World Academy of Science, Engineering and Technology
International Journal of Computer and Systems Engineering
Vol:3, No:8, 2009

# Predicting the Impact of the Defect on the Overall Environment in Function Based Systems

Parvinder S. Sandhu, Urvashi Malhotra, and Ebru Ardil

***Abstract*—**There is lot of work done in prediction of the fault proneness of the software systems. But, it is the severity of the faults that is more important than number of faults existing in the developed system as the major faults matters most for a developer and those major faults needs immediate attention. In this paper, we tried to predict the level of impact of the existing faults in software systems. Neuro-Fuzzy based predictor models is applied NASA's public domain defect dataset coded in C programming language. As Correlation-based Feature Selection (CFS) evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them. So, CFS is used for the selecting the best metrics that have highly correlated with level of severity of faults. The results are compared with the prediction results of Logistic Models (LMT) that was earlier quoted as the best technique in [17]. The results are recorded in terms of *Accuracy*, Mean Absolute Error (*MAE*) and Root Mean Squared Error (*RMSE*). The results show that Neuro-fuzzy based model provide a relatively better prediction accuracy as compared to other models and hence, can be used for the modeling of the level of impact of faults in function based systems.

***Keywords*—**Software Metrics, Fuzzy, Neuro-Fuzzy, Software Faults, Accuracy, MAE, RMSE.

## I. Introduction

IN the literature [1] [2], [3], [4], [5], [6] made prediction of fault prone modules in software development process and mostly used the metric based approach with machine learning techniques to model the fault prediction in the software modules. Khoshgoftaar [7] used zero-inflated Poisson regression to predict the fault-proneness of software systems with a large number of zero response variables. Munson and Khoshgoftaar [8, 9] also investigated the application of multivariate analysis to regression and showed that reducing the number of "independent" factors (attribute set) does not significantly affect the Accuracy of software quality prediction. Menzies, Ammar, Nikora, and Stefano [10] compared decision trees, naïve Bayes, and 1-rule classifier on the NASA software defect data. Emam, Benlarbi, Goel, and Rai [11] compared different case-based reasoning classifiers and concluded that there is no added advantage in varying the combination of parameters (including varying nearest neighbor and using different weight functions) of the classifier to make the prediction Accuracy better. Many modeling techniques have been developed and applied for software quality prediction [12], [13], [14], [15]. The software quality may be analyzed with limited fault proneness data [16].

In [17], the author has used various machine learning techniques for an intelligent system for the software maintenance prediction and proposed the logistic model Trees (LMT) and Complimentary Naïve Bayes (CNB) algorithms on the basis of Mean Absolute Error (*MAE*), Root Mean Square Error (*RMSE*) and *Accuracy* percentage.

Neuro-Fuzzy systems have proven to be of great practical value in a variety of application domains especially in case of non linear modeling of systems. In this paper, we tried to predict the level of impact of the existing faults in software systems. Neuro-Fuzzy based predictor models are applied NASA's public domain defect dataset coded in C programming language. The results are compared with the prediction results of Logistic Models (LMT) that was earlier quoted as the best technique in [17].

This paper is organized as: Section two describes the Methodology part of work done, which shows the steps used in order to reach the objectives and carry out the results. In the section three, results of the implementation are discussed. In the last section, on the basis of the discussion various Conclusions are drawn and the future scope for the present work is discussed.

## II. Proposed Methodology

### *A. Find the Structural Code and Design Attributes*

The first step is to find the structural code and design attributes of software systems i.e. software metrics that are relevant in modeling the impact of severity of faults in function oriented systems. The real-time defect data sets are taken from the NASA's MDP (Metric Data Program) data repository. The dataset is related to the safety critical software systems being developed by NASA.

Urvashi Malhotra is working as Lecturer in Computer Science & Engineering Department, Rayat Institute of Engineering & Information Technology, Rail Majra, Punjab, India.

Parvinder S. Sandhu is Professor with Computer Science & Engineering Department, Rayat & Bahra Institute of Engineering & Bio-Technology, Sahauran, Distt. Mohali (Punjab)-140104 India (phone: +91-98555-32004; e-mail: parvinder.sandhu@gmail.com).

### *B. Collection of Metric Values*

The suitable metrics values of different product modules is collected and observed severity of faults in those modules is noted.

### *C. Refinement of Metrics using CFS*

A central problem in machine learning is identifying a representative set of features from which to construct a classification model for a particular task. In the step the metrics are refined using Correlation based Feature Selection (CFS) technique. This CFS addresses the problem of feature selection for machine learning through a correlation based approach. The central hypothesis is that good feature sets contain features that are highly correlated with the class. A feature evaluation formula, based on ideas from test theory, provides an operational definition of this hypothesis. In other words, Correlation-based Feature Selection evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them.

