NIST Technical Note 2218

DeepFit: automated distribution fitting for building stochastic models

Siham Khoussi
Alan Heckert
Abdella Battou
Saddek Bensalem

This publication is available free of charge from:
https://doi.org/10.6028/NIST.TN.2218

National Institute of
Standards and Technology
U.S. Department of Commerce

NIST Technical Note 2218

DeepFit: automated distribution fitting for building stochastic models

Siham Khoussi
Communications Technology Laboratory

Alan Heckert
Statistical Engineering Division
Information Technology Laboratory

Abdella Battou
Communications Technology Laboratory

Saddek Bensalem
University of Grenoble Alpes (UGA)
Grenoble, France

This publication is available free of charge from:
https://doi.org/10.6028/NIST.TN.2218

April 2022

U.S. Department of Commerce
Gina M. Raimondo, Secretary

National Institute of Standards and Technology
Laurie E. Locascio, NIST Director and Undersecretary of Commerce for Standards and Technology

National Institute of Standards and Technology Technical Note 2218
Natl. Inst. Stand. Technol. Tech. Note 2218, 20 pages (April 2022)
CODEN: NTNOEF

This publication is available free of charge from:
https://doi.org/10.6028/NIST.TN.2218

Abstract

Statistical model checking (SMC) is a formal verification method that combines simulations with statistical techniques to provide quantitative answers on whether a stochastic system satisfies some requirements with a controllable accuracy. SMC takes three inputs: a stochastic model, a linear-time/Metric Temporal Logic property to verify and a set of required confidence parameters. The stochastic model is generally obtained by modeling the functional behavior of a system then adding probabilistic variables to it, which are updated via probability distributions (PD). The latter is, typically, obtained by analyzing measurements from the system's execution using statistical tests to select the best fit distribution. However, this task requires a good statistical background and familiarity with several distributions, which is beyond the expertise of some analysts. Hence, in the case of SMC, assuming an incorrect distributional model for the data can lead to inappropriate statistical analysis as well as inaccurate verification of the system under study. As such, this paper presents DeepFit, a tool that uses deep learning in addition to traditional statistics to automate the distributional modeling process. DeepFit was evaluated against synthetic data and real-world data, and it can perform comparably to using maximum likelihood estimation with an Anderson-Darling, Kolmogorov-smirnov and Probability plot correlation coefficient plot goodness of fit tests.

Key words

Deep Learning; neural networks; distribution fitting; data normalization.

Table of Contents

List of Tables

List of Figures

1. Introduction

Statistical model-checking (SMC) [1–3] is a formal verification method that combines simulations with statistical reasoning to provide answers on whether a stochastic system, under certain assumptions, satisfies some requirements with a fixed confidence. It emerged as a solution to the state space explosion problem associated with using the classical model checking on probabilistic models. SMC consists of only exploring a sub-part of the state space and mainly relying on statistics and simulations to generalize, under certain assumptions, the partial results that are obtained by simulating the system a number of times to the whole system with a fixed confidence and a controllable accuracy. Given a stochastic model and a formal property, SMC answers two questions:

1. **Qualitative:** Can the model satisfy the specified requirement with a probability p such that $p \geq \theta$, respectively $p \leq \theta$, where θ is a certain threshold?

2. **Quantitative:** What is the probability that the model satisfies the requirement specification?

SMC takes three inputs: a stochastic model, a Temporal Logic property to verify and a set of required confidence parameters. In order to build the stochastic model, measurements from the system under study are analyzed and characterized in the form of probability distributions. Mapping a set of empirical observations into the corresponding probability distribution is referred to as the distribution fitting process. It is important to emphasize that distribution fitting is a tedious statistical task that is typically done by expert analysts as it requires a good statistical knowledge and familiarity with many commonly used distributions, mostly because the statistical methods used involve human interpretation of graphs or numerical tests which is above the expertise of some analysts. In the context of SMC, it is really important to fit the system data to accurate probability distributions in order to build rich stochastic performance models and to be able to quantitatively verify them. Any misconception on the data or inaccurate fitted probability distributions will result in wrong verification results of the system.

Distribution fitting is usually conducted by following a four-step method as in Figure 1. The first one consists of pre-screening the data for randomness and autocorrelation. Generally, this step is assessed graphically using plots such as the auto-correlation plot and the the lag plot [4] or numerically using tests such as the Ljung-Box test. The second step consists of graphically summarizing the underlying distribution from a histogram or a kernel density plot (kdp). These two plots are used to determine the basic shape of the distribution that fits the data by searching for properties such as the skewness, the presence of multiple modes in the data, symmetry, presence of upper/lower tails, etc. Note that, a good statistical background and experience working with many distributions are essential in these first two steps, mostly because the statistical methods used involve human interpretation of graphs or numerical tests which is subjective and changes depending on the statistical experience of the analyst. Although a poorly chosen distributional model

may suffice for measuring and assessing the uncertainty of averages, this will not be the case for data that show signs of asymmetry and/or extended tails.

Once a candidate distribution model has been identified from the second step, the parameters for the selected distribution (e.g., location, scale and shape) must be estimated in the third step. There are many methods, both numerical and graphical, for estimating the parameters of a probability distribution. This includes the maximum likelihood method [5], the method of moments, the Least Squares method, etc. However, some estimators have better statistical properties than others. This step requires a deeper knowledge of these statistical estimators in order to determine the best method for the selected probability distribution. Finally, the last step consists of assessing the accuracy of the best fit model. The latter relies on applying one or more goodness of fit tests to produce a verdict on the appropriateness of the fit. There are many goodness of fit tests such as the Anderson-Darling test, the Kolmogorov-Smirnov test, the Cramer-Von Mises test or information criteria such as AIC and BIC, etc.

