1)} - r^2(\mathbf{x}, \mathbf{c}^{(j)}, \boldsymbol{\sigma}^{(j)})\right).$$

For either type of RBF, the i^{th} discriminant function is the following function of the radial basis functions:

$$D_i(\mathbf{x}) = f\left(b_i^{(2)} + \sum_{j=1}^{N^{(1)}} w_{ij}\phi_j(\mathbf{x})\right).$$

The centers $\mathbf{c}^{(j)}$, widths $\boldsymbol{\sigma}^{(j)}$, hidden-node bias weights $b_j^{(1)}$ (RBF2 only), output-node bias weights $b_i^{(2)}$, and output-node weights w_{ij} may be collectively thought of as the trainable "weights" of the RBF network. They are trained initially using the cluster means (from a "K-means" algorithm applied to the prototype set) as the center vectors $\mathbf{c}^{(j)}$. The width vectors $\boldsymbol{\sigma}^{(j)}$, are set to a single tunable positive value. More sophisticated methods of determining RBF parameters may be found in [25] [26]. The output layer weights are set such that each output node is connected with a positive weight to hidden nodes of its class (that is, hidden nodes whose initial center vectors are means of clusters from its class), and connected with a negative weight to hidden nodes of other classes. Training proceeds by optimization identical to that described for the MLP.

2.3.3 Probabilistic Neural Net

This classifier is proposed in a recent paper by Specht [27]. Each training prototype defines the center of a kernel whose value is maximum at the center and decreases monotonically with distance from the example in the feature space. An unknown x is classified by computing, for each class i, the sum of the values of the class-i kernels at x, multiplying these numbers by compensatory factors involving the estimated *a priori* probabilities, and picking the class whose resulting discriminant value is highest. Many forms are possible for the kernel functions; we have obtained our best results using radially symmetric Gaussian kernels. The resulting discriminant functions are of the form

$$D_i(\mathbf{x}) = \frac{\hat{p}(i)}{M_i}\sum_{j=1}^{M_i} \exp\left(-\frac{1}{2\sigma^2} d^2\left(\mathbf{x}, \mathbf{x}_j^{(i)}\right)\right),$$

where σ is a scalar "smoothing parameter" that is empirically optimized. The value used throughout this study was 3.

3 Test and Training Data

The classifiers described in this paper were trained and tested using feature vectors derived from the digit images of NIST Special Database 3 [28]. This database consists of binary 128 by 128 pixel raster images segmented from the sample forms of 2100 writers published on CD as [29]. Other results on segmentation and recognition of this database have been reported [30]. The relative difficulties of the NIST OCR databases have been discussed in [31]. For this study samples are drawn randomly from the first 250 writers to yield a training set of 7480 digits with *a priori* class probabilities all equal to 0.1. Even for digits, depending on the application, certain classes may be more prevalent; in banking tasks, for example, "0" is more common. The test set is similarly constructed from the second 250 writers yielding 23140 samples. The images are size normalized by pixel deletion, stroke width bounded by binary erosion and dilation, and consistent orientation is effected by shearing rows by an amount determined by the leftmost and rightmost pixels in the first and last rows defining a vertical line. The resulting image is 32 pixels high and its width is less than or equal to its height. The training and testing data are identical to the data used in [1].

4 Results

4.1 K-L Feature Extraction

One of the advantages of applying the K-L transform to the BDM data set is that insight into problem difficulty can be obtained directly from the KNN classification accuracy for each BDM as shown in table 1 and by plotting the first two K-L features of some typical problems. The problems chosen were separation of "0" and "1", of "6" and "8" and of "3" and "8". The "0-1" problem is the canonical easy task [32], the "6-8" problems is the one which involves the easy ("6") and hard ("8") digits and the "3-8" problem is one that can be difficult on the test data set even for humans.

Examination of table 1 shows that when one K-L feature is used even the hard problem can be solved with 10.6% error and the errors of the two easy problems are less than 5%. This demonstrates that very simple low dimensional methods can produce results comparable to those found on the global problem in [1, 15] by complex networks. There is also a substantial difference in error rates for the easy and hard problems for any number of features. The easy problems, "0-1" and "6-8", start with 4% error and fall to 0.4% error. The hard "3-8" problem starts at 10.6% and never falls below 1.4%. All three tasks have reached their peak performance using 32 features which is less than that required for the global problem[1].

Examination of figure 1 shows that most of the "0" and "1" points are clearly separated but there are some nearest neighbors of the opposite class. The data in table 1 shows that these points are usually

Feature	"0-1"	"6-8"	"3-8"
1	4.29	4.58	10.60
2	4.01	1.55	9.09
3	1.42	1.31	9.05
4	0.97	1.21	5.89
5	0.47	1.03	5.66
8	0.34	0.54	3.43
16	0.43	0.43	1.90
24	0.41	0.36	1.46
32	0.38	0.30	1.44
48	0.38	0.38	1.44

Table 1: Error in separating the test digits for selected BDM machines using KNN as a function of feature dimensionality.

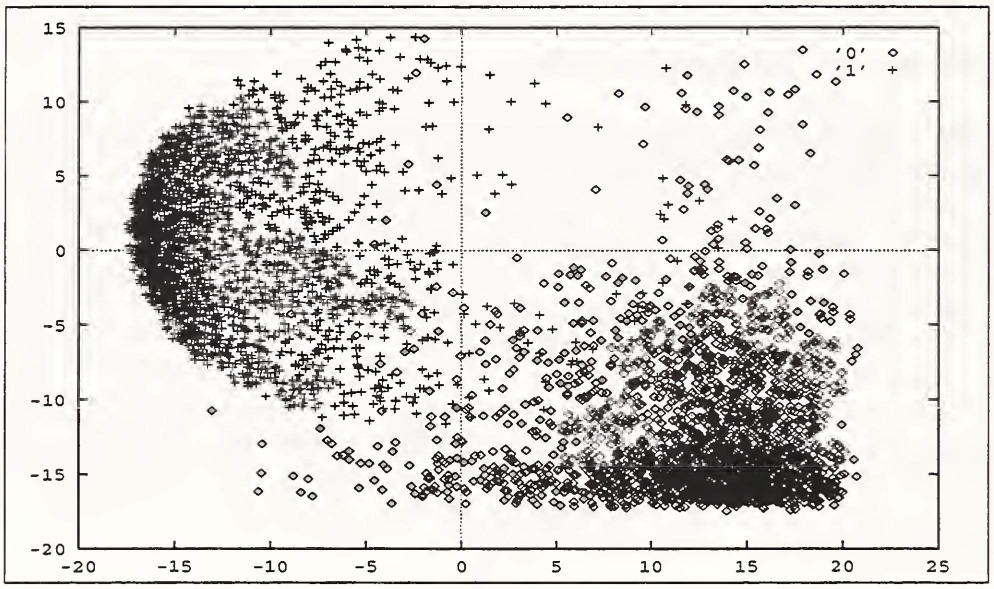

Figure 1: The separation of "0" and "1" testing data using the first two K-L features from "0" and "1" training data. 2314 examples of each digit are shown. The class designations are in at top right.

