Contents list available at JMCS

Journal of Mathematics and Computer Science

Journal Homepage: www.tjmcs.com

Aspect-Oriented Software Maintainability Assessment Using Adaptive Neuro Fuzzy Inference System (ANFIS)

Hossein Momeni
Computer Engineering Department, Golestan University, Gorgan, Iran
h.momeni@gu.ac.ir
Shiva Zahedian
Mazandaran University of Sciences and Technology, Mazandaran, Iran
zahedian@ustmb.ac.ir

Article history:
Received May 2014
Accepted July 2014
Available online August 2014

Abstract

Aspect-oriented development is a relatively new approach that emphasizes dealing with crosscutting concerns. In aspect-oriented programming, concern networks and requirement networks are independent and can easily be added to or removed from a model of system; therefore maintenance and modifying in aspect-oriented system models are easier than object-oriented ones. Software maintenance is an important activity in software development and one of the most expensive activities. Also, its vagueness in prediction at early stage of development makes the process more complex. Researchers and developers are working on devising various techniques/ algorithms for better prediction. The aim of the paper is to show that ANFIS can more accurately predict maintainability as compared to other models such as Fuzzy Logic. For this we selected four metrics and used them for training, testing and validation.

Keywords: Maintainability, Assessment, Fuzzy Logic, Adaptive Neuro Fuzzy Inference System (ANFIS), Aspect-Oriented, maintenance.

1. Introduction

Software maintenance is one of the most expensive activities that consume about 50–70 percent of the development cost [1]. Therefore, it is a very important activity that requires much attention. For this reason, people have attempted to find ways to minimize maintenance costs by introducing better development methodologies that can minimize the effects of change, simplify the understanding of programs, facilitate the early detection of faults, and so forth.

Software maintenance, in general, refers to the set of activities that are performed to keep a system operational, as software changes after the system has been deployed. Software maintenance begins as soon as a system has been released to users for the first time. Maintainability in a system should be considered as one of the most important quality aspect. A lot of research work has been carried out to measure the maintainability. To measure it, one need to identify the set of attributes / factors that bear on the efforts needed to make specified modifications. As we know that maintenance time of software is always greater than its development time, so it is important to measure the maintainability of software to reduce the maintenance operational time. However, it may be very difficult to measure these factors and combines them to get final value of maintainability.

The Accurate prediction of software maintainability can be useful because it helps project managers in comparing the productivity and costs among different projects; provides managers with information for more effectively planning the use of valuable resources; helps managers in taking important decision regarding staff allocation; guides about maintenance process efficiency; helps in keeping future maintenance effort under control; enables the developers to identify the determinants of software quality so that they can improve design and coding; enables the developers to identify the determinants of software quality so that they can improve design and coding; helps practitioners to improve the quality of software systems and thus optimize maintenance costs and the threshold values of various metrics which drastically affect maintainability of software can be checked and kept under control so as to achieve least maintenance cost. Software metrics can be classified as product, resource, or process metrics [2]. Different researchers have proposed different metrics [3–5, 6, 7] for assessing the impact of AOP on software quality. The proposed metrics can be classified as either concern level metrics or structural complexity metrics.

2. Related works

There are several models which have been proposed to predict the software maintainability. To increase systems functionality and systems complexity, many researchers have given different terminology and methods to evaluate maintainability but there is an inconsistency and vagueness in the terminology and methods involved with maintenance of the software applications.

Kvale et al. [8] conducted a study to show how AOP can be used to improve the maintainability of COTS-based systems. In their study, they argue that if the code for calling the COTS libraries is scattered all over the glue code, the maintenance of the system will be difficult. They showed that if the glue code is built using AOP, then the COTS-based system can be easily maintained. They used the Java Email Server in their experiments. They showed that AOP improves the changeability of a COTS-based system as the code that has to be modified is minimal in cases where AOP is used in the glue code. They used size-based metrics in their study.

