INTERNATIONAL JOURNAL ON ORANGE TECHNOLOGIES
www.journalsresearchparks.org/index.php/IJOT e- ISSN: 2615-8140|p-ISSN: 2615-7071
Volume: 03 Issue: 05 | May 2021

Adaptive neuro - fuzzy model for assessing the reliability of component software systems

O'rinov Nodirbek Toxirjonovich,
Teacher, Department of Information Technology,
Andijan State University
E-mail:nodirbekurinov1@gmail.com
Nabiyev Sherzodbek Nurmuhammad o'g'li
Teacher, Department of Computer Engineering,
Andijan State University
E-mail: sherzodbekn1994@gmail.com

----------***----------

ABSTRACT

Although many algorithms and methods have been developed for assessing the reliability of component software systems (CBSS), much more research is needed. An accurate assessment of the reliability of CBSS is difficult because it depends on two factors: the reliability of the components and the reliability of the glue code. Moreover, reliability is a real-world phenomenon associated with many real-time problems. Soft computing techniques can help solve problems for which solutions are uncertain or unpredictable. A number of soft computing approaches have been proposed to assess the reliability of CBSS. These methods learn from the past and capture existing patterns in the data. The two main elements of soft computing are neural networks and fuzzy logic. In this article, we propose a model for assessing the reliability of CBSS, known as Adaptive Neural Fuzzy Inference System (ANFIS), which builds on these two basic elements of soft computing, and compare its performance to that of a simple FIS (Fuzzy Inference System) for different datasets.

Key words: neuro-fuzziness; Component Software Systems (CBSS); Fuzzy; Reliability; Reliability model;

1. Introduction

There are usually two user requirements for software: reliability and availability. Reliability is required when product inefficiencies will have the greatest impact, and availability is required when downtime will have the greatest impact. While it is formally difficult to define reliability, we cannot simply define it as a binary notion by saying that if a program is correct, its reliability is 1, and if it is wrong, its reliability is 0. Instead, reliability can usually be measured probabilistically as

$$R_{SYS} = (1 - \text{probability of failure})$$

In particular, software reliability is defined as the probability that a software system will not fail over a specified period of time in a specified environment. As the complexity of software applications continues to grow, there is an increasing emphasis on reuse. Thus, CBSS-based applications have emerged.

Component Software Development (CBSE) is a specialized form of software reuse associated with the creation of software from existing components, including commercial off-the-shelf (COTS) components, by assembling them for interoperability. Building a highly reliable software application is difficult, even when high quality, pre-tested and validated software components are combined. Therefore, several

methods for analyzing the reliability of component applications have appeared. They are divided into two groups:

- System-level reliability assessment: The reliability of the application as a whole is assessed.
- Component-based reliability assessment: Application reliability is assessed based on the reliability of individual components and their interactions.

Traditional approaches to software reliability analysis consider the software as a whole and use test data at the software testing stage to simulate only the interaction of the software with the outside world. These are known as black box models. However, black box models ignore the structure of software built from components as well as the reliability of individual components and are therefore not suitable for modeling CBSS applications. Recently, soft computing methods have appeared. Since reliability is a real issue, many runtime parameters are related to reliability. This makes soft computing methods ideal for assessing the reliability of CBSS, as these methods mainly deal with uncertainty. The two main methods of soft computing are neural networks and fuzzy logic. In this article, we combine these two methods to assess the reliability of CBSS.

This article presents an adaptive neuro- fuzzy inference system (ANFIS) model for assessing the reliability of CBSS. In a fuzzy inference system (FIS), if-then rules are formulated based on expert advice. However, this is a relatively laborious process. The advantage of ANFIS over the FIS model is that it combines fuzzy logic with neural network (NN) training capabilities to solve this problem. The goal of our proposed model is to integrate the best characteristics of fuzzy logic and neural networks into the CBSS reliability assessment model. Our results demonstrate how much of an improvement this is compared to FIS.

The rest of this article is organized as follows. Section 2 discusses related work. Section 3 presents the structure for our proposed model, and Section 4 presents the model. Section 5 presents our experiments to evaluate the proposed model. Section 6 discusses the experimental results, and Section 7 presents our findings.

2. Related work

Researchers have proposed a number of models for assessing the reliability of CBSS.

We can divide the approaches into three types:

- Architecture-based reliability models.
- Mathematical models for assessing the reliability of CBSS.
- Soft computing techniques for assessing the reliability of CBSS.

