Soft Computing Approaches for Prediction of Software Maintenance Effort

Dr. Arvinder Kaur
University School of Information Technology
GGS Indraprastha University
Delhi

Kamaldeep Kaur
University School of Information Technology
GGS Indraprastha University
Delhi

Dr. Ruchika Malhotra
University School of Information Technology
GGS Indraprastha University
Delhi

ABSTRACT
The relationship between object oriented metrics and software maintenance effort is complex and non-linear. Therefore, there is considerable research interest in development and application of sophisticated techniques which can be used to construct models for predicting software maintenance effort. The aim of this paper is to evaluate and compare the application of different soft computing techniques – Artificial Neural Networks, Fuzzy Inference Systems and Adaptive Neuro-Fuzzy Inference Systems to construct models for prediction of Software Maintenance Effort. The maintenance effort data of two commercial software products is used in this study. The dependent variable in our study is maintenance effort. The independent variables are eight Object Oriented metrics . It is observed that soft computing techniques can be used for constructing accurate models for prediction of software maintenance effort and Adaptive Neuro Fuzzy Inference System technique gives the most accurate model.

Categories and Subject Descriptors
D.4.8 [**Performance**]: Modeling and Prediction

Keywords
Software Maintenance Effort Prediction, Soft Computing, Object Oriented(OO) Metrics, Artificial Neural Networks(ANNs), Fuzzy Inference Systems(FIS), Adaptive Neuro-Fuzzy Inference System(ANFIS).

General Terms
Measurement and Performance.

1. INTRODUCTION
There are several empirical studies which show that there is a strong relationship between Object Oriented(OO) metrics and OO software quality attributes such fault proneness[1], maintenance effort[2] and testability[3]. We focus on maintenance effort in this study. Models that predict maintenance effort have been built using statistical techniques [2] and ANNs(Artificial Neural Networks) [4,5]. In our knowledge there are no significant research studies showing application of FIS(Fuzzy Inference Systems) and ANFIS(Adaptive Neuro-Fuzzy Inference Systems) to Software Maintenance Effort in OO paradigm. ANNs, FIS and ANFIS are universal approximators as they are capable of approximating

general non-linear relationships to high degree of accuracy [6][7][8]. ANNs possess advantages of low-level computational features, whereas FIS have advantages of reasoning on a high semantic level [9]. FIS allows relationships in data to be modeled by "if-then" rules that are easy to understand and verify. ANNs and FIS complement each other[9]. ANFIS introduced by Jang et al [8] is an FIS implemented in the framework of adaptive networks. It uses a hybrid learning procedure that combines optimizing the premise membership functions by gradient descent with optimizing the consequent equations by linear least square estimation[8]. But which of these three approaches is best suited for software maintenance effort prediction ? An attempt has been made in this research to find the best suited approach of these three approaches for constructing software maintenance effort prediction models.

Khoshgotaar et al. [10,11] introduced the use of ANNs and Fuzzy systems in empirical studies in software engineering. Fuzzy subtractive clustering has been used by them to generate rules for an FIS to predict number of faults in a legacy telecommunication system in the procedural paradigm[12] . Our study is conducted in the OO paradigm where principal components of OO design metrics are input to ANNs, FIS and ANFIS as predictors of maintenance effort. Various ANN architectures and training algorithms have been proposed in ANN literature. We also compare the accuracy of feedforward backpropagation neural networks trained with different training algorithms in this paper. The application of radial basis functions and generalized neural networks to software maintenance effort is also studied and compared. We use maintenance effort data of two commercial software products QUES(Quality Evaluation System) and UIMS(User Interface System)[2]. Our models aims to predict software maintenance effort by estimating number of lines changed per class. A change of line could be addition or deletion. A change in content of line is counted as addition followed by deletion.

This paper is organized as follows. Section 2 presents related work in this area. Section 3 presents the metrics studied and describes the source of the data. Section 4 presents the experimental methodology used in this study. The results are presented in Section 5.