Kumar et al. [9] undertook a study on the change impact in AOP systems. In their study, they used AOP systems that have been refactored from their OO versions. The systems that they used were refactored from 149 OO modules into 129 modules. For the OO versions, the module refers to the classes, and for the AOP version, module refers to classes and aspects. Their study used the metrics and tool for collecting metrics data as defined in [3]. In their study, they found out that the change impact is lower in AOP systems as compared to the OO systems. Also, they found out that if the concerns which are not crosscutting are moved to aspects, then the impact of change for these modules will be higher. They assessed the maintainability of the AOP based on the changeability of the system, hence their assessment was done at the module level.

Bandini et al., [10] considered three independent factors, namely; design complexity, maintenance task and programmer's ability to predict the maintenance performance for object-oriented systems. To measure design complexity, interaction level, interface size and operation argument complexity were chosen. Perfective and corrective maintenance were selected to represent maintenance task. Correlation analysis was performed to conclude that the selected attributes were able to predict the maintenance efforts of the systems.

Several problems were discussed by Kajkoet al. [11] related with the maintenance of the software systems and proposed two maintainability models separately for product and process. Product aspects, consisted of common characteristics, variable characteristics, maturity level, traceability characteristics and others, while in process aspects the common tasks were to manage maintainability throughout the whole software life - cycle.

Fuzzy model is used to measure the maintainability of the software system by Aggarwal et al. [4]. The authors considered four factors affecting maintainability, namely, average number of live variable, average life span of variables, average Cyclomatic complexity and the comments ratio. All inputs are classified into fuzzy sets viz. low, medium and high, while maintainability is classified as very good, good, average, poor and very poor. All the inputs and outputs are fuzzified and in total 81 rules are proposed for the model. Model is validated against the software projects developed by undergraduate engineering students. However, the model is not validated on real life complex projects.

In [12] the authors Kauret al. tries to evaluate and compare the application of different soft computing techniques like Artificial Neural Network, Fuzzy Inference System and Adaptive Neuro Fuzzy Inference System for predication of software maintenance effort. The author uses OO paradigm with OO design metric for the input to ANN, FIS and ANFIS as predictors of maintenance effort. Authors concluded that Adaptive Neuro Fuzzy Inference System is best. And hence Soft Computing Techniques can be successfully used for prediction of software maintenance effort.

3. Proposed method

The In this paper we have 4 inputs: CAE, CDA, WOM and CFA. In our proposed methods, output is the maintainability rate of aspect-based software, and output is classified in five classes. As was stated before, the output is measured in terms of four parameters: The first input parameter is CAE which shows the number of aspects containing advices triggered by the execution of operations in a given module. This is the number of inward arrows from aspects to a particular module [13].This metric is used to measure the dependence of the operation on the advices; hence a change in the advice might impact the operation [3]. A higher value of this metric for a given module means that the module is coupled with more aspects. The second is CDA which shows the number of modules affected by pointcuts and introductions in a given aspect. This measures the number of modules that are affected by an aspect. And WOM that shows the number of operations (methods or advices) in a module. This is equivalent to the weighted operations per class (WMC) of the CK metrics [3]. A class with a higher number of operations is considered to be more complex, and hence it is fault prone [3, 14]. The complexity of the operations is considered to be equal. Also, more effort is needed to test a class with a higher WOM value. A lower value of WOM is desired per module and then, CFA that shows the number of modules that have fields that are called by a given module. This metric measures the coupling between modules based on field access. A higher value of CFA implies tight coupling between the modules which indicates complexity, increase in the module being fault prone, and also decrease in the testability. A lower value of CFA is desired.

