International Journal of Technical Research and Applications e-ISSN: 2320-8163,
www.ijtra.com Volume 4, Issue 1 (January-February, 2016), PP. 180-184

DEVELOPMENT OF EFFICIENT MODEL FOR SOFTWARE DEFECT ANALYSIS

Ravendra Ratan Singh[1], Deepak Tiwari[2]
[1]Assistant Professor & HOD CS dept. Saroj Institute of Technology and Management, Lucknow
[2]Saroj Institute of Technology and Management, Lucknow
[1]raveandraratansingh@yahoo.co.in
[2]deepakt745@gmail.com

***Abstract*— Faults in software systems continue to be a major problem. High quality of software is ensured by Software reliability and Software quality assurance. A software fault is a defect that causes software failure in an executable product. A variety of software fault predictions techniques have been proposed, but none has proven to be consistently accurate. The objective in the construction of models of software error prediction is to use measures that may be obtained relatively early in the software development life cycle to provide reasonable initial estimates of quality of an evolving software system. In the present work an Adaptive Neuro Fuzzy Inference System (ANFIS) Approach has been reviewed for the development of an efficient predictive model using Subtractive Clustering Algorithm. The datasets are taken from NASA Metrics Data Program (MDP) data repository. Through the work presented, it was shown that models developed in this paper using ANFIS technique could be used to effectively address these issues. Low Root Mean Square Error (RMSE) has been obtained, both for training and testing datasets.**

***Keywords*— Software Fault, Subtractive Clustering Algorithm, RMSE, ANFIS, MDP.**

I. INTRODUCTION

Faults in software systems continue to be a major problem [3]. High quality of software is ensured by Software reliability and Software quality assurance. Both these concepts are drawn in throughout the software development and maintenance process. The activities like the performance analysis, functional tests, quantifying time and budget along with measurement of metrics are used to ensure quality. A software bug is an error, flaw, mistake, failure, or fault in a computer program that prevents it from behaving as intended (e.g., producing an incorrect result) [4]. A software fault is a defect that causes software failure in an executable product. Most bugs arise from mistakes and errors made by people in either a program's source code or its design, and a few are caused by compilers producing incorrect code. Knowing the causes of possible defects as well as identifying general software process areas that may need attention from the initialization of a project could save money, time and work. The possibility of early estimating the potential faultiness of software could help on planning, controlling and executing software development activities [5].

Over the past decades years, several empirical studies have been carried out to predict the fault proneness models. And none of the techniques have achieved widespread applicability in the software industry due to several reasons, including the limitation of testing resource, the lack of software tools to automate this software defect prediction, the unwillingness to collect the software defect data, many methods based on the private software data, and the other practical problems [6]. Adaptive Neuro Fuzzy Inference System (ANFIS) proposed by R. Jang,[9,10] is an evolutionary artificial intelligence technique[9,10] and has been applied into many areas including software defect prediction. It has the advantage of allowing the extraction of fuzzy rules from numerical data or expert knowledge and adaptively constructs a rule base. Moreover, it can adapt the complicated conversion of human intelligence to fuzzy systems. In the present work an ANFIS technique using subtractive clustering algorithm has been used applied to solve the problem of software defect prediction.
The rest of the paper is organized as follows: Section II describes the Literature Review. Section III deals with data used, IV the methodology part of work done, followed by results and discussions in section V. In the last section, on the basis of the discussion various conclusions are drawn.

II. Literature Review

Norman Fenton et.al. [4], described a probabilistic model for software defect prediction. This model can not only be used for assessing ongoing projects, but also for exploring the possible effects of a range of software process improvement activities. Ahmet Okutan, et.al.(2012)[5], proposed a novel method using Bayesian networks to explore the relationships among software metrics and defect proneness. Mrinal Singh Rawat et. al.(2012)[6], identified causative factors which in turn suggest the remedies to improve software quality and productivity. The paper also showcases on how the various defect

prediction models are implemented resulting in reduced magnitude of defects. Supreet Kaur, et.al. (2012)[7], studied the performance of the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is evaluated for Fault prediction in Java based Object Oriented Software systems and C++ language based software components. Xiao-dong Mu et. al.,(2012)[8], in order to improve the accuracy of software defect prediction proposed a co-evolutionary algorithm based on the competitive organization. N Fenton, et. al. (2008)[9], in their paper reviewed the use of Bayesian networks (BNs) in predicting software defects and software reliability. The approach allows analysts to incorporate causal process factors as well as combine qualitative and quantitative measures, hence overcoming some of the well known limitations of traditional software metrics methods. Jie Xu, et. al. (2010)[10], used Several statistical techniques together with machine learning method are utilized to verify the effectiveness of software metrics. Manu Banga, (2013) [11], used new computational intelligence sequential hybrid architectures involving Genetic Programming (GP) and Group Method of Data Handling (GMDH) viz. GPGMDH.

