International Journal of Research and Development in Applied Science and Engineering (IJRDASE)
ISSN: 2454-6844

# *Applying Computational Intelligence Technique for Software Defect Prediction During Early Stage of Software Development*

**Preetika Verma**
Computer Science and Engg
B.I.E.T., Lucknow
preetikaverma.smsit@gmail.com

**Sameer Awasthi**
Computer Science and Engg
B.I.E.T., Lucknow
sameer.awasthi@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 paper an Adaptive Neuro Fuzzy Inference System (ANFIS) Approach has been applied for the development of an efficient predictive model using Substractive Clustering Algorithm. For this NASA's Metrics Data Program (MDP) containing software metric data and error data at the function/method level has been used to validate the algorithm. The experimental results show that the proposed algorithm is effective for software defect prediction.

***Keywords*- Software Defect, RMSE, ANFIS, MDP**

## 1. INTRODUCTION

Software metrics-based quality prediction models can be effective tool for identifying the number of defects of the modules. The use of such models prior to each release planning or re-planning of the system, or even deployment of that can considerably improve system quality. A defect prediction model is calibrated using metrics from a past release or similar project, and is then applied to modules currently under development. Subsequently, a timely prediction of which modules needs more effort to remove the defects, can be obtained. Over the past decades years, several empirical studies have been carried out to predict the fault proneness models. Software defect prediction studies can be categorized as statistical and machine learning (ML) approaches. And the use of machine learning approaches to fault prediction modeling is more popular[7]. Unfortunately, this problems of software defect prediction have not resolved thoroughly. 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,[10] is an evolutionary artificial intelligence technique[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 and the future scope for the present work is discussed.

## 2. Literature Survey

Various techniques, such as linear regression, discriminate analysis, decision trees, neural networks etc. have been developed and applied to predict defects in software. Munson et al. [1] investigate linear regression models and discriminate analysis to conclude the performance of the latter is better. Catal *et al* [2] developed an Eclipse-based software fault prediction tools for Java programs, and naïve bays chosen as the plug-in for the tools. Norman Fenton et al.[3] used dynamic Bayesian nets for predicting software defects. Rather than depending only on data from previous versions, his method makes use of causal models of the Project Manager's understanding and covering mechanisms. Bullard et al. [4] employ a rule-based classification model in a telecommunication system and reported that their model produces lower false positives, which are considered as high cost classification errors.. Ahmet Okutan, et. al., (2012)[6] used Bayesian networks to determine the probabilistic influential relationships among software metrics and defect proneness. Mrinal Singh Rawat, et. al.,(2012)[7] identified causative factors which in turn suggest the remedies to improve software quality and productivity. It also showed on how the various defect prediction models are implemented resulting in reduced magnitude of defects. Supreet Kaur, et. al., (2012)[8] showed that 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)[9] In order to improve the accuracy of software defect prediction, a coevolutionary algorithm based on the competitive organization is put forward for software defect prediction. experiment based on the five datasets from NASA is used to validate the method. The experimental results show that the proposed method is effective.

### 3. DATA USED

The software metrics and dataset used in this study are mission critical NASA software projects [5], which are all high assurance and complex real-time system. They are taken from PROMISE Software Engineering Repository data set made publicly available in order to encourage repeatable, verifiable, refutable, and/or improvable predictive models of software engineering. They are Class-level data for KC1. This one includes a numeric attribute (NUMDEFECTS) to indicate defectiveness. The descriptions of the features are taken from http://mdp.ivv.nasa.gov/mdp_glossary.html.

**Table 1. Input and Output variables for ANFIS model**

| | |
|---|---|
| **Input Variable** | LOC_BLANK<br>BRANCH_COUNT<br>LOC_CODE_AND_COMMENT<br>LOC_COMMENTS<br>CYCLOMATIC_COMPLEXITY<br>DESIGN_COMPLEXITY<br>ESSENTIAL_COMPLEXITY<br>LOC_EXECUTABLE<br>HALSTEAD_CONTENT<br>HALSTEAD_DIFFICULTY<br>HALSTEAD_EFFORT<br>HALSTEAD_ERROR_EST<br>HALSTEAD_LENGTH<br>HALSTEAD_LEVEL<br>HALSTEAD_PROG_TIME<br>HALSTEAD_VOLUME<br>NUM_OPERANDS<br>NUM_OPERATORS<br>NUM_UNIQUE_OPERANDS<br>NUM_UNIQUE_OPERATORS<br>LOC_TOTAL |
| **Output Variable** | NUMDEFECTS |