Architecture-based reliability models, such as state-based and path-based models, are most appropriate for CBSS (Popostojanova and Tvedi , 2001; Cai et al ., 2003; Gokhle , 2007). The reliability of CBSS depends not only on the architecture, but also on the working profile for the login. Heiko and Koziolek (Koziolek and Becker , 2005) describe the role of the component as a converter from one working profile to another. In government-based models (Cheung , 1980; Wang et al , 2006; Gokhle . Et al , 1998; Littlewood , 1979), flow control between components is taken into account. Components are assumed to fail independently. These models treat

passing between components as Markov behavior, which means that the component's current behavior is independent of its previous behavior. These models can be represented in two ways, namely as composite models or as hierarchical models. Path-based models (Shooman,1976; Krishnamurthy and Mathur,1997) look at possible execution paths to assess the reliability of an application. A sequence of components along different paths is obtained either by algorithmic or experimental testing.

Jakub etal. (2004) propose an approach to reliability analysis called scenario-based reliability analysis. This approach introduces component dependency graphs (CDGs) that can be extended to complex distributed systems. Using the same algorithm, sensitivity can be analyzed as a function of component reliability and communication reliability. This approach is based on scenarios that can be captured using sequence diagrams, which means the approach can be automated. A limitation of this approach is that it does not account for failure dependencies between components. Zhang et al . (2008) also proposed a model based on CDG. With this approach, a working profile of the system is given. It is assumed that the control flow passes from component *i* to component *j*, and the reliability of component *j* is calculated as $T_{ij} = R_{ij} \times W_{ij}$, where

- T_{ij} = probability of transition from component *i* to component *j*,
- R_{ij} = vector of reliability for each subdomain of component j, and
- W_{ij} = weight vector for each subdomain of component *j* when passing from component *i* to component *j*.

This approach can characterize the robustness of the components for an application in response to changes in the system's operating profile.

Dong et al. (2008) proposed a new model for assessing the reliability of CBSS, which analyzes various complex component relationships. The Markov model is used to address these complex relationships, which have a large impact on the reliability of the system. The results were used to develop a new tool for calculating the reliability of software applications. Huang et al. (2008) proposed an algebra-based method that provides a framework for describing the syntax and predicting the reliability of CBSS, implemented by Seth et al . (2010) proposed a minimum spanning tree approach to assess the reliability of CBSS.

Goswami and Acharya (2009) proposed an approach to CBSS reliability analysis that takes into account component utilization, which is the time allotted for a component to run outside of the overall application runtime. Mathematical formulas are used to calculate the utilization rate. Due to the flexibility of component utilization, this approach can be used in real-time applications.

Fiondella et al . (2013) suggested approach based on correlated failures of components (COCOF). This article proposes an effective approach to assessing the reliability of a software application by considering component reliability, correlation, and application architecture. The proposed approach is based on an algorithm that converts the multivariate Bernoulli distribution (MVB) into a joint distribution of component outcomes.

Palviainen et al . (2011) proposed a method for evaluating the reliability, predicting and measuring CBS. The proposed approach

explains that reliability is a key factor for critical safety systems such as healthcare systems and traffic controllers. This document provides an assessment of the reliability of software at the design and implementation stages. The contribution of this approach lies in the integration of reliability at the component level. There is a need for a systematic approach that helps software developers both build robust component-based software systems and ensure that the software architecture, the selected components, and the built software system are consistent with reliability goals. The approach combines reliability heuristics, model-based reliability prediction and component-level model-based reliability measurements and system-level reliability prediction activities to support the gradual and iterative development of robust CBSS. The proposed approach is not applicable to distributed systems.

Hu et al . (2013) proposed a software reliability assessment method using modified adaptive testing (MAT) to test the reliability of CBS. In this article, a set of advanced metrics, based on the Nelson Software Reliability Model, takes into account information from the user's perspective regarding the severity of observed failures. In this approach, the use of test history information allows the resulting testing process to be adaptive in selecting tests within a limited testing budget. This approach can improve software reliability assessment testing by guiding the test case selection process, providing more descriptive and accurate results.

Dimov and Sasikumar (2010) proposed a fuzzy reliability model for CBSS based on fuzzy logic and possibility theory. A mathematical fuzzy model based on necessity and capability is proposed to predict the reliability of the CBSS. Like many other models, this model does not require component failure data as it is based on uncertainty. However, a mechanism is needed to model the propagation of failure between components and the behavior of the failure.