2. RELATED WORK
There is some existing work on the use of soft computing techniques for predicting software quality in procedural and OO paradigm. Khoshgoftaar et al[10] used one hidden layer neural

network with backpropagation training algorithm to classify modules as fault prone or not. In their work [12] they used Fuzzy Subtractive Clustering to predict the number of faults. Aggarwal et al[13] have developed a fuzzy model for measuring software maintainability. The inputs to the model are comment ratio, average live variable , average life span and average cyclomatic complexity. The output of the model is average corrective maintenance time. They have used Mamdani style inference. Gyimothy et al[14] empirically validated Chidamber and Kamerer metrics on Open Source Software for fault prediction. They employed regression(linear and logistic) and machine learning methods(ANNs and decision trees) for predicting fault proneness. Thwin and Quah have used a Generalized Regression Neural Network for predicting software development faults [17] and software readiness[18]. However, none of the above studies compare the various soft computing techniques or make use of hybrid ANFIS approach. In this study we turn our attention to the use of hybrid approach which combines the advantages of ANNs and FIS. Moreover, we compare the predictive accuracy of three soft computing approaches.

3. RESEARCH BACKGROUND

3.1 Metrics Studied

The OO metrics are independent variables in our study and are summarized as under:-

LCOM(Lack of Cohesion)[15]. It counts number of null pairs of methods that do not have common attributes

DIT(Depth of Inheritance Tree)[15] The depth of a class with in the inheritance hierarchy is the maximum number of steps from the class node to the root of the tree and is measured by the number of ancestor classes.

WMC(Weighted Methods Per Class)[15] The WMC is a count of sum of complexities of all methods in a class. Consider a class K1, with methods M1,…….. Mn that are defined in the class. Let C1,…….Cn be the complexity of the methods.

$$\mathrm{WMC} = \sum_{i=1}^{n} C_i$$

NOC(Number of Children)[15] The NOC is the number of immediate subclasses of a class in a hierarchy

RFC(Response for a Class)[15] The response set of a class (RFC) is defined as set of methods that can be potentially executed in response to a message received by an object of that class. It is given by RFC=|RS|, where RS, the response set of the class, is given by

$$\mathrm{RS} = M_i \cup_{all\ j} \{R_{ij}\}$$

DAC (Data Abstraction Coupling)[2] Data Abstraction is a technique of creating new data types suited for an application to be programmed.

DAC = number of ADTs defined in a class.

MPC (Message Passing Coupling)[2] It counts the number of send statements defined in a class.

NOM (Number of Methods per Class) It counts number of methods defined in a class.

3.2 Empirical Data

The commercial software products UIMS (User Interface System) and QUES (Quality Evaluation System) data are used in this investigation, which is presented in [2]. The maintenance effort is measured by using the number of lines changed per class. A line change could be an addition or a deletion. A change of the content of a line is counted as a deletion followed by an addition. This measurement is used in this study to estimate the maintenance effort of the OO systems. UIMS system consists of 39 classes and QUES system consists of 71 classes.

4. EXPERIMENTAL METHODOLOGY

The maintenance effort data was normalized using z-score normalization technique so that input OO metrics and target(maintenance effort) have means of zero and standard deviations of one. Since the OO metrics have high correlation with each other, to uncorrelate the inputs, Principal Component Analysis(PCA) of the normalized input metrics was performed using singular value decomposition technique. Only those components were retained that contribute more than two percent to the variance in the data set. The application of this technique retained six principal components. The Principal components of the OO metrics are called domain metrics[10]. The transformed domain metrics data set was divided into training set and validation set in the ratio 3:1. ANNs, FIS and ANFIS models for Maintenance Effort Prediction were constructed using the training set and validated using validation set.

4.1 Maintenance Effort Modeling Using FeedForward Backpropagation Neural Networks

The first investigation was to compare the predictive accuracy of Feed – Forward Neural Network trained with variants of backpropagation training . The network architecture was empirically determined and was as follows:-

Table 1

Feed Forward Backprop Neural Network Architecture	
Hidden Layers	1
Hidden neurons	5
Activation function hidden layer	Hyperbolic Tangent
Output neurons	1
Activation function output neuron	Pure linear

