ICIC Express Letters
Volume **12**, Number **12**, December **2018**

ICIC International ©2018 ISSN 1881-803X
pp. **1213–1222**

SOFT COMPUTING APPROACH FOR PREDICTION OF SOFTWARE RELIABILITY

Kavita Sahu* and R. K. Srivastava

Department of Computer Science
Dr. Shakuntala Mishra National University
Lucknow, Uttar Pradesh 226017, India
*Corresponding author: kavi9839@gmail.com; rks100664@gmail.com

Received June 2018; accepted September 2018

Abstract. *The paper is based on Fuzzy Logic (FL) and Neural Network (NN) techniques to predict the software reliability using the MATLAB toolbox. There are four methods used in this paper to predict reliability of the dataset retrieved from John Musa of Bell Laboratories. These methods are fuzzy method, neural network, fuzzy-neural network and neural-fuzzy. After the assessment of data the results we achieved were best from the fuzzy-neural method among all proposed methods. In fuzzy-neural method the Levenberg-Marquardt algorithm is used for training the neurons. The performance of our proposed approaches has been tested using the testing data, and 15% of the data are from failure data set.*
Keywords: Software reliability, SRGM (Software Reliability Growth Model), Fuzzy-neural

1. **Introduction.** With the ever-increasing role of software in our real life systems, concerns have steadily grown over quality of software products. Therefore, reliability has become a primary concern from both software developers and software user's point of view. Software reliability is a subject of growing importance. Software reliability defined by ANSI is the probability of failure-free software operation for a specified period of time in a specified environment [1]. It is an important factor for quantitatively characterizing software quality and estimating the duration of software. Reliability is one of the most important non-functional requirements for software [20-22]. Accurately estimating reliability for service oriented system is not possible.

Software Reliability Growth Model (SRGM) aims to quantify software reliability status and behavior, help to develop reliable software and predict when reliability has grown enough to warrant product release [23]. Typically two broad categories of Software Reliability Growth Models (SRGMs) include parametric models and nonparametric models. Most parametric models depend on priori assumptions about the probability of individual failures occurring, development environments and the nature of software failures. They include non-homogeneous Poisson process model, Goel-Okumoto model, SHOOMAN model, etc. Nonparametric models predict reliability metrics only based on failure history without the assumptions of parametric models [24-26]. Software reliability has remained a thrust area of research over the past forty years, but still there are flaws in the modeling of software reliability. In the last two decades there has been lots of work done in the area of soft computing using the fuzzy logic, neural network, genetic algorithms and their hybrid approaches [15,16]. Both the statistical and soft computing techniques give noteworthy results in predicting reliability but it varies as the type of data changes. There are various prediction techniques for software reliability, but before using these techniques,

DOI: 10.24507/icicel.12.12.1213

one must thoroughly go through different techniques according to their research question [17-19].

Neural networks can offer noble approaches to software reliability prediction and modeling. Fuzzy logic based reliability estimation models are more appropriate when vague and imprecise information is to be accounted for. Such models usually rely on expert knowledge, which is however, often too general to fit a particular data set because different data sets have different characteristics. We present an innovative neuro-fuzzy reliability prediction modeling technique. Dataset provided by John Musa of Bell Laboratories is used for validation of the model. It was observed that the neuro-fuzzy reliability prediction model provided significantly better estimations than Mohanty et al. [7].

2. **Software Reliability.** In traditional reliability theory, both the system and its components are allowed to take only two possible states: either working or failed. In general software reliability is defined as 'how well the software meets its requirements' and also 'the probability of failure free operation for the specified period of time in a specified environment' [4-6]. The literature exposes many unpleasant happenings related to failure of software in the health and defense sectors [2-4,7,8] due to that many people lose their lives. One of the major causes behind all these misshaping is the presence of unreliable software. Software reliability is an important factor related to defects and faults. It differs from hardware reliability in that it reflects the design perfection, rather than manufacturing perfection. The principal factors that affect software reliability are (i) fault introduction, (ii) fault removal and (iii) the environment. The basic parameters to be considered during prediction of reliability are MTBF (Mean Time Between Failures), MTTR (Mean Time To Repair), FITS (Total no of Failures), Availability, Downtime, MTTF (Mean Time To Failure), Cumulative Errors.

3. Methods for Prediction.

3.1. Reliability prediction through fuzzy-neural approach.

Neural networks and fuzzy logic represent two different methodologies to deal with uncertainty. Each of those has its own advantages and disadvantages for using in prediction. Neural network can be used in complex non-linear relationships while fuzzy logic is used for imprecision and uncertain data. The fuzzy set theory has emerged as an alternative to capture the vagueness, uncertainty and imprecision present in the information. Therefore, in early stages, where the data is inadequate or is present in form of 'knowledge', use of fuzzy logic would be more appropriate. Any models based on fuzzy techniques help in the prediction of software residual defects. Here a hybrid approach of neuro-fuzzy combination is used for predicting reliability. The proposed model of prediction is depicted in Figure 1.