Si et al . (2011) proposed a scheme for assessing reliability using a component composition mechanism. The approach offers five main mechanisms for composing components and methods for assessing their reliability. After calculating the reliability for each composition, the procedure evaluates the overall reliability of the application based on the component composition mechanisms and the frequency of component usage. Additional composition mechanisms can be recognized.

Many approaches to software computing are based on support vector machines (SVMs), genetic algorithms (GA), and fuzzy logic. Lo (2010) proposed a software reliability assessment model based on SVM and GA. This model indicates that recent failure data alone is sufficient to assess reliability. The parameters for evaluating the reliability of SVM are determined by the GA. This model is less dependent on failure data than other models.

Hsu and Huang (2011) proposed a model that is an adaptive approach for testing path robustness estimates for complex CBSS. Three methods have been proposed to assess the reliability of a path: they use sequence, branch and loop structures. The suggested path robustness can be used to assess the robustness of the entire application. Traction and Sharma (2012) proposed a fuzzy logic approach to assess the reliability of CBSS. In this approach, four critical factors for assessing the reliability of the CBSS were identified and used to develop the FIS for the assessment.

INTERNATIONAL JOURNAL ON ORANGE TECHNOLOGIES
www.journalsresearchparks.org/index.php/IJOT e- ISSN: 2615-8140|p-ISSN: 2615-7071
Volume: 03 Issue: 05 | May 2021

3. The basis for our proposed model.

This article proposes a neuro - fuzzy reliability model. The difference between this model and other proposed models is that this model is completely based on the characteristics of the CBSS. Fuzzy logic is widely studied in various fields of technology with varying degrees of success. FIS often consists of "if-then" rules. These rules run in parallel: when the if conditions are met, the consequence of these rules is triggered to the extent that the preceding rules are met. One of the main challenges in creating a FIS is defining the fuzzy sets and fuzzy rules to be used. This requires deep domain knowledge from human experts, and fine-tuning fuzzy sets and rules can be very time-consuming. The solution to this problem is to combine FIS with NN learning capabilities, resulting in ANFIS. The proposed model is an improvement on the fuzzy model proposed by Tyagi and Sharma (2012) .

3.1. Neural networks and fuzzy logic

NN and fuzzy logic are complementary concepts. Networked networks have learning abilities: they can learn from data and feedback. These are black box models. Fuzzy logic models are rule-based models where rules are usually in the form of "if-then". Fuzzy models do not have the learning opportunities and learning they need to make techniques of the other methods. Therefore, the combination of the two technologies is natural. Fuzzy logic is based on a combination of the four concepts shown in Figure 1 .

3.2. ANFIS

ANFIS was first proposed by Jang (1992) in 1992. They are adaptive networks with functions similar to FIS. The goal of ANFIS is to combine the best capabilities of fuzzy systems and networked networks. The advantages of ANFIS over FIS are as follows:

- ANFIS can model and analyze the mapping relationship between input and output using a learning algorithm to optimize the parameters of a given FIS.
- FIS is completely dependent on its membership functions. ANFIS includes networks that combine nodes and directed links, and some learning rules are associated with these networks. Networks study the relationship between inputs and outputs.
- ANFIS can be trained without any of the expertise usually required for standard fuzzy logic design.

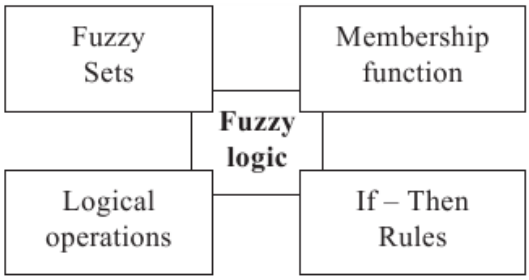

Figure 1. Operation of a fuzzy system.

4. Proposed model

In this article, we propose an automated CBSS reliability assessment model that uses the ANFIS approach. ANFIS is a class of adaptive networks that are functionally equivalent to FIS. This is an extension of the fuzzy model proposed by Tyagi and Sharma (2012) . We take into account all four factors proposed in this model to compare the results of both models.

The four factors suggested by Tyagi and Sharma (2012) are as follows. There

are two main factors in assessing the reliability of a component: the reusability of a component and its performance profile. __

4.1. Component reusability

Reusability is one of the most basic concepts in component development (CBD). As the name suggests, reusability refers to how often a component is used across different applications. We chose reusability as a factor in assessing component reliability because components that have been used in many applications are believed to be highly reliable. Therefore, the reliability of a component is directly proportional to its reusability: the reliability of a component is its reusability.