separated, if they can be separated, using 5 features. Examination of figure 2 shows that most of the "6" and "8" points are clearly separated and there are few near neighbor points of the opposite class. The data in table 1 shows that these points are well separated using 2 features. The simplification of this problem using two features shows why it is an easy problem at low dimension but table 1 also shows that at 5 features it is intermediate in difficulty between "0-1" and "3-8". Examination of figure 3 shows that most of the "3" and "8" points are not separated; their are many near neighbor points of the opposite class. The data in table 1 shows that these points are never separated as well as they are for easier problems.

These results show that the K-L transform reduces the difficult OCR problem to a relatively simple BDM problem so long as the BDMs are only asked to classify characters of the optimal class. We will show that the global problems is still difficult when each machine is required to classifly digits from other classes.

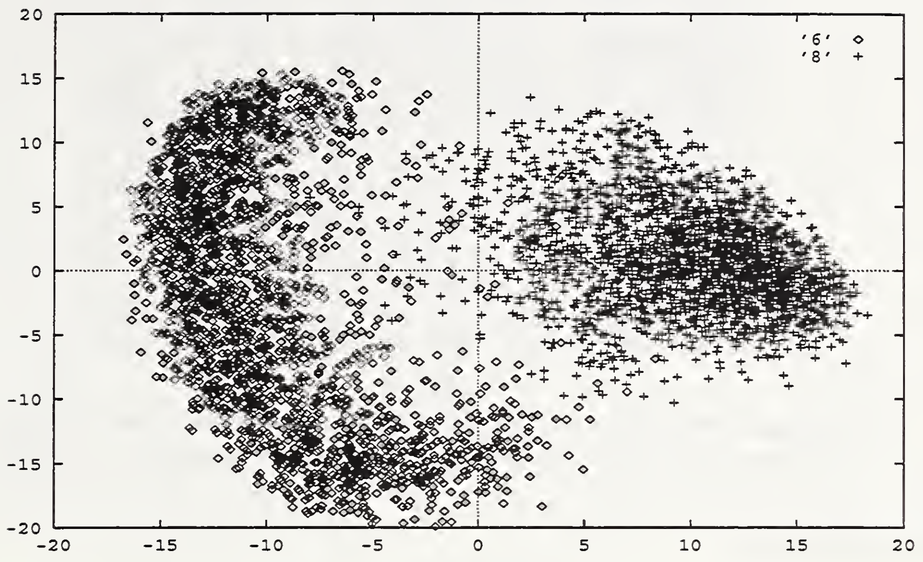

Figure 2: The separation of "6" and "8" testing data using the first two K-L features from "6" and "8" training data. 2314 examples of each digit are shown.

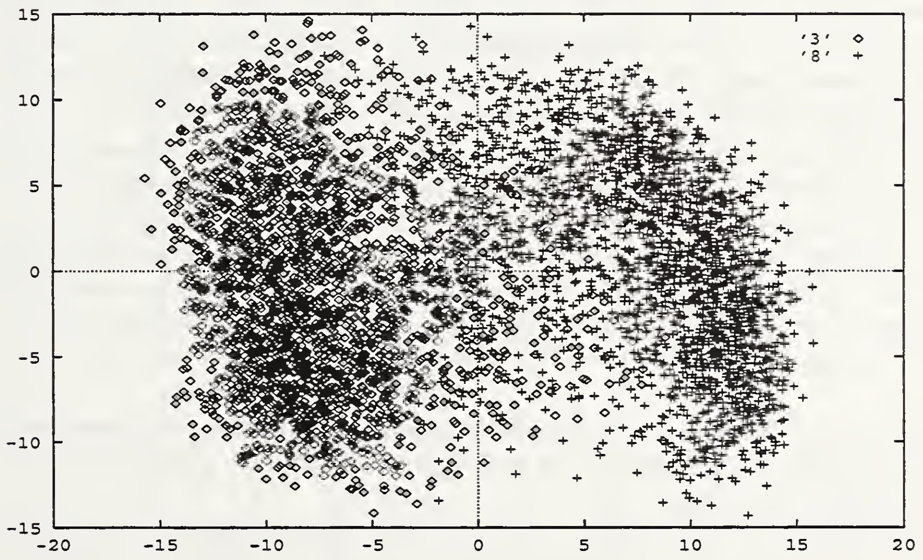

Figure 3: The separation of "3" and "8" testing data using the first two K-L features from "3" and "8" training data. 2314 examples of each digit are shown.

Classifier	Number of features						
	8	12	16	20	24	28	32
MLP1-ERR	3.9	3.2	3.0	2.9	3.0	2.8	3.2
MLP1-REJ	1.6	1.1	3.0	0.9	1.0	0.9	1.0
MLP2-ERR	3.9	3.2	3.2	2.6	2.9	2.9	2.9
MLP2-REJ	1.6	1.0	0.9	0.9	0.9	0.9	0.9
PNN-ERR	3.7	3.0	2.8	2.7	2.7	2.6	2.6
PNN-REJ	0.4	0.3	0.2	0.1	0.1	0.1	0.1
RBF1-ERR	6.8	6.3	6.1	5.9	5.8	5.8	6.4
RBF1-REJ	1.5	1.5	1.5	1.6	1.6	1.6	1.5
RBF2-ERR	6.4	5.2	5.0	5.1	4.7	4.6	3.4
RBF2-REJ	1.4	1.5	1.5	1.6	1.7	1.6	1.6

Table 2:
Voting Rule Error and Reject Percentages for Classifiers and Number of Features. Reject Percentages based on the number of tied votes.

4.2 Classification

The results are grouped by output classification paradigm. The performance from the voting schemes is given first and is followed by the larger tables from the trained classifiers.