Advantage of fuzzy-neural over neural

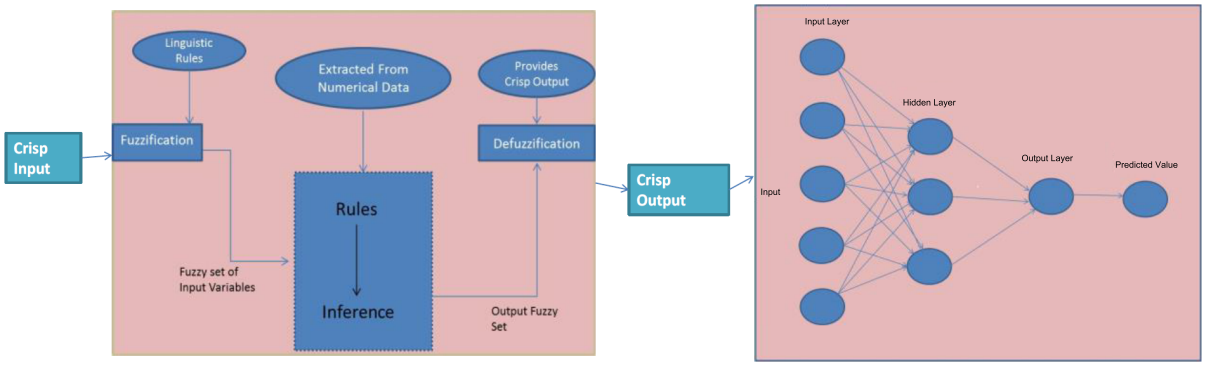


FIGURE 1. Implementation of fuzzy-neural approach

Method of Fuzzification and Prediction. Method of prediction through fuzzy has been obtained by Srivastava [8]. Fuzzification process starts with defining the universe of discourse U, which contains the historical data and upon which the fuzzy sets are defined. The study deals with the number of faults occurred during the software reliability checking with assumption that it includes some vagueness incurred due to statistical sampling. The algorithm for application of fuzzy method for software reliability prediction comprises the following steps.

Step 1: Let A_{min} and A_{max} be minimum and maximum production. Based upon A_{min} and A_{max}, we define the universe of the discourse U as $[A_{min} - A_1, A_{max} + A_2]$, where A_1 and A_2 are two proper positive numbers and accordingly, the universe of discourse U = [3-15]. Further the universe of discourse U is partitioned into six intervals of equal length as follows:

$U_1 = [3\text{-}5]$ $A_1 = 4$
$U_2 = [5\text{-}7]$ $A_2 = 6$
$U_3 = [7\text{-}9]$ $A_3 = 8$
$U_4 = [9\text{-}11]$ $A_4 = 10$
$U_5 = [11\text{-}13]$ $A_5 = 12$
$U_6 = [13\text{-}15]$ $A_6 = 14$

1. Universe of discourse: all existing matter and space considered as a whole; the cosmos

2. The values of Amin and Amax are predefined, derived from different sources

3. The values of A1, A2, A3, etc. are mid points, according to which the range of U1, U2, U3, etc respectively, is calculated

4. The A1 and A2 values given above are different from the A1, A2, A3, etc values given for the ranges. These A1 and A2 just ranges given to the min and max values as fault tolerance.

Step 2: Fuzzy sets A_1, A_2, A_3, A_4, A_5, A_6 on universe of discourse, having linguistic values as:

A_1 = not good, A_2 = not too good, A_3 = satisfactory good,
A_4 = good, A_5 = fairly good, A_6 = very good

This can be confusing but we can understand it this way: the higher the number of faults detected, the better, probably because the faults are not going unseen, which could be the case for lower number of faults. LOL it's linguistic, so let's just accept and move on

are to be defined. U_1, U_2, U_3, U_4, U_5, U_6 are chosen as elements of these fuzzy sets. The membership grades of U_1, U_2, U_3, U_4, U_5, U_6 to

Membership Grades == Degree of Membership

$A_1 = \{U_1/1, U_2/.5, U_3/0, U_4/0, U_5/0, U_6/0\}$
$A_2 = \{U_1/.5, U_2/1, U_3/.5, U_4/0, U_5/0, U_6/0\}$
$A_3 = \{U_1/0, U_2/.5, U_3/1, U_4/.5, U_5/0, U_6/0\}$
$A_4 = \{U_1/0, U_2/0, U_3/.5, U_4/1, U_5/.5, U_6/0\}$
$A_5 = \{U_1/0, U_2/0, U_3/0, U_4/.5, U_5/1, U_6/.5\}$
$A_6 = \{U_1/0, U_2/0, U_3/0, U_4/0, U_5/.5, U_6/1\}$

To understand this shit, I looked for [8], here it is: https://www.csjournals.com/IJITKM/PDF%204-2/Article_11.pdf

If you look at page 3, you will get a better understanding of how things are derived.

Basically, the values of A1, A2, etc are given. The values found from the survey are compared with all the ranges, and accordingly assigned a set. Look at the sample dataset in appendix A.

The fuzzy logical relationships are

The transition of 1 element to the other is represented as Ai -> Aj. It means that the next transition after Ai was Aj.