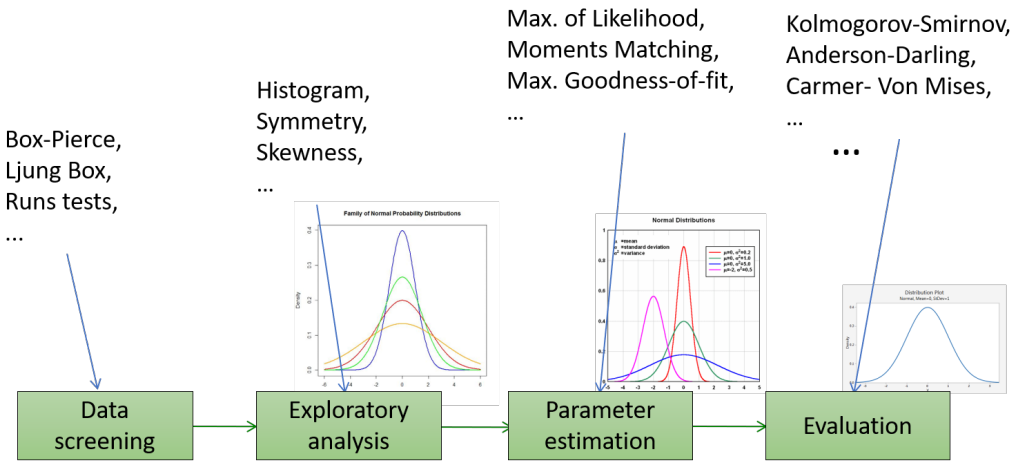

Fig. 1. Traditional approach of conducting distribution fitting

There is a variety of software designed to automate the distribution fitting process. This includes: Matlab, DataPlot [6], FitDistPlus [7], BestFit, ExpertFit, EasyFit [8], StatFit and R. Most of these tools rely on the traditional workflow presented in Figure 1 to identify a good representation of the data in the form of probability distributions (Figure 1). However, most of them are still not fully automated and are still in need of the analyst's own interpretation of the numerical and graphical outputs to properly deduce the suitable distribution, which is in fact the most sensitive and essential step in distributional modeling. In this context of SMC, choosing an inaccurate fit leads to wrong decision-making and faulty judgments on the performance of the system being evaluated.

As such, we investigate the use of deep learning as an alternative to the traditional

methodology of conducting distributional modeling in order to save the analysts' time and avoid tools and methods that leave room for their interpretation and uneducated guesses. In this paper, we present DeepFit, a tool we developed which combines deep learning in addition to the traditional statistical approach of conducting distribution fitting. First, a neural networks (NN) model that we previously trained and tested on synthetic and real-world data [9] is used to guess the best candidate distribution for the collected measurement. Then, statistical techniques such as the maximum likelihood is used to estimate the parameters of the selected distribution. Additionally, to evaluate and validated the predictions made by the NN classifier, DeepFit applies several goodness of fit tests. Users can choose to opt out of the neural networks prediction and rely solely on the implemented goodness of fit tests to rank all the supported distributions from the best fit to the last.

This paper is organized as follows: in section 2, we overview the architecture of DeepFit and in section 3 we present a simple use case that demonstrates the capabilities of the tool. Then in section 4, we demonstrate how DeepFit can be integrated in the context of SMC and finally we conclude this paper with lessons learned and future work.

2. Architecture

DeepFit is a tool for automating the distribution fitting process for uncensored and unbinned uni-variate data. It relies on neural networks in order to identify the 'best' candidate model from a set of commonly used distributions.[1] Then, it applies traditional statistical techniques such as the maximum likelihood and many goodness of fit tests to estimate the parameters of the selected distribution and assess its appropriateness. DeepFit has five modules (Figure 1):

1. Data screening;
2. Neural networks classification;
3. Parameter estimation;
4. Evaluation;
5. Best Fit ranking.

We rely on the study we published in [9] for the first four and add support for module 5. Each module will be investigated in details in the next subsections.

[1] uniform, 2: normal, 3: logistic, 4: exponential, 5: double-exponential, 6: half-normal, 7: half-logistic, 8: gumbel-min, 9: gumbel-max

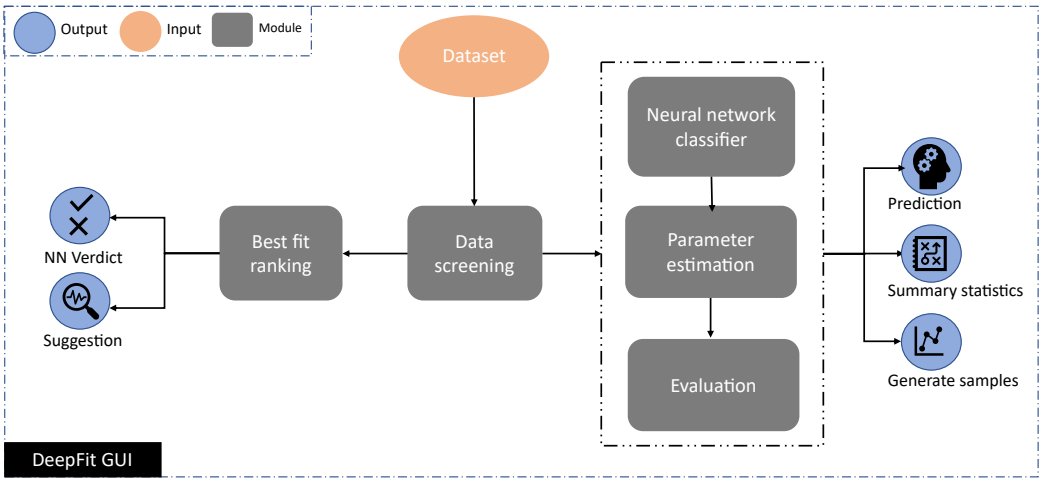

Fig. 2. DeepFit architecture

2.1 Data screening

The first step in using DeepFit is to pre-process and validate the input data for the neural network classifier.