III. Data Used

The software metrics and dataset used in this study are mission critical NASA software projects [6], which are all high assurance and complex real-time system. They are taken from PROMISE Software Engineering Repository data set publicly available for research use in software engineering. They are CM1 data, a NASA spacecraft instrument written in "C". This one includes a linguistic attribute – yes/no to indicate defectiveness. It has 498 instances and 22 attributes, given in Table:1 below. The descriptions of the features are taken from http://mdp.ivv.nasa.gov/mdp_glossary.html.

Table 1. Data Attributes used

Input Variable	LOC_BLANK BRANCH_COUNT LOC_CODE_AND_COMMENT LOC_COMMENTS CYCLOMATIC_COMPLEXITY DESIGN_COMPLEXITY ESSENTIAL_COMPLEXITY LOC_EXECUTABLE HALSTEAD_CONTENT HALSTEAD_DIFFICULTY HALSTEAD_EFFORT HALSTEAD_ERROR_EST HALSTEAD_LENGTH HALSTEAD_LEVEL HALSTEAD_PROG_TIME HALSTEAD_VOLUME NUM_OPERANDS NUM_OPERATORS NUM_UNIQUE_OPERANDS NUM_UNIQUE_OPERATORS LOC_TOTAL
Output Variable	DEFECTS

IV. ANFIS Model Development

ANFIS [9,10] is a judicious integration of FIS and ANN, capable of learning, high-level thinking and reasoning and it combines the benefits of these two techniques into a single capsule. Identification of the rule base is the key of a FIS. To generate a FIS using ANFIS [9, 10], it is significant to choose proper parameter, inclusive of the number of membership functions (MFs) for each individual antecedent variables. It is also important to select proper parameters for learning and refining process, including the initial step size (ss). In the present work the commonly used rule extraction method applied for FIS identification and refinement is subtractive clustering. MATLAB Fuzzy Logic Toolbox [11] has been used to simulate the ANFIS.

Here the initial parameters of the ANFIS are identified using the subtractive clustering method [7]. However, the parameters of the subtractive clustering algorithm still need to be specified. The clustering radius is very important parameter in the subtractive clustering algorithm and is optimally determined through a trial and error procedure. For other parameters default values are used in the subtractive clustering algorithm. Gaussian membership functions are used for each fuzzy set in the fuzzy system. The number of membership functions and fuzzy rules required for a particular ANFIS is determined through the subtractive clustering algorithm. Gaussian membership function parameters are optimally determined

using the hybrid learning algorithm. ANFIS training is done for 20-50 epochs.

In MATLAB[11] *genfis2* generates a Sugeno-type FIS structure using subtractive clustering. Since there is only one output, *genfis2* has been used to generate an initial FIS for ANFIS training. *genfis2* achieves this by extract a set of rules that models the data performance. The rule extraction method initially uses the *subclust* function to determine the number of rules and antecedent membership functions and then uses linear least squares estimation to determine each rule's consequent equations.

The parameters used in the model for training ANFIS are given in Table 2 and the rule extraction method used are given in Table 3. Table 4 summerizes the results of types and values of model parameters used for training ANFIS.

Table.2 Parameters used in all the models for training ANFIS

Rule extraction method used	Subtractive clustering
Input MF type	Gaussian membership ('gaussmf')
Input partitioning	variable
Output MF Type	Linear
Number of output MFs	one
Training algorithm	Hybrid learning
Training epoch number	20 TO 50
Initial step size	0.01

Table 3 Rule extraction method used for training ANFIS

Rule Extraction Method	**Type**
And method	'prod'
Or method	'probor'
Defuzzy method	'wtever'
Implication method	'prod'
Aggregation method	'max'

Table 4 Types and values of parameters used for training ANFIS model

No. of nodes	112
No. of linear parameters	44
No. of non-linear parameters	84
Total no. of parameters	128
No. of training data pairs	300
No. of testing data pairs	100
No. of fuzzy rules	2

V. RESULTS AND DISCUSSIONS

ANFIS model having four input variables are trained and tested by ANFIS method and their performances compared and evaluated based on training and testing data. The best fit model structure is determined according to criteria of performance evaluation. The performances of the ANFIS model are shown in Fig. 1 & 2 below and their RMSE values both for training and testing data are shown in Table 5 below**.** Fig. 3 is the graphical representation of the improved model output for testing datasets after ANFIS Application. Fig. 4 is the plot of model output against checking data to ascertain overfitting. Fig. 5 is the comparative plot of FIS against test data.

