IJRRS
International Journal of Reliability, Risk Saftey: Theory and Application

Online ISSN: 2676-3346

Vol. 5/ Issue 1/ 2022/ pp. 43-48
DOI: 10.30699/IJRRS.5.1.6
Received: 2021.09.02, Accepted: 2022.09.30

Original Research Article

A Novel Method for Software Reliability Assessment via Neuro-Fuzzy System

Elham Babaie[1]*

1. Department of Computer Engineering, Ayatollah Amoli Branch, Islamic Azad University, Mazandaran, Iran

***elhambabaie@ymail.com**

Abstract

Nowadays, the utilization of software engineering in various areas of technology is remarkably increased. As a matter of fact, it is used in many critical applications such as eye surgery, autopilot systems of airplanes, centralized traffic control (CTC), and so on. Therefore, the reliability of software is very important, and it plays an essential role in the lifetime of the software. Software reliability is one of the main characteristics of software quality. Moreover, the rapid assessment of the reliability of the application is essential during the software life cycle. In this paper, I use the neuro-fuzzy methods to assess the software's reliability in order to cope with uncertainties in measuring the actual parameters of the software. By designing neuro-fuzzy inference systems and applying four parameters of the ISO/IEC 9126quality model(i.e., the maturity of software, fault-tolerant, recoverability, and reliability compliance) and finding the parameters of a fuzzy system by exploiting approximation techniques from neural networks, I present an integrated assessment model for evaluation of software reliability. The case study used in this paper to evaluate the proposed method is the software income tax calculator. By applying the input parameters, I observe that the software reliability is 0.65. software reliability in our proposed method is more exact than software reliability in the fuzzy multi-criteria and fuzzy method because The weights of the input parameters have been set by experts and software developers, and simulations are carried out using MATLAB tool (ANFIS). Simulations confirm that the proposed method provides acceptable results.

Keywords: software quality, software reliability, Neuro-Fuzzy System, software maturity, fault tolerance, recoverability, and reliability compliance reliability.

1. Introduction

Because of the complexity and sensitivity of the software, certain parameters, such as the quality of the software, have to be taken into account. It is worth noting that the reliability of software is a subset of software quality. Software Reliability is the probability of failure-free software operation for a specified period of time in a specified environment. Software Reliability is also an important factor affecting system reliability [1]. Due to the increasing development and usage of software engineering in various fields of technology, software reliability plays a critical role in determining the software's lifetime.

Software reliability is the main characteristic of software quality, and also it is the most important evaluation metric for software clients. The reliability of software is reduced by failures that occur at run-time. If a program fails to function repeatedly, then the software quality factor is considerably decreased. So evaluating the reliability of a program in the life cycle is essential. Maturity, fault tolerance, recoverability, and compliance reliability are the important factors affecting the reliability of the application [1]. There are many models addressed in the technical literature to estimate software reliability, each of which has its own pros and cons. Some of them will be briefly studied in the next section. In this paper, a novel, efficient model for assessing software reliability is presented using a neuro-fuzzy-based method. In order to implement the proposed system in the neuro-fuzzy space, we need the software's reliability metrics. The final output of the system shows the efficiency of the software reliability in neuro-fuzzy.

The Neuro-Fuzzy approach is the combination of concepts based on fuzzy logic and a neural network. The integration of these two paradigms overcomes the limitations of each technique's isolated concept [2]. Neuro-fuzzy is an intelligent system that combines the human-like reasoning style of fuzzy systems with the learning and connectionist structure of neural networks. This type of system is characterized by a fuzzy system where fuzzy sets and fuzzy rules are adjusted using input-output patterns [3]. Neural networks can only come into play if the problem is expressed by a sufficient amount of observed examples. On the one hand, no prior knowledge about the problem needs to be given. On the other hand, however, it is not straightforward to extract comprehensible rules from the neural network's structure.