This neural network was trained and validated for various feedforward backprop training algorithms available in Matlab Neural Network toolbox[20]. Fifteen algorithms are available in the toolbox. The Bayesian regularization algorithm trainbr was used in the work of [5]. We evaluate fourteen algorithms here. The inputs to the neural network were domain metrics. The output was maintenance effort. The predictive accuracy of training algorithms was compared using following performance

measures[19,20]:-
Mean Absolute relative error(MARE)[19]- This is the preferred measure used by software engineering researchers and is given as

$$MARE = \left(\sum_{i=1}^{n} \left| \frac{estimate - actual}{actual} \right| \right) \div n \quad (1)$$

where:

estimate is the network output for each observation n is the number of observations

Mean Relative Error (MRE)[19] – This measure is used to estimate whether models are biased and tend to overestimate or underestimate and is calculated as follows

$$MRE = \left(\sum_{i=1}^{n} \frac{estimate - actual}{n} \right) \div n \quad (2)$$

A large positive MRE would suggest that the model over estimates the number of lines changed per class, whereas a large negative value will indicate the reverse.

Correlation-"Correlation coefficient (R-value)[20] between the outputs and targets. It is a measure of how well the variation in the output is explained by the targets. If this number is equal to 1, then there is perfect correlation between targets and outputs".

p-values[20] –" p-values are used for testing the hypothesis of no correlation. Each p-value is the probability of getting a correlation as large as the observed value by random chance, when the true correlation is zero. If p is small, say less than 0.05, then the correlation R is significant".

The performance measures obtained on the validation data are summarized in table 2.

Table 2

Validation Results of Maintenance Effort Prediction accuracy with different Training Algorithms

Training Algorithm	MARE	MRE	R	p-value
Trainb	0.656	0.222	0.356	0.063
Trainbfg	0.714	-0.029	0.473	0.011
Trainc	0.374	-0.061	0.660	0.000
Traincgb	0.403	-0.159	0.681	0.000
Traincgf	0.514	0.221	0.549	0.003
Traincgp	0.456	0.065	0.541	0.003
Traingd	0.795	0.332	0.343	0.074
Traingda	0.441	0.091	0.560	0.002
Traingdm	0.808	0.293	0.303	0.117
Traingdx	0.368	-0.030	0.669	0.000
Trainlm	0.769	0.059	0.505	0.006
Trainoss	0.474	-0.126	0.665	0.000
Trainrp	0.521	0.172	0.372	0.051
Trainscg	0.479	0.109	0.370	0.053

For analyzing the results of Table 2 only those algorithms with a p-value less than 0.05 are selected as only these algorithms have significant correlations, as shown in Figure 1

From Table 2 and Figure 1 it is observed that out of the fourteen algorithms evaluated, traingdx algorithm, which is a gradient descent momentum and adaptive learning rate algorithm [20], has the best accuracy for maintenance effort prediction with a lowest MARE of 0.368, MRE of -0.030 and a correlation of 0.669.

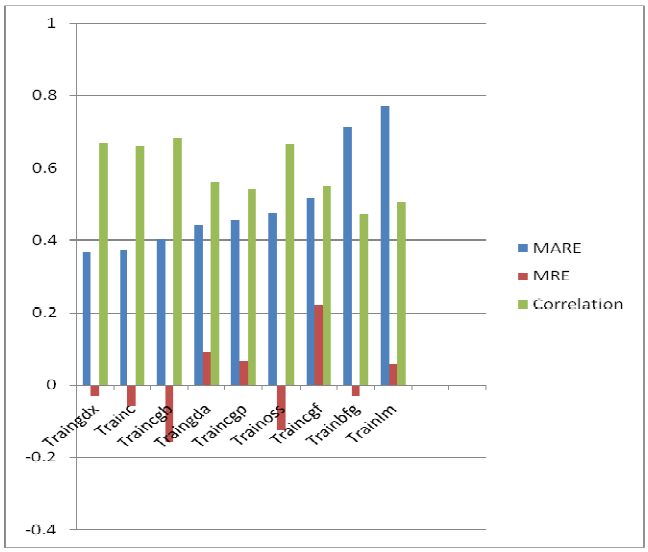

Figure 1 A Comparison of Performance Measures of different training algorithms for predicting software maintenance effort