$A_1 \rightarrow A_4$
$A_2 \rightarrow A_4, A_3, A_4, A_2, A_4, A_3$
$A_3 \rightarrow A_3, A_4, A_3, A_4, A_4, A_3, A_3, A_3, A_4, A_3, A_3, A_5, A_3, A_2, A_1, A_4, A_4, A_4, A_6, A_3, A_5$
$A_4 \rightarrow A_4, A_3, A_4, A_5, A_3, A_3, A_4, A_3, A_3, A_4, A_2, A_4, A_4, A_4, A_3, A_3, A_5, A_4, A_2, A_3, A_4, A_5, A_4, A_3, A_4, A_5, A_4, A_4, A_5, A_5, A_5, A_5$
$A_5 \rightarrow A_5, A_2, A_4, A_4, A_3, A_5, A_5, A_4, A_5, A_5, A_5, A_5, A_4, A_6, A_4, A_6, A_6, A_5, A_5, A_5, A_5, A_6, A_5, A_6, A_4\ A_6 \rightarrow A_6, A_3, A_6, A_6, A_5, A_6, A_5, A_5, A_3, A_5$

From here we get

$A_1 = 7$
$A_2 = 8.2857$
$A_3 = 9.1305$
$A_4 = 9.4285$
$A_5 = 12$
$A_6 = 12.1818$

The values given here are derived from the entries in "Predicted Values" given in the sample data in appendix A. I could not find how they predicted the values based on the set, but it is probably beyond the scope of this research paper.

These values (A1 = 7 etc) are the crisp outputs received from fuzzy network, as these values are given below the column name " Predicted value (fuzzy)"

I feel like they have abstracted the crisp output to predicted value conversion part (basically neural network part).
I do not understand the purpose of finding the relationships and what exactly we achieve by doing so :D like if I give this network a value say 8, what exactly will be the output? A3? 9.13? 8.30?

3.2. **Neural network approach.** Artificial neural networks are a computational metaphor inspired by studies of the brain and nervous systems in biological organisms [5]. Neural networks are likened to non-parametric models in the statistical literature. It communicates through the connections between processing elements called neurons. Knowledge

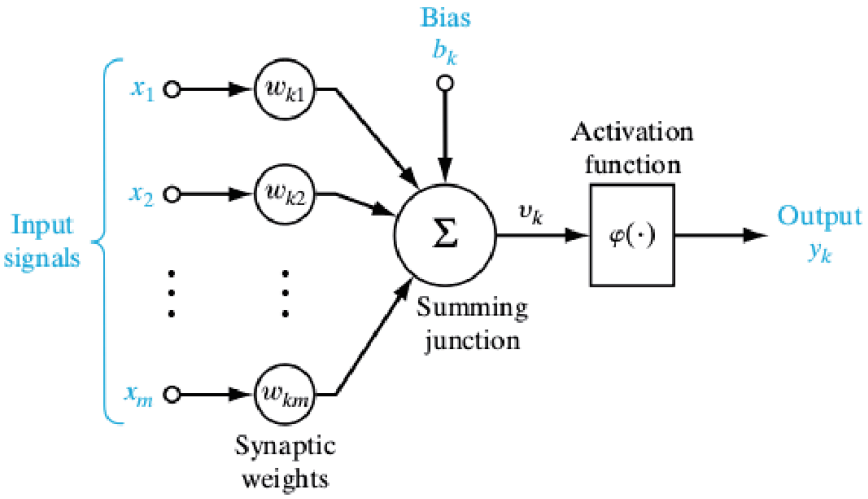


FIGURE 2. Process of neural network

is encoded into the network through the strength of the connections between different neurons, called weights, w which can be modified so as to model synaptic learning. The unit computes some function f of the weighted sum of its inputs. While designing the neural network the individual element inputs are $x_1, x_2, \ldots, x_m$ multiplied by the weights $w_{k1}, w_{k2}, \ldots, w_{km}$ and the weighted values are fed to the summing junction. Their sum is simply wx, the dot product of the single row matrix w and the vector x. m is the number of elements in the input vector.

LOL they make it sound so complicated. It's just the basic dot product of weight and inputs + bias

$$a = \left(\sum w_1, m \times m\right) + b \tag{1}$$

The weighted sum is $\sum w_1$, and $m \times m$ is called the **net input** to unit 1, often written as net1.

Note that w_1, m refers to the weight from unit m to unit 1 and b refers to the bias neuron.

The function φ is the unit's **activation function**. In the simplest case, φ is the sigmoid function, and the unit's output is

$$\varphi(n) = 1/(1 + e^{-n}) \tag{2}$$

Neural networks learn by example. The *learning rule* is provided with a set of examples (the *training set*) of proper network behavior $\{x_1, t_1\}, \{x_2, t_2\}, \ldots, \{x_Q, t_Q\}$ where x_Q is an input to the network, and t_Q is the corresponding correct (*target*) output. As the inputs are applied to the network, the network outputs are compared to the targets. The learning rule is then used to adjust the weights and biases of the network in order to move the network outputs closer to the targets.