For these factors, four linguistic terms has been defined in table 1 as the following:

Table 1: fuzzy membership functions of input variables

Range of Membership function	Membership function	range	metrics
0.0- 0.27	Low	0-1	CAE CDA CFA WOM
0.25-0.52	Medium		
0.50-0.77	High		
0.75-1.00	Very High		

The membership function for each linguistic term is expressed by the following equation:

$$\mu(\mathrm{CAE}) = \mu(\mathrm{CDA}) = \mu(\mathrm{CFA}) = \mu(\mathrm{WOM}) = x$$
$$= \begin{cases} \text{Low} & \text{if} \quad 0 \le x \le 0.27 \\ \text{Medium} & \text{if} \quad 0.25 \le x \le 0.52 \\ \text{High} & \text{if} \quad 0.50 \le x \le 0.77 \\ \text{VeryHigh} & \text{if} \quad 0.75 \le x \le 1.00 \end{cases} \qquad (1)$$

The rate of maintainability of an aspect oriented is between 0 an d 1in this paper. And the member ship functions for this output variable are Very Low, Low, Medium, high and Very High. We use the Gaussian membership function for the output variable. This function can determined by two parameters (c,σ) as the following equation:

$$\mu(x) = \mathrm{gaussian}(x, c, \sigma) = e^{-1/2((x-c)/\sigma)2} \qquad (2)$$

In (2), c is the center and σ determines the width of membership function. The range of linguistic variables of output (maintainability) is shown in table 2.

Table 2: The range of linguistic variables of output

Membership function	Range
Very Low	0.00-0.23
Low	0.20-0.43
Medium	0.40-0.63
High	0.60-0.83
Very High	0.80-1.00

3.1. The proposed method

In this paper two new approaches for assessing the maintainability of aspect oriented systems are presented and then compared RMSE values obtained with fuzzy logic approach and Adaptive Neuro Fuzzy Inference System (ANFIS) together. The following shows how to calculate the RMSE.

$$\sqrt{\frac{e_1{}^2 + e_2{}^2 + \cdots + e_n{}^2}{n}} \qquad (3)$$

3.2. Fuzzy approach for evaluating the maintainability

As was stated in the previous section, the membership functions for input variables are selected of a trapezoidal membership function. These functions are shown in Figure 1.