### 3. ANFIS MODEL DEVELOPMENT

#### 3.1 Parameter Selection

ANFIS [11],[14] 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. The problems are (1) there are no standard methods for transforming human knowledge or experience into rule base; and (2) it is required to further tune the MFs to minimise the output error and to maximise the performances. Thus when generating a FIS using ANFIS, it is important to select proper parameters, including 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. The ANFIS is simulated using the MATLAB Fuzzy Logic Toolbox [12].

Here the initial parameters of the ANFIS are identified using the subtractive clustering method [13]. However, the parameters of the subtractive clustering algorithm still need to be specified. The clustering radius is the most important parameter in the subtractive clustering algorithm and is optimally determined through a trial and error procedure. By varying the clustering radius $r_a$ between 0.1 and 1 with a step size of 0.01, the optimal parameters are sought by minimizing the root mean squared error obtained on a representative validation set. Clustering radius $r_b$ is selected as 1.5 $r_a$. Default values are used for other parameters 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. Parameters of the Gaussian membership function are optimally determined using the hybrid learning algorithm. Each ANFIS is trained for 1000 epochs.

Gaussian membership function has been used as the input membership function and linear membership function for the output function. Here separate sets of input and output data has been used as input arguments. In MATLAB *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* accomplishes this by extracting a set of rules that models the data behaviour. The rule extraction method first 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. This function returns a FIS structure that contains a set of fuzzy rules to cover the feature space.

Table 1 shows the input and output model parameters used for model development. 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 | 1000 |
| 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. Values of parameters used for training ANFIS**

| | |
|---|---|
| No. of nodes | 244 |
| No. of linear parameters | 110 |
| No. of non-linear parameters | 210 |
| Total no. of parameters | 320 |
| No. of training data pairs | 90 |
| No. of testing data pairs | 55 |
| No. of fuzzy rules | 5 |

## 4. RESULTS AND DISCUSSIONS

Here the ANFIS model has been trained tested by ANFIS method and their performance for the best prediction model are evaluated and compared for training and testing data sets separately. The best RMSE performances of the ANFIS model ( for cluter radius r=0.75) both for training and testing datasets have been plotted separately in Fig. 1 & Fig.2 and their corresponding RMSE values for training and testing datasets for different clustering radius are summarized in Table 5.

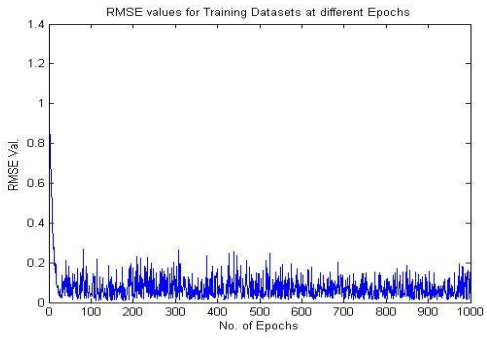


**Fig. 1 Graphical plot of RMSE value variation during training**

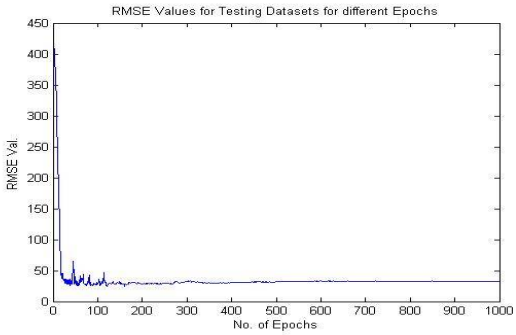


**Fig. 2 Graphical plot of RMSE value variation during testing**

**Table 5. RMSE values for different Clustering radius**

| RMSE for diff. radius of influence ( r) | | | |
|---|---|---|---|
| | **r=0.5** | **r=0.75** | **r=1.0** |
| **Trg. Data** | 0.014 | 0.0101 | 1.844 |
| **Tst. Data** | 25.197 | 13.256 | 31.372 |
| **Overall Data** | 15.518 | 8.164 | 19.2 |