We selected components from www.ejbeans.com; Recycling measures for these components are listed on the website. Sharma et al . (2009) proposed another method for calculating reusability using artificial neural networks.)

Operational Profile for a Component: The Operational Profile (OP), a quantitative measure of how software will be used, is essential in any software reliability engineering application. OP is a complete set of operations with an indication of the probabilities of their occurrence. The reliability can be different for different OPs. The OP for any component describes the input to be fed into the component.

Other factors can affect the reliability of the component, but these are the two factors that have the greatest impact on the reliability of the component. Since the components are platform independent, the measurements of these two factors will be the same for all applications.

Two main factors are used to assess the reliability of an interface:

Component Dependency: In CBSS, various components are connected together to form a larger application. Components depend on each other to perform their functions, that is, the output of one component can serve as an input for another component. This dependence plays an important role in assessing the reliability of the entire application. If the components are highly dependent on each other, the reliability of the system will be low.

There are several mechanisms for characterizing dependencies between components. For example, representing dependencies as an adjacency matrix indicates what dependencies exist between components. However, this view does not explain the interactions between components. These interactions play an important role in terms of component dependencies and therefore their reliability, so we need a different way to represent interactions. Sharma et al . (2009) proposed a dependency view that is based on linked lists and implemented using a Hash Map in Java . This view can store dependencies along with other information such as provided and requested information that can be used to analyze multiple interoperability and dependency issues.

Application Complexity: The complexity of any CBSS application can be defined in terms of the number of components in that application and how they are interconnected. An application with a lot of components is considered more complex and therefore less reliable. Therefore, the complexity of an application is inversely proportional to its reliability:

$$\text{Application complexity} \propto \frac{1}{reliability}$$

The complexity of an application can be measured by the number of components in the application. Where N is the number of components in the CBSS, we define the complexity of the AC application as

AC ∞ N:

Our proposed model is additive. It does not explicitly take into account the CBSS architecture, but the measure of the complexity of the application will depend on the platform because it comes from the glue code (integration code).

The FIS studied in this paper has four input factors as described above, namely reusability, application complexity, component dependency, and operational profile. For the first-order fuzzy Sugeno model, the set of rules with fuzzy if-then rules looks like this:

Assumptions:

(1) R = reusable
(2) CD = component dependency
(3) AC = Application Complexity
(4) OP = work profile

Rule *1*:

If R is $_i$ and AC is B $_i$ and compact is C $_i$ and OP is D $_i$, then

f = p_i R + q_i AC + r_i CD + n_i OP + m_i, where A_i, B_i, C_i and D_i are fuzzy sets, and p_i, q_i, r_i, n_i, and m_i are parameters that are determined during training.

In total, there are 81 rules in which *low* (L), *medium* (M) and *high* (H) are linguistic variables.

Nodes in the same layer have similar functions. ANFIS has the following five levels:

Level 1: Each node at this level is an adaptive node with a node function. $O_{1,i}$ - degree of membership of a fuzzy set. The parameters for this layer are known as *room* parameters.

$O_{1,i}$ = μ_A_i (R) for i = 1,2,3

where μA_i is the membership function for R.

Level 2: Each node at this level is a fixed node labeled Π whose output is the product of all incoming signals.

$O_{2,i} = w = \mu\_A\_i(R)\mu\_B\_i(AC)\mu\_C\_i(CD)\mu\_D\_i(OP)$ for $i = 1,2,3$

The output of each node represents the strength of the rule triggering.

Level 3: Each node at this level is a fixed node labeled N. The *i*th node calculates the ratio of the triggering force of the *i*th rule to the sum of all the triggering powers of the rule. Output of this layer is called *a normalized firing strength*.

Level 4: Each node at this level is an adaptive node with a node function.

$O_{4,i} = w_i f_i = w_i(p_i R + q_i AC + r_i CD + n_i OP + m_i)$

INTERNATIONAL JOURNAL ON ORANGE TECHNOLOGIES
www.journalsresearchparks.org/index.php/IJOT e- ISSN: 2615-8140|p-ISSN: 2615-7071
Volume: 03 Issue: 05 | May 2021

The parameters of this node are called *sequential* parameters.