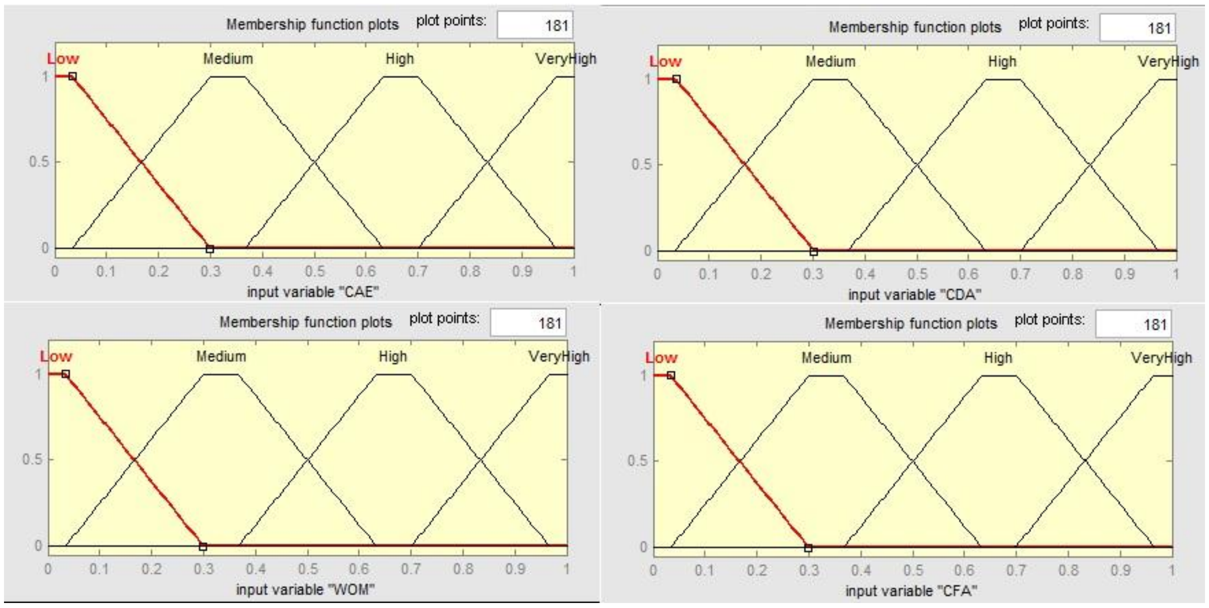

Figure 1. Membership functions for input and output parameters

The rule base of this system is shown in following:

Rule1: If (CAE is Low) and (CDA is Low) and (WOM is Low) and (CFA is Low) then (MNABTY is VeryLow)

Rule2: If (CAE is Low) and (CDA is Low) and (WOM is Low) and (CFA is Medium) then (MNABTY is Low)

Rule3: If (CAE is Low) and (CDA is Low) and (WOM is Low) and (CFA is High) then (MNABTY is Medium)

Rule4: If (CAE is Low) and (CDA is Low) and (WOM is Midium) and (CFA is Low) then (MNABTY is Low)

Rule5: If (CAE is Low) and (CDA is Low) and (WOM is Midium) and (CFA is Medium) then (MNABTY is Low)

Rule6: If (CAE is Low) and (CDA is Low) and (WOM is Midium) and (CFA is High) then (MNABTY is Medium)

Rule7: If (CAE is Low) and (CDA is Low) and (WOM is High) and (CFA is Low) then (MNABTY is Medium)

Rule8: If (CAE is Low) and (CDA is Low) and (WOM is High) and (CFA is Medium) then (MNABTYis Medium)

.

.

.

Rule80: If (CAE is Low) and (CDA is Low) and (WOM is High) and (CFA is Medium) then (MNABTY is VeryHigh)

Rule81: If (CAE is Low) and (CDA is Low) and (WOM is High) and (CFA is High) then (MNABTY is VeryHigh)

The results of the prediction and estimation are visible in Figure 2.

