ISSN No. 0976-5697

Volume 3, No. 7, Nov-Dec 2012

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Adaptive Neuro Fuzzy Inference System for Software Development Effort Estimation

Rama Sree P*
Dept. of Computer Science &Engineering,
Aditya Engineering College,
Surampalem, Andhra Pradesh, India,
ramasree_p@rediffmail.com

Prasad Reddy P.V.G.D
Dept. of Computer Science and Systems Engineering,
Andhra University,
Visakhapatnam, Andhra Pradesh, India,
prasadreddy.vizag@gmail.com

Sudha K.R
Dept. of Electrical Engineering,
Andhra University,
Visakhapatnam, Andhra Pradesh, India,
arsudhaa@gmail.com

Abstract: Software estimation such as cost estimation, effort estimation, quality estimation and risk analysis is a major challenge for Software Projects. The literature shows several algorithmic cost estimation models such as Boehm's COCOMO(Constructive Cost Model), Albrecht's' Function Point Analysis, Putnam's SLIM(Software Lifecycle Management), ESTIMACS(Macro Estimation Model) etc., where each model has its own pros and cons for estimation, there is still a need to find a model that gives accurate estimates. This paper is a modest attempt in explaining the soft computing models using Adaptive Neuro Fuzzy Inference System (ANFIS) which are designed to improve the performance of the network that suits to the COCOMO Model for software development effort prediction. ANFIS Models are created using Triangular, GBell, Trapezoidal and Gauss membership functions. A case study based on NASA 93 projects compares the proposed models with the Intermediate COCOMO. In the results which were analyzed using five different criterions Variance Accounted For (VAF), Mean Absolute Relative Error (MARE), Variance Absolute Relative Error(VARE), Mean Balance Relative Error (Mean BRE) and Prediction ,it is observed that the proposed ANFIS models combined with the neural network adaptive capabilities and the fuzzy inference system indicate a high level of efficiency with an accuracy of 99% and particularly ANFIS Model using Triangular Membership function provided better results.

Keywords: Effort Estimation; COCOMO; Fuzzy Logic; Neural Nets; ANFIS Models; NASA-93 Dataset

I. INTRODUCTION

The mathematical formulae which are derived based on some historical data are used [16] to predict costs and efforts in algorithmic cost estimation. The best known algorithmic cost model called COCOMO (COnstructive COst MOdel) developed from the analysis of 63 software projects was proposed in 1981 by Barry Boehm [6]. In the present paper, the main focus is on the Intermediate COCOMO which falls in Boehm's proposed three levels of the model called Basic COCOMO, Intermediate COCOMO and Detailed COCOMO.

A. *Intermediate COCOMO:*

The Basic COCOMO model is based on the relationship: Development Effort, DE = a*(SIZE)b where, SIZE is measured in KLOC. The constants a, b are dependent upon the 'mode' of development of projects. DE is measured in man-months or person/months. Boehm proposed 3 modes of projects:

a. ***Organic mode*** – simple projects that engage small teams working in known and stable environments.
b. ***Semi-detached mode*** – projects that engage teams with a mixture of experience. It is in between organic and embedded modes.
c. ***Embedded mode*** – complex projects that are developed under tight constraints with changing requirements.

The accuracy of Basic COCOMO [6] is limited because it does not consider the factors like hardware, personnel, use of modern tools and other attributes that affect the project cost. Further, Boehm proposed the Intermediate COCOMO that adds accuracy to the Basic COCOMO by multiplying 'Cost Drivers' into the equation with a new variable: EAF (Effort Adjustment Factor) has shown in Table I.

Table: 1 Development Effort (DE) For the Intermediate COCOMO

Development Mode	*Intermediate Effort Equation*
Organic	DE = EAF * 3.2 * $(SIZE)^{1.05}$
Semi-detached	DE = EAF * 3.0 * $(SIZE)^{1.12}$
Embedded	DE = EAF * 2.8 * $(SIZE)^{1.2}$

The EAF term is the product of 15 Cost Drivers [6] that are listed in Table III. The multipliers of the cost drivers are Very Low, Low, Nominal, High, Very High and Extra High. For example, for a project, if RELY is Low, DATA is High, CPLX is extra high, TIME is Very High, STOR is High and rest parameters are Nominal then EAF = 0.75 * 1.08 *1.65*1.30*1.06 *1.0. If the category values of all the 15 cost drivers are "Nominal", then EAF is equal to 1.