Level 5: The only node at this level - a fixed node labeled, which computes the overall output signal as the sum of all the incoming signals (see. Figure 2.).

Total output = $O_{5,1} = \sum w_i f_i / \sum w_i$

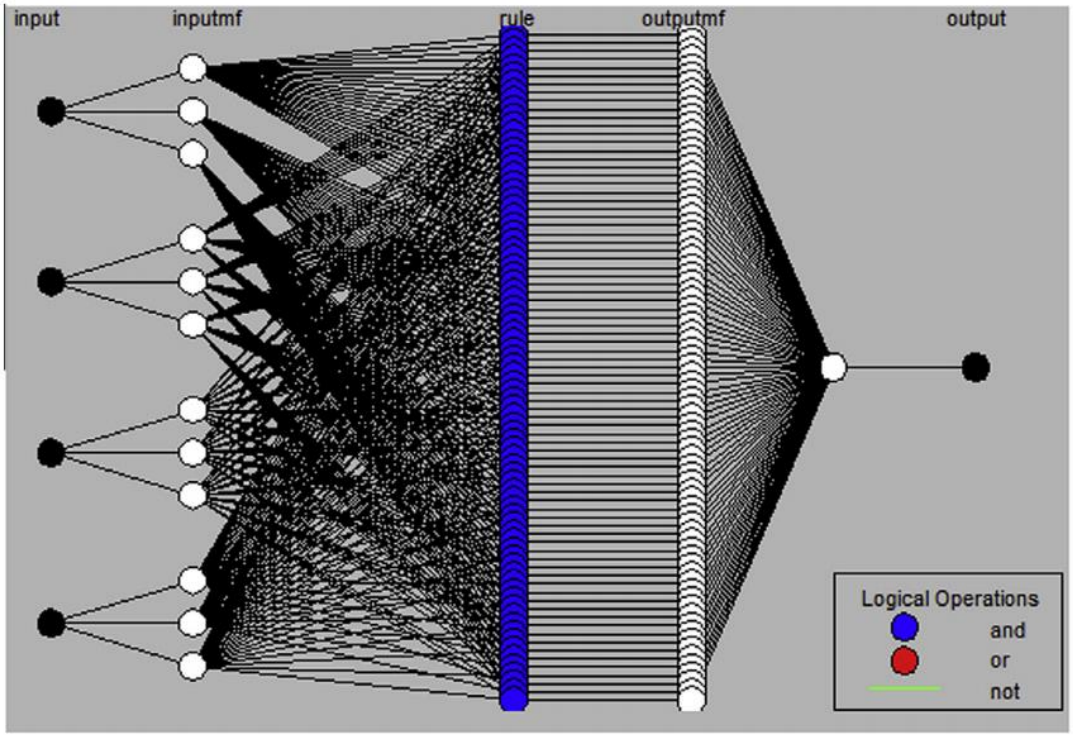

Figure 2. Proposed system architecture.

Adaptive Neural Fuzzy Model for Evaluation

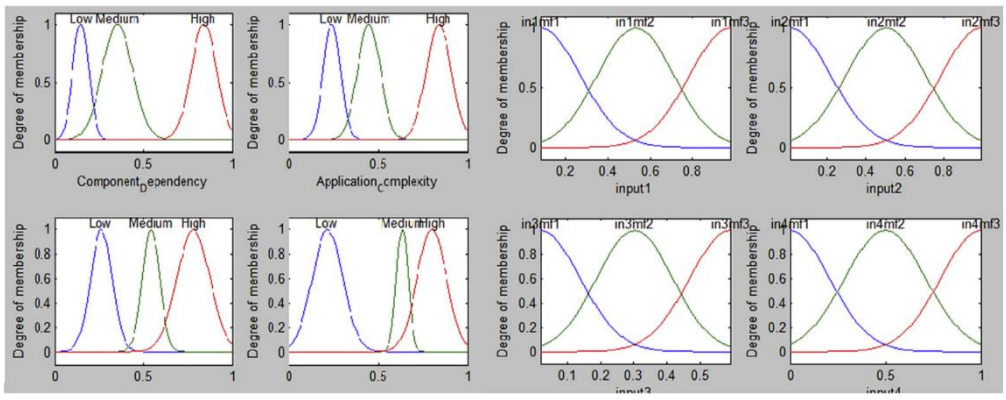

Figure 3. Membership function before and after training.