4.2 Maintenance Effort modeling using Radial Basis Function(RBF) Neural Networks

The second investigation was to construct maintenance effort prediction models and compare their predictive accuracy using different radial basis function neural networks available in Matlab Neural Network toolbox[20] . Three radial basis functions are available in the toolbox. They are

i) "Exact design radial basis networks[20]- This method finds an exact solution. The function *newrbe* creates radial basis networks with as many radial basis neurons as there are input vectors in the training data".

ii) "Efficient design radial basis networks[20]-This method finds the smallest network that can solve the problem within a given error goal. Typically, far fewer neurons are required by newrb than are returned newrbe".

iii) "Generalized Regression Neural Network(GRNN) [20] – This ANN is often used for function approximation. It has a radial basis layer and a special linear layer It has been shown that, given a sufficient number of hidden neurons, GRNNs can approximate a continuous function to an arbitrary accuracy".

Three different RBF neural networks were created using the Matab Neural Network Toolbox Functions *newrbe, newrb* and *newgrnn*[20] using training data and tested using Matab Neural Network Toolbox Function *sim.*

A plot of actual and predicted maintenance effort on validation data is shown in figure 2

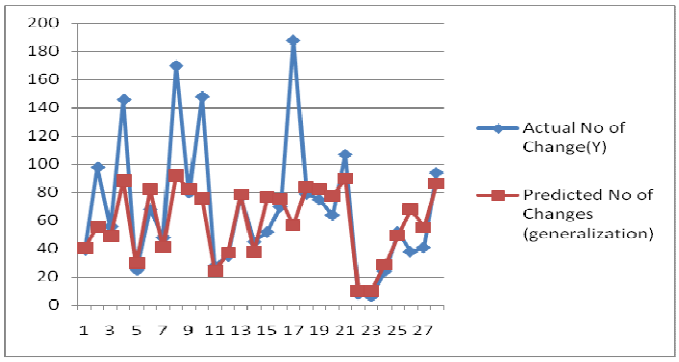

Figure 2 Actual Versus Predicted Maintenance Effort using a GRNN

The performance measures obtained on the validation data are summarized in Table 3.

Table 3

Validation Results of Maintenance Effort Prediction accuracy with different RBF Networks

RBF Network	MARE	MRE	R	p-value
Generalized Regression	0.255	0.017	0.669	0.000
RBF (Exact Design)	0.510	-0.250	0.280	0.148
RBF(Efficient Design)	0.487	0.186	0.482	0.007

From Table 3 it is observed that out of the three types of RBF networks available in Matlab Toolbox[20] ,GRNN has the best accuracy for maintenance effort prediction with an MARE of 0.255, MRE of 0.017 and a correlation of 0.669. The predictive accuracy of GRNN is better than a Feed –Forward Back Prop ANN trained with trainbr algorithm [20] (MARE 0.265, MRE 0.09) [5] and traingdx algorithm [20] (MARE 0.368, MRE -0.03) (Table 2).

4.3 Maintenance Effort modeling Using Fuzzy Inference System (FIS)

The third investigation was to construct a Maintenance Effort model using an FIS. The domain metrics data was divided in the ratio 3:1 for generating and evaluating the FIS. It is a general practice that rules for an FIS are generated from expert knowledge. In our study, expert knowledge regarding number of changes in each class of QUES and UIMS system was not available , so we considered generating rules from training data set. Our approach is to extract rules based on cluster estimation. Since we did not have any clear idea on how many clusters could be there in the data set so we used *Subtractive clustering*, [16], a fast, one-pass algorithm for estimating the number of clusters and the cluster centers in the data set. An important advantage of using clustering method for rule generation is that the resultant rules are more tailored to input data than they are in an FIS generated without clustering[20]. The FIS was generated using the Matlab Fuzzy logic Toolbox[20] Function *genfis2*[20]. Figure 3 shows the rules generated using this method. A total of 58 rules is generated. This reduces the problem of combinatorial explosion of rules when data has high dimensionality[20].