The 15 cost drivers [1] are broadly classified into 4 categories as in Table II.

Depending on the projects, multipliers of the cost drivers will vary and thereby the EAF may be greater than or less than 1, thus affecting the Effort. To decrease effort increase these cost drivers: ACAP, PCAP, AEXP, MODP, TOOL, VEXP, and LEXP. To decrease effort decrease these cost drivers: STOR, DATA, TIME, TURN, VIRT, CPLX and RELY.

Table: 2 COCOMO Cost Drivers

S.No	*Category*	*Cost Drivers*
1	PRODUCT	RELY- Required software reliability
		DATA- Data base size
		CPLX- Product complexity
2	PLATFORM	TIME - Execution time
		STOR - main storage constraint
		VIRT - virtual machine volatility
		TURN - computer turnaround time\
3	PERSONNEL	ACAP - analyst capability
		AEXP - applications experience
		PCAP - programmer capability
		VEXP - virtual machine experience
		LEXP - language experience
4	PROJECT	MODP - modern programming
		TOOL - use of software tools
		SCED–Required Development Schedule

Table: 3 Intermediate COCOMO Cost Drivers with multipliers

S. No	*Cost Driver Symbol*	*Very low*	*Low*	*Nominal*	*High*	*Very high*	*Extra high*
1	RELY	0.75	0.88	1.00	1.15	1.40	—
2	DATA	—	0.94	1.00	1.08	1.16	—
3	CPLX	0.70	0.85	1.00	1.15	1.30	1.65
4	TIME	—	—	1.00	1.11	1.30	1.66
5	STOR	—	—	1.00	1.06	1.21	1.56
6	VIRT	—	0.87	1.00	1.15	1.30	—
7	TURN	—	0.87	1.00	1.07	1.15	—
8	ACAP	—	0.87	1.00	1.07	1.15	—
9	AEXP	1.29	1.13	1.00	0.91	0.82	—
10	PCAP	1.42	1.17	1.00	0.86	0.70	—
11	VEXP	1.21	1.10	1.00	0.90	—	—
12	LEXP	1.14	1.07	1.00	0.95	—	—
13	MODP	1.24	1.10	1.00	0.91	0.82	—
14	TOOL	1.24	1.10	1.00	0.91	0.83	—
15	SCED	1.23	1.08	1.00	1.04	1.10	—

II. MACHINE LEARNING TECHNIQUES

In this section the main focus is on the categories of Machine learning like Fuzzy Logic, Neural Networks and then the proposed ANFIS Models.

A. *Fuzzy Logic:*

When the systems are not suitable for analysis by conventional approach or when the available data is uncertain, inaccurate or vague, a fuzzy model is used [17]. The Fuzzy logic [8] maps an input space to an output space using a list of if-then statements called rules [18]. All rules are evaluated in parallel not bothering the order of the rules. For writing the rules, the inputs and outputs of the system are to be identified. The inputs to the Intermediate COCOMO model data used for developing the Fuzzy Inference System (FIS) [11][19] are MODE and SIZE. The output is Fuzzy Development Effort.

a. *Advantages:*

a) The main advantage of using the fuzzy ranges is that it predicts the effort for projects that do not come under a precise mode i.e. comes in between 2 modes, where the situation cannot be handled using the COCOMO.
b) Fuzzy logic is tolerant of imprecise data.
c) Fuzzy logic is based on natural language.

b. *Disadvantages:*

a) As the whole work has to be redefined for a newer dataset it is hard to maintain a degree of meaningfulness
b) As the answers are confined to what is written in its rule base, it is incapable to generalize.
c) Demands the presence of an expert to write the rules.