A number of graphs are provided to help the analyst determine various characteristics of the data (e.g., is the data symmetric? if the data is skewed in which direction is the skewness? are there extreme observations?). These graphs can also help identify outliers. In this context, the purpose of identifying outliers is simply to help the analyst determine whether an observation is erroneous (e.g., is it mis-coded?) and not to perform formal outlier analysis. Formal outlier analysis is based on assuming that the underlying distribution is known (most outlier tests are based on assuming the data is normally distributed). However, DeepFit assumes that the underlying distribution is unknown and in fact the purpose of the tool is to determine an appropriate distributional model. An ”extreme” point may indicate a bad data value or it may be a reflection of the underlying distribution of the data, so it is recommended that an observation be removed only if it can reasonably be determined to be erroneous. It should be noted that ”noisy” data is an indication that the observations do not come from a single common distribution. This type of data may be more appropriately modeled with a mixture distribution which is beyond the current implementation of DeepFit at the time of this writing. However, further use cases and more distributions will be supported soon. More specifically, DeepFit generates four plots [10] that are designed to check whether the data are independent and draws from a common distribution with fixed location and fixed scale. Datasets that do not satisfy these assumptions should not be modeled with a single distributional model. The analyst is engaged in understanding their data while locating and removing any mis-coded points that could be considered as outliers. The four recommended plots are:

1. The run sequence plot is performed on the unsorted data and is generally used as a test to identify whether the fixed location and fixed scale assumptions are reasonable.

Moreover, it can detect any obvious trends in the observations. Note that, the presence of a trend in this plot indicates a lack of independence in the data. Typically, this plot provides answers to these questions: Does the ”location” or ”scale” shift? Are there any trends in the data over time? and are there any ”outliers”?

2. The lag plot is used as a test for first-order autocorrelation. In fact, data that shows significant autocorrelation is not independent. This plot should basically look like a random blob. Patterns in the data are an indication that the data is not independent. Note that, this plot must be performed on the unsorted data.

3. The third plot is used to give the analyst an idea of the distribution of the data. Generally, a histogram or a kernel density plot is used in this context. As an example, if a bell-shaped plot is observed, then the analyst could assume that the underlying distribution is symmetric and perhaps approximately normal. The next modules can confirm or reject this assumption.

4. The normal probability plot: although this specifically checks for normality, in the context of screening the data it can be useful for identifying outlying points.

This module identifies some underlying assumptions (e.g., normality, shape, scale and location, existence of outliers) about the input set and the process from which the measurements were collected. DeepFit only advocates removing an observation from the set if it can be determined that the observation is in fact a bad data point as opposed to simply being an outlier relative to a normal distribution. Once the analyst is satisfied with the data, he/she can proceed to the next module which uses the neural networks classifiers for fitting.

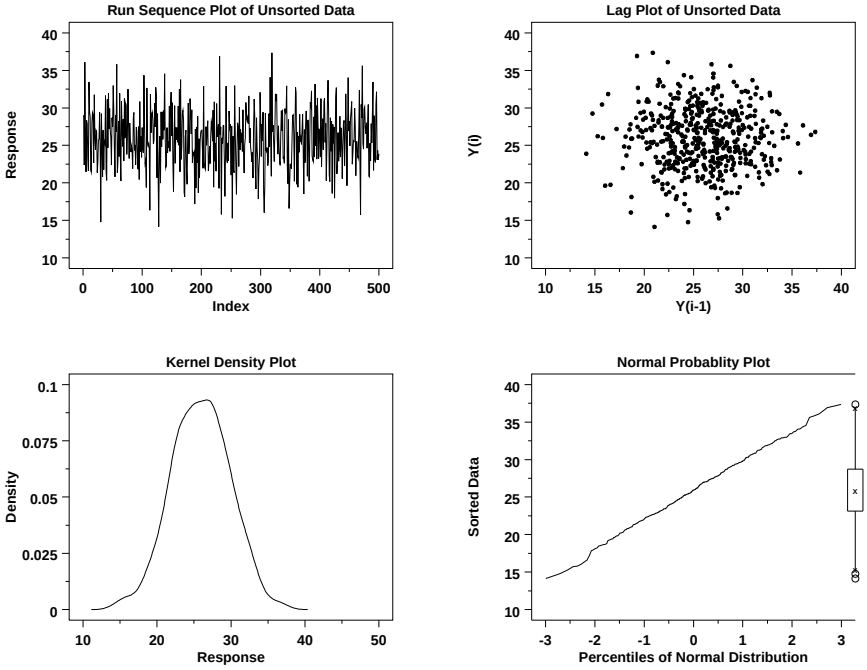

Fig. 3. 4-plot of 500 random normal.