On the contrary, a fuzzy system demands linguistic rules instead of learning examples as prior knowledge. Furthermore, the input and output variables have to be described linguistically. If the knowledge is incomplete, wrong, or contradictory, then the fuzzy system must be tuned. Since there is not any formal approach to it, the tuning is performed in a heuristic way. This is usually very time-consuming and error-prone [4]. I used this method because Combining both approaches should unite advantages and exclude disadvantages.

In this paper, I use the neuro-fuzzy methods to assess the software's reliability in order to cope with uncertainties in measuring the actual parameters of the software. By designing neuro-fuzzy inference systems and applying four parameters of the ISO/IEC 9126 quality model (i.e., the maturity of software, fault-tolerant, recoverability, and reliability compliance) and finding the parameters of a fuzzy system by exploiting approximation techniques from neural networks, I present an integrated assessment model for evaluation of software reliability. The case study used in this paper to evaluate the proposed method is the income tax calculator software. By applying the input parameters, I observe that the software reliability is 0.65. Simulation results confirmed that the proposed model is more accurate than previous ones because the weights of the input parameters have been set by experts and software developers.

The remainder of the paper is as follows: In section 2, the previous work is studied. In section 3, the ISO/IEC 9126 Model is reviewed. In section 4 Neuro-Fuzzy System is reviewed. In section 5, the proposed model is presented. Simulation results are shown in section 6. Finally, conclusions and Future Work are made in sections 7&8.

2. Related Work

Adnan and Yaacob showed that artificial neural networks (ANNs), together with fuzzy logic systems, can predict software reliability more effectively [5]. Aljahdali and Shetahave implemented a novel fuzzy model based on Takagi-Sugeno's previous model, and also they scrutinized their developed fuzzy model using three kinds of applications such as control systems, military, and real-time operating [6]. Georgieva and Dimov introduce a fuzzy logic approach to estimate the software reliability using standard, reliable growth of data obtained during the testing of the system [7].

Challa Paul et al. present an algorithm to quantify the software quality parameters using the fuzzy multi-criteria approach. The proposed model has been clearly illustrated with case studies in [8]. The quantified software quality with respect to the user's, developer's, and project manager's perspectives have been obtained [8]. Dadhich and Mathur have made an attempt to quantify the reliability of aspect-oriented software using the ISO/IEC 9126 Model. Due to the unpredictable nature of software quality attributes, the fuzzy multi-criteria approach has been used to mature the quality of the software [1].

Bonthu Kotaiah proposed an approach for assessing software reliability by using different machine learning techniques like neural networks, fuzzy logic, and so forth to measure the software errors to improve the software reliability at different phases of the software development life cycle (SDLC) [9]. Kotaiah, Prasad, et al. proposed a software reliability assessment method using neuro-fuzzy-based systems and their effect in assessing software reliability. They used the normalized root mean square error (NRMSE) as an evaluation criterion [10]. Zavvar and Ramezani proposed a method using fuzzy logic and the above aspects of the software to measure the software's reliability. Given that the proposed model uses four metrics, and each of these metrics is also three linguistic variables, therefore, the proposed model uses 81 laws [11]. Milovancevic, Dimov, et al. analyzed several reliability models by a soft computing approach called adaptive neuro-fuzzy inference system (neuro-fuzzy) in order to estimate the models' capability based on root mean square errors (RMSE). Various aspects of the accuracy of some of the well-known software reliability models have been used in their work [12]. Gandhi, Khan, et al. proposed an efficient algorithm that can be used for the prediction of software reliability. The proposed algorithm is implemented using a hybrid approach named Neuro-Fuzzy Inference System and has also been applied to test data. In this work, a comparison between different techniques of soft computing has been performed [2]. Elghamry, Eldamcesse et al. discussed a new algorithm for evaluating the fuzzy reliability of a fuzzy linear consecutive k-out-of-n: F system (Lin/C (k,n: F)) with independent, un-repairable, and non-identical components[13].