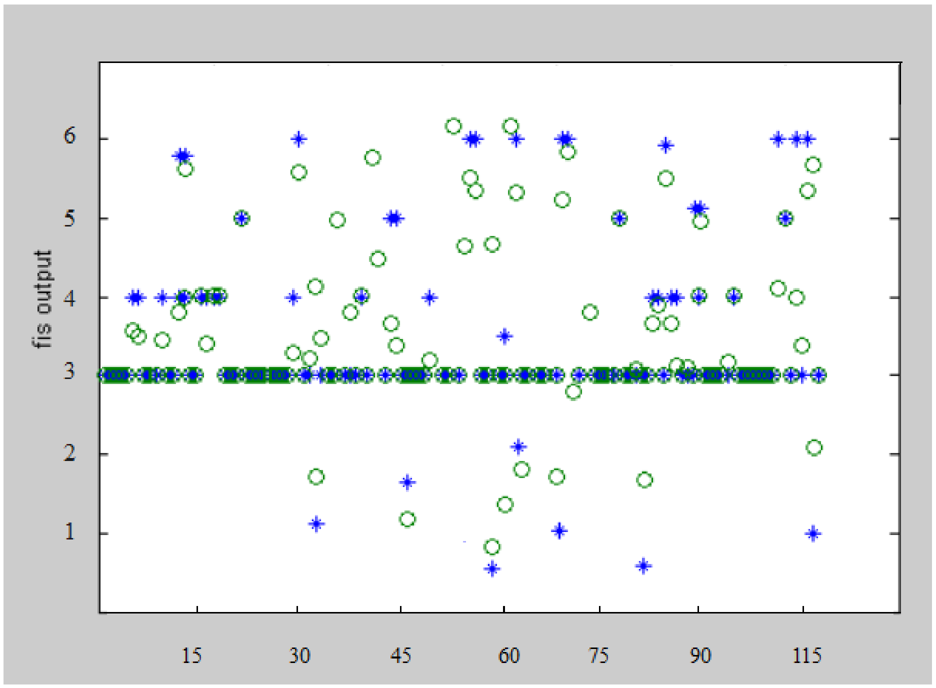

Figure 2.The results of the classifying by fuzzy system

RMSE values for this method was 0.780. As indicated in Figure 2, the classification of inputs to the specified categories (target classes) has not been carefully performed by us in the fuzzy system.

3.3. The Adaptive Neuro Fuzzy Inference System (ANFIS) for evaluation

The structure of Adaptive Neuro Fuzzy Inference System is shown in Figure 3. In this figure, the output of the nodes of the first layer is the degree of linguistic variables. Typically, bell membership functions are used in the layer.

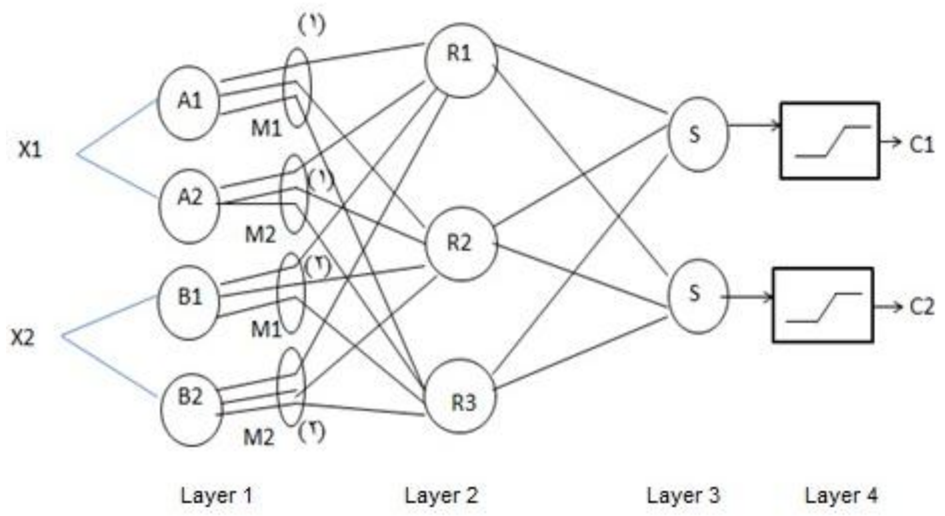

Figure 3.Structure of fuzzy artificial neural network

The structure of Bell membership function is:

$$f(x) \qquad = \exp\left[\frac{-1}{2}\left(\frac{x - xi_1}{bi_1}\right)^2\right] \qquad (4)$$

The second layer in fuzzy neural network is the law layer. The condition part of rules is calculated by the fuzzy operator of min, and the result is applied as the degree of law enforcement. Learning activities in this layer are done by changing the degree of law enforcement regarding the training data which are injected into the network. In the third layer of the linear combination, resultant rate law is used to determine the membership degree in a specific category. In the fourth layer, the Sigmund membership function is used. Network training vector is injected into the Network based on the following relationship:

$$\{(x^k, y^k)\ k = 1,2, \dots, K\} \qquad (5)$$

Here xk refers to the kth pattern in input vector, and then we will have:

$$y^k = \begin{cases} (1,0) & \text{if } x^k \text{ belongs to class 1} \\ (0,1) & \text{if } x^k \text{ belongs to class 2} \end{cases} \qquad (6)$$

Error function for pattern k is calculated from the following equation.

$$E^k = \frac{1}{2}\left[({o_1}^k - {y_1}^k)^2 + ({o_2}^k - {y_2}^k)^2\right] \qquad (7)$$

Where yk and ok are the desired output and the calculated output, respectively. In the implementation of this method, the number of Membership functions for input variable, CAE, CDA, WOM and CFA are respectively 4,4,4 and 4. Membership functions for Triangular input and for the output variables are linear. Number of training courses is considered. 20 and 126 samples of training data for the network In Figure 4, the results of the classification of training input data in ANFIS are shown.