From the perusal of the data given in tables 5 it is inferred that for three different clustering radius 0.5, 0.75 and 1.0, the prediction model performed best for r=0.75, having RMSE value of 0.01 and 13.256 for for training and testing phase. The same has been plotted in Fig. 1 & 2, whose analysis reveals that during training phase (Fig.1), there is a sudden fall in the RMSE value during the first 20 epochs, after which there is more or less gradual change in RMSE values, having a minimum value of 0.0101 and a maximum value of 1.0267. On the other hand, during testing phase (Fig.2) of ANFIS training initially there is sudden fall in RMSE value, after which there is more or less constant value upto epochs 1000. Thus there is a minimum and maximum RMSE value of 24.66 and 415.29. Also from the perusal of the data given in Table 5 it can be inferred that ANFIS has performed better during training phase than testing phase but its overall RMSE value is 8.164.

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.

In order to depict how well ANFIS has performed, a comparative plot of actual defect versus predicted effort, using ANFIS technique, has been shown in Fig. 5 & 6, using data given in Table 6 & 7. From the graph it is seen that ANFIS model line almost closely follows the actual defect line. This again depicts the superiority of ANFIS technique for defect prediction.

**Table 6. Summerised Results of Sctual and Predicted Defect Values for Training Datasets**

| Act. Defect | Pre. Defect | 0 | -0.01251398 |
|---|---|---|---|
| 23 | 22.9890937 | 1 | 0.991823328 |
| 16 | 15.98918948 | 0 | 0.001242293 |
| 3 | 3.003743258 | 4 | 4.000252911 |
| 19 | 18.98897328 | 9 | 8.984964791 |
| 6 | 6.001008371 | 0 | -0.00352579 |
| 3 | 2.99819667 | 0 | 0.000276154 |
| 3 | 2.996777391 | 8 | 7.999837797 |
| 3 | 2.997890354 | 101 | 100.9740268 |
| 4 | 3.990260445 | 0 | -0.01669314 |
| 3 | 3.010712475 | 0 | 0.002790826 |
| 5 | 4.998229282 | 0 | -0.0001285 |
| 0 | -0.00010444 | 0 | -0.00096789 |
| 4 | 3.997512361 | 0 | 0.006739904 |
| 0 | 0.024282289 | 0 | 0.000744705 |
| 0 | -0.00089097 | 0 | -0.00807808 |
| 23 | 22.99145436 | 0 | -0.00575344 |
| 20 | 19.99665302 | 0 | -0.01186826 |
| 0 | -0.00109047 | 2 | 1.994642146 |
| 0 | 0.00366517 | 0 | -0.00845106 |
| 0 | -0.0012657 | 0 | -0.00584906 |
| 2 | 2.001206132 | 0 | 3.76E-05 |
| 0 | 0.000810968 | 0 | 0.001169137 |
| 0 | -0.02887525 | 0 | -0.00052485 |
| 0 | -0.00010768 | 0 | -9.01E-05 |
| 14 | 13.99297704 | 0 | -0.00510395 |
| 8 | 7.999988641 | 0 | 0.0044353 |
| 7 | 7.000460668 | 0 | -0.01761119 |
| 22 | 21.99823083 | 0 | 0.010061132 |
| 5 | 4.996054261 | 0 | 0.031007541 |
| 0 | -0.00552051 | 0 | -0.04604462 |
| 0 | 0.00107111 | 7 | 7.000149208 |
| 4 | 3.984978301 | 0 | -0.00170446 |
| 4 | 3.976680944 | 0 | -1.43E-05 |
| 0 | -0.00311922 | 42 | 41.97857439 |
| 0 | -0.00032679 | 24 | 23.99989789 |
| 0 | -0.00068376 | 17 | 16.99907937 |
| 0 | 0.012474512 | 8 | 7.999107988 |
| 0 | -3.14E-05 | 4 | 4.00324772 |
| 0 | 0.013382224 | 6 | 5.999952047 |
| 4 | 3.996189869 | 3 | 3.000162951 |
| 0 | 0.000201082 | 17 | 16.99814297 |
| 2 | 1.999998524 | 9 | 9.000538086 |
| 0 | 0.000460854 | 0 | 0.000111185 |
| 7 | 7.001433515 | 0 | 0.000732842 |
| | | 32 | 31.9974817 |