2.1.The ISO/IEC 9126 Model

The ISO 9126 software quality model identifies 6 main quality characteristics, including Functionality, Reliability, Usability, Efficiency, Maintainability, and Portability [14]. Model ISO/IEC 9126, in comparison with other presented models, is complete and has eliminated the disadvantages of previous models. Therefore, it is chosen as a model which is used in this paper. The reasons for choosing this model are the special feature of this model; the most important are comprehensive qualitative features, the ability to understand the hierarchical structure, phrases, and topics of a common, clear definition of components, and measurement standards.

Software reliability is one important characteristic that should be regarded both during the development and usage of software [12]. Reliability is the probability that a system or component will fail within a given period of time. Reliability is further subdivided into various sub-characteristics that include maturity, recoverability, fault tolerance, and reliability compliance**.**

Maturity: describes the frequency of failure of the software by faults.

Fault Tolerance: evaluates the robustness of the software. It describes the software attributes that describe the ability of the software to maintain a specified level of performance in cases of software faults or the violation of its specified interface**.**

Recoverability: describes the capability of the software to reestablish its level of performance and to recover the data directly affected in case of failure and the time and effort needed for it.

Reliability compliance: determines whether the software adheres to the compliance standards of reliability or not.

3. Neuro-Fuzzy System

The techniques of artificial intelligence based on fuzzy logic and neural networks are frequently applied together. The reasons to combine these two paradigms come out of the difficulties and inherent limitations of each isolated paradigm. Generically, when they are used in a combined way, they are called Neuro-Fuzzy Systems. This term, however, is often used to assign a specific type of system that integrates both techniques. This type of system is characterized by a fuzzy system where fuzzy sets and fuzzy rules are adjusted using input-output patterns. There are several different implementations of neuro-fuzzy systems, where each author defines their own model [3].

Both neural networks and fuzzy systems have some things in common. They can be used for solving a problem (e.g., pattern recognition, regression, or density estimation) if there does not exist any mathematical model of the given problem. They solely do have certain disadvantages and advantages, which almost completely disappear by combining both concepts. Neural networks can only come into play if the problem is expressed by a sufficient amount of observed examples. These observations are used to train the black box. On the one hand, no prior knowledge about the problem needs to be given. On the other hand, however, it is not straightforward to extract comprehensible rules from the neural network's structure.

On the contrary, a fuzzy system demands linguistic rules instead of learning examples as prior knowledge. Furthermore, the input and output variables have to be described linguistically. If the knowledge is incomplete, wrong, or contradictory, then the fuzzy system must be tuned. Since there is not any formal approach to it, the tuning is performed in a heuristic way. This is usually very time-consuming and error-prone. It is desirable for fuzzy systems to have an automatic adaption procedure that is comparable to neural networks. Combining both approaches should unite advantages and exclude disadvantages [4].

In generic terms, the bibliography points out that fuzzy neuro systems that implement Takagi-Sugeno-type fuzzy inference systems get more accurate results than the approaches that implement neuro-fuzzy inference systems of the Mamdani type. For instance, consider that the FIS has two inputs, x and y, and one output z. For the first-order Sugeno fuzzy model, a typical rule set with two fuzzy if-then rules can be expressed as [15, 16]

Rule1: if xis A_1 and yis B_1 then $f_1 = p_1 x + q_1 y + r_1$ (1)

Rule2: if xis A_2 and yis B_2 then $f_2 = p_2 x + q_2 y + r_2$ (2)

The general structure of the ANFIS is presented in Figure 1. Selection of the FIS is the major concern when designing an ANFIS to model a specific target system.

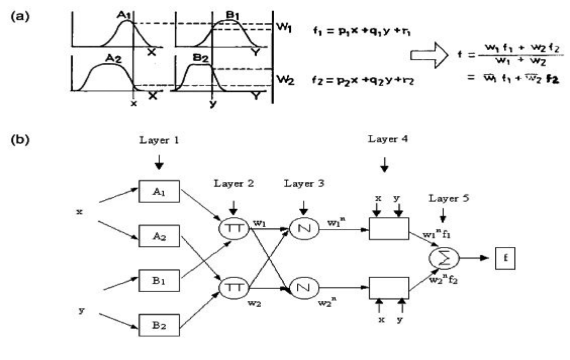


Figure 1. **(a)** Fuzzy inference system. **(b)** Equivalent ANFIS architecture [15, 16]