Figure 3 shows an example of the 4-plot method applied to 500 normally distributed data points. Using these plots, we see no obvious patterns nor trends on the lag plot and the run sequence plot. This indicates that the data is independent and comes from a random process. Additionally, from the normal probability plot we learn that there are no visible outliers to remove. Moreover, from the kernel density plot (kdp), we can already interpret that the data is normally distributed since the kdp has a bell-like shape and is symmetric. But since we assume that the analysts are unfamiliar with the shape of many probability distributions, they can proceed with the next step and let our trained NN classifiers predict the 'correct' model for them (our published study [9] explains the details of the neural networks used in DeepFit).

We also show another example of data sampled from the gumbel-max distribution in Figure 4. Similar conclusions from Figure 3 about the data independence still apply. We can also confidently confirm the absence of any extreme points from the normal probability plot. However, we can't clearly identify the shape of the distribution from the kernel density plot except that it's an upper tailed distribution. Therefore, the analyst is advised to use our neural networks classifier to predict the 'best' model for their data.

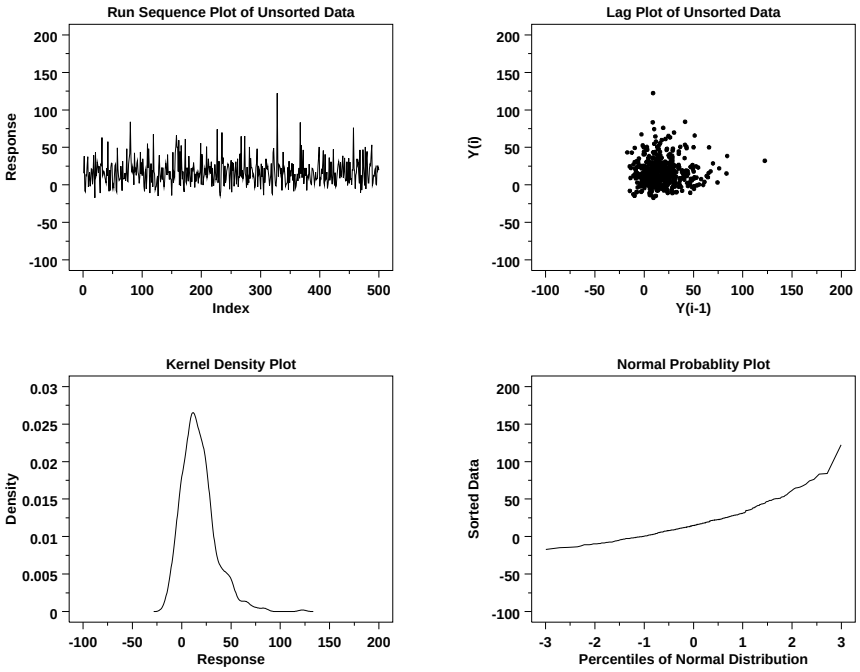

Fig. 4. 4-plot of 500 random gumbel-max points.

2.2 Neural networks classification

In this module, an initial transformation is applied to the input dataset then it's fed to the neural network classifier which was trained on a large database consisting of commonly used distributions and continuous measurements where the data are not binned, censored or truncated. This classifier takes as input a kernel density plot of the data to fit and makes a prediction on the best fit distribution model for it. We refer the reader to our study [9] for further details on the trained model. We experimented with several transformation algorithms but, only two yielded promising results, that is the kernel density normalization and the u-score normalization:

1. The u-score normalization, also referred to as the Min-Max scaler, transforms the observations to a (0,1) scale according to the following mathematical formulation:

$$u\_score = \frac{x - min(x)}{max(x)) - min(x)} \tag{1}$$

   x is the original observation value, u_score is the normalized value, $min(x)$ and $max(x)$ are respectively the minimum and maximum values observed in this data.

2. The kernel density normalization transforms the kernel density heights to integrate to 1 on the 1 to 256 x-coordinate scale

$$k\_score = \frac{x}{\sum_{i=1}^{256} x_i} \tag{2}$$

where x is the original value and k_score is the normalized value.

In our study [9], we found that these two techniques are non-distorting of the shape of the kernel density plot and preserve the form of the probability distribution regardless of the location and scale values

2.3 Parameter estimation

Once the neural network classifier has identified the "best" distributional model, the parameters of the distribution are estimated in this module via the maximum likelihood method [5]. This module estimates three different output sections:

1. Some basic summary statistics for the observations (e.g. minimum, maximum, range, skewness, kurtosis, etc.).

2. The parameter estimates (location and scale). Note that, we are continuously adding more distributions to the tool in order to support the ones with the shape parameters.

3. Confidence intervals for the estimated parameters.

2.4 Evaluation

This module includes traditional statistical goodness of fit techniques to determine if the distribution model suggested by the neural networks is in fact appropriate for the data. Generally, there are three basic categories of the goodness of fit tests:

1. The first category is based on comparing the empirical cumulative distribution function (CDF) (i.e., based on the data) to the theoretical CDF function. This includes tests such as the Kolmogorov-Smirnov (KS) test, the Anderson-Darling (AD).

2. The second category relies on the percent point function (PPF). Tests in this category compare the differences between the empirical PPF to the theoretical PPF. This includes the probability plot correlation coefficient test (PPCC) [11].

3. The third category relies on the likelihood function. As the name "maximum likelihood" implies, this module searches for the distribution that provides the maximum value of the likelihood function. Note that, it is more common to use "information critieria" which is also based on the value of the likelihood function. Examples of this category include the Akaike's Information Criterion (AIC) and the Bayesian Information Criterion (BIC). The latter is more commonly used.