### *D.Experimentation with Neuro-Fuzzy based System*

The Neuro-Fuzzy based system is used in prediction of level of impact of faults. Neural Network is used to train the Sugeno based Fuzzy Inference System. The results are calculated in terms of the different criteria explained in the next point.

### *E. Comparison Criteria*

The comparisons of machine learning algorithms are made on the basis of following criteria:

- *Accuracy*

  The percentage of the predicted impact values that match with the expected values of the level impact of faults in the modules.

- *Mean Absolute Error*

Mean absolute error, *MAE* is the average of the difference between predicted and actual value in all test cases; it is the average prediction error [21]. The formula for calculating *MAE* is given in equation shown below:

$$\frac{|a_1 - c_1| + |a_2 - c_2| + \ldots + |a_n - c_n|}{n} \qquad (1)$$

Assuming that the actual output is *a*, expected output is *c*.

- *Root Mean-Squared Error*

*RMSE* is frequently used measure of differences between values predicted by a model or estimator and the values actually observed from the thing being modeled or estimated [21]. It is just the square root of the mean square error as shown in equation given below:

$$\sqrt{\frac{(a_1 - c_1)^2 + (a_2 - c_2)^2 + \ldots + (a_n - c_n)^2}{n}} \qquad (2)$$

The best system is that having the highest *Accuracy* and *least* values *of MAE and RMSE.*

### *F. Conclusions Drawn*

The conclusions are made on the basis of the comparison made in the previous section.

## III. Results & Discussion

The real-time defect data set used is taken from the NASA's MDP (Metric Data Program) data repository, the details of that dataset contains 178 modules of C Programming language with different values of software fault severity labeled as 1, 2 and 3. The level 1 represents the highest severity, level 2 represents the medium and level 3 represents the minor fault that can be overlooked to save time. Details of the type of faults existing in different number of modules of the Dataset are shown in bar chart of Fig. 1.

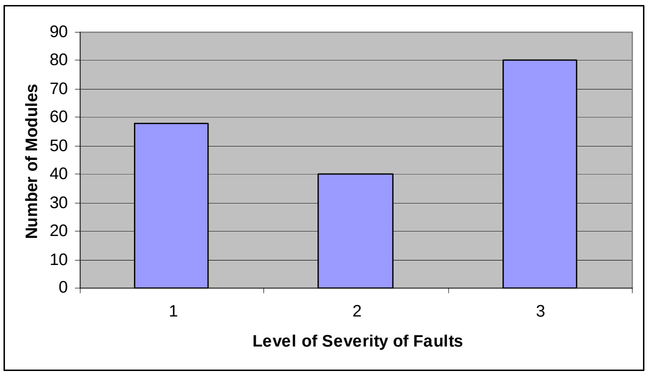
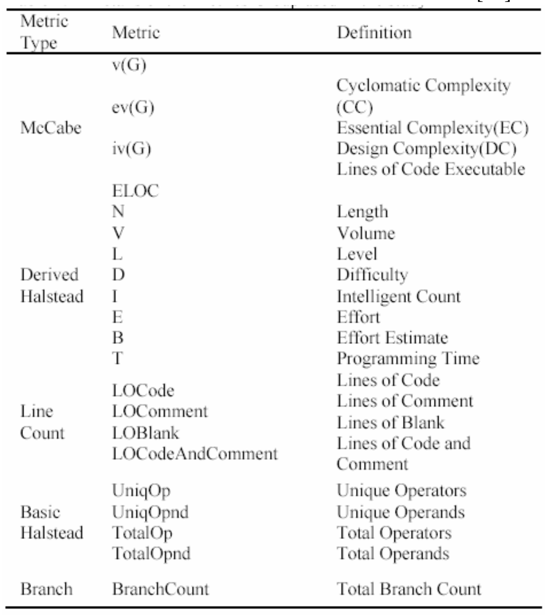


Fig. 1 Graphical Representation of Details of the Type of Modules in the Dataset

There are total 21 metrics in the dataset of NASA. The detail of the metrics is shown in the following table:

TABLE I
Details of the Metric Group of NASA Data [17]



| Metric Type | Metric | Definition |
|---|---|---|
| McCabe | v(G) | Cyclomatic Complexity (CC) |
| | ev(G) | Essential Complexity(EC) |
| | iv(G) | Design Complexity(DC) |
| | ELOC | Lines of Code Executable |
| Derived Halstead | N | Length |
| | V | Volume |
| | L | Level |
| | D | Difficulty |
| | I | Intelligent Count |
| | E | Effort |
| | B | Effort Estimate |
| | T | Programming Time |
| Line Count | LOCode | Lines of Code |
| | LOComment | Lines of Comment |
| | LOBlank | Lines of Blank |
| | LOCodeAndComment | Lines of Code and Comment |
| Basic Halstead | UniqOp | Unique Operators |
| | UniqOpnd | Unique Operands |
| | TotalOp | Total Operators |
| | TotalOpnd | Total Operands |
| Branch | BranchCount | Total Branch Count |

In order to incorporate the correlation of independent variables, a correlation based feature selection technique (CFS) is applied to select the best predictors out of independent variables in the datasets [19]. The best combinations of independent variable are searched through all possible combinations of variables. CFS evaluates the best of a subset of variables by considering the individual predictive ability of each feature along with the degree of redundancy between them [20]. The following are the five metrics that are further used for the NF based modeling:

- LOC_CODE_AND_COMMENT: The number of lines which contain both code and comment in a module.
- HALSTEAD_LENGTH: The Halstead length metric of a module.
- NUM_OPERANDS: The number of operands contained in a module.
- NUM_OPERATORS: The number of operators contained in a module.
- LOC_TOTAL: The total number of lines for a given module.

Given separate sets of input and output data, subtractive clustering algorithm is used generates a Fuzzy Inference system (FIS). The inputs and output of the FIS are shown in Fig. 2. It is accomplished by extracting a set of rules that models the data behavior. The rule extraction method first determines 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 an FIS structure that contains a set of fuzzy rules to cover the feature space.

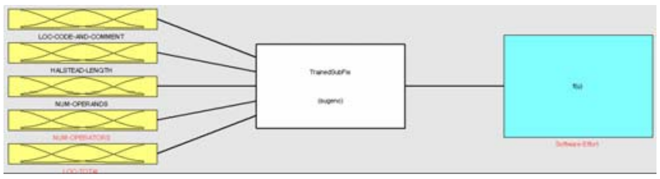
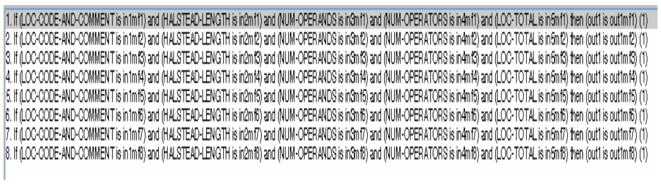


Fig. 2 Fuzzy Inference System Generated using Subtractive Clustering



```
1. If (LOC-CODE-AND-COMMENT is in1mf1) and (HALSTEAD-LENGTH is in2mf1) and (NUM-OPERANDS is in3mf1) and (NUM-OPERATORS is in4mf1) and (LOC-TOTAL is in5mf1) then (out1 is out1mf1) (1)
2. If (LOC-CODE-AND-COMMENT is in1mf2) and (HALSTEAD-LENGTH is in2mf2) and (NUM-OPERANDS is in3mf2) and (NUM-OPERATORS is in4mf2) and (LOC-TOTAL is in5mf2) then (out1 is out1mf2) (1)
3. If (LOC-CODE-AND-COMMENT is in1mf3) and (HALSTEAD-LENGTH is in2mf3) and (NUM-OPERANDS is in3mf3) and (NUM-OPERATORS is in4mf3) and (LOC-TOTAL is in5mf3) then (out1 is out1mf3) (1)
4. If (LOC-CODE-AND-COMMENT is in1mf4) and (HALSTEAD-LENGTH is in2mf4) and (NUM-OPERANDS is in3mf4) and (NUM-OPERATORS is in4mf4) and (LOC-TOTAL is in5mf4) then (out1 is out1mf4) (1)
5. If (LOC-CODE-AND-COMMENT is in1mf5) and (HALSTEAD-LENGTH is in2mf5) and (NUM-OPERANDS is in3mf5) and (NUM-OPERATORS is in4mf5) and (LOC-TOTAL is in5mf5) then (out1 is out1mf5) (1)
6. If (LOC-CODE-AND-COMMENT is in1mf6) and (HALSTEAD-LENGTH is in2mf6) and (NUM-OPERANDS is in3mf6) and (NUM-OPERATORS is in4mf6) and (LOC-TOTAL is in5mf6) then (out1 is out1mf6) (1)
7. If (LOC-CODE-AND-COMMENT is in1mf7) and (HALSTEAD-LENGTH is in2mf7) and (NUM-OPERANDS is in3mf7) and (NUM-OPERATORS is in4mf7) and (LOC-TOTAL is in5mf7) then (out1 is out1mf7) (1)
8. If (LOC-CODE-AND-COMMENT is in1mf8) and (HALSTEAD-LENGTH is in2mf8) and (NUM-OPERANDS is in3mf8) and (NUM-OPERATORS is in4mf8) and (LOC-TOTAL is in5mf8) then (out1 is out1mf8) (1)
```