Layer 1: Each node in this layer generates membership grades of an input variable. The node output OPli is defined by:

$$OP^1_i = \mu_{Ai}(\chi)\ for\ i = 1,2 \tag{3}$$

$$OP^1_i = \mu_{Bi-1}(y) for\ i = 3,4 \tag{4}$$

Layer 2: Every node in this layer multiplies the incoming signals, denoted as π, and the output OPli That represents the firing strength of a rule is computed as:

$$OP^2_i = w_i = \mu_{Ai}(\chi)\mu_{Bi}(y)\ for\ i = 1,2 \tag{5}$$

Layer 3: The ith node of this layer, labeled as N, computes the normalized firing strengths as:

$$OP^3_i = \varpi_i = \frac{w_i}{W1+W2}\ for\ i = 1,2 \tag{6}$$

Layer 4: Node i in this layer computes the contribution of the i th rule towards the model output with the following node function:

$$OP^4_i = \varpi_i f_i = \varpi_i (p_i x + q_i y + r_i) \tag{7}$$

Layer 5: The single node in this layer computes the overall output of the ANFIS as:

$$OP^5_i = \sum_i \varpi_i f_i = \frac{\sum_i f_i w_i}{\sum_i w_i} \tag{8}$$

The hybrid learning algorithm, which combines the back propagation gradient descent and least squares method, can be used for an effective search of the optimal parameters of the ANFIS. More specifically, in the forward pass of the hybrid learning algorithm, the node output goes forward until layer 4, and the consequent parameters are identified by the least squares method. In the backward pass, the error signal propagates backward, and the premise parameters are updated by gradient descent. As mentioned earlier, the consequent parameters thus identified are optimal under the condition that the premise parameters are fixed. Accordingly, the hybrid approach converges much faster since it reduces the dimension of the search space of the original back-propagation method [15].

4. The Proposed Model

The proposed model is the designing of a neuro-fuzzy system that can evaluate the software's reliability. The hybrid learning algorithm is used to train the fuzzy neural system. The FIS inputs are the parameters of software reliability that have been extracted from the quality model ISO/IEC 9126 (i.e., the maturity of software, fault-tolerant, recoverability, and reliability compliance), and output will be the overall reliability of the software. The case study used in this paper to evaluate the proposed method is a software income tax calculator; this is openly available software that has been developed by the Government of India. It's commonly used across the country. This software is used to calculate income tax based on various input parameters such as total income, investments, savings, etc. [8]. Challa Paul et al. evaluated the software quality with a fuzzy multi-criteria approach for software "calculating income tax."

The Fuzzy input data is provided from this case study, and The weights of the input parameters and fuzzy rules have been set by experts and software developers.

With Considering all Possible combinations for inputs, 375 rules are presented in the knowledge base. These weights have been acquired via a questionnaire [8].

The system implementation is done using ANFIS toolbox, existing in MATLAB. Figure 2 shows FIS with 4 inputs and one output variable.

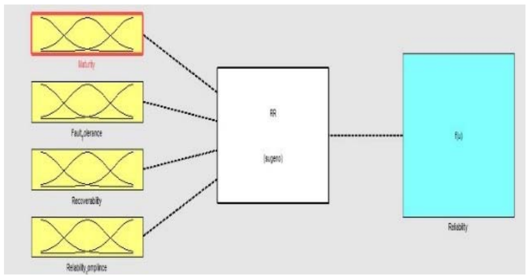


Figure 2. The number and names of inputs and outputs of FIS

70% of the data was used as training data, and 30% of the data as check data. After determining the parameters of the Sugeno fuzzy system for training, a fuzzy inference system should load train data used for modeling in the first step. Then should load the check data. After loading data, the generate f has used an option to set the membership functions and defined mf type of input as gbell mf and mf type of output as linear, and several mfs as (5 5 5 3). thus were loaded data in ANFIS. Figure 3 and Figure 4 show train data and check data, respectively.