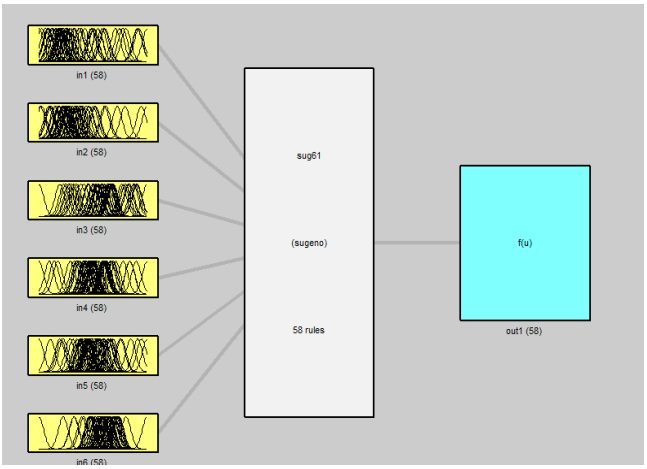

Figure 3 Rules for Fuzzy Inference System for Maintenance Effort Prediction

A plot of actual and predicted maintenance effort on validation data is shown in Figure 4

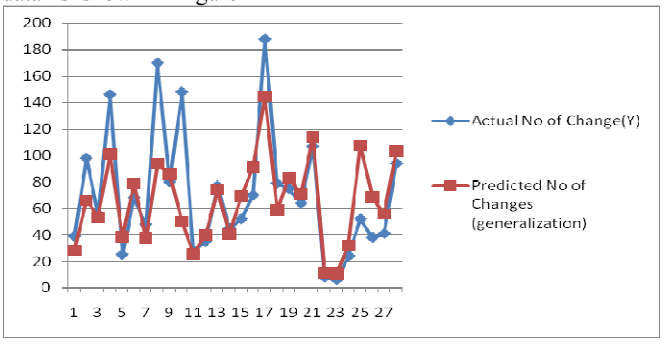

Figure 4 Actual Versus Predicted Maintenance Effort

using an FIS

The performance measures obtained on the validation data are summarized in Table 4.

Table 4

Validation Results of Maintenance Effort Prediction accuracy with Fuzzy Subtractive Clustering

MARE	0.308
MRE	0.093
Correlation coefficient	0.747
p-value	0.0000

From Table 4 it is observed that an FIS based on subtractive clustering for maintenance effort prediction with an MARE of 0.308 and correlation of 0.747 has better predictive accuracy than a Feed –Forward Back Prop ANN trained with trainbr algorithm[20] (MARE 0.265, MRE 0.09) [5] and traingdx[20] algorithm [20] (MARE 0.368, MRE -0.03) (Table 2) where as a predictive accuracy of a GRNN(MARE 0.255, MRE 0.017)(Table 3) is better than that of an FIS.

4.4 Maintenance Effort modeling Using Adaptive Neuro-Fuzzy Inference System(ANFIS)

The fourth investigation was to construct maintainability model using neuro-fuzzy technique. Neuro-fuzzy techniques provide a method for the fuzzy modeling procedure to learn information about a data set, in order to compute the membership function parameters that best allow the associated fuzzy inference system to track the given input/output data. "ANFIS is an adaptive network which permits the usage of neural network topology together with fuzzy logic. It not only includes the characteristics of both methods, but also eliminates some disadvantages of both when used separately[8]".

The Matlab Fuzzy Logic Toolbox function that accomplishes this membership function parameter adjustment is called *anfis[20]*.

The overall ANFIS model structure is shown in Figure 5. This model shows Six inputs that are the domain metrics, 58 rules, and one output, ,i.e, Maintenance Effort

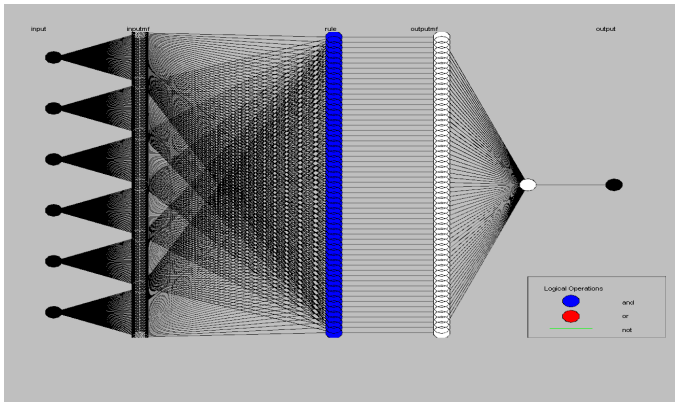

Figure 5 ANFIS Model Structure

A plot of actual and predicted maintenance effort on validation data are shown in Figure 6