Table 2 shows the results of using the direct voting scheme to classify the output of the neural networks. The error rate and the percent rejected due to ties are given. Note that the PNN machines have both the lowest error and reject rates. As is typical there is a decrease in error as more features are used.

The following tables give by class error rates for one particular input BDM type. The bold face entries in the tables indicate the combination of features and networks in the output network which had the lowest error for that digit. Blank entries in the tables usually indicate that training failed for that network. This is characterized by the optimization algorithm yielding a set of weights that pathologically give very large error rates. It is generally true throughout the tables that class "8' has the highest error rate while class "6" has the lowest. It appears that MLP and RBF are usually superior to PNN as second stages for any given input BDM layer. Often 20 features are sufficient to obtain best classification, although the clustered RBF second stages are notably more parsimonious.

The second set of tables gives the dependence of error rate on feature dimensionality for each type of BDM and output stages. The bold face indicates the second stage classifier and the dimensionality that gave minimum classification error over all classes. The error rate of the best combination, MLP2 BDMs and the voting classifier, is 2.6%. This rate is comparable with the best nonparametric classifiers in [1], and is obtained with 20 instead of 40 K-L components. With the exception of RBF type BDM input layers, the voting classifier outperforms the neural network second stages by about 0.5%. However as discussed in the next section both the voting and PNN output networks provide poor reject accuracy results.

Several of the neural BDM and classifier combinations gave a lower error rate than their counterparts in the classifier study [1]. For MLP's this superiority is 1% and for RBF about 0.9% and obtained with significantly fewer K-L components.

These results also show a strong effect associated with the saturation of the sigmoid function in the MLP1, MLP2, RBF1, and, to a lesser extent, the RBF2 network. In tables 4-7 and tables 9-12, commection of any of these input networks to a PNN output network results in abnormally high errors. This result is caused by the saturation of the outputs of MLP and RBF input networks providing only near zero and near unity values to the PNN output networks. The PNN networks only function well when continuous input values, such as the K-L features, are provided. This effect was previously noted in the reject performance of other types of neural networks [33] and is discussed in the next section

n	DIGIT	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"
8	MLP:48	7.1	4.1	7.4	14.5	23.3	5.5	8.0	7.7	8.0	13.0
	PNN	1.5	1.4	3.6	4.6	2.8	4.0	1.0	2.8	7.1	4.6
	RBF1:30	0.9	1.7	7.5	4.2	4.4	4.4	2.1	2.2	9.4	4.3
16	MLP:48	2.0	**1.0**	4.0	**2.2**	4.2	2.9	2.5	2.8	9.4	4.4
	PNN	1.5	1.5	3.4	4.3	3.3	3.7	1.1	4.8	**7.0**	3.9
	RBF1:30	1.5	1.6	4.2	3.1	3.3	3.8	0.7	2.6	8.5	**3.8**
	RBF2:30	1.8	2.1	4.0	2.7	2.8	4.4	**0.5**	2.2	6.9	4.1
20	MLP:48	2.5	1.3	5.1	5.0	4.0	2.9	2.6	**1.0**	9.5	11.0
	PNN	1.4	1.3	4.0	4.2	3.2	3.8	1.1	2.4	7.4	3.8
	RBF1:30	1.4	3.1	5.0	2.5	3.1	3.9	0.5	1.6	8.9	4.3
	RBF2:30	1.5	1.8	4.2	2.8	2.9	3.9	0.5	2.2	10.4	4.3
32	MLP:48	2.8	1.2	4.3	7.3	2.5	**2.2**	1.0	1.8	11.5	7.5
	PNN	**0.1**	1.6	7.5	5.2	3.8	5.4	1.7	2.5	10.1	4.1
	RBF1:30	1.5	1.6	3.1	4.4	3.3	2.8	0.6	2.0	8.4	4.1
	RBF2:30	1.3	2.2	**3.0**	2.5	**2.0**	3.1	0.6	1.5	12.5	5.0

Table 3:
By class error rates for PNN BDMs and various output stages. The left column gives the K-L dimensionality. The integer suffix for the output stages gives the number of hidden units for MLPs and the number of cluster centers for RBF networks.

n	DIGIT	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"
16	MLP:48	1.6	2.1	4.7	**3.4**	2.2	4.2	1.7	2.2	7.3	5.1
	PNN	12.0	5.0	20.0	6.0	9.5	12.3	1.3	4.7	26.7	7.7
	RBF1	1.4	2.0	4.7	3.5	2.4	4.3	1.6	2.5	7.3	4.7
	RBF2	1.5	2.2	4.9	3.5	2.4	4.5	1.4	2.3	7.3	5.1
20	MLP:48	1.6	1.9	4.1	3.6	**1.9**	3.7	1.6	2.1	7.2	**4.2**
	PNN	10.2	8.3	24.6	6.4	22.1	13.6	**0.4**	9.8	27.1	8.0
	RBF1	**1.2**	2.0	4.3	3.5	2.4	**3.8**	1.4	**2.0**	7.3	4.5
	RBF2	1.6	**1.9**	4.2	3.7	2.4	3.9	1.7	2.2	**7.0**	4.6
24	MLP:48	1.5	2.2	4.0	3.6	2.3	4.6	1.3	2.8	7.3	4.7
	PNN	10.5	7.2	25.6	7.0	20.1	16.5	0.4	8.3	30.7	8.4
	RBF1	1.6	2.3	**4.0**	3.8	2.2	4.8	1.0	2.5	7.2	4.4
	RBF2	1.5	2.2	4.1	3.8	2.0	4.5	1.5	2.6	7.4	4.4
32	MLP:48	1.6	2.3	4.7	4.2	2.5	4.7	1.4	2.3	8.4	4.9
	PNN	8.2	7.3	27.7	7.8	19.2	15.5	0.5	8.9	27.9	7.5
	RBF1	1.6	2.4	5.0	4.0	2.5	4.4	1.2	2.4	8.5	4.6
	RBF2	1.6	2.5	5.4	3.9	2.5	4.6	1.1	2.3	8.7	4.6

Table 4:
By class error rates for MLP1 BDMs and various output stages. The left column gives the K-L dimensionality. The integer modifier to the MLP nets indicates number of hidden units.