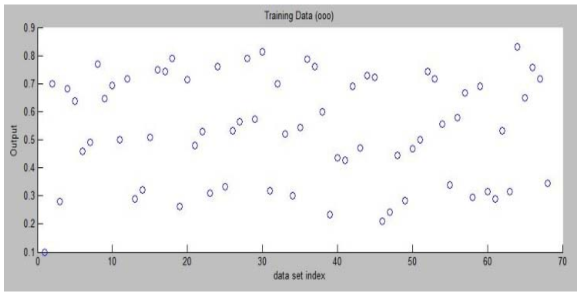


Figure 3. shows train data

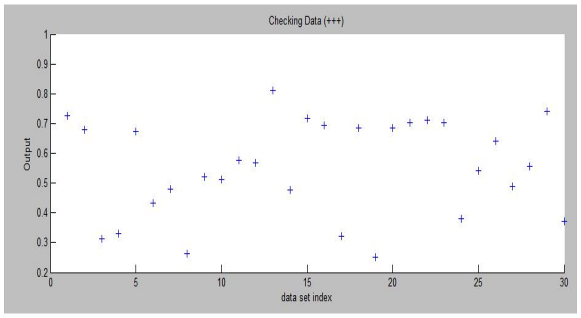


Figure 4. shows the check data

After loading train data, the hybrid method was used to train ANFIS. The hybrid learning algorithm, which combines the back propagation gradient descent and least squares method, can be used for an effective search of the optimal parameters of the ANFIS, then defined epochs and error tolerance. Our proposed model considered error tolerance as 0.001% and epochs as 35. training is stopped by reaching the epochs or the error tolerance. Figure 5 shows The least error in the training epoch. The model is trained according to these data, then modeled the data behavior, and The model according to the trained data is predicted the data.

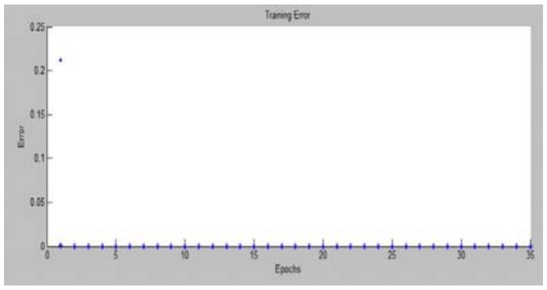


Figure 5. shows The least error in the train epoch

Trained data and modeled data were indicated with a circle (O) and star (*), respectively, in Figure 6. The average testing error of this model is 0.00000122668%. And it shows that ANFIS has done this modeling with a certainty of 100 %.

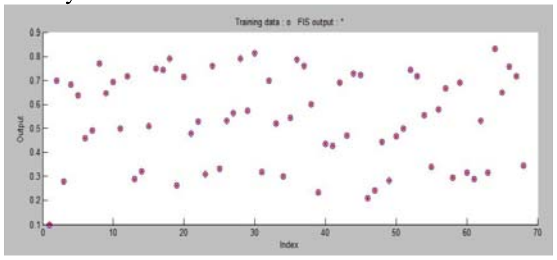


Figure 6. shows Trained data and modeled data

Checked data and predicted data were indicated with summation (+) and star (*), respectively, in Figure 7. The average testing error that was predicted is 0.21176 %. And it shows that ANFIS has done this predicting with a certainty of 99.78824 %. Therefore, it can be predicted software reliability by the neuro-fuzzy system.

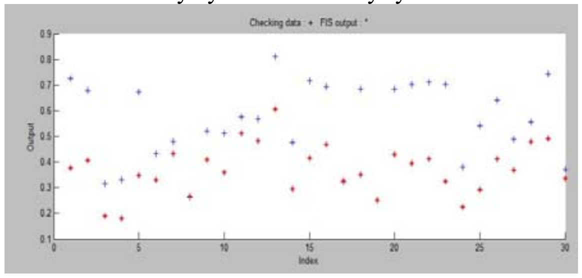


Figure 7. shows checked data and modeled data

4.1. Evaluating The Proposed Method

In order to further study and also ensure the integrity of the proposed method I use a case study reported in [8]. It evaluates the software quality with a fuzzy multi-criteria approach for software "calculating income tax." This software is used publicly and commonly in some countries. This software is used for the calculation of income tax based on various input parameters such as total income, assets, savings, etc. Table 1 shows the input parameters of software reliability.