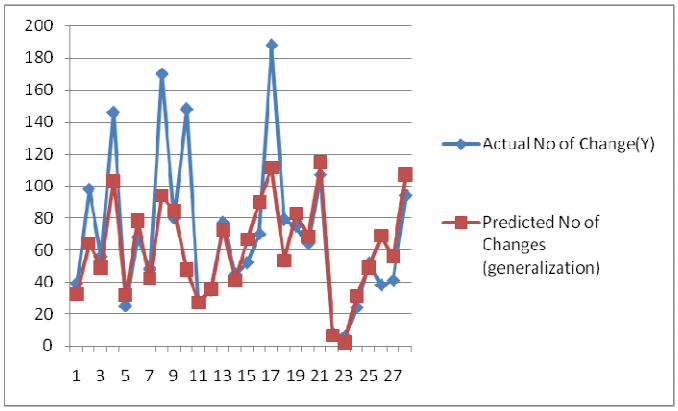

Figure 6 Actual Versus Predicted Maintenance Effort using an ANFIS

The performance measures obtained on the validation data are summarized in Table 5.

Table 5

Validation Results of Maintenance Effort Prediction accuracy with ANFIS

MARE	0.24235
MRE	-0.03025
Correlation coefficient	0.7578
p-value	0.0000

From Table 5 it is observed that an ANFIS for maintenance effort prediction with an MARE of 0.24235 and correlation of 0.7578 has better predictive accuracy than a Feed –Forward Back Prop ANN trained with trainbr algorithm[20] (MARE 0.265, MRE 0.09) [5] and traingdx[20] algorithm [20] (MARE 0.368, MRE -0.03) (Table 2) . This predictive accuracy is also

better than that of a GRNN(MARE 0.255, MRE 0.017)(Table 3) and also better than that of an FIS(MARE 0.308, MRE 0.093)(Table 3) .

5. CONCLUSION

In this work we evaluate and compare different soft computing techniques for prediction of software maintenance effort of commercial software systems. The graph in Figure 7 shows the MARE obtained with different soft computing approaches to compare their accuracy for prediction of software maintenance effort . The results show that the MARE of Feed forward Backbrop Artificial Neural Network model is 36.8% where as that of GRNN is 25.5% .The MARE of FIS model is 30.8%, while that of ANFIS model is 24.2%. It is concluded that the results of ANFIS are the best followed by that of a GRNN. Hence it is concluded that soft computing techniques can be successfully used for prediction of software maintenance effort. However, our results need to be generalized by conducting similar studies on maintenance data of other software systems.

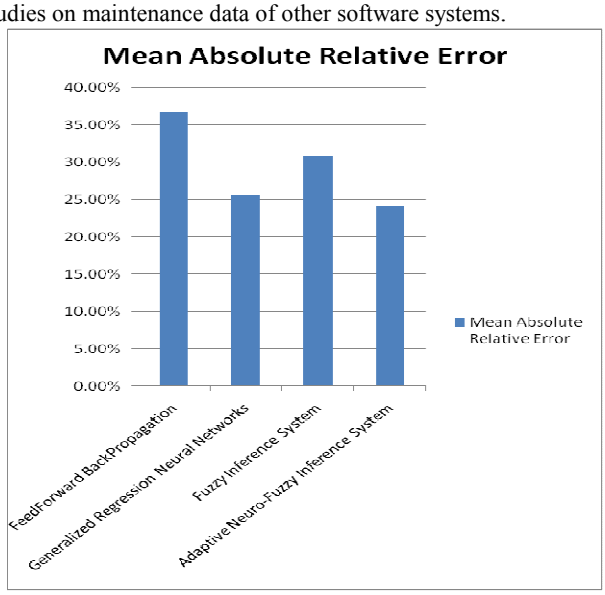

Figure 7 MARE using different soft computing techniques

6. REFERENCES

[1] V.Basili, L.Briand, W.Melo, "A Validation of Object-Oriented Design Metrics as Quality Indicators", IEEE Transactions on Software Engineering, vol. 22 no.10, pp. 751-761, 1996

[2] W.Li and S.Henry, "Object Oriented Metrics that Predict Maintainability", Journal of Systems and Software, vol 23 no.2, pp.111-122, 1993.