We have used the MATLAB for neural network tool. We used Levenberg-Marquardt algorithm for training and testing of results. LM algorithm combines the advantages of gradient-descent and Gauss-Newton methods. LM steps are linear combination of gradient-descent and Gauss-Newton steps based on adaptive rules. Gradient-descent dominated steps until the canyon is reached, are followed by Gauss-Newton dominated steps. The training dataset is 70% of the actual data while testing and validation set are 15% each.

https://en.wikipedia.org/wiki/Levenberg%E2%80%93Marquardt_algorithm
https://en.wikipedia.org/wiki/Gauss%E2%80%93Newton_algorithm

4. **Results and Discussion.** The proposed methods of fuzzy approach and fuzzy-neural approach have been implemented on the very well-known data set of John Musa of Bell Laboratories received from IEEE repository. We have applied results achieved from fuzzy

prediction into the ANFIS tool of MATLAB. Membership functions for ANFIS tool input are

```
Name='input1'
Range=[1 15]
MF1='NG':'trimf',[1 2 3]
MF2='NTG':'trimf',[3 4.5 6]
MF3='SG':'trimf',[5.5 7 8.5]
MF4='G':'trimf',[8.5 9.5 10.5]
MF5='FG':'trimf',[10.48 11.488 12.48]
MF6='VG':'trimf',[12.6 13.68 15]
```

Output membership functions are

```
Range=[0 15] Num

MFs=3

MF1='NG':'trimf',[0.167 2.92 5.257]
MF2='FG':'trimf',[4.503 7.16 10]
MF3='VG':'trimf',[9 12.043 15]
```

Fuzzy Min-Max networks are useful for handling problems where the decision boundaries between classes are not well-defined or when there is uncertainty in the data.

Centroid method is probably referring to finding the crisp output by finding the value of the centroid of the graph achieved in MATLAB.

Fuzzy min-max method is used for fuzzification and centroid method is used for defuzzification while back propagation algorithm is used in getting the better results. The fuzzy if-then rules, membership functions and ANFIS process is depicted in the figure below.

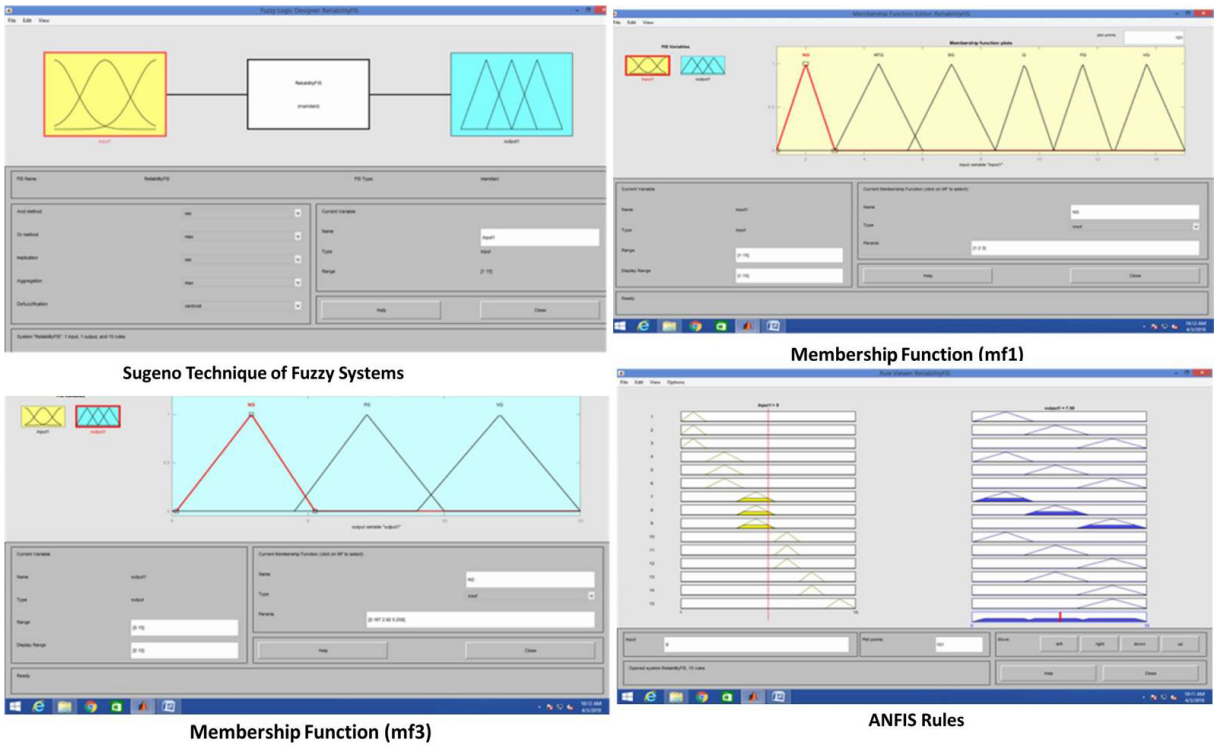


FIGURE 3. Process in ANFIS tool of MATLAB

Reliability prediction values in form of time to failure achieved from different techniques that are fuzzy, fuzzy-neural, neural network and neural-fuzzy are shown in Appendix A.