n	DIGIT	“0”	“1”	“2”	“3”	“4”	“5”	“6”	“7”	“8”	“9”
16	MLP:48	1.5	2.0	4.7	3.9	2.6	4.3	1.5	2.3	8.1	4.8
	PNN	2.5	2.2	8.3	5.0	1.9	5.0	0.9	2.3	16.5	6.1
	RBF1	1.6	2.3	4.4	3.8	2.5	3.9	1.6	2.5	8.2	5.0
	RBF2	1.4	2.2	4.6	3.9	2.3	4.0	1.5	2.5	8.2	4.9
20	MLP:48	**0.3**	1.9	3.5	**3.5**	2.2	**3.7**	1.1	1.9	8.1	**4.1**
	PNN	1.2	2.3	7.9	4.7	2.4	4.7	**0.8**	**1.8**	16.5	6.6
	RBF1	0.1	**1.8**	3.2	3.7	2.3	3.8	1.2	2.1	8.0	4.6
	RBF2	0.1	1.9	**3.1**	3.6	2.5	3.7	1.2	2.1	8.2	4.5
24	MLP:48	1.5	2.4	4.2	3.6	2.3	4.1	1.4	2.1	7.3	4.6
	PNN	2.5	2.0	8.2	4.4	**1.7**	5.2	1.0	2.6	16.9	6.0
	RBF1	1.6	2.2	4.1	3.8	2.1	4.1	1.2	2.3	**6.1**	4.4
	RBF2	1.6	2.8	4.2	3.5	2.3	3.9	1.2	2.2	7.6	4.3
32	MLP:48	1.2	2.3	4.0	3.9	2.4	4.1	0.9	2.0	7.7	4.3
	PNN	2.5	2.0	7.9	5.1	2.0	4.6	0.9	2.2	15.6	6.0
	RBF1	1.2	2.2	4.0	3.8	2.2	4.3	1.0	2.2	7.6	4.4
	RBF2	1.5	2.2	4.2	3.7	2.2	4.3	1.0	2.2	7.7	4.4

Table 5:
By class error rates for MLP2 BDMs and various output stages. The left column gives the K-L dimensionality.

n	DIGIT	“0”	“1”	“2”	“3”	“4”	“5”	“6”	“7”	“8”	“9”
12	MLP:48	**2.0**	**2.1**	4.6	4.5	3.4	4.1	1.4	2.5	**7.9**	5.3
	PNN	3.0	3.8	13.1	5.7	10.0	9.9	2.2	5.5	27.5	9.8
	RBF1	2.5	2.4	4.8	4.8	3.4	4.4	2.8	2.8	10.1	5.8
	RBF2	2.5	2.5	4.2	**3.9**	3.3	4.2	2.4	**2.4**	8.9	5.9
20	MLP:48	2.0	2.2	**4.2**	4.4	2.6	3.8	**1.4**	3.2	8.4	**5.0**
	PNN	2.7	3.4	13.4	5.8	9.5	8.5	2.4	5.4	27.6	8.9
	RBF1	2.7	2.5	5.3	4.6	3.4	4.1	2.1	2.6	9.7	5.6
	RBF2	2.3	2.4	5.2	4.5	3.0	4.4	2.1	2.9	8.8	5.0
24	MLP:48	2.1	2.2	4.4	5.0	3.0	**3.4**	1.5	3.0	8.7	5.7
	PNN	2.8	3.3	13.3	5.6	7.9	9.1	2.2	5.4	28.3	8.2
	RBF1	2.6	2.2	5.1	4.9	2.9	4.5	1.9	3.2	10.1	5.9
	RBF2	2.1	2.2	4.7	4.7	**2.5**	4.3	2.1	2.9	9.2	5.3

Table 6:
By class error rates for RBF1 BDMs and various output stages. The left column gives the K-L dimensionality.

n	DIGIT	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"
8	MLP:48	2.1	3.1	5.1	3.9	4.8	4.4	2.0	3.9	9.9	5.6
	PNN	3.0	2.7	12.7	6.3	9.5	8.2	2.0	5.0	26.4	9.6
12	MLP:48	1.9	2.4	5.2	4.6	2.9	4.1	2.0	2.8	15.3	5.2
	PNN	3.1	2.3	**2.0**	5.4	6.6	8.8	1.8	3.7	20.8	8.7
16	MLP:48	**1.8**	2.1	5.0	4.3	2.6	4.3	2.0	3.2	**7.8**	5.1
	PNN	2.7	2.2	10.9	5.3	6.1	7.8	1.9	3.1	23.0	7.2
20	MLP:48	1.9	2.5	5.0	4.0	3.0	4.7	1.9	3.3	8.7	5.3
	PNN	2.8	1.9	12.3	4.9	6.1	7.8	1.9	3.5	22.0	7.7
24	MLP:48	1.9	2.8	4.6	4.2	**2.2**	4.1	2.2	3.0	8.1	5.0
	PNN	2.6	2.0	12.4	4.5	6.8	6.6	**1.8**	4.3	23.8	6.8
28	MLP:48	2.0	2.3	5.7	4.4	2.3	4.4	1.9	**2.7**	8.3	5.2
	PNN	2.6	**2.0**	12.4	4.9	6.5	7.9	1.9	3.7	22.8	6.1
32	MLP:48	1.7	2.3	5.6	**3.8**	2.7	**3.6**	3.2	4.1	12.4	**4.6**
	PNN	2.5	2.0	11.6	4.8	5.4	7.6	1.9	3.7	16.6	6.2

Table 7:
By class error rates for the RBF2 BDMs and various output stages. The left column gives the K-L dimensionality. The integer modifier to the MLP nets indicates number of hidden units.

PNN Binary Decision Machines							
Output Stage	Number of features						
	8	12	16	20	24	28	32
VOTE	3.7	3.4	2.8	2.7	2.7	2.6	**2.6**
MLP:48	8.5	3.2	3.6	4.5	4.9	3.9	4.2
PNN	3.4	3.2	3.2	3.3	3.4	3.8	4.3
RBF1:60	4.1	3.4	3.3	3.3	3.3	3.1	3.2
RBF2:60		3.6	3.3	3.4	3.4	3.5	3.4

Table 8:
Error rates for PNN BDMs. The integer attached to the MLP and RBF labels is the number of hidden units.

in that context. When layers of networks are combined, the saturation of networks in upper layers can seriously degrade the performance of networks in the following layers if these networks depend on continuous input signals for their proper operation. The MLP and RBF network concentrate all their output near the corners of an n-dimensional cube. PNN networks are not well suited to determining relationships between the n-dimensional vertices. Both MLP and RBF nets, on the other hand, operate well in a mode where pattern in sequences of saturated inputs are learned.