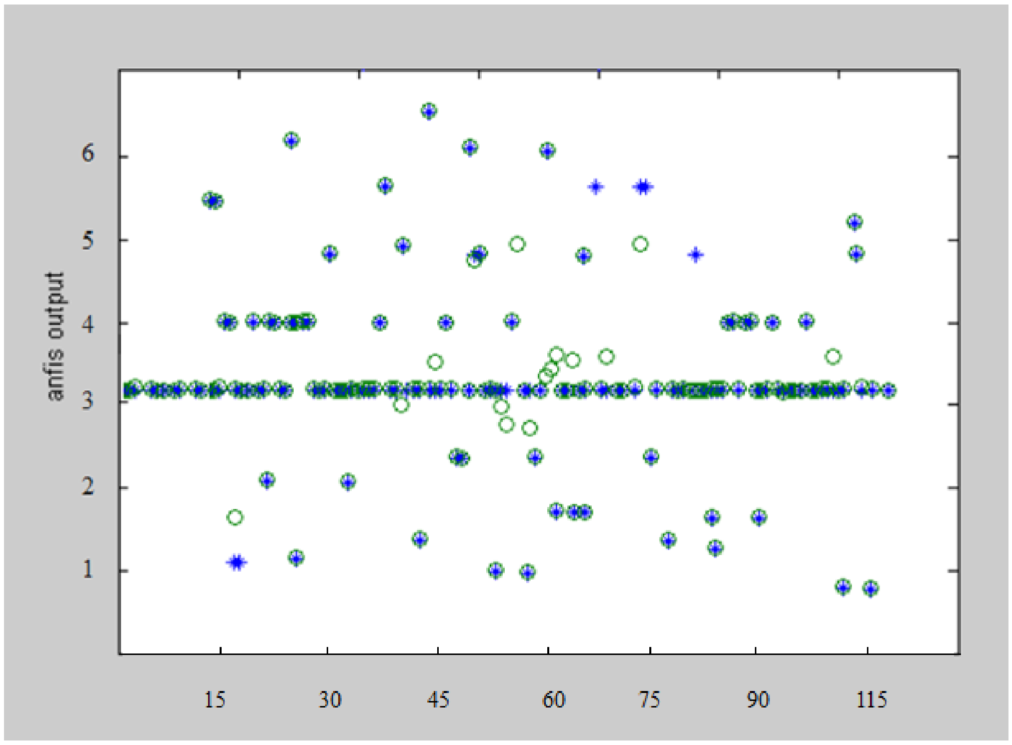

Figure 4: Results of the classification of maintainability by ANFIS system

4. Evaluation

Results of implementation and experiments under the operating system of Windows 7 with the Processor of AMD Turion (tm) X2 Dual-Core Mobile RM-75 2.20 GHz and 4 GB of RAM memory in MATLAB software, version 7.8.1 indicated that the fuzzy approach had a value of RMSE equal to 0.780 , and this means that this method of classification and estimation of the maintainability of aspect-oriented software has been weaker than the adaptive fuzzy neural network-based methods and also its prediction error is greater. For the adaptive fuzzy neural network approach after 20 network training courses, the calculated RMSE was 0.210 which is more efficient than the value calculated for the fuzzy method and it means that the estimation of this method for the maintainability of aspect-oriented software code is better and more accurate. Figure 5 shows RMSE values for each transition of the networks.

