Volume 2, Issue 10, October 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

# A Review of Estimating Development Time and Efforts of Software Projects by Using Neural Network and Fuzzy Logic in MATLAB

**Surendra Pal Singh*, Prashant Johri**
*Department of Computer Science & Engineering*
*Nims Institute of Engineering and Technology, NIMS University, Rajasthan, Jaipur (India)*

***Abstract:*** ***Software estimation accuracy is among the greatest challenges for software developers. Software development effort estimation is one of the most major activities in software project management. A number of models have been proposed to make software effort estimations but still no single model can predict the effort accurately. The need for accurate effort estimation in software industry is still a challenge. It can analyze the project decisions like resource allocation and bidding which can be used to complete the project with respect to time/within the scope of the time. It gives estimation about the cost and time required for software development. It can be implemented through various estimation techniques and estimation models. In this paper, we proposed a new model using fuzzy logic in-order to estimate the most important factors of software effort estimation such as cost and time and neural network models used for carrying out the effort estimations for developing a software project & this field of Soft Computing is suitable in effort estimations. We use MATLAB to determine the parameters of various time estimation models. The performance of model is evaluated on published software projects data. A simple review of our models with existing ubiquitous models is shown in this paper.***

***Keywords*— *Neural Network, Fuzzy Logic, Software Estimation, Soft Computing, Matlab.***

## 1. INTRODUCTION

Software effort estimation is the process of predicting the amount of time (Effort) required to build a software system. Software time estimating has been an important but difficult task since the beginning of the computer era in the 1940s. As software applications have grown in size and importance, the need for accuracy in software time estimating has grown, too. In recent years, software has become the most expensive component of computer system projects.

ANY existing research papers have proposed various estimation techniques, but no single software development estimation technique is the best for all situations [1]. A careful comparison of the results of the several approaches is most likely to choose the best one and produce realistic estimates [2]. The neural network research started in the 1940s, and the fuzzy logic research started in the 1960s, but the neuro-fuzzy research area is relatively new [3]. The objective of this paper is to present a feasible way of combining fuzzy logic and neural networks for achieving higher accuracy.

In order to perform cost-benefit analysis, time estimation is to be performed by client or developer. Cost Estimation is achieved in terms of person-months (PM), which can be translated into actual dollar cost. Estimation carries inherent risk and this risk leads to obscurity. The different obscurity factors are project complexity, project size etc. The concept of software cost estimation has been growing rapidly due to practicality and demand for it.

Today the peoples are expecting high quality software with a low cost, the main objective of software engineering. So many popular cost estimation models like COCOMO81, COCOMOII, SLIM, FP, Delphi, Halsted Equation, Bailey-Basili, Doty, and Anish Mittal Model had came into existence. These models are created as a result of regression analysis and power regression analysis methods applied to historical data. A recent review of surveys on software cost estimation found that software projects have cost overruns.

Currently, no cost estimation model has integrated in its modeling process the above three criteria. In our recent research, we have developed an estimation model based on reasoning by analogy, fuzzy logic, and possibility theory to satisfy to two first criteria. Further research work has been initiated to look at the integration of the third criterion, concerning learning capabilities, in our model.

## 2. NEURAL NETWORK MODEL

Neural network techniques are based on the principle of learning from historical data, whereas fuzzy logic is a method used to make rational decisions in an environment of uncertainty and vagueness. However, fuzzy logic alone does not enable learning from the historical database of software projects. Once the concept of fuzzy logic is incorporated into

 

neural network, the result is a neuro-fuzzy system that combines the advantages of both techniques [4]. A software tool (MATLAB 7.4) was used to process the fuzzy logic, neural network and neuro-fuzzy systems.

Artificial neural networks can model complex non-linear relationships and approximate any measurable function. They can be used as an effective tool for pattern classification and clustering [8]. They are particularly useful in problems where there is a complex relationship between an input and output. It has been established that a one hidden layer feed forward network with (sufficient number of) sigmoidal nodes can approximate any continuous mapping with arbitrary precision [2-4]. The feed forward multi layer network is a network in which no loops occur in the network path. A learning rule is defined as a procedure for modifying the weight and biases of a network with the objective of minimizing the mismatch between the desired output and the obtained output from the network for any given input. The learning rule / network training algorithm is used to adjust the weights and biases of the network in order to move the network outputs close to the targets. The classical backpropagation algorithm was the first training algorithm developed [9]. The simplest implementation of backpropagation learning updates the network weights and biases in the direction in which the performance function decreases most rapidly - the negative of the gradient [10] though second order optimization algorithms like the conjugate gradient, the Levenberg-Marquardt and Baysian learning algorithms have also been developed.