B. *Neural Networks:*

Neural network, a massive parallel distributed processor [7] made up of simple processing units which has a natural propensity for storing experimental knowledge [3] and making it available for use, resembles the brain in two respects [10]:

a) Knowledge is acquired by the network from its environment through a learning process.
b) Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge.

a. *Advantages:*

a) Artificial neural networks is very useful in problems where there is a complex relationship between inputs and outputs as it can model complex non-linear relationships and approximate any measurable function.
b) Many different algorithms are available to choose from.

b. *Disadvantages:*

a) There is no clear guidance on how to design neural nets like for example how many hidden layers are to be present.
b) Accuracy depends on larger training dataset which is not always available.
c) They are effectively black boxes- once given the inputs; the generated outputs have to be accepted.

C. *Neuro Fuzzy Model:*

ANFIS , the acronym for Adaptive Neuro Fuzzy Inference System [8], using a given input/output data set, constructs a fuzzy inference system (FIS) whose membership function parameters are tuned (adjusted) using either a back propagation algorithm alone or in combination with a least squares type of method. This adjustment allows the fuzzy systems to learn from the data they are modeling. To interpret the input/output map, a network-type structure similar to that of a neural network, which maps inputs through input membership functions and associated parameters, and then through output membership functions and associated parameters to outputs, can be used.

The parameters associated with the membership functions whose computation is facilitated by a gradient vector changes through the learning process. This gradient vector provides a measure of how well the fuzzy inference system is modeling the input/output data for a given set of parameters. When the gradient vector is obtained, any of several optimization routines can be applied in order to adjust the parameters to reduce some error measure which is usually defined by the sum of the squared difference between actual and desired outputs.

The hybridization of neural networks and fuzzy logic, the basic idea behind the Neuro Fuzzy system is done in two ways: Fuzzy Neural Networks (FNN) and Neuro Fuzzy Systems (NFS). FNN is a neural network equipped with the capability of handling fuzzy information. NFS is a fuzzy system augmented by neural networks to enhance some characteristics like flexibility and adaptability. This paper mainly throws light on the second approach.

The Takagi-Sugeno Neuro Fuzzy system is the main ingredient of the paper which makes use of a mixture of back propagation to learn the membership functions and least mean square estimation to determine the coefficients of the linear combination in the rule's conclusions. The Takagi-Sugeno Neuro fuzzy system schema is depicted in Fig.1.

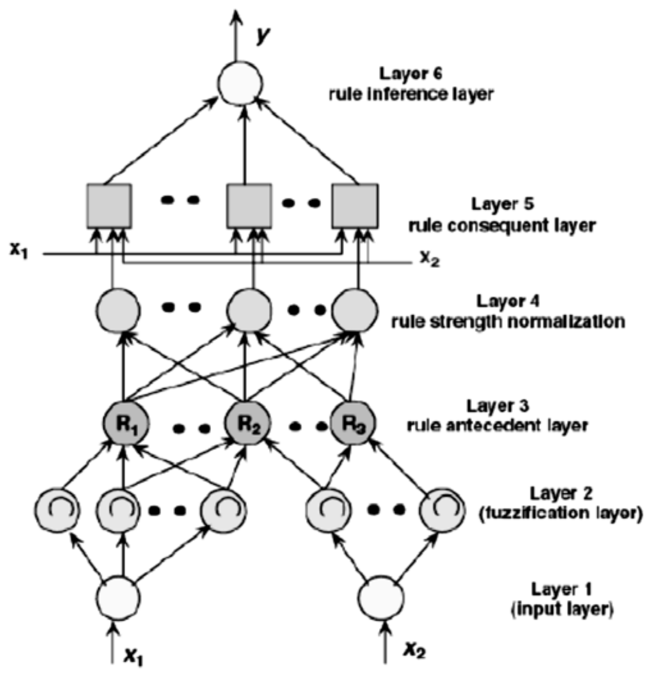

Figure 1. Takagi-Sugeno Neuro Fuzzy system

The first integrated hybrid Neuro Fuzzy model is ANFIS [13], and also due to Takagi-Sugeno rules implementation in ANFIS [2]; it has lowest Root Mean Square Error (RMSE) among the other Neuro Fuzzy models. So ANFIS was used here for implementing Neuro Fuzzy model [4]. In ANFIS, the adaptation (learning) process is only concerned with parameter level adaptation within fixed structures. The objective of the parameter-learning phase is to adjust parameters of the fuzzy inference system (FIS) such that the error function during training dataset reaches minimum or is less than a given threshold.