Currently DeepFit includes four goodness of fit statistics from the categories described above: Kolmogorov-Smirnov (KS), Anderson Darling (AD), PPCC and the BIC information criterion. This choice is justified by the fact that AD and KS tests are more powerful for the type of distributions supported by the neural network classifier at the moment[2]. We also suggest the analyst to follow up with the probability plot correlation coefficient test (PPCC) from the second category. For the list of commonly used distributions considered in this study, using BIC is equivalent to just using the likelihood value since all the distributions have the same number of parameters. However, it will be more useful as we continue supporting additional distributions with one or more shape parameters in the neural networks models.

2.5 Best Fit ranking

This module uses several goodness of fit tests to rank the supported distributions from the best match to the last match.

3. Tool assessment

DeepFit was evaluated on synthetic data [9] and real-world data [12] and successfully modeled several commonly used distributions. In this section, we use one example of real measurements obtained from a published study on Heat Flow Meter Calibration & Stability Analysis [12] to demonstrate the functionalities of DeepFit.

Figure 5 presents the 4-plots method implemented in the first module of DeepFit (i.e., data screening). The first two plots (i.e., the run sequence and the lag plots) show no obvious trends in the data which indicates that this dataset comes from a random process. Additionally, the kernel density plot looks symmetric and the normal probability plot is linear. This suggests that the normal probability distribution is probably a good fit for this data. In Figure 6, the pre-screened data is normalized by selecting one of the two methods: the u-score and the kernel normalization methods before it is passed through the neural networks classifier. The latter predicts the best candidate model for the data from the currently supported distributions. In this example, the normal distribution was selected which corresponds to the initial assumption made in the first module (Figure 5). Next is the parameter estimation module as shown in Figure 7. In this module, the parameters of the distribution that was previously selected by the neural networks classifier are estimated. That is, the location, the scale and the shape[3] parameters as well as their associated confidence intervals. For this example, the location and scale parameters are estimated for the normal probability distribution as in Figure 7. In this step, the analyst can also generate random samples from the selected distribution and store it locally into a proper format or plot both the original

[2]Uniform, normal, logistic, exponential, half-normal, half-logistic, double-exponential, gumbel-max and gumbel-min

[3]Currently not supporting families of distribution with one or more shape parameters