4.3 Error-Rejection Rates

In addition to forced decision accuracy, The error reject characteristics of the various combination of Input Network and Output networks were examined. The two most successful of these were PNN Input Networks and MLP input networks. The RBF Input Networks produced only marginal improvements in forced decision accuracy and has less effective reject accuracy performance.

Figure 4 shows the percent error versus the percent reject for the 24-feature PNN BDM's. The percent error for the PNN BDM voting rule is based on the output signal size. A machine is rejected if its signal is less then a given threshold value. The pattern is rejected if all 45 BDMs are rejected or if there is a tie. Each of the graphs show that the voting rule starts with a lower error rate, but the RBF1 and RBF2 have a much better rejection rate over the voting rule.

Figure 5 shows the log percent error versus the percent reject of the 24-feature MLP1 BDMs. The

MLP1 Binary Decision Machines							
Output	Number of features						
Stage	8	12	16	20	24	28	32
VOTE	3.9	3.2	3.0	2.9	2.9	**2.8**	3.2
MLP:32	4.3	3.7	3.5	3.3	3.4	3.4	3.7
PNN	14.1	11.5	9.9	13.1	13.5	13.4	13.1
RBF1:30	4.6	3.8	3.5	3.3	3.4	3.3	3.7
RBF2:30	4.5	3.8	3.5	3.3	3.4	3.3	3.7

Table 9:
Error rates for MLP1 BDMs. The integer attached to the MLP and RBF labels is the number of hidden units.

MLP2 Binary Decision Machines							
Output	Number of features						
Stage	8	12	16	20	24	28	32
VOTE	3.9	3.2	3.2	**2.6**	2.9	2.9	2.9
MLP:48	4.3	3.6	3.5	3.1	3.3	3.3	3.3
PNN	6.1	5.2	5.1	4.9	5.1	5.0	11.9
RBF1:30	4.5	3.8	3.6	3.1	3.3	4.3	3.3
RBF2:30	4.5	3.6	3.6	3.1	3.3	3.3	3.4

Table 10:
Error rates for MLP2 BDMs. The integer attached to the MLP and RBF labels is the number of hidden units.

RBF Type 1 Binary Decision Machines							
Output	Number of features						
Stage	8	12	16	20	24	28	32
VOTE	6.8	6.3	6.1	5.9	5.8	5.8	6.4
MLP:48	13.4	**3.7**	3.8	3.7	3.9	3.9	3.9
PNN	28.7	9.0	8.7	8.8	8.6	8.7	9.6
RBF1:50	14.2	4.5	4.5	4.5	4.5	4.3	
RBF2:50	13.8	4.0	4.1	4.2	4.0	4.0	

Table 11:
Error rates for RBF1 BDMs. The integer attached to the MLP and RBF labels is the number of hidden units.

RBF Type 2 Binary Decision Machines							
Output	Number of features						
Stage	8	12	16	20	24	28	32
VOTE	6.4	5.2	5.0	5.1	4.7	4.6	3.4
MLP	4.5	4.6	3.8	4.0	**3.0**	3.9	4.4
PNN	9.5	7.3	7.0	7.0	7.2	7.1	6.2
RBF1:3	5.2	4.7	4.6	4.5	4.7	4.6	7.2
RBF2:3	4.7	4.5	4.2	4.3	4.2	4.3	7.8

Table 12:
Error rates for RBF2 BDMs. The integer attached to the MLP and RBF labels is the number of hidden units.