D. Membership Functions:

A membership function (MF) is a curve that defines how each point in the input space is mapped to a membership value (or degree of membership) between 0 and 1 [12]. The input space is also called as the universe of discourse. For our problem, we have used 4 types of membership functions:

a) Triangular membership function(TriMF)
b) GBell membership function(GBellMF)
c) Trapezoidal Membership function(TrapMF)
d) Gauss Membership function(GaussMF)

III. VARIOUS CRITERIONS FOR ASSESSMENT OF ESTIMATION MODELS

A. Variance Accounted For (VAF):

$$VAF\ (\%) = \left(1 - \frac{\operatorname{var}\left(E - \hat{E}\right)}{\operatorname{var} E}\right) \times 100 \qquad (1)$$

B. Mean Absolute Relative Error (MARE):

$$MARE\ (\%) = \frac{\sum f(R_E)}{\sum f} \times 100 \qquad (2)$$

C. Variance Absolute Relative Error (VARE):

$$VARE\ (\%) = \frac{\sum f(R_E - meanR_E)}{\sum f} \times 100 \qquad (3)$$

D. Prediction (n)

Prediction at level n is defined as the % of projects that have absolute relative error less than n.

E. Balance Relative Error (BRE):

$$BRE = \frac{\left|E - \hat{E}\right|}{\min(E, \hat{E})} \qquad (4)$$

E= estimated effort $\hat{E}$ = actual effort

$$\text{Absolute Relative Error (RE)} = \left|\frac{\hat{E} - E}{\hat{E}}\right| \qquad (5)$$

A model which gives higher VAF is better than that which gives lower VAF [9]. A model which gives lower MARE [8] is better than that which gives higher MARE [5] .A model which gives lower VARE is better than that which gives higher VARE. A model which gives lower BRE is better than that which gives higher BRE [5]. A model which gives higher Pred (n) is better than that which gives lower Pred (n) [14][20][21].

IV. EXPERIMENTAL STUDY

Out of 93 projects chosen from different NASA Centre's of the NASA 93 database [15], 83 projects are randomly selected and used as training data. The Model is trained for maximum of 200 epochs. The Model is tested using the entire dataset. The estimated efforts using Intermediate COCOMO, ANFIS Models using TriangularMF, GBellMF, Trapezoidal MF and GaussMF are shown for some sample projects in Table IV. The Effort is calculated in man-months. Table V and Fig.2 & Fig.3 show the comparisons of various models basing on different criterions.

Table: 4 Maintaining the Integrity of the Specifications

Project ID	*Actual Effort*	*COCOMO Effort*	*TriMF Effort*	*GBellMF Effort*	*TrapMF Effort*	*GaussMF Effort*
2	117.6	95.2	120.7	114.9	127.4	111.7
4	36.0	27.8	39.9	41.7	40.8	42.2
6	8.4	6.4	9.0	9.7	10.4	9.1
8	352.8	290.5	352.8	352.7	352.8	352.6
11	24.0	10.5	27.0	27.7	28.9	27.8
19	48.0	29.4	78.3	75.2	73.5	75.8
24	90.0	62.2	78.3	75.2	73.5	75.8
26	48.0	31.0	51.9	49.9	49.7	50.3
33	18.0	17.8	24.3	24.2	26.3	24.4
39	42.0	35.4	38.7	40.6	39.7	41.0
40	114.0	85.5	87.7	84.3	83.8	85.6
59	4560.0	24726	4559	4560	4559	4560
72	300.0	464.0	298.6	298.2	201.5	296.3
77	1200	2727	1200	1200	1200	1200
85	4178	3555	4178	4178	4178	4178
93	38.0	37.1	37.9	38.0	38.0	38.0