**Table 7:- Summerised Results of Sctual and Predicted Defect Values for Testing Datasets**

| Act. Defect | Pre. Defect | 0 | 14.80862444 |
|---|---|---|---|
| 20 | 18.31263759 | 0 | -0.01856829 |
| 14 | -3.64604818 | 0 | -0.03721321 |
| 3 | -6.02204598 | 26 | 2.284127638 |
| 2 | 7.487105594 | 4 | -13.2145005 |
| 0 | -12.6343929 | 0 | -16.0773878 |
| 6 | 29.53264229 | 0 | 3.530103455 |
| 19 | 28.79311684 | 6 | -13.5319316 |
| 13 | 31.89483623 | 0 | 3.075540284 |
| 0 | -16.4369616 | 11 | 38.17036328 |
| 0 | -45.9344396 | 11 | 24.14581483 |
| 0 | 0.074209395 | 3 | 3.409076182 |
| 0 | -5.12362697 | 0 | 0.117551486 |
| 6 | 13.86958413 | 13 | 2.415202415 |
| 0 | 1.542898 | 0 | 1.470573004 |
| 2 | 0.318371452 | 0 | 0.49834135 |
| 0 | 1.070233258 | 0 | 15.72977697 |
| 1 | -0.8881197 | 0 | 0.051507761 |
| 0 | 24.6483655 | 0 | 0.134772179 |
| 0 | -0.398183 | 0 | -19.1858159 |
| 0 | 0.072438033 | 0 | 0.009541944 |
| 0 | 16.48991558 | 0 | 0.374123026 |
| 0 | 15.39989893 | 0 | -0.27895808 |
| 0 | -1.31037766 | 0 | -0.02811841 |
| 0 | -2.06176251 | 0 | 0.00491892 |
| 0 | 23.14399417 | 0 | -0.02811841 |
| 1 | 24.10270153 | 0 | 3.276273884 |
| | | 0 | -0.02811841 |
| | | 0 | 3.276273884 |

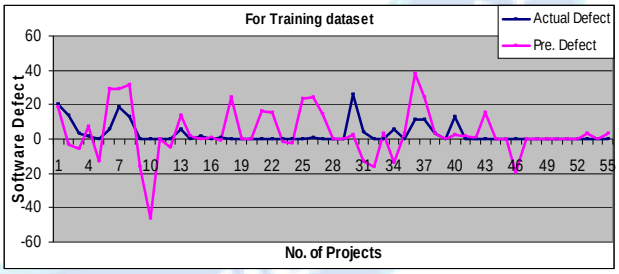


**Fig 5 Comparative plot of Actual and Predicted Software Defect for Training Datasets**

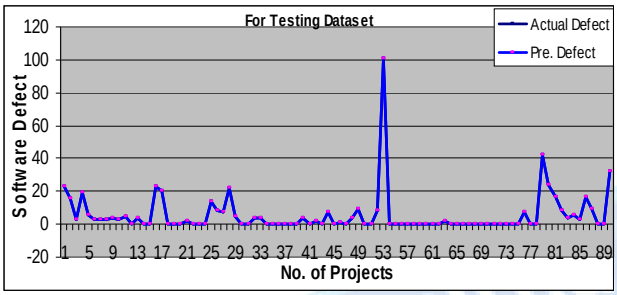


**Fig 6 Comparative plot of Actual and Predicted Software Defect for Testing Datasets**

Finally, Figure 7, shows the scatter plot of Actual defect versus Estimated defect using ANFIS. The figures wisely demonstrate that (1) the model performance is in general accurate in case of ANFIS, where all data points roughly fall onto the line of agreement; (2) model using ANFIS is consistently superior for software defect prediction.