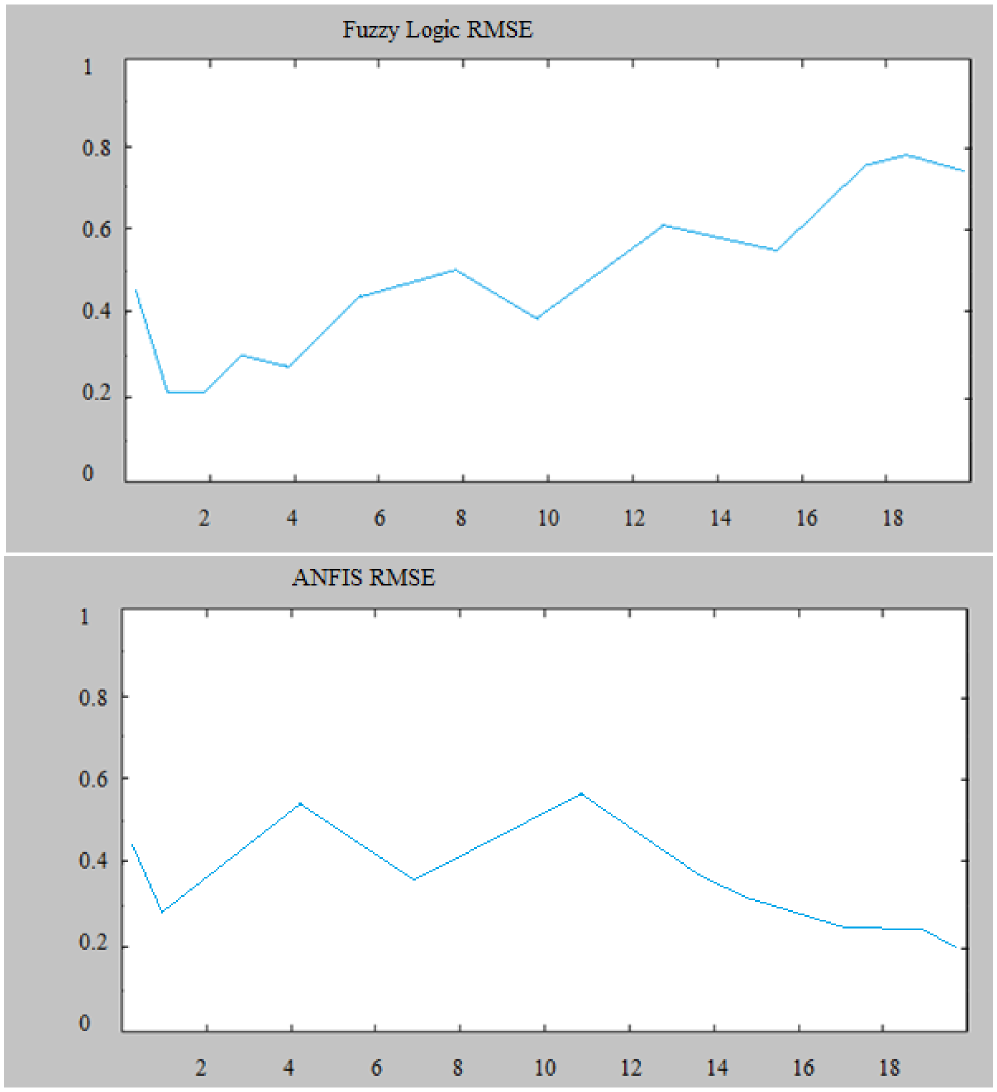

Figure 5. RMSE of Fuzzy Logic and ANFIS Approaches

5. Conclusion and Future Work

Type An efficient and accurate method for measuring aspect-oriented software maintainability has presented in this paper. Adaptive Neuro Fuzzy Inference System was used for predicting the maintainability of aspect-oriented software by means of a number of factors and the final output of the system was estimated amount of maintainability of aspect-oriented software, that the output was classified in 5 classes. After the Execution of operation, the values of each of these factors was mapped in to fuzzy values and used as inputs for the proposed system. The results are expressed as following:

- For the Mamdani Fuzzy approach with 4 membership functions for each effective factor on maintainability and 256 rules for inference, the value of root-mean-square error was 0.780.