Table: 5 Comparison of Various Model

Model	*VAF*	*MARE*	*VARE*	*Mean BRE*	*Pred(30)%*
COCOMO	33.64	47.22	46.89	0.77	53
ANFIS-TriMF	99.14	15.06	5.91	0.20	81
ANFIS-GBellMF	99.09	15.78	6.56	0.20	77
ANFIS-TrapMF	98.70	29.86	171.49	0.35	73
ANFIS-GaussMF	99.13	15.14	5.99	0.20	78

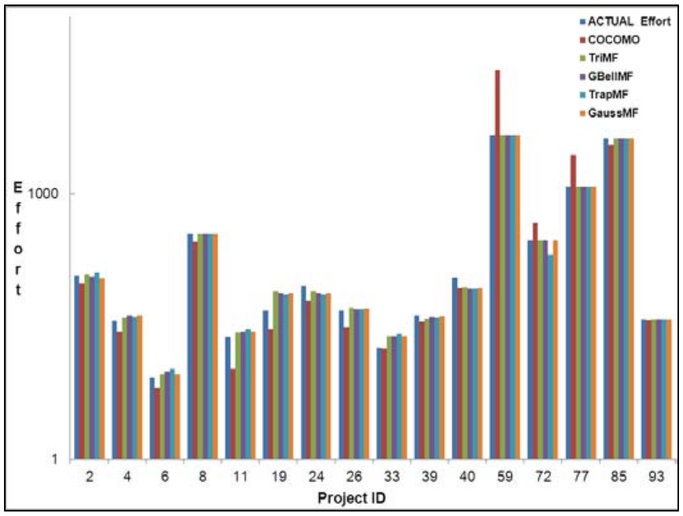

Figure 2. Actual Effort versus Estimation Effort using various models for selected Projects

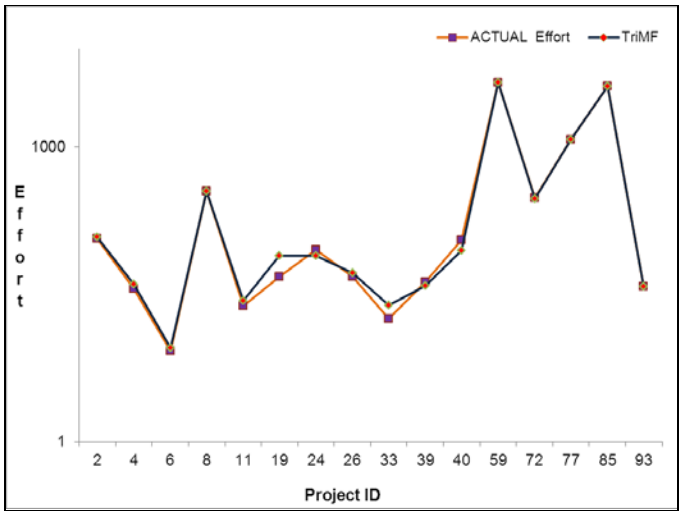

Figure 3. Actual Effort versus Estimation Effort using TriMF for selected Projects

V. CONCLUSION

From Table V, it is clear that Neuro Fuzzy Model using TriMF yields better results for maximum criterions when compared with the other models. As per results based on VAF, MARE, VARE, Mean BRE, & Pred (30), ANFIS Model using TriMF is best suitable. To conclude, it would be better to create an ANFIS Model using Triangular MF for some training data and use it for effort estimation for all the other projects.

VI. REFERENCES

[1] A. Idri, T. M. Khoshgoftaar, & A. Abran, Can neural networks be easilyinterpreted in software cost estimation, , IEEE Trans, Software Engineering, Vol. 2, pp. 1162 – 1167, 2000.