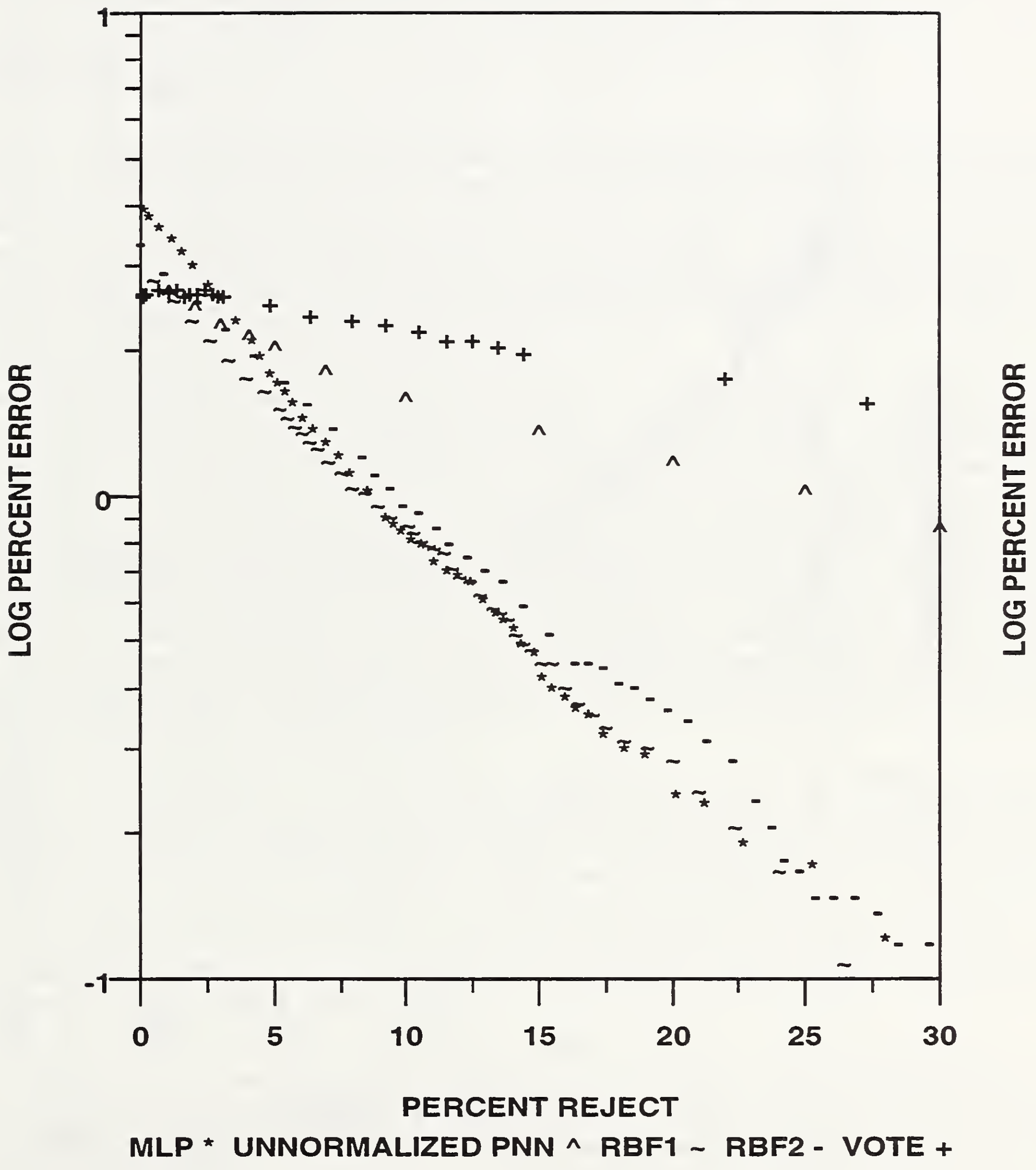

Figure 4: 24 Feature PNN BDM Error vs Reject for different output networks.

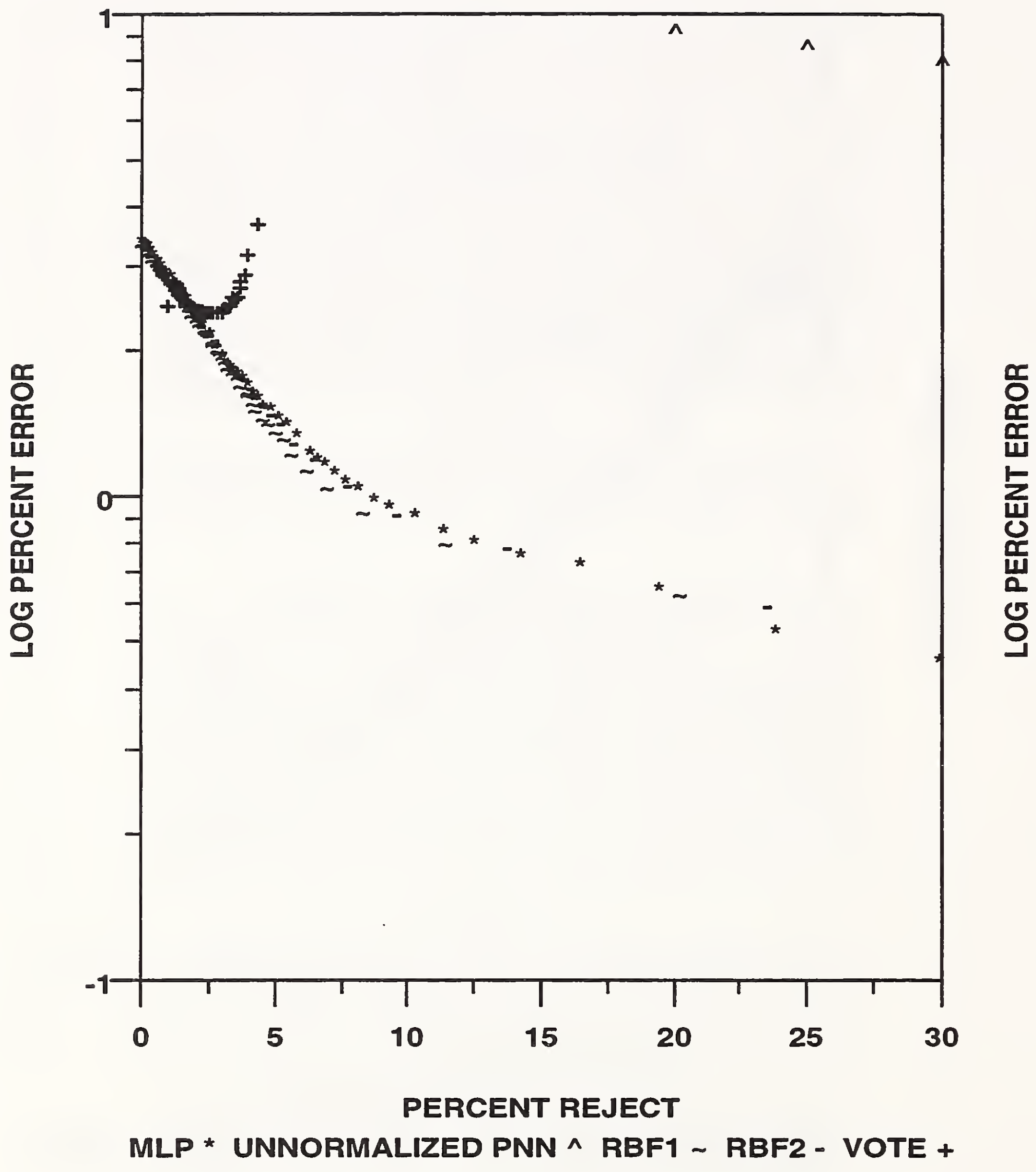

Figure 5: 24 Feature MLP1 BDMs Error vs Reject for different output networks.

percent error for the MLP1 BDMs voting rule is based on the output signal's magnitude. A machine is rejected if its signal is less then a given threshold value. The pattern is rejected if all 45 BDMs have been rejected based on the threshold or if the resulting vote is a tie. The curve for the MLP1 voting network is the result of both the number of machines rejected per pattern and the number of tie votes which result.

Although the PNN and voting rule based systems give good forced decision accuracy, they provide poor reject accuracy performance. The best combination of input and output networks is one which uses PNN BDMs and an RBF1 output. This provides 3.1% forced decision accuracy and 1% accuracy at about 8% rejection. For rejection rates between 1% and 7% an all MLP based system will provide better reject accuracy since at 7% rejection rate 1% accuracy is achieved.

5 Conclusions

The structure consists of three layers of processing, the K-L feature processing, the input network layer and the output network layer. As we demonstrated in table 1 the K-L layer can provide accuracy approaching [1] on the binary digit problem. We also demonstated in table 2 that a simple voting mechanism can get good results using the outputs of the second layer. Unfortunately, neither of these mechanisms provides satisfactory estimates of the confidence of its result so that the output layer is required to provide good reject accuracy performance. This high accuracy on forced decision coupled with poor reject performance, was also observed in [15].