Fig. 3 Rules of Fuzzy Inference System Generated using Subtractive Clustering

During the generation of the FIS using subtractive clustering radii is an important parameter. Radii is a vector that specifies a cluster center's range of influence in each of the data dimensions, assuming the data falls within a unit hyperbox. If radii is a scalar, then the scalar value is applied to all data dimensions, i.e., each cluster center will have a spherical neighborhood of influence with the given radius. The rules extracted from the dataset are shown in Fig. 3.

The FIS is trained using Neural Network methodology. The structure of the Neuro-Fuzzy system is shown in Fig. 4.

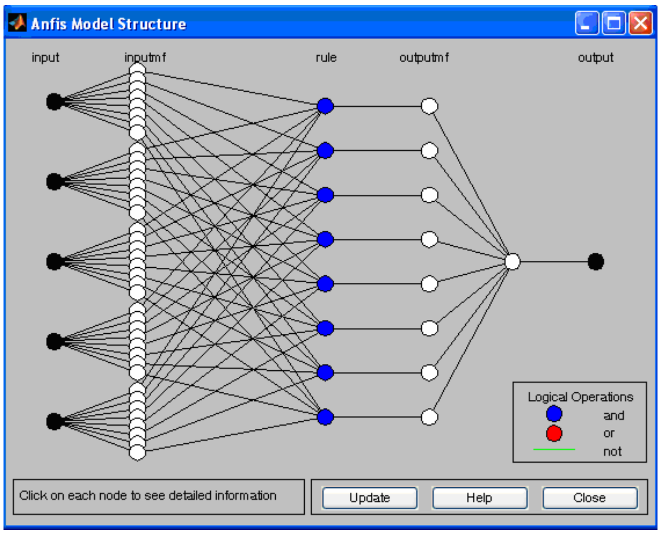


Fig. 5 Neuro-Fuzzy Structure for training of FIS

TABLE I
PERFORMANCE OF THE NF IN PREDICTING THE IMPACT OF FAULTS AT DIFFERENT RADII VALUES

| Radii Value | Performance Criteria | | |
|---|---|---|---|
| | Accuracy | MAE | RMSE |
| 0.15 | 71.91 | 0.382 | 0.53 |
| 0.17 | 72.47 | 0.39 | 0.539 |
| 0.2 | 67.41 | 0.426 | 0.536 |
| 0.25 | 66.85 | 0.454 | 0.605 |

During the testing phase the performance of the NF in predicting the Impact of Faults at different Radii Values is noted down as shown in the Table I. The proposed system shows optimal performance at radii 0.17 with 66.85, 0.454 and 0.605 as Accuracy (%), MAE and RMSE values.

In literature [17] has mentioned that Logistic Model Trees (LMT) is best among other machine learning algorithms used in that study for the prediction of maintenance severity. So, Logistic Model Trees (LMT) is apllied on the polished dataset it has shown *Accuracy*, *MAE* and *RMSE* values as 56.74, 0.2269, and 0.334 respectively.

## IV. CONCLUSION

According to [17], on comparing all the classes of WEKA's machine learning algorithms, it is observed that Logistic Model Trees algorithm is best prediction techniques as compared with other classes of machine learning algorithms in prediction of severity of faults in software systems. But, the proposed Adaptive Neuro-fuzzy based prediction technique has outperformed Logistic Model Trees technique on the basis of the testing data with 66.85, 0.454 and 0.605 as *Accuracy*, *Mean Absolute Error* and *Root Mean Square Error* values.

It is therefore, concluded the model is implemented and the best algorithm for classification of the software components into different level of severity of impact of the fault is found to be Neuro-Fuzzy based technique. The algorithm can be used to develop model that can be used for identifying modules that are heavily affected by the faults and those can be debugged at appropriate time.

## REFERENCES

[1] Saida Benlarbi,Khaled El Emam, Nishith Geol (1999), “Issues in Validating Object-Oriented Metrics for Early Risk Prediction”, by Cistel Technology 210 Colonnade Road Suite 204 Nepean, Ontario Canada K2E 7L5.