To have a comparison of the NRMSE calculated values of our proposed models and shown in Figure 2. The Root-Mean-Square Error (RMSE) is a frequently used measure of the differences between values (sample and population values) predicted by a model or an

estimator and the values actually observed. The RMSE represents the sample standard deviation of the differences between predicted values and observed values. The RMSE of prediction of a method is measure of accuracy of that prediction method. Normalizing the RMSE facilitates the comparison between datasets or models with different scales. Though there is no consistent means of normalization in the literature, common choices are the mean or the range (defined as the maximum value minus the minimum value) of the measured data:

$$NRMSE = \frac{RMSE}{\max - \min}$$

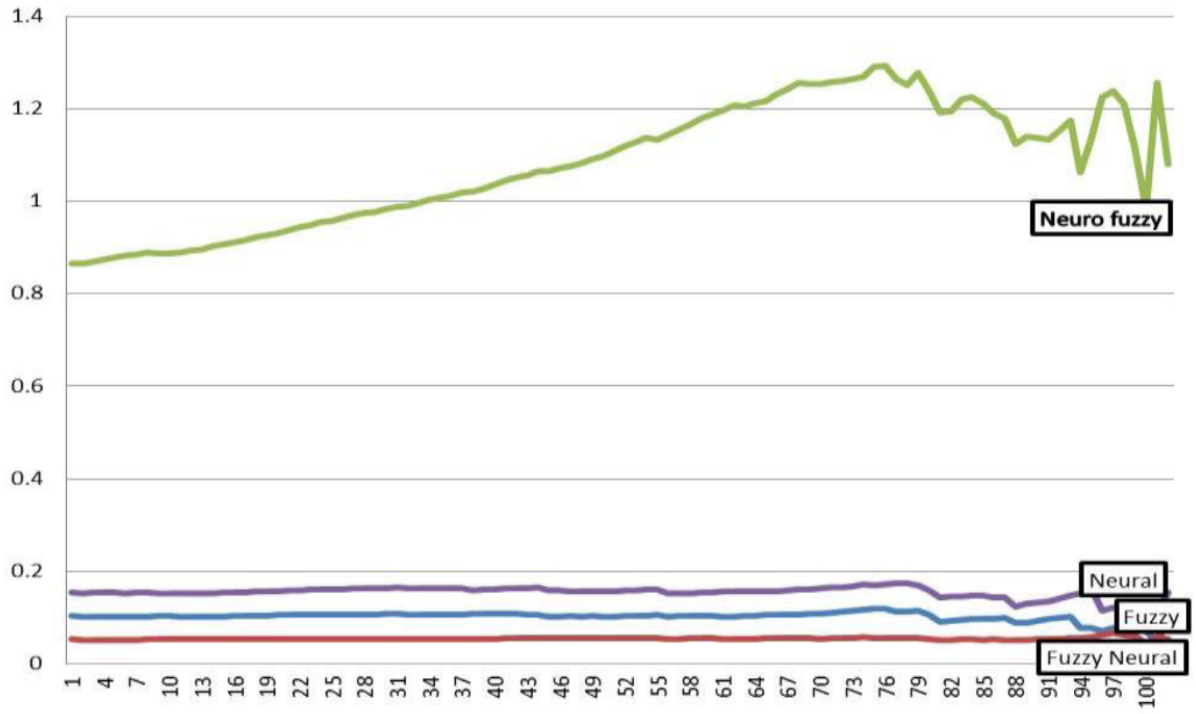
where

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(P_i - O_i)^2}{n}}$$

where P_i is predicted value and O_i is observed value. n is the number of observations. NRMSE is often expressed as a percentage, where lower values indicate less residual variance. In many cases, especially for smaller samples, the sample range is likely to be affected by the size of sample which would hamper comparisons. The NRMSE values attained by the above results are

Fuzzy	Fuzzy-neural	Neuro-fuzzy	Neural
0.102111	0.05463183	1.081001141	0.15418

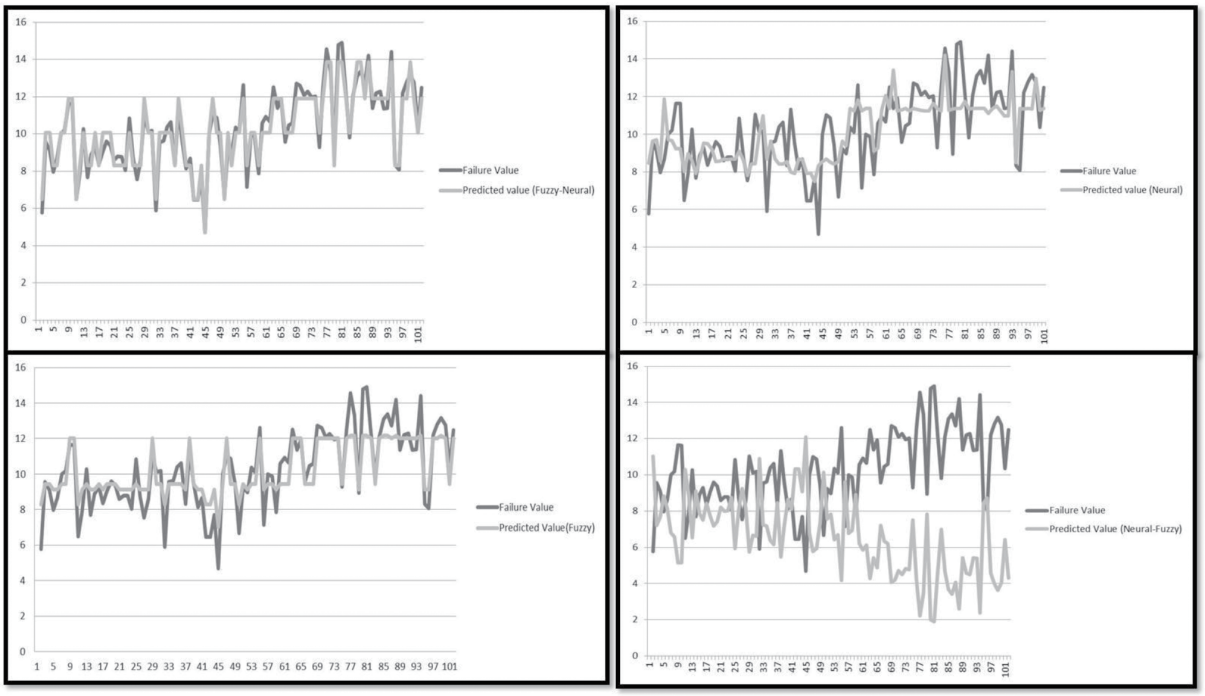
The average NRMSE values achieved by different methods and their effects are depicted by graph in following Figure 4.