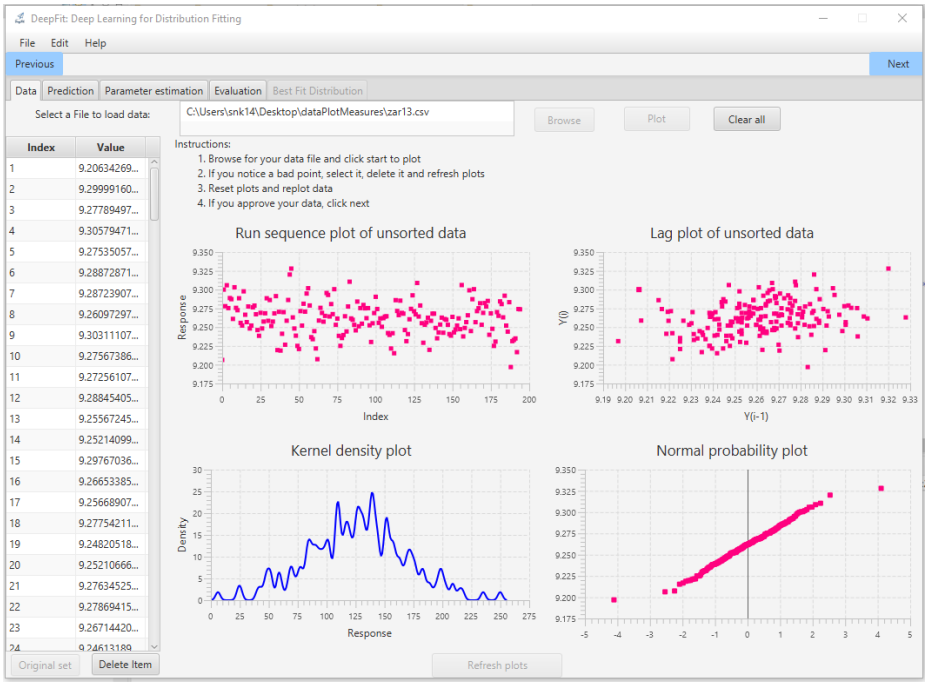

Fig. 5. Module 1: Data screening

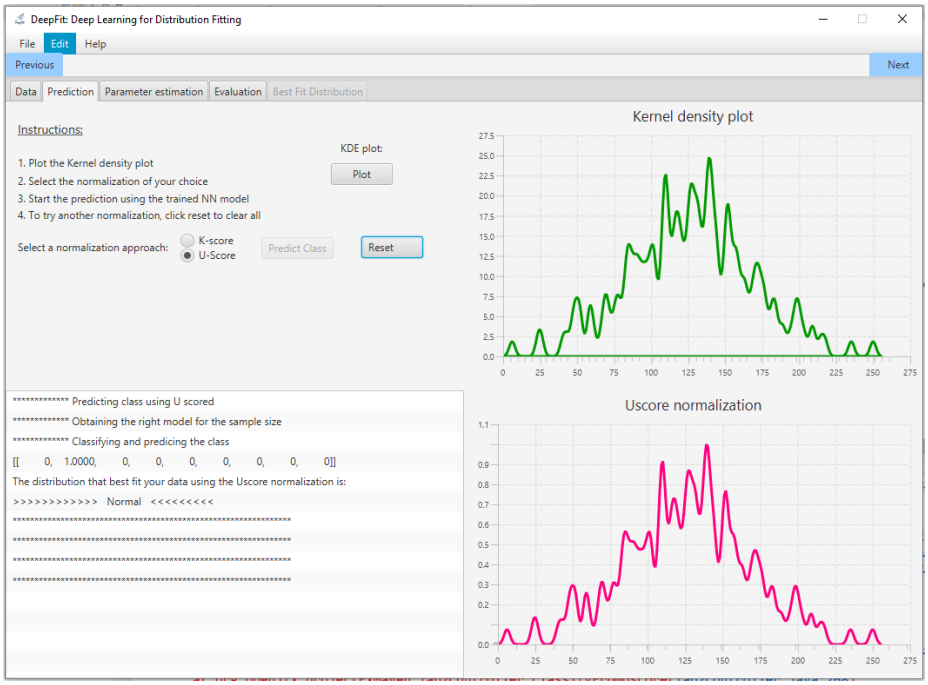

Fig. 6. Module 2: Neural networks classification

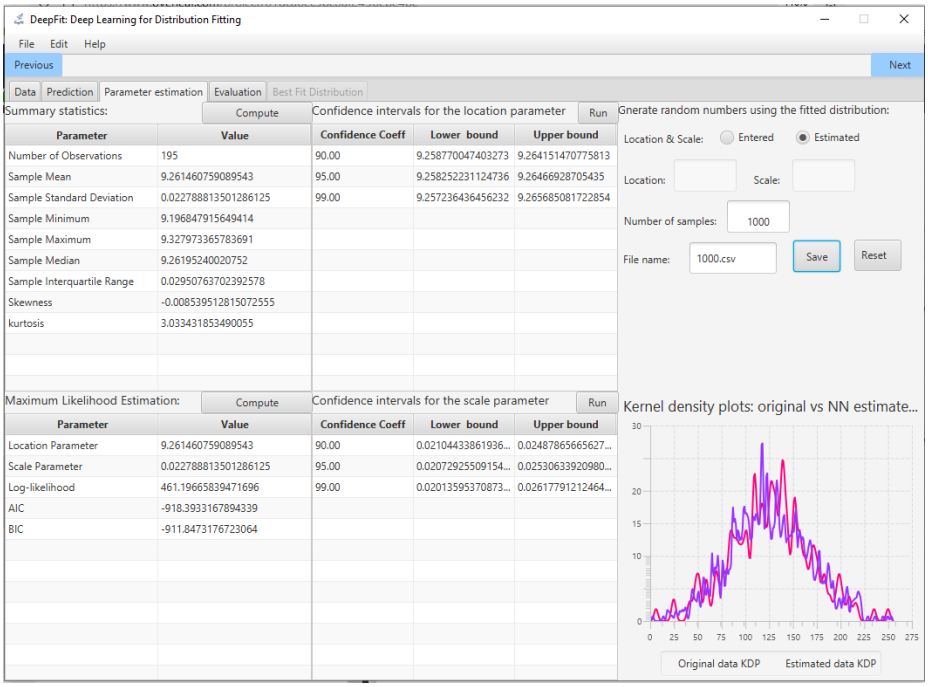

Fig. 7. Module 3: Parameter estimation

kdp of the input data and the kdp of N random samples generated from the NN estimation to see if they are similar. Figure 7 indicates that both plots look almost identical.

For further assessment, the analyst might choose to run several goodness of fit tests, supported in DeepFit, to confirm the neural networks classification or to test different distributions from the provided drop-down menu as in Figure 8. The currently supported goodness of fit tests include the Anderson Darling (AD), the Kolmogorov–Smirnov (KS) and the probability plot confidence coefficient (PPCC) tests. In certain cases, these tests could reach different conclusions because each of them is evaluating specific features of the data. As an example, the AD, KS and PPCC can be sensitive to different types of departures from the hypothesized distributions (e.g., AD is more sensitive to differences in the tails unlike the KS test, the PPCC test is a lower tailed test).

Finally, Figure 9 presents the fifth module of the tool which includes the option to bypass the neural networks and simply run a few goodness of fit tests to rank the supported distributions from the best match to the last. When applied to the Heat Flow Meter Calibration & Stability Analysis dataset, the normal distribution ranked first which matches the NN prediction made in module 2 (Figure 6).