[2] Lanubile F., Lonigro A., and Visaggio G. (1995) “Comparing Models for Identifying Fault-Prone Software Components”, Proceedings of Seventh International Conference on Software Engineering and Knowledge Engineering, June 1995, pp. 12-19.

[3] Fenton, N. E. and Neil, M. (1999), “A Critique of Software Defect Prediction Models”, Bellini, I. Bruno, P. Nesi, D. Rogai, University of Florence, IEEE Trans. Softw. Engineering, vol. 25, Issue no. 5, pp. 675-689.

[4] Giovanni Denaro (2000), ”Estimating Software Fault-Proneness for Tuning Testing Activities” Proceedings of the 22nd International Conference on Software Engineering (ICSE2000), Limerick, Ireland, June 2000.

[5] Manasi Deodhar (2002), “Prediction Model and the Size Factor for Fault-proneness of Object Oriented Systems”, MS Thesis, Michigan Tech. University, Dec. 2002.

[6] Bellini, P. (2005), “Comparing Fault-Proneness Estimation Models”, 10th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS'05), vol. 0, 2005, pp. 205-214.

[7] Khoshgoftaar, T.M., K. Gao and R. M. Szabo ( 2001), “An Application of Zero-Inflated Poisson Regression for Software Fault Prediction. Software Reliability Engineering”, ISSRE 2001. Proceedings of 12th International Symposium on, 27-30 Nov. (2001), pp: 66 -73.

[8] Munson, J. and T. Khoshgoftaar, (1990) “Regression Modeling of Software Quality: An Empirical Investigation”, Information and Software Technology, 32(2): 106 - 114.

[9] Khoshgoftaar, T. M. and J. C. Munson, (1990). “Predicting Software Development Errors using Complexity Metrics”, IEEE Journal on Selected Areas in Communications, 8(2): 253 -261.

[10] Menzies, T., K. Ammar, A. Nikora, and S. Stefano, (2003), “How Simple is Software Defect Prediction?”, Journal of Empirical Software Engineering, October (2003).

[11] Eman, K., S. Benlarbi, N. Goel and S. Rai, (2001), “Comparing case-based reasoning classifiers for predicting high risk software components”, Journal of Systems Software, 55(3): 301 – 310.

[12] Hudepohl, J. P., S. J. Aud, T. M. Khoshgoftaar, E. B. Allen, and J. E. Mayrand, (1996), “Software Metrics and Models on the Desktop”, IEEE Software, 13(5): 56-60.

[13] Khoshgoftaar, T. M., E. B. Allen, K. S. Kalaichelvan, and N. Goel, (1996), “Early quality prediction: a case study in telecommunications”, IEEE Software (1996), 13(1): 65-71.

[14] Khoshgoftaar, T. M. and N. Seliya, (2002), “Tree-based software quality estimation models for fault prediction”, METRICS 2002, the Eighth IIIE Symposium on Software Metrics, pp: 203-214.

[15] Seliya N., T. M. Khoshgoftaar, S. Zhong, (2005), “Analyzing software quality with limited fault-proneness defect data”, Ninth IEEE international Symposium, Oct 12-14, (2005).

[16] Munson, J. C. and T. M. Khoshgoftaar, (1992), “The detection of fault-prone programs”, IEEE Transactions on Software Engineering, 18(5): 423-433.

[17] Sandhu, Parvinder Singh, Sunil Kumar and Hardeep Singh, (2007), “Intelligence System for Software Maintenance Severity Prediction”, Journal of Computer Science, Vol. 3 (5), pp. 281-288, 2007

[18] Jang, J.-S. R., Sun, C.-T. and Mizutani, E., (2004), “Neuro-Fuzzy and Soft Computing- A Computational Approach to Learning and Machine Intelligence”, Pearson Education (Singapore) Pvt. Ltd., 1st Edition, 2004.

[19] M. Hall, “Correlation-based feature selection for discrete and numeric class machine learning”, In proceedings of the 17th International conference on Machine learning, 2000, pp. 359-366.

[20] Kaur, A. Malhotra, R., "Application of Random Forest in Predicting Fault-Prone Classes", ICACTE '08. International Conference on Advanced Computer Theory and Engineering, 2008, Phuket, Dec. 20-22, 2008, pp. 37-43

[21] Challagulla, V.U.B. , Bastani, F.B. , I-Ling Yen , Paul,( 2005) “Empirical assessment of machine learning based software defect prediction techniques”, 10th IEEE International Workshop on Object-Oriented Real-Time Dependable Systems, WORDS 2005, 2-4 Feb 2005, pp. 263-270.