LMAO

FIGURE 4. Graph of NRMSE between different methods

5. **Conclusions.** The proposed fuzzy-neural method has been implemented to have software reliability. The prediction of reliability has many factors such as mean time to failure, and time to failure. We have considered the time to failure data of bell laboratories. The motivation of the study is that there is no single method which can be used in all kinds of data of reliability for prediction. Hence this research focused on hybrid methodology



Nice images we can use

FIGURE 5. Comparison of achieved results from different methods

of fuzzy neural. In this method we implemented data of reliability on fuzzy methodology manually and the results are then applied to the MATLAB tool of neural network. Further in neural network we used Levenberg-Marquardt algorithm to predict reliability.

To evaluate the performance of reliability prediction model, average normalized RMSE error was computed. From the four methods proposed here it comes minimum with the fuzzy-neural method and it is 0.05463183 which is good in comparison to the other methods.

Based on the results achieved we may conclude that

- The suitability of the proposed method is examined and it is found that the fuzzy-neural method of prediction of software reliability is superior in terms of accuracy and robustness.
- Fuzzy based algorithm proposed here with the MATLAB implementation of neural network can be used as effective model for prediction of reliability with failure dataset.
- In the proposed work we have considered only one factor of reliability. In future results may vary with the different factors consideration.

REFERENCES

[1] J. D. Musa, *Software Reliability Engineering: More Reliable Software, Faster Development and Testing*, Mcgraw-Hill, New York, 2004.

[2] D. N. Arnold, *Two Disasters Caused by Computer Arithmetic Errors*, http://www.ima.umn.edu/*arnold/455.f96/disasters.html.

[3] M. R. Lyu, Software reliability engineering: A road map, *Future of Software Engineering*, pp.153-170, 2007.

[4] D. K. Yadav, S. K. Chaturvedi and R. B. Misra, Early software defects prediction using fuzzy logic, *International Journal of Performability Engineering*, vol.8, no.4, pp.399-408, 2012.

[5] J. L. Lions, *ARIANE 5 Flight 501 Failures-Report by the Inquiry Board*, http://www.di.unito.it/*damiani/ariane5rep.html.

[6] J. D. Musa, Software reliability data, *IEEE Computer Society Repository*, 1979.

[7] R. Mohanty, V. Ravi and M. R. Patra, Hybrid intelligent systems for predicting software reliability, *Applied Soft Computing*, vol.13, no.1, pp.189-200, 2013.