One of the main limitations of this ANFIS is that it only uses Sugeno style fuzzy inference systems. The difference between the mechanisms of withdrawal of Sugeno and Mamdani is the method of defuzzification. FIS type Mamdani using a method of defuzzification for fuzzy output, and FIS type Sugeno uses the weighted average value to calculate accurate output data. Mamdani FIS also has output membership functions, while Sugeno FIS has no output membership functions.

5. Experiments and model evaluation.

We have developed our ANFIS for Sugeno FIS. For ANFIS training, we collected data from 47 classroom projects. Some of these projects are complex CBSS sand.

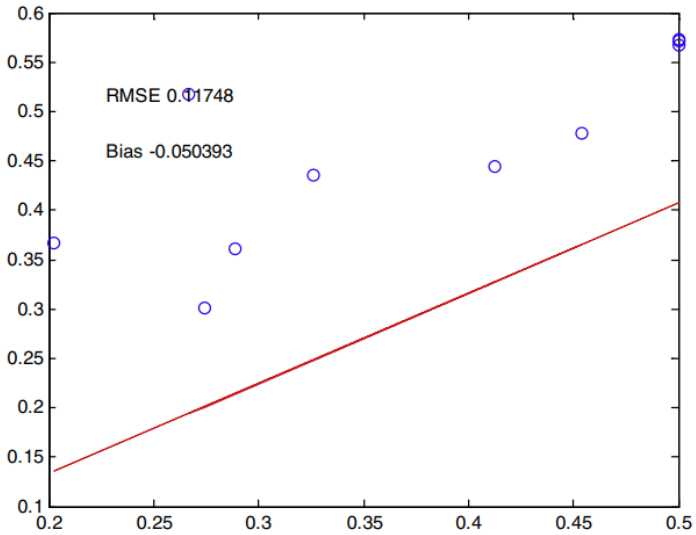

Figure 4. FIS root mean square error.

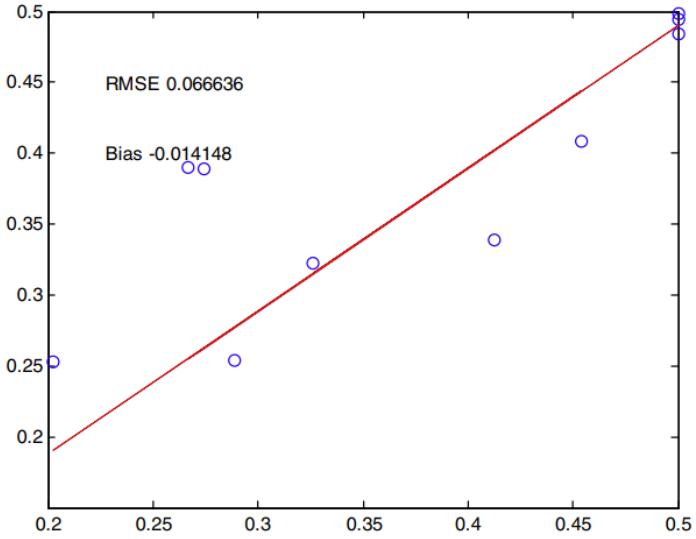

Figure 5. Root mean square error of ANFIS.

Table 1 Performance analysis of FIS and ANFIS.

Inputs				Output reliability		
CD	AC	*R*	OP	Original	FIS	ANFIS
.0202	.0202	.0202	.0202	.5000	.5667	.4837
.2424	.2424	.2424	.2424	.3263	.4351	.3228
.2323	.2323	.2323	.2323	.4127	.4438	.3390
.3131	.3131	.3131	.3131	.2025	.3672	.2526
.3434	.3434	.3434	.3434	.2891	.3604	.2541
.6970	.6970	.6970	.6970	.4543	.4772	.4079
.6667	.6667	.6667	.6667	.2667	.5178	.3896
.6465	.6465	.6465	.6465	.2740	.3013	.3890
.9798	.9798	.9798	.9798	.5000	.5734	.4942
.9293	.9293	.9293	.9293	.5000	.5710	.4984

some of them are simple CBSS. ANFIS has been trained and tested using data from these projects. Some data was used for training and testing, and some for results. A total of 81 rules were formed based on the available data. The rules divide the input factors into three variables: low, medium, and high.

To evaluate our proposed model, a training dataset was loaded and ANFIS was trained using this dataset. To validate the model, we loaded the testing dataset and then plotted the testing error. The FIS membership function before and after training is shown in Figure 3 .