Table 1. shows the input parameters of software reliability

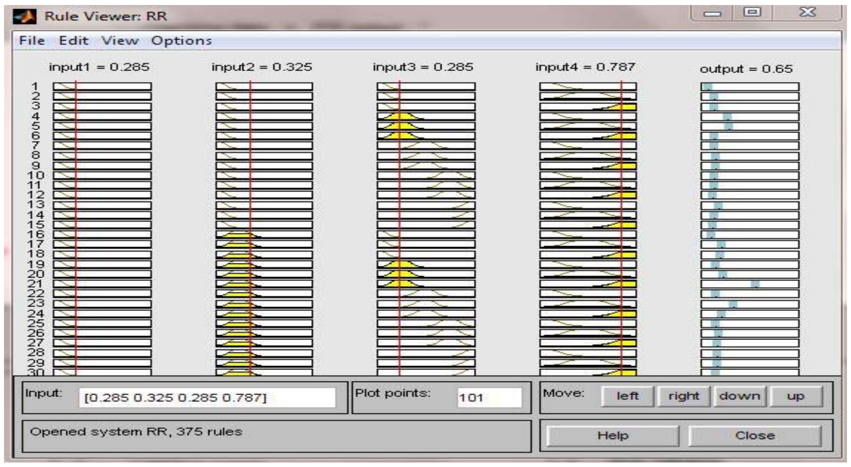
Parameters value	Parameters
0.285	Maturity
0.325	Fault tolerance
0.285	Recoverability
0.78	Reliability compliance

By applying the values of input parameters of the software reliability, the software reliability is obtained at 0.65, as shown in (Figure 8). Our proposed method is easier and more acceptable than other methods to evaluate software reliability. Because the values of the weight of the input parameters have been set by experts and software developers, and also computations are performed with tools of MATLAB instead of manual calculations.

The rating of reliability is another metric that is computed using Equation (9) [8]. Where the symbol 'r' denotes the rate, and 'w' denotes the weight.

$$r_{\text{reliability}} = r_{\text{maturity}} \times w_{\text{maturity}} + r_{\text{fault-tolerance}} \times w_{\text{fault-tolerance}} + r_{\text{recoverabilitiy}} \times w_{\text{recoverabilitiy}} + r_{\text{Reliability compliance}} \times w_{\text{Reliability compliance}} \quad (9)$$

The simulation results are tabulated in Table 2. Simulation results confirm that the proposed model is more exact than the previous model presented in [8] and the fuzzy method. In fact, the absolute value of the difference between the proposed model and the value computed using Equation (9) is 0.0096 while the difference value for previous work is 0.1396 respectively. In other words, the proposed model is more exact compared to the previous work.



Input: [0.285 0.325 0.285 0.787]

Opened system RR, 375 rules

Figure 8. inference for input (0.78,0.285,0.325,0.285)

Table 2. Simulation results for software reliability

Models	Previous Model[8]	Proposed Model(neuro-fuzzy)	Reliability Results using Eq. (9)
Reliability Value	0.78	0.65	0.6404

5. Conclusions

This paper presented a novel model to quantify the Software Quality Parameters using the neuro-fuzzy approach. The case study used in this paper to evaluate the proposed method is the income tax calculator software. By applying the input parameters, I observe that the software reliability is 0.65. software reliability in our proposed method is more exact than software reliability in the fuzzy multi-criteria and fuzzy method because Experts and software developers have set the weights of the input parameters, and simulations are carried out using MATLAB tool (ANFIS). Simulation results confirmed that the proposed model is more accurate than previous ones. This work may be extend in the future as follows:

- Considering some more factors to quantify the software reliability more exactly.
- Using Artificial Intelligence (AI), Neural Networks (NNs), and Genetic Algorithm(GA) more extensions can be done.

6. Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

7. Acknowledgment

I am grateful to all those who helped me to improve the quality of the paper with their valuable suggestions.

8. References

[1] R. Dadhich and B. Mathur, "BM, Measuring Reliability of an Aspect Oriented Software Using Fuzzy Logic Approach," *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 1, no. 5, pp. 233-237, 2012.

[2] P. Gandhi, M. Z. Khan, R. K. Sharma, O. H. Alhazmi, S. Bhatia, and C. Chakraborty, "Software Reliability Assessment Using Hybrid Neuro-Fuzzy Model," *Comput. Syst. Sci. Eng.,* vol. 41, no. 3, pp. 891-902, 2022.

[3] J. Vieira, F. M. Dias, and A. Mota, "Neuro-fuzzy systems: a survey," *in 5th WSEAS NNA international conference on neural networks and applications, Udine, Italia*, pp. 1-6, 2004.

[4] R. Kruse, "Fuzzy neural network," *Scholarpedia,* vol. 3, no. 11, p. 6043, 2008.

[5] W. A. Adnan and M. H. Yaacob, "An integrated neural-fuzzy system of software reliability prediction," *in Proceedings of 1st International Conference on Software Testing, Reliability and Quality Assurance (STRQA'94)*, IEEE, pp. 154-158, 1994.

[6] S. Aljahdali and A. F. Sheta, "Predicting the reliability of software systems using fuzzy logic," *in Eighth International Conference on Information Technology: New Generations (ITNG)*, IEEE, pp. 36-40, 2011.

[7] O. Georgieva and A. Dimov, "Software reliability assessment via fuzzy logic model," *in Proceedings of the 12th International Conference on Computer Systems and Technologies*, pp. 653-658, 2011.

[8] J. S. Challa, A. Paul, Y. Dada, V. Nerella, P. R. Srivastava, and A. P. Singh, "Integrated software quality evaluation: a fuzzy multi-criteria approach," *Journal of Information Processing Systems*, vol. 7, no. 3, pp. 473-518, 2011.

[9] B. Kotaiah and R. Khan, "Software Reliability Assessment by using Neural Networks with Fuzzy Logic based systems‖," *in International Conference on Advances in Computer Science (AET-ACS)*, Elsevier,2013.

[10] B. Kotaiah, M. Prasad, and R. Khan, "An analysis of software reliability assessment with neuro-fuzzy based expert systems," *Procedia Computer Science*, vol. 62, pp. 92-98, 2015.

[11] M. Zavvar and F. Ramezani, "A method based on fuzzy system for assessing the reliability of software based aspects," *Advances in Science and Technology. Research Journal*, vol. 9, no. 27, 2015.

[12] M. Milovancevic, A. Dimov, K. B. Spasov, L. Vračar, and M. Planić, "Neuro-Fuzzy Evaluation of the Software Reliability Models by Adaptive Neuro Fuzzy Inference System," *Journal of Electronic Testing,* vol. 37, no. 4, pp. 439-452, 2021.

[13] E. Elghamry, M. A. Eldamcesse, and M. S. Nayel, "Reliability Model of Fuzzy Consecutive k-out-of-n: F System," *International Journal of Reliability, Risk and Safety: Theory and Application,* vol. 2, no. 1, pp. 1-4, 2019.

[14] Coallier, F. "Software engineering–Product quality–Part 1: Quality model." *International Organization for Standardization: Geneva, Switzerland*, 2001.

[15] P. C. Nayak, K. Sudheer, D. Rangan, and K. Ramasastri, "A neuro-fuzzy computing technique for modeling hydrological time series," *Journal of Hydrology*, vol. 291, no. 1-2, pp. 52-66, 2004.

[16] M. Zounemat-Kermani and M. Teshnehlab, "Using adaptive neuro-fuzzy inference system for hydrological time series prediction," *Applied soft computing*, vol. 8, no. 2, pp. 928-936, 2008.