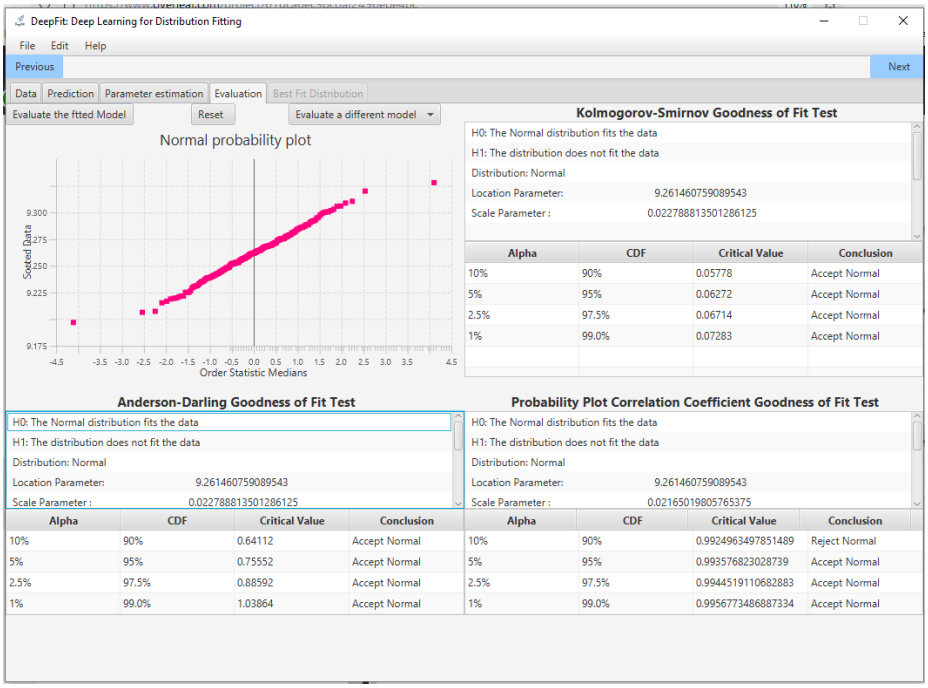

Fig. 8. Module 4: Evaluation

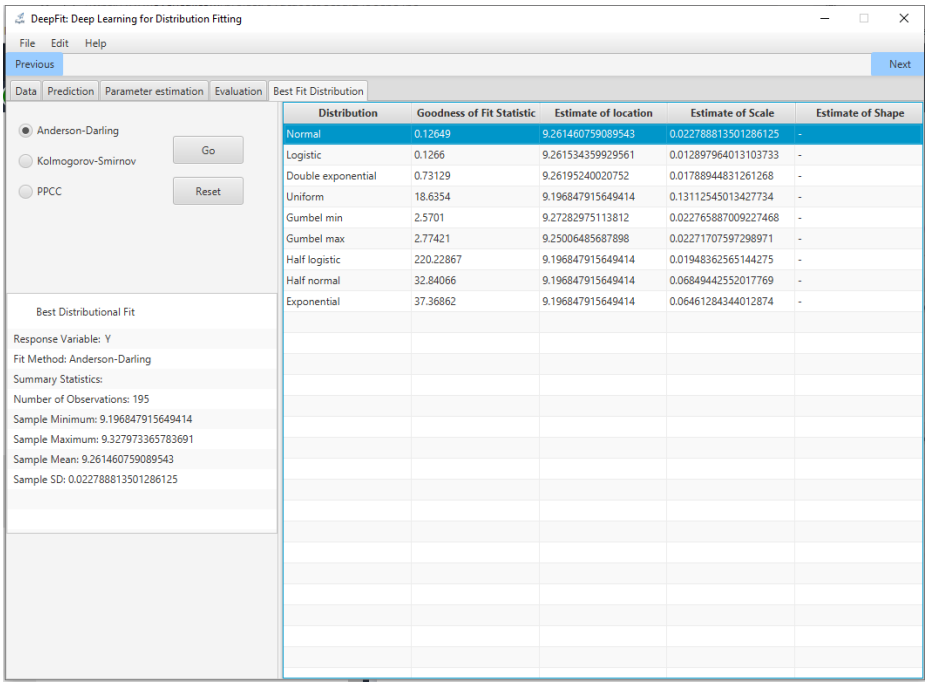

Fig. 9. Module 5: Best fit ranking

4. SMC context

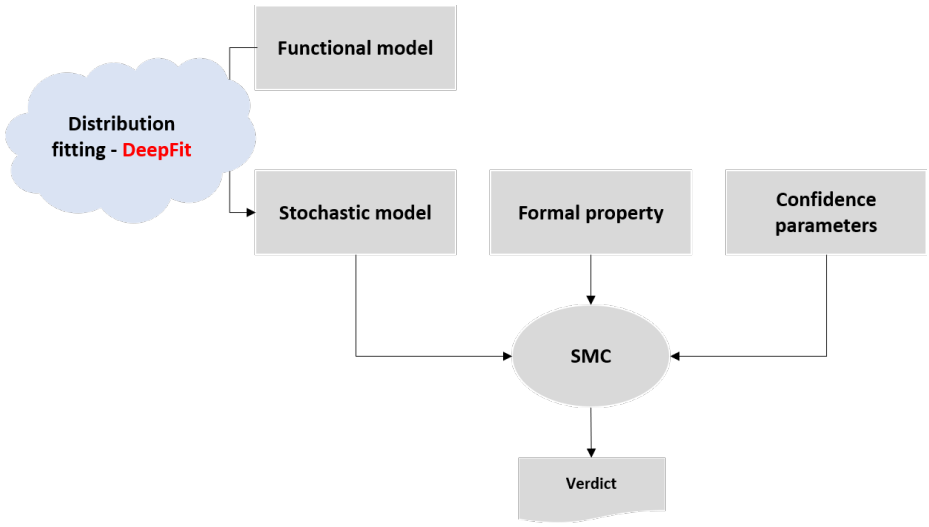

Fig. 10. DeepFit in the context of SMC

In this paper, we propose a tool for data analysis to be used in the context of SMC verification. SMC takes as input a stochastic model a property to verify and a set of confidence parameters to control the accuracy of the evaluation. The stochastic model is obtained by calibrating the functional behavior of a system with probabilistic variables, which are updated via probability distributions (PD). A PD is, typically, obtained by collecting and analyzing measurements from the system's execution using traditional statistical tests to select the best fit distribution (i.e., distribution fitting). Distribution fitting is crucial for the correct assessment via SMC and it's an important preliminary step in science and engineering, in general. However, this task requires a good statistical background and familiarity with several distributions which is beyond the expertise of some analysts. Therefore, we propose to use DeepFit, to automate the distribution fitting process as part of the workflow presented in Figure 10.