It was also possible to improve the recognition performance, over that obtained with the same test and training sets in [1], of both the MLP and RBF methods by the use of BDMs. The performance of the local methods, PNN, is not improved by this process. The improvement in MLP performance is greater than the improvement in RBF performance. This clearly indicated that the methods improved in a way which is proportional to the amount that they are converted form global to local methods. RBF, as used here, is preclustered and is therefore partly local and partly global. RBF is improved but not as much as the MLP networks are. MLPs are usually global but by converting them to local methods, even when it results in a smaller training set as it does in this case, has made them perform as well as local methods. This is an issue of intrinsic dimensionality of the type discussed in [14]. Local dimensionality has long been recognized as a critial factor in neighbor-based methods and we now conclude that it is equally important for neural networks methods.

This intrinsic variability of rank of the feature set is further seen in the distribution of character-by-character errors seen with feature set size and output network type and in the large difference in errors for the BDMs associated with different digits. In tables 3-7 typical errors for the "6" class are 0.5% and typical errors for the "8" class are 7.0% so that in this sense the classification of "8" is 14 times harder than the classification of "6". This variability of classification accuracy by character type has also been seen using an image-based method [34, 9] where the number of memories needed to recognize digits is highly class dependent. This is a clear indication that the features used to select a "6" are much more efficient than the features used to select an "8".

Another indication of the local rank effects in the problem structure for OCR is provided by the clear division of Tables 3-7 into two groups distinguished by the amount of change in feature set size for maximum accuracy for each digit. Tables 3, 6, and 7 for PNN, RBF1, and RBF2 networks show large changes in optimum feature set size and output network type for different digits. Tables 4 and 5 for MLP1 and MLP2 networks show a global optimum feature set size of 20 for most digit types. This is less than half the optimum feature set size for the global solution to this same problem given in [1]. The change in optimum feature set size and the sharp decrease in feature set size for BDM networks demonstrates that the global solution is rank deficient and that clustering of the global problem into local problems reduces the rank of the optimum feature set and therefore makes global optimization, used for the MLP networks, more effective.

The need for global rank reduction is further supported by Boltzmann pruning studied of large MLP networks. In [35, 36] it was shown that in these networks up to 80% of the weights can be removed without affecting the network performance and that the remaining weights typically have 9-11 bits of information content. As the weights are removed from the network the removal of low variance weights connected to K-L features associated with smaller eigenvalues is strongly favored. Weight pruning provides strong evidence of rank deficiency in the global problem and the relatively small number of digits present in even the pruned weights demonstrates that calculations using these weight values may experience significant problems.

The networks that are trained using optimization methods [23, 22] are, during training, dynamic systems subject to the same convergence and stability problems which are present in other dynamic systems [37, 38, 39]. All of these methods of training nonlinear networks are directly or indirectly dependent on the stability of the Jacobian matrix of the local linearization of the system. The equivalent linear problem is dependent on the rank of the covariance matrix as is the computation of the K-L transform. If the number of bits used in the variables which form this matrix is lower than is practical for the condition number of the matrix used in the training process the process will be both dynamically and numerically unstable. This does not mean that all solutions found are useless but only that some of these solutions are selected at random based on rounding error not on input data. In problems which are seriously rank deficient the training process is a Boltzmann machine where numerical rounding error and input signal noise serve as the random number generator. Clustering, using BDMs or other methods, should be used to provide an effective method for rank reduction and improve system stability.

In [1] and [15] many different OCR systems were presented which achieved 5% error rates and several were presented which have 2%-3% error rates. By reducing the problem to a series of BDMs we have shown that several different neural network methods can achieve 3% error rates on the relatively small training set used in [1]. These systems also can exhibit error reject behavior comparable to the best presented in [15]. Analysis of the digit-by-digit performance with respect to feature set size and network type shows that much of this improvement is associated with local rank reduction in the feature set.

This changes the way the problem of character recognition is stated. We now know that the binary decision process for two characters using learning data that consists of other characters from the binary set is relatively easy. The "3-8" data using K-Ls and KNN can in general get errors of 7.6%. So "3-8" is three times harder than "0-1" and "8"s are 5 times harder than "1"s. The increasing difficulty is caused not by the primary seraration of digits but by the ability of the BDMs to reject characters of other classes. This is particulary important when an OCR system is constructed since the most sucessful methods of segmentation depend on deliberate over segmentation and reconstruction [40]. The over segmentation process generates numerious partial and merged sections of digits which must be rejected for OCR to suceed.

Another way of considering the rejection problem is to consider the number of images that are near any 32 by 32 binary image in image space. If character images should be recognized after reversing m bits of an n bit image then each character has $n! - (n-m)!$ neighbors which must be recognized and $(n-m)!$ derived images that should be rejected. Both these numbers are very large compared to any projected OCR test or training set and indicate the redundancy of even simple character images. The more redundant the image set, the more difficult the classification process is, since many small variations in the image yield no useful classification data. Larger and larger training set are more rather than less redundant since they will contain more examples of common character types.

From these arguments we would expect neither larger feature sets nor larger training sets to eliminate the remaining sources of OCR error. Larger feature sets can only be effective if they increase the rank of the feature set. Larger training sets can only be effective if they provide new protypes which are not redundant.

Acknowledgement

The authors would like to acknowledge Jack Rabinow for suggesting the BDM approach and Craig Watson and Jerry Candela for helping with various aspects of the calculations.

References

[1] J. L. Blue, G. T. Candela, P. J. Grother, R. Chellappa, and C. L. Wilson. Evaluation of Pattern Classifiers for Fingerprint and OCR Applications. *Pattern Recognition*, 27(4):485–501, 1994.

[2] G.A. Carpenter and S. Grossberg. A massively parallel architecture for a self organizing neural pattern recognition machine. *Neural Networks*, 2:169–181, 1989.

[3] G. A. Carpenter and S. Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, 37:54–115, 1987.

[4] G. A. Carpenter and S. Grossberg. Art 2: self-organization of stable category recognition codes for analog input patterns. *Applied Optics*, 26:4919–4930, 1987.

[5] J. A. Hartigan. *Clustering Algorithms*, pages 84–108. New York: John Wiley & Sons, Inc., 1975.

[6] D. L. Alkon, K. T. Blackwell, G. S. Barbour, A. K. Rigler, and T. P. Vogl. Pattern-recognition by an artificial network derived from biological neuronal systems. *Biological Cybernetics*, 62:363–376, 1990.