[2] A. Kabla, Adaptive Neuro-Fuzzy Inference System for Financial Trading using Intraday Seasonality Observation

Model, World Academy of Science, Engineering and Technology 58, 2009.

[3] Alain Abran, Ali Idri, & Samir Mbarki,. Validating and Understanding Software Cost Estimation Models based on Neural Networks, IEEE, 0-7803-8482-2/04, 2004.

[4] Ammar A. Aldair, & Weiji Wang, FPGA Based Adaptive Neuro FuzzyInference Controller for Full VehicleNonlinear Active Suspension Systems, International Journal of Artificial Intelligence & Applications (IJAIA), Vol.1, No.4, October 2010.

[5] Angelis L, Stamelos I & Morisio M, Building software cost estimation model based on categorical data. Software Metrics Symposium, Seventh International, Volume 4-15, 2001.

[6] B.W. Boehm, Software Engineering Economics. Prentice-Hall, Englewood Cli4s, NJ, 1981.

[7] Dawson, C.W., A neural network approach to software projects effort estimation. Transaction: Information and Communication Technologies, Volume 16, pages 9, 1996.

[8] Dr. Arvinder Kaur, Kamaldeep Kaur, & Dr. Ruchika Malhotra, Soft Computing Approaches for Prediction of Software Maintenance Effort. International Journal of Computer Applications,Vol.1, No. 16, 2010.

[9] Harish Mittal, & Pradeep Bhatia, Optimization Criteria for Effort Estimation using Fuzzy Technique. CLEI Electronic Journal, Vol 10, No 1, Paper 2, 2007.

[10] LiMin Fu., Neural Networks in Computer Intelligence. Tata McGraw-Hill Edition pp.94-97, 2003.

[11] Prasad Reddy P.V.G.D, Sudha K.R, Rama Sree P, & Ramesh S, Fuzzy Based Approach for Predicting Software Development Effort. International Journal of Software Engineering, Vol 1,Issue 1,pp 1-11, 2010.

[12] Prasad Reddy P.V.G.D, Sudha K.R , Rama Sree P., Ramesh S, Software Effort Estimation using Radial Basis and Generalized Regression Neural Networks. Journal of Computing, Vol 2, Issue 5, pp 87-92, 2010.

[13] R. Jang, ANFIS: Adaptive network-based fuzzy inference system. IEEETransactions on Systems, Man and Cybernetics, Vol. 23 (3) pp 665-685, 1993.

[14] Xishi Huang, Danny Ho, Jing Ren, & Luiz F. Capretz, Improving the COCOMO model using a neuro-fuzzy approach. Science Direct, Applied Soft Computing 7, 29–40, 2007.

[15] NASA-93DatasetSource: http://promisedata.org/repository/data/nasa93/nasa93.arff, Jairus Hihn, JPL, NASA, Manager SQIP Measurement &Benchmarking Element, Feb 8 2006

[16] Jainendra K. Navlakha, “Choosing a Software Cost Estimation Model for Your Organization: A Case Studv ”, Elsevier Science Publishers B.V. (North-Holland), 1990,255-261

[17] R. Babuska, Fuzzy Modeling For Control, Kluwer Academic Publishers, Dordrecht, 1999

[18] Moshood Omolade Saliu, Adaptive Fuzzy Logic Based Framework for Software Development Effort Prediction, King Fahd University of Petroleum & Minerals, April 2003

[19] Iman Attarzadeh and Siew Hock Ow, Software Development Effort Estimation Based on a New Fuzzy Logic Model, IJCTE, Vol. 1, No. 4, October 2009.

[20] Prasad Reddy P.V.G.D, Sudha K.R, Rama Sree P, “Application of Fuzzy Logic Approach to Software Effort Estimation”, International Journal of Advanced Computer Science & Applications, Vol 2, Issue 5, pp 87-92, May 2011.

[21] Prasad Reddy P.V.G.D, Sudha K.R, Rama Sree P, “ Prediction of Software Development Effort Using RBNN And GRNN”, International Journal of Computer Science Engineering and Technology, Vol 1, Issue 4, pp 185-190, May 2011.