[8] R. K. Srivastava, Fuzzy-neural techniques for short term forecast of food grains production, *International Journal of Information Technology*, vol.4, no.2, pp.385-390, 2011.
[9] N. Rajkiran and V. Ravi, Software reliability prediction using wavelet neural networks, *International Conference on Computational Intelligence and Multimedia Application (ICCIMA 2007)*, pp.195-197, 2007.
[10] Y.-S. Su and C.-Y. Huang, Neural-network-based approaches for software reliability estimation using dynamic weighted combinational models, *The Journal of Systems and Software*, vol.80, no.4, pp.606-615, 2007.
[11] T. M. Khoshgoftaar, E. B. Allen, W. D. Jones and J. P. Hudepohl, Classification – Tree models of software quality over multiple releases, *IEEE Trans. Reliability*, vol.49, no.1, pp.4-11, 2000.
[12] N. Karuanithi, D. Whitely and K. Malaya, Predictions of software reliability using connectionist models, *IEEE Trans. Software Engineering*, vol.18, no.7, pp.563-574, 1992.
[13] L. A. Zadeh, Fuzzy sets, *Information and Control*, vol.8, pp.338-353, 1965.
[14] L. A. Zadeh, Fuzzy logic and soft computing: Issues, contentions and perspectives, *Proc. of IIZUKA'94: The 3rd International Conference on Fuzzy Logic, Neural Nets and Soft Computing*, Iizuka, pp.1-2, 1994.
[15] Y. Zhang and H. Chen, Predicting for MTBF failure data series of software reliability by genetic programming algorithm, *Proc. of the 6th International Conference on Intelligent Systems Design and Applications (IEEE Computer Society)*, Washington, USA, 2006.
[16] E. O. Costa, A. T. R. Pozo and S. R. Vergilio, A genetic programming approach for software reliability modeling, *IEEE Trans. Reliability*, vol.59, no.1, 2010.
[17] Z. Al-Rahamneh, M. Reyalat, A. F. Sheta, S. Bani-Ahmad and S. Al-Oqeili, A new software reliability growth model: Genetic-programming-based approach, *Journal of Software Engineering and Applications*, vol.4, pp.476-481, 2011.
[18] M. Benaddy and M. Wakrim, Simulated annealing neural network for software failure prediction, *International Journal of Software Engineering and Its Applications*, vol.6, no.4, 2012.
[19] D. K. Yadav, S. K. Chaturvedi and R. B. Misra, Early software defects prediction using fuzzy logic, *International Journal of Performability Engineering*, vol.8, no.4, pp.399-408, 2012.
[20] R. Kumar, S. A. Khan and R. A. Khan, Durable security in software development: Needs and importance, *CSI Communication*, pp.34-36, 2015.
[21] K. Sahu, Rajshree and R. Kumar, Risk management perspective in SDLC, *International Journal of Advanced Research in Computer Science and Software Engineering*, vol.4, no.3, pp.1247-1251, 2014.
[22] K. Sahu and R. K. Srivastava, Revisiting software reliability, in *Data Management, Analytics and Innovation, Advances in Intelligent Systems and Computing*, V. Balas, N. Sharma and A. Chakrabarti (eds.), Springer, 2018.
[23] C. Jin and S. W. Jin, Software reliability prediction model based on support vector regression with improved estimation of distribution algorithms, *Applied Soft Computing*, vol.15, pp.113-120, 2014.
[24] P. Roy, G. S. Mahapatra, P. Rani, S. K. Pandey and K. N. Dey, Robust feedforward and recurrent neural network based dynamic weighted combination models for software reliability prediction, *Applied Soft Computing*, vol.22, pp.629-637, 2014.
[25] S. W. A. Rizvi, R. A. Khan and V. K. Singh, Software reliability prediction using fuzzy inference system: Early stage perspective, *International Journal of Computer Applications*, vol.145, no.10, pp.16-23, 2016.
[26] Y. Shi, M. Li, S. Arndt and C. Smidts, Metric-based software reliability prediction approach and its application, *Empirical Software Engineering*, vol.22, no.4, pp.1579-1633, 2017.

Appendix A.

	Failure value	Fuzzified value	Predicted value (Fuzzy)	Predicted value (Fuzzy-Neural)	Predicted value (Neural)	Predicted value (Neural-Fuzzy)
1	5.7683	A2	8.2857	6.4892	8.45502	11.0187
2	9.5743	A4	9.4285	10.0662	9.616975	7.2127
3	9.105	A4	9.4285	10.0662	9.701643	7.682
4	7.9655	A3	9.1304	8.3018	8.579047	8.8215
5	8.6482	A3	9.1304	8.3018	11.85793	8.1388
6	9.9887	A4	9.4285	10.0662	9.69367	6.7983
7	10.1962	A4	9.4285	10.0662	9.658796	6.5908
8	11.6399	A5	12	11.901	9.22112	5.1471
9	11.6275	A5	12	11.901	9.237244	5.1595
10	6.4922	A2	8.2857	6.4892	8.012412	10.2948
11	7.901	A3	9.1304	8.3018	8.962842	8.886
12	10.2679	A4	9.4285	10.0662	8.602759	6.5191
13	7.6839	A3	9.1304	8.3018	7.91716	9.1031
14	8.8905	A3	9.1304	8.3018	8.667106	7.8965
15	9.2933	A4	9.4285	10.0662	9.52884	7.4937
16	8.3499	A3	9.1304	8.3018	9.505547	8.4371
17	9.0431	A4	9.4285	10.0662	9.205988	7.7439
18	9.6027	A4	9.4285	10.0662	8.546327	7.1843
19	9.3736	A4	9.4285	10.0662	8.581358	7.4134
20	8.5869	A3	9.1304	8.3018	8.682687	8.2001
21	8.7877	A3	9.1304	8.3018	8.649329	7.9993
22	8.7794	A3	9.1304	8.3018	8.677301	8.0076
23	8.0469	A3	9.1304	8.3018	8.683874	8.7401
24	10.8459	A4	9.4285	10.0662	9.09289	5.9411
25	8.7416	A3	9.1304	8.3018	8.60319	8.0454
26	7.5443	A3	9.1304	8.3018	7.794003	9.2427
27	8.5941	A3	9.1304	8.3018	8.400581	8.1929
28	11.0399	A5	12	11.901	8.434395	5.7471
29	10.1196	A4	9.4285	10.0662	9.960807	6.6674
30	10.1786	A4	9.4285	10.0662	10.98305	6.6084
31	5.8944	A2	8.2857	6.4892	8.676776	10.8926
32	9.546	A4	9.4285	10.0662	9.637993	7.241
33	9.6197	A4	9.4285	10.0662	8.68233	7.1673
34	10.3851	A4	9.4285	10.0662	8.422882	6.4019
35	10.6301	A4	9.4285	10.0662	8.398752	6.1569
36	8.3333	A3	9.1304	8.3018	8.546156	8.4537
37	11.315	A5	12	11.901	7.978447	5.472
38	9.4871	A4	9.4285	10.0662	7.910388	7.2999
39	8.1391	A3	9.1304	8.3018	8.665771	8.6479
40	8.6713	A3	9.1304	8.3018	8.648707	8.1157
41	6.4615	A2	8.2857	6.4892	7.90437	10.3255
42	6.4615	A2	8.2857	6.4892	7.947771	10.3255
43	7.6955	A3	9.1304	8.3018	7.495981	9.0915
44	4.7005	A1	7	4.7005	8.338976	12.0865
45	10.0024	A4	9.4285	10.0662	8.550078	6.7846
46	11.0129	A5	12	11.901	8.665771	5.7741
47	10.8621	A4	9.4285	10.0662	8.52336	5.9249
48	9.4376	A4	9.4285	10.0662	8.413915	7.3494
49	6.6644	A2	8.2857	6.4892	8.534675	10.1226
50	9.2294	A4	9.4285	10.0662	9.617057	7.5576