[7] Richard Granger, Jose Ambros-Ingerson, and Gary Lynch. Derivations of encoding characteristics of layer II cerebral cortex. *Journal of Cognitive Neuroscience*, 1(1):67–87, 1989.

[8] Jose Ambros-Ingerson, Richard Granger, and Gary Lynch. Simulation of paleocortex performs hierarchical clustering. *Science*, 247:1344–1348, 1990.

[9] C. L. Wilson. FAUST: a vision based neural network multi-map pattern recognition architecture. In *Proceedings: Applications of Artificial Neural Networks III.* Orlando, SPIE, April 1992.

[10] Y. Le Cun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2, pages 598–605. Morgan Kaufman, 1990.

[11] G. L. Martin. Centered-object integrated segmentation and recognition for visual character recognition. In J. Moody, S. Hanson, and R. Lippmann, editors, *Advances in Neural Information Processing Systems*, volume 4, pages 504–511. Morgan Kaufmann, Denver, December 1991.

[12] G. Martin and J. Pittman. Recognizing hand-printed letters and digits. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2, pages 405–414. Morgan Kaufmann, 1990.

[13] G. Martin and J. Pittman. Recognizing handprinted letters and digits using backpropagation. *Neural Computation*, 3:258–267, 1991.

[14] K. Fukunaga. *Introduction to Statistical Pattern Recognition.* New York: Academic Press, second edition, 1990.

[15] R. A. Wilkinson, J. Geist, S. Janet, P. J. Grother, C. J. C. Burges, R. Creecy, B. Hammond, J. J. Hull, N. J. Larsen, T. P. Vogl, and C. L. Wilson. The First Optical Character Recognition Systems Conference. Technical Report NISTIR 4912, National Institute of Standards and Technology, August 1992.

[16] Michael D. Garris, James L. Blue, Gerald T. Candela, Darrin L. Dimmick, Jon Geist, Patrick J. Grother, Stanley A. Janet, and Charles L. Wilson. NIST Form-Based Handprint Recognition System. Technical Report NISTIR 5469, National Institute of Standards and Technology, July 1994.

[17] P. J. Grother. Karhunen Loève feature extraction for neural handwritten character recognition. In *Proceedings: Applications of Artificial Neural Networks III.* Orlando, SPIE, April 1992.

[18] T. P. Vogl, K. L. Blackwell, S. D. Hyman, G. S. Barbour, and D. L. Alkon. Classification of Japanese Kanji using principal component analysis as a preprocessor to an artificial neural network. In *International Joint Conference on Neural Networks*, volume 1, pages 233–238. IEEE and International Neural Network Society, 7 1991.

[19] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 332:533–536, 1986.

[20] R. Fletcher and C. M. Reeves. Function minimization by conjugate gradients. *Computer Journal*, 7:149–154, 1964.

[21] E. M. Johansson, F. U. Dowla, and D. M. Goodman. Backpropagation learning for multi-layer feed-forward neural networks using the conjugate gradient method. *IEEE Transactions on Neural Networks*, 1991.

[22] M. F. Møller. A scaled conjugate gradient algorithm for fast supervised learning. Technical Report PB-339, Aarhus University, 1990.

[23] J. L. Blue and P. J. Grother. Training Feed Forward Networks Using Conjugate Gradients. In *Conference on Character Recognition and Digitizer Technologies*, volume 1661, pages 179–190, San Jose California, February 1992. SPIE.

[24] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990.

[25] M. T. Musavi, W. Ahmed, K. H. Chan, K. B. Faris, and K. M. Hummels. On the training of radial basis function classifiers. *Neural Networks*, 5:595–603, 1992.

[26] D. Wettschereck and T. Dietterich. Improving the performance of radial basis function networks by learning center locations. In J. E. Moody, S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 4*, pages 1133–1140, San Mateo, 1991. Morgan Kaufmann.

[27] Donald F. Specht. Probabilistic neural networks. *Neural Networks*, 3(1):109–118, 1990.

[28] M. D. Garris and R. A. Wilkinson. Handwritten segmented characters database. Technical Report Special Database 3, **HWSC**, National Institute of Standards and Technology, February 1992.

[29] C. L. Wilson and M. D. Garris. Handprinted character database. Technical Report Special Database 1, **HWDB**, National Institute of Standards and Technology, April 1990.

[30] R. G. Casey and H. Takahashi. Experience in Segmenting and Recognizing the NIST Database. In *Proceedings of the International Workshop on Frontiers of Handwriting Recognition*, France, 1991.

[31] Patrick J. Grother. Cross Validation Comparison of NIST OCR Databases. In D. P. D'Amato, editor, , volume 1906. SPIE, San Jose, 1993.

[32] C. L. Wilson and J. L. Blue. Neural network methods applied to character recognition. *Social Science Computer Review*, 10:173–195, 1992.

[33] M. D. Garris and C. L. Wilson. Reject Mechanisms for Massively Parallel Neural Network Character Recognition Systems. In Su-Shing Chen, editor, *Neural and Stochastic Methods in Image and Signal Processing*, volume 1766. SPIE, San Diego, 1992.

[34] C. L. Wilson. A New Self-Organizing Neural Network Architecture for Parallel Multi-Map Pattern Recognition - FAUST. *Progress in Neural Networks*, 4, 1993. to be published.

[35] O. M. Omidvar and C. L. Wilson. Information Content in Neural Net Optimization. *Journal of Connection Science*, 6:91–103, 1993.

[36] O. M. Omidvar and C. L. Wilson. Optimization of Neural Network Topology and Information Content Using Boltzmann Methods. In *Proceedings of the IJCNN, volume IV*, pages 594–599, June 1992.

[37] Morris W. Hirsch and Stephen Smale. *Differential Equations, Dynamical Systems, and Linear Algebra.* Academic Press, New York, NY, 1974.

[38] John Guckenheimer and Philip Holmes. *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields.* Springer-Verlag, New York, NY, 1983.

[39] Morris W. Hirsch. Convergent activation dynamics in continuous time networks. *Neural Networks*, 2:331–349, 1989.

[40] J. Geist, R. A. Wilkinson, S. Janet, P. J. Grother, B. Hammond, N. W. Larsen, R. M. Klear, M. J. Matsko, C. J. C. Burges, R. Creecy, J. J. Hull, T. P. Vogl, and C. L. Wilson. The Second Census Optical Character Recognition Systems Conference. Technical Report NISTIR 5452, National Institute of Standards and Technology, May 1994.