5. Lessons learned

In this paper, we propose DeepFit, a combined effort between neural networks and statistical techniques for data analysis and distributional fitting. DeepFit provides a preliminary step of data screening to remove bad data points (e.g., data is mis-coded or there is an assignable cause for why the observation is in error). Then uses a neural networks classifier that was previously trained on a large set of commonly used distributions in order to select the 'best' candidate model given a set of empirical observations. Moreover, the tool incorporates a variety of traditional statistics designed to compute the parameters of the selected distribution as well as assess it's goodness of fit. Additionally, DeepFit has

the advantage of being used as a standalone tool for fitting data or can be included in the workflow of SMC which was the initial motivation behind this work. We explained that one of the inputs to SMC is the stochastic model for the system to verify which is obtained by calibrating the functional model with probabilistic variables that are updated via probability distributions (PD). A PD is obtained by analyzing measurements from the system's execution using traditional statistical tests via a process called distribution fitting. These tests generally require a deeper understanding and familiarity with many probability distributions in order to interpret their numerical and graphical outputs. However, some analysts aren't equipped with such statistical background and are at risk of making faulty judgments or uneducated guesses of the underlying distribution from the data, hence leading to incorrect verification of the system via SMC. As such, we suggest to use our tool in this context to automate the distribution fitting process.[4]

6. Future work

Currently, DeepFit[5] supports a limited list of commonly used distributions in science and engineering to serve as a proof of concept of the viability of our approach. In the future, we plan to extend the number of supported distributions to also include families of distributions, such as the Weibull, Lognormal, Gamma distributions and other use cases as well as incorporate the ability to make more specific classifications (e.g., distinguish between Weibull or Lognormal) and compare this to approaches such as the likelihood ratio test [13], [14]. We also plan to explore a different type of neural networks such as the Long Short-Term Memory (LSTM) networks [15] [16]. These networks are a type of recurrent neural network, with the ability to learn order dependence in sequence prediction problems.

References

[1] Legay A, Lukina A, Traonouez LM, Yang J, Smolka SA, Grosu R (2019) Statistical Model Checking. *Computing and Software Science: State of the Art and Perspectives* (Springer, Cham, Switzerland), , pp 478–504. https://doi.org/10.1007/978-3-319-91908-9_23

[2] Hérault T, Lassaigne R, Magniette F, Peyronnet S (2004) Approximate Probabilistic Model Checking. *International Conference on Verification, Model Checking, and Abstract Interpretation, VMCAI'04*, , pp 73–84.

[3] Younes HLS (2005) *Verification and Planning for Stochastic Processes with Asynchronous Events*. Ph.D. thesis. Carnegie Mellon, .

[4] The identification of any commercial product or trade name does not imply endorsement or recommendation by the National Institute of Standards and Technology, nor is it intended to imply that the materials or equipment identified are necessarily the best available for the purpose.

[5] A link to the repository will be provided soon.

[4] Filliben JJ (2003) *Lag Plot* (National Institute of Standards and Technology), . Available at https://www.itl.nist.gov/div898/handbook/eda/section3/lagplot.htm.

[5] Filliben JJ (2003) *Maximum Likelihood* (National Institute of Standards and Technology), . Available at https://www.itl.nist.gov/div898/handbook/eda/section3/eda3652.htm.

[6] Dataplot homepage. Available at https://www.itl.nist.gov/div898/software/dataplot/homepage.htm.

[7] Delignette-Muller M, Dutang C (2015) fitdistrplus: An R Package for Fitting Distributions. *Journal of Statistical Software* 64(4):1–34.

[8] Schittkowski K (2002) EASY-FIT: a software system for data fitting in dynamical systems. *Structural and Multidisciplinary Optimization* 23:153–169.

[9] Khoussi S, Heckert A, battou A, Bensalem S (2021) Neural networks for classifying probability distributions. *NIST* https://doi.org/10.6028/NIST.TN.2152. Available at https://nvlpubs.nist.gov/nistpubs/TechnicalNotes/NIST.TN.2152.pdf

[10] Filliben JJ (2003) *4-Plot* (National Institute of Standards and Technology), . Available at https://www.itl.nist.gov/div898/handbook/eda/section3/4plot.htm.

[11] Filliben JJ (1975) The Probability Plot Correlation Coefficient Test for Normality .

[12] Khoussi S (2021) Some real data for testing. *GitHub repository* .

[13] Dumonceaux R, Antle CE, Haas G (1973) Likelihood Ratio Test for discrimination between two models with unknown scale and location parameters. *Technometrics* 15(1):19.

[14] Dumonceaux R, Antle CE (1973) Discrimination Between the Log-Normal and the Weibull Distributions. *Technometrics* 15(4):923–926.

[15] Sherstinsky A (2020) Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. *Physica D* 404:132306. https://doi.org/10.1016/j.physd.2019.132306

[16] Staudemeyer RC, Morris ER (2019) Understanding LSTM – a tutorial into Long Short-Term Memory Recurrent Neural Networks. *arXiv* 1909.09586 Available at https://arxiv.org/abs/1909.09586v1.