This is not the bell dataset that they are talking about non stop.

	Failure value	Fuzzified value	Predicted value (Fuzzy)	Predicted value (Fuzzy-Neural)	Predicted value (Neural)	Predicted value (Neural-Fuzzy)
51	8.9671	A3	9.1304	8.3018	9.381895	7.8199
52	10.3534	A4	9.4285	10.0662	11.35835	6.4336
53	10.0998	A4	9.4285	10.0662	11.23909	6.6872
54	12.6078	A5	12	11.901	11.81392	4.1792
55	7.1546	A3	9.1304	8.3018	11.2449	9.6324
56	10.0033	A4	9.4285	10.0662	11.37073	6.7837
57	9.8601	A4	9.4285	10.0662	11.37085	6.9269
58	7.8675	A3	9.1304	8.3018	9.094755	8.9195
59	10.5757	A4	9.4285	10.0662	9.263998	6.2113
60	10.9294	A4	9.4285	10.0662	11.28957	5.8576
61	10.6604	A4	9.4285	10.0662	12.04592	6.1266
62	12.4972	A5	12	11.901	11.37079	4.2898
63	11.3745	A5	12	11.901	13.40372	5.4125
64	11.9158	A5	12	11.901	11.22857	4.8712
65	9.575	A4	9.4285	10.0662	11.28411	7.212
66	10.4504	A4	9.4285	10.0662	11.36819	6.3366
67	10.5866	A4	9.4285	10.0662	11.23313	6.2004
68	12.7201	A5	12	11.901	11.3612	4.0669
69	12.5982	A5	12	11.901	11.31239	4.1888
70	12.0859	A5	12	11.901	11.25808	4.7011
71	12.2766	A5	12	11.901	11.23101	4.5104
72	11.9602	A5	12	11.901	11.23142	4.8268
73	12.0246	A5	12	11.901	11.62888	4.7624
74	9.2873	A4	9.4285	10.0662	11.36039	7.4997
75	12.495	A5	12	11.901	11.23383	4.292
76	14.5569	A6	12.1818	13.8476	14.18953	2.2301
77	13.3279	A6	12.1818	13.8476	11.25622	3.4591
78	8.9464	A3	9.1304	8.3018	11.33068	7.8406
79	14.7824	A6	12.1818	13.8476	11.37087	2.0046
80	14.8969	A6	12.1818	13.8476	11.37086	1.8901
81	12.1399	A5	12	11.901	11.74921	4.6471
82	9.7981	A4	9.4285	10.0662	11.35478	6.9889
83	12.0907	A5	12	11.901	11.36261	4.6963
84	13.0977	A6	12.1818	13.8476	11.37056	3.6893
85	13.368	A6	12.1818	13.8476	11.3686	3.419
86	12.7206	A5	12	11.901	11.37805	4.0664
87	14.192	A6	12.1818	13.8476	11.10511	2.595
88	11.3704	A5	12	11.901	11.3445	5.4166
89	12.2021	A5	12	11.901	11.51131	4.5849
90	12.2793	A5	12	11.901	11.24131	4.5077
91	11.3667	A5	12	11.901	10.96682	5.4203
92	11.3923	A5	12	11.901	10.96682	5.3947
93	14.4113	A6	12.1818	13.8476	13.32296	2.3757
94	8.3333	A3	9.1304	8.3018	8.492889	8.4537
95	8.0709	A3	9.1304	8.3018	11.37076	8.7161
96	12.2021	A5	12	11.901	11.33752	4.5849
97	12.7831	A5	12	11.901	11.35914	4.0039
98	13.1585	A6	12.1818	13.8476	11.33241	3.6285
99	12.753	A5	12	11.901	12.95231	4.034
100	10.3533	A4	9.4285	10.0662	11.26392	6.4337
101	12.4897	A5	12	11.901	11.35678	4.2973