The low RMSE value obtained during testing phase clearly shows that the model so developed has been able to address the issue, i.e. it has performed well for software defect prediction. Further, analysis of the plots given in Fig. 3 it is inferred that the ANFIS model for testing data has shown improvement, whereas from Fig. 4 it is clearly seen that the model has not been overfitted as the observed data and predicted data almost fall in line. This can be further confirmed from Fig. 1 and 2, where the RMSE values follow more or less similar trend. Thus, it is clear that proper selection of influential radius which affects the cluster results directly in ANFIS using substractive clustering rule extraction method, has resulted in reduction of RMSE both for training and testing data sets. Hence, it is seen that for small size training data, ANFIS has performed well.

Table 5:- RMSE Values for Datasets after using ANFIS

Model Development Stages		**RMSE**	
		Trg. Data	Chk. Data
1	System generated using ANFIS	1.6568e-015	2.2408e-015
2	Optimised Output value	4.9431e-006	9.0379e-005
3	Testing the overfitting	2.5722e-007	3.8947e-007

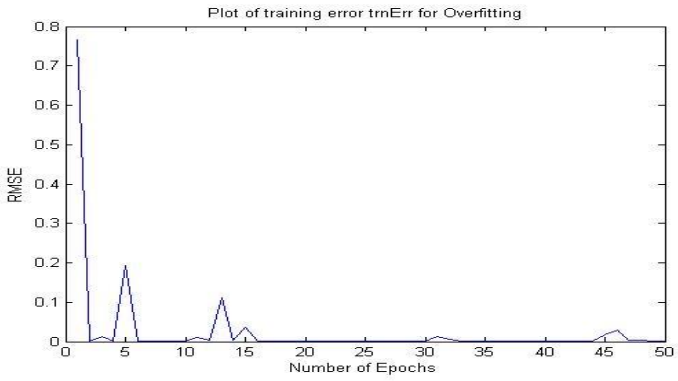

Fig.:- 1 RMSE Plot of Training Datasets during ANFIS Training

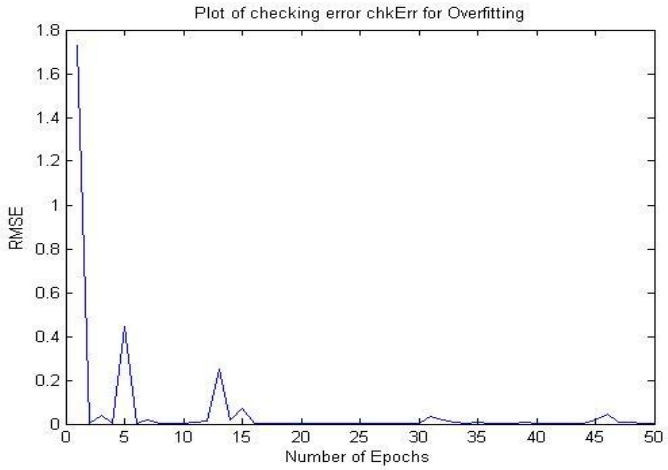

Fig.:- 2 RMSE Plot of Testing Datasets during ANFIS Training

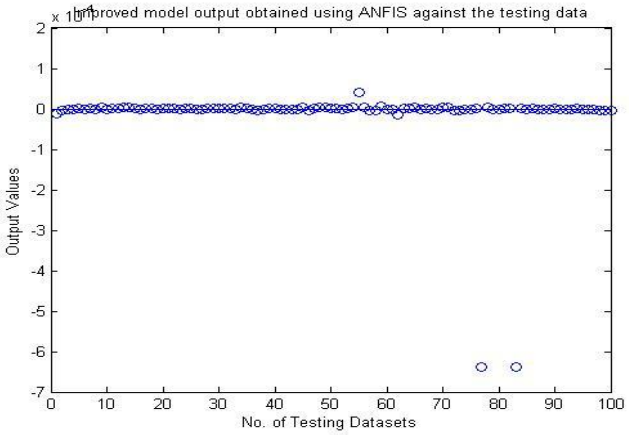

Fig. 3: Improved Model Output after ANFIS Application

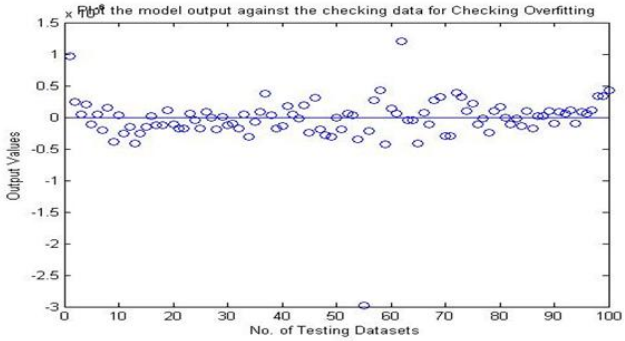

Fig. 4: Plot of Model Output against Checking Data to ascertain Overfitting

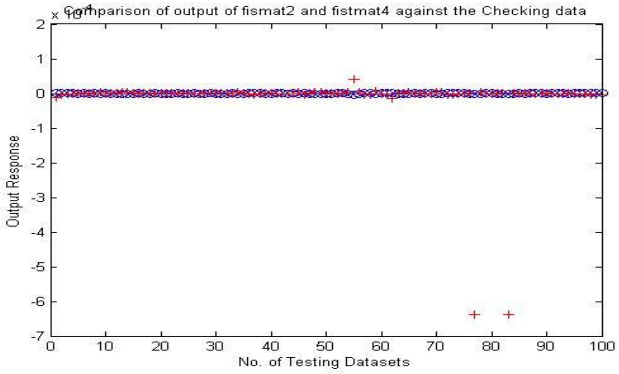

Fig. 5: Comparative plot of FIS against Test Data

VI. CONCLUSION

In the present paper an attempt has been made to develop ANFIS model for predicting software errors. The study has been carried out using MATLAB simulation environment. In all twentyone input variable were used, consisting of various software metrics and one output variable as software defect. For this subtractive clustering algorithm