Adaptive Neural Fuzzy Model for Evaluation

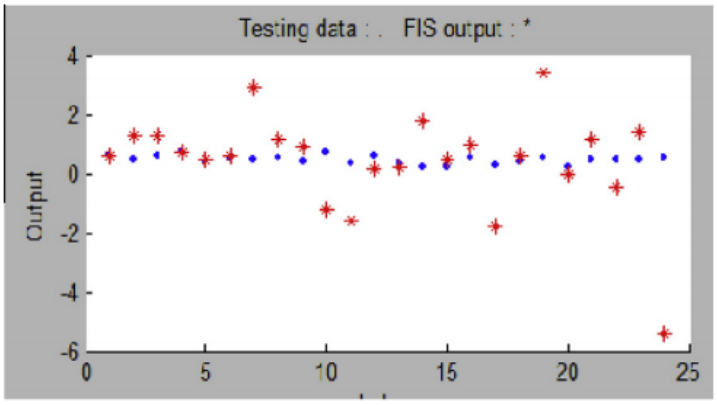

Figure 6. Error testing sugeno FIS mapping to ANFIS.

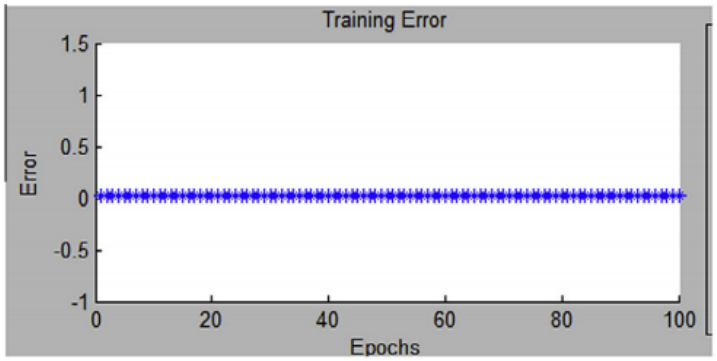

Figure 7. Displaying sugeno FIS learning errors in ANFIS.

6. Results and discussion.

After creating the ANFIS model, we compared the output reliability values for different input sets with the original values. We have calculated the root mean square error (RMSE) for the output from the FIS and the output from the ANFIS with the original output. Membership levels have undergone significant changes (see Figures 4 and 5).

You can see the difference between the two RMSEs. Using only FIS, the error in the results was 11.74%, but ANFIS reduces the error to 6.66%. Hence ANFIS performs better than FIS. Using ANFIS, we first trained FIS and rules were formed from the training data to produce the output of the trained model. The only limitation of this model is the complexity of its implementation for large datasets. Our results show that the ANFIS model provides a more accurate measure of reliability than the FIS model (Table 1).

Testing error and learning error obtained from MATLAB FIS TOOL-BOX are shown in Fig.6 and 7.

7. Conclusion and further work

Any software system has two user requirements: reliability and availability. Reliability is required when product performance will have the greatest impact. This article proposes a neuro - fuzzy model for assessing the reliability of CBSS. Many approaches to the reliability of CBSS have been proposed, some based on mathematical formulas and others based on soft computing techniques. Our proposed method is based on ANFIS, which is a soft computing method. It is a hybrid method that requires less computational time than traditional approaches and the previously proposed FIS approach. The CBSS used to test the models are cool projects. Our results show that ANFIS improves the reliability score of FIS vehicles.

A limitation of our proposed model is that although the four factors for which the model was implemented are the most important factors, there may be other relevant factors that should be used. One such factor is the failure rate, but we currently only have data for four factors, so adding this factor is left for future work.

References

[1.] Tsai , K., Bai, K., Zhong , X., 2003. An Introduction to Reliability Models of a Component Software System. J. Xi'an Jiaotong Univ . 37 (6), 560-564.

[2.] Cheung, RC, 1980. A User-Centered Software Reliability Model. IEEE Trans . Softtv . English. 6 (2), 118-125 . Dimov, Aleksandar, Sasikumar, Punnekkat, 2010. Fuzzy Reliability Model for Component Software Systems, 36th EUROMICRO Conference on Software Engineering and Advanced Applications, pp. 39–46. Dong, W., Huang, N., Min, Y., 2008. Reliability Analysis of Component Software Based on Component Interconnection, IEEE Web Services Conference, pp. 814–815.