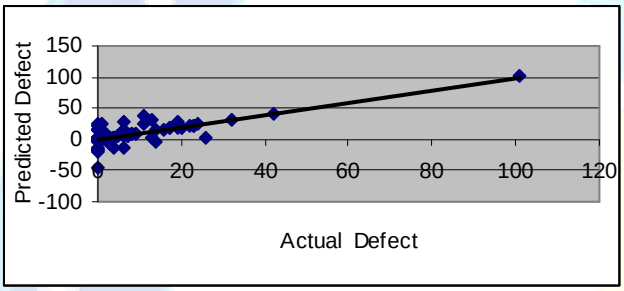


**Fig 7 Scatter Plot of Actual Vs. Predicted Defect using ANFIS**

## 5. CONCLUSION

In the present paper, applicability and capability of ANFIS techniques for software defect prediction has been investigated. It is seen that ANFIS models are very robust, characterised by fast computation, capable of handling the noisy and approximate data that are typical of data used here for the present study. Due to the presence of non-linearity in the data, it is an efficient quantitative tool to predict effort estimation. The studies has been carried out using MATLAB simulation environment. In all twenty input variable were used, consisting of various software metrics and one output variable as software defect.

Here the initial parameters of the ANFIS are identified using the subtractive clustering method. Gaussian membership functions ( given in earlier section ) 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. Parameters of the Gaussian membership function are optimally determined using the hybrid learning algorithm. Each ANFIS has been trained for 1000 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. The RMSE value obtained from ANFIS model for r=0.75 is 8.164, which is lower than those for r=0.5 & 1.0. Further from Fig. 5, & 6 and Table 7 it is seen that ANFIS model line almost closely follows the actual defect line. This again depicts the superiority of ANFIS technique as a predictor tool.

## 6. REFERENCES

[1]. John C. Munson and Taghi M. Khoshgoftaar, "The Detection of Fault-Prone Programs" , IEEE Transactions on Software Engineering, vol. 18, pp. 423-433, 1992. 13

[2]. Cagatay Catal, Ugur Sevim and Banu Diri, "Practical development of an Eclipse-based software fault prediction tool using naïve bayes algorithm", Expert System with application, vol. 38, no. 3, pp. 2347~2353, 2011. 16

[3]. Norman Fenton, Martin Neil, William Marsh, Peter hearty, David Marquez, Paul Krause and Rajat Mishra, "Predicting software defect in varying development lifecycles using Bayesian nets", Information and Software Technology, vol. 49, pp. 32~43,2007. 18

[4]. Lofton A. Bullard, Taghi M. Khoshgoftaar and Kehan Gao, "An application of a rule-based model in software quality classification" , In Proceeding of Sixth International Conference on Machine Learning and Applications, pp. 204 ~210, 2007. 21

[5]. NASA metrics data program, http://mdp.ivv.nasa.gov/mdp_glossary.html. 2012. 22

[6]. Ahmet Okutan, et. al., (2012), "Software defect prediction using Bayesian networks", Empir Software Eng (2014) 19:154–181

[7]. Mrinal Singh Rawat, et. al.,(2012), "Software Defect Prediction Models for Quality Improvement: A Literature Study", IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 5, No 2,

[8]. Supreet Kaur, et. al., (2012) "Software Fault Prediction in Object Oriented Software Systems Using Density Based Clustering Approach", International Journal of Research in Engineering and Technology (IJRET) Vol. 1 No. 2, pp 111-117.

[9]. Xiao-dong Mu, et. al.,(2012), "Software Defect Prediction Based on Competitive Organization CoEvolutionary Algorithm", Journal of Convergence Information Technology(JCIT) Volume7, Number5

[10]. Hammouda, K. A., *"Comparative Study of Data Clustering Techniques"*.

[11]. Jang, J-S. R., (1992), *"Neuro-Fuzzy Modeling: Architecture, Analyses and Applications"*, P.hd. Thesis.

[12]. Fuller, R., (1995), *"Neural Fuzzy Systems"*, ISBN 951-650-624-0, ISSN 0358-5654.17

[13]. Chen, D.W. And Zhang, J.P., (2005), *"Time series prediction based on ensemble ANFIS"*, Proceedings of the fourth International Conference on

Machine Learning and Cybernatics, IEEE, pp 3552-3556.10

[14]. Jang, J-S. R., (1993), *"ANFIS-Adaptive-Network Based Fuzzy Inference System"*, IEEE Transactions on Systems, Man and Cybernatics, 23(3), pp 665-685.