has been used for parameter identification in ANFIS. Both Gaussian and Linear membership functions have been used for the input and output variables. Parameters of the Gaussian membership function are optimally determined using the hybrid learning algorithm. Each ANFIS has been trained for 50 epochs.

From the analysis of the above results, given under heading Results and Discussions, it is seen that the software defect prediction model developed using ANFIS technique has been able to perform well. This can be concluded from the analysis of the results given in Table 5.

REFERENCES

1. Ahmet Okutan, et. al., (2012), "Software defect prediction using Bayesian networks", Empir Software Eng (2014) 19:154–181 6
2. Jang, J-S. R., (1992), "Neuro-Fuzzy Modeling: Architecture, Analyses and Applications", P.hd. Thesis. 10
3. Parvinder S. Sandhu, Sunil Khullar, Satpreet Singh, Simranjit K. Bains, Manpreet Kaur, Gurvinder Singh, "A Study on Early Prediction of Fault Proneness in Software Modules using Genetic Algorithm", World Academy of Science, Engineering and Technology, 2010, pp. 648-653. 1
4. http://puretest.blogspot.com/2009/11/1.html 2
5. Bibi S., Tsoumakas G., Stamelos I., Vlahavas I., "Software Defect Prediction Using Regression via Classification", IEEE International

Conference on Computer Systems and Applications, Issue Date: March 8, 2006, pp.330 - 336, 3

6. http://mdp.ivv.nasa.gov/mdp_glossary.html.
7. CHIU, S., (1994), "Fuzzy Model Identification based on cluster estimation", Journal of Intelligent and Fuzzy Systems, 2 (3), pp 267–278. 8
8. Jang, J-S. R., (1992), "Neuro-Fuzzy Modeling: Architecture, Analyses and Applications", P.hd. Thesis. 9
9. Jang, J-S. R., (1993), "ANFIS-Adaptive-Network Based Fuzzy Inference System"**,** IEEE Transactions on Systems, Man and Cybernatics**,** 23(3), pp 665-685.
10. Jang, J-S. R., SUN, C.-T., (1995), "Neuro-fuzzy modeling and control"**,** Proceedings IEEE**,.** 83 (3), pp 378–406.
11. "Fuzzy Logic Toolbox", MATLAB version R2013a. 12