[3.] Fiondella, Lance, Rajasekaran, Sangutever, Gokhale , Swapana 2013. Effective software reliability analysis with correlated component failures. IEEE Trans . Reliable . 62 (1), 244-255.

[4.] Gokhle SS, 2007. Analysis of software reliability based on architecture: overview and limitations. IEEE Trans . Reliable secure computing . 4 (1), 32-40.

[5.] Gohle, SS, Dong, VE, Trivedi, KS, Horgan, JR, 1998. An Analytical Approach to Predicting Software Reliability Based on Architecture, Proc. Third International Symposium on Computer Performance and Reliability, Durham, NC, pp. 13-22.

[6.] Goswami V., Acharya Y.B., 2009. Method for assessing the reliability of software systems based on COTS components, International Symposium on Software Reliability Design.

[7.] Xu , K., Huang , K., 2011. Adaptive Reliability Analysis Using Path Testing for Complex Component-Based Software Systems . IEEE Trans . Reliable . 60 (1), 158-170 . ____

[8.] Hai Hu, Chang- Hai Jiang, Kai-Yuan Tsai , W. Eric Wong, Aditya P. Mathur, 2013. Improving Software Reliability Estimates Using Modified Adaptive Testing, Elsevier Information Journal and Software Technologies , p. 288 - 300.

[9.] Huang, N., Wang, D., Jia , X., 2008. QUICK ABSTRACT: An Algebra-Based Approach to Reliability Prediction for Composite Web Services, 19th International Symposium on Software Reliability Engineering, p. 285-286.

[10.] Jung, JR, 1992. ANFIS: A Fuzzy Inference System Based on Adaptive Networks, IEEE Trans. Systems, man and cybernetics.

[11.] Koziolek H., Becker S., 2005. Transforming Work Profiles of Software Components to Predict Service Quality, Proc.10th WCOP'05,pp.1-8.

[12.] Krishnamurti , S. Mathur , AP, 1997. On an estimate of the reliability of the software system with the reliability of its components and materials of the eighth International Symposium on Software Reliability, Albuquerque, New Mexico, pp. 146- 155

[13.] Littlewood , B., 1979. A software reliability model for a modular program structure. IEEE Trans . Reliable . 28 (3), 241- 246.

[14.] Lo, J., 2010. Early Prediction of Software Reliability Based on Support Vector Machines with Genetic Algorithms, Fifth IEEE Industrial Electronics and Applications Conference, pp. 2221–2226.

[15.] Palviainen , Markov, Evesti , Antti , Ovaska , Eila 2011. Reliability Assessment, Prediction and Measurement of Component Software. J. Syst . Softtv ., 1054-1070.

[16.] Popostyanova , KG, Trivedi , KS, 2001. An architectural approach to assessing the reliability of software systems. Execute. Eval . J. 45 (2), 179-204.

[17.] Seth, K., Sharma, A., Seth, A., 2010. A minimal spanning tree approach for assessing the reliability of COTS- based software applications . IUP J. Comput . Sci . 4 (4), 13–21.

[18.] Sharma, A., Grover , PS, Kumar , R., 2009. Dependency Analysis for Component Software
Systems. ACMSIGSOFT Softw . English. Notes 34 (4), 1-6.

[19.] Shooman, M., 1976. Structural Models for Predicting Software Reliability, Proceedings of the Second International Conference on Software Engineering, San Francisco, CA, pp. 268–280.

[20.] Xi Yuanjie, Xiaohu Yang, Xinyu Wang, Chao Huang, Alexander J. Kavs, 2011. An Architecture-Oriented Framework for Reliability Assessment Using Component Composition Mechanisms, 2nd International Conference on Computers, Engineering and Technology, p. 165 - 170.

[21.] Tyagi , K., Sharma, A., 2012. A rule-based approach for assessing the reliability of component systems. Adv . English. Softtv . 54, 24-29.

[22.] Wang, W., Pan, D., Chen , M.H., 2006. Architecture-Based Software Reliability Modeling. J. Syst . Softtv . 79 (1), 132-146.

[23.] Yakub,S., Chukich,B., Ammar,H., 2004. A scenario-based approach to reliability analysis for component systems. IEEE Trans. Reliable. 53 (4), 465-480.

[24.] Zhang, F., Zhou, X., Chen, J., Dong, Y., 2008. A New Model for Component Analysis of Software Reliability, 11th IEEE High Assurance Systems Engineering Symp o\sium, pp. 303– 309.