[3] S. R. Chidamber and C. F. Kemerer, "A Metrics Suite for Object Oriented Design," IEEE Transactions on Software Engineering, vol. 20, pp 476-493,1994.

[4] Thet Thwin Mie Mie, Tong-Seng Quah, "Application of Neural Networks for Estimating Software Maintainability Using Object-Oriented Metrics", Proceedings of the 15th International Conference on Software Engineering and Knowledge Engineering, 2003, pp. 69-73, San Francisco, U.S.A

[5] K. K. Aggarwal, Yogesh Singh, Arvinder Kaur, and Ruchika Malhotra, "Application of Artificial Neural Network for Predicting Maintainability Using Object Oriented Metrics," Proceedings of World Academy of Science, Engineering and Technology, vol 15, pp 285-289 2006.

[6] S. Haykins, "A Comprehensive Foundation on Neural Networks ," Prentice Hall, 1999

[7] L.-X. Wang and J. M. Mendel, "Fuzzy basis function, universal approximation, and orthogonal least squares learning," ***IEEE Trans. Neural Networks,*** vol. 3 no. **5,** pp. 807-814, Sept. 1992.

[8] Jyh Shing , Roger Jang, "ANFIS: Adaptive Network based Fuzzy Inference System", IEEE Transactions on Systems, Man and Cybernetics , vol. 23 ,no 3, pp 665-685,1993.

[9] S.N. Sivanadam, S. Sumathi, S.N. Deepa , "Introduction to Neural Networks Using Matlab 6.0 " , Tata McGraw Hill , NewDelhi, 2006.

[10] T.M. Khoshgoftaar, E.B.Allen, J.P. Hudephol and S.J. Aud," E.B.Allen, J.P. Hudephol and S.J. Aud ," Application of neural networks to quality modeling of a very large telecommunication system", IEEE Transactions on Neural Networks, vol.8,pp. 902-909,1997

[11] A.I Tagi, M. Khoshgoftaar, A. Abran, Can Neural Networks be easily interpreted in Software Cost Estimation, IEEE Transactions on Software Engineering, 1162-1167 ,Feb 2002

[12] Yuan X. , Khoshgoftaar T.M., Allen E.B., Ganesan K., " An Application of Fuzzy Clustering to Software Quality Prediction" Proceedings of IEEE Symposium on Application Specific Systems and Software Engineering Technology, pp85-90,2000

[13] K.K. Aggarwal, Yogesh Singh, Pravin Chandra and Manimala Puri," Measurement of Software Maintainability Using a Fuzzy Model**",** Journal of Computer Sciences,pp 538-542,2005

[14] Tibor Gyimothy, Rudolf Fernec, and Istvan Siket , " Empirical Validaion of Object –Oriented Metrics on Open Source Software for Fault Prediction" , IEEE Transactions on Software Engineering, vol. 31, no. 10, October 2005

[15] S.Chidamber, C. Kemerer, "Towards a Metrics Suite for Object Orienteddesign". Proc. Conference on Object-Oriented Programming: Systems,Languages and Applications (OOPSLA'91). Published in SIGPLAN Notices, vol 26 no. 11, pp.197-211, 1991

[16] Chiu S.L. , "Extracting Fuzzy Rules from Data for Function approximation and pattern Classification,"

Chapter 9 in Fuzzy Information Engineering: A Guided Tour of Applications John Wiley &Sons, 1997

[17] Jon T. S. Quah and Mie Mie Thet Thwin, "Prediction of Software Readiness Using Neural Network", Proceedings of 1st International Conference on Information Technology & Applications (ICITA 2002)

[18] Mie Mie Thet Thwin and Tong-Seng Quah, "Application of Neural Network for Predicting Software Development Faults Using Object-Oriented Design Metrics", Proceedings of ninth InternationalConference on Neural Information Processing, Singapore, 18-22 Nov, 2002, vol.5, pp. 2312-2316, 2002.

[19] G.Finnie and G. Witting, "AI Tools for Software Development Effort Estimation", International Conference on Software Engineering: Education and practice, 1996.

[20] www.mathworks.com.