- For the Sugeno fuzzy approach with 4 linguistic variables and 20 network training courses, the root- mean-square error was 0.210.
- Since the Adaptive Nero Fuzzy Inference system has an error amount less than Mamdani fuzzy system, it is more accurate to evaluate the rate of aspect-oriented software maintenance.

In our future work, we would like to evaluate the maintainability of AO software by concern-level metrics and compare it with the estimated value obtained by means of s structural complexity metrics as we did in this paper. Also with the help of ANFIS, other qualitative parameters and factors such as reusability, reliability, and etc. could be measurable, which we expect more appropriate results will be achieved by using ANFIS instead of many soft computing methods.

6. References

[1] G. Tiwari, A. Sharma, "Maintainability Techniques for Software Development Approaches – A Systematic Survey", Special Issue of International Journal of Computer Applications (0975 – 8887) on Issues and Challenges in Networking, Intelligence and Computing Technologies – ICNICT 2012, November 2012.

[2] M. Ceccato and P. Tonella, "Measuring the effects of software aspectization," in Proceedings of the 1st Workshop on Aspect Reverse Engineering (WARE '04), Delft, The Netherlands, 2004.

[3] H. Osser, W. H. Harrison, S. S.Jr. N degrees of separation P. Tarr, "Multidimensional separation of concerns", in Proceedings of the international Onference on Software Engineering, IEEE Computer Society Press, 1999, pp.107-119.

[4] Lionel Seinturier Renaud Pawlak, "Foundations of AOP for J2EE Development, "Springer Verlag New York, no. ISBN: 1-59059-6, 2005.

[5] DONG, J. "An Improved Fuzzy Synthesis Evaluation Algorithm for Software Quality". International Conference on Information Management, Innovation Management and Industrial Engineering, 2009.

[6] J. McCall and G. Walters. Factors in Software Quality. The National Technical Information Service, Springfield, VA, USA, 1977.

[7] R. Burrows, F. C. Ferrari, A. Garcia, and F. Ta¨ıani, "An empirical evaluation of coupling metrics on aspect-oriented programs," in Proceedings of the ICSE Workshop on Emerging Trends in SoftwareMetrics (ICSE '10), pp. 53–58,ACM,CapeTown, South Africa, May 2010.

[8] Grady, Robert, Caswell, Deborah (1987), Software Metrics: Establishing a Company-wide Program. Prentice Hall. pp. p. 159. ISBN 0138218447.

[9] IEEE, "IEEE Standard Glossary of Software Engineering Terminology," IEEE Std 610.12-1990, 1990.

[10]Schauerhuber A., Schwinger W., Kapsammer E., Retschitzegger W., Wimmer M., "Towards a Common Reference Architecture for AspectOriented Modeling", Proceedings of the 8th International Workshop on Aspect-Oriented Modeling (AOM), Bonn, Germany, 20-24 March 2006.

[11]w. Abdelmoez, H. Khater, N. El-shoafy, "Comparing Maintainability Evolution of Object-Oriented and Aspect-Oriented Software Product Lines", The 8th International Conference on INFOrmatics and Systems (INFOS2012), Advances in Software Engineering Track, 14-16 May 2012.

[12]ISO 9126-1 Software Engineering - Product Quality - Part 1: Quality Model, 2009.

[13]C. Babu and R. Vijayalakshmi, "Metrics-based design selection tool for aspect oriented software development," SIGSOFT Software Engineering Notes, vol. 33, no. 5, pp. 1–10, 2008.

[14]S. R. Chidamber and C. F. Kemerer, "Metrics suite for object oriented design," IEEE Transactions on Software Engineering, vol. 20, no. 6, pp. 476–493, 1994.