A four input and one output network is used. The network uses only one hidden layer. The activation functions at the hidden layer and the output layers are the tangent - hyperbolic (tanh) function. The network inputs are (a) The function point count for projects, (b) the team size, (c) the level of the language used in development and (d) the function point standard. The block diagram of the network used is shown in Fig.1.

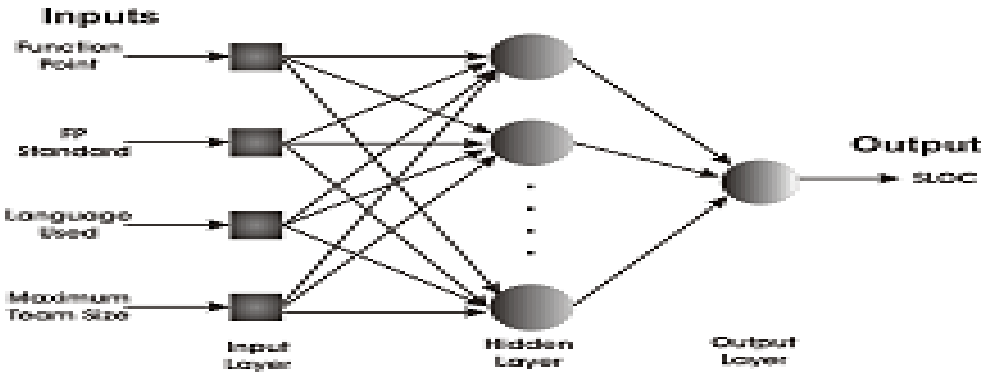


**Fig.1: Neural Network Model**

Fig. 1 shows a model whose inputs are function points, language used, FP standard and maximum team size. The output (target) is source lines of code.

## 3. FUZZY LOGIC MODEL

The theory of Fuzzy Logic was first raised by the mathematician Lotfi A. Zadeh in 1965. This theory is a result of the insufficiency of Boolean algebra to many problems of the real world. As most of the information in the real world is imprecise, and one of humans' greatest abilities is to effectively process imprecise and "'fuzzy" information.

According to the Oxford English Dictionary, the word Fuzzy is defined as blurred, indistinct, imprecisely defined, confused or vague. Fuzzy systems are knowledge based or rule based systems. The heart of a fuzzy system is a knowledge base consisting of the so called fuzzy IF-THEN rules. A fuzzy IF-THEN rule is an IF-THEN statement in which some words are characterized by continuous membership functions.

During the early nineties, fuzzy logic was firmly grounded in terms of its theoretical foundations and application in the various fields in which it was being used, such as robotics, medicine, and image processing. A fuzzy set [11] is a set with a graded membership function, □ , in the real interval [0, 1]. This definition extends the one of a classical set where the membership function is in the couple {0, 1}. Fuzzy sets can be effectively used to represent linguistic values such as *low*, *young*, and *complex*, in the following two ways:

**3.1 Fuzzy sets** that model the gradual nature of properties, i.e., the higher the membership that a given property x has in a fuzzy set A, the more it is true that x is A. In this case, the fuzzy set is used to model the vagueness of the linguistic value represented by the fuzzy set A.

**3.2 Fuzzy sets** that represent incomplete states of knowledge. In this case, the fuzzy set is a possibility distribution of the variable X, and consequently, is used to model the associated uncertainty. When considering that it is only known that x is A, and x is not precisely known, the fuzzy set A can be considered as a possibility distribution, i.e., the higher the membership x' has in A, the higher the possibility that x = x'.

 

Two fuzzy models [18, 22] are available to measure maintainability but the first one [18] has been used here.

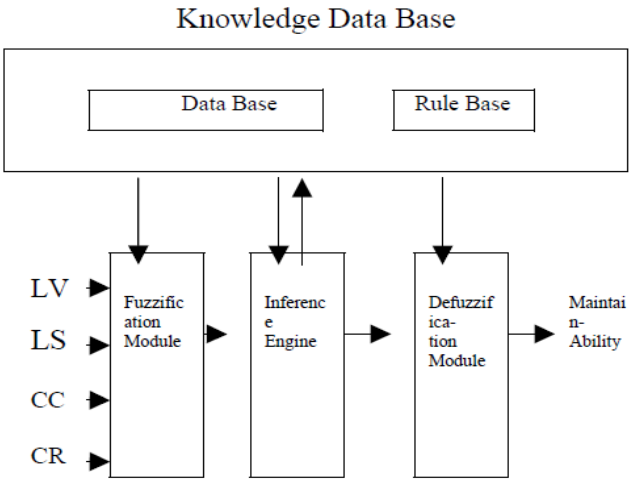


**Fig 2: Fuzzy Model for Software maintainability**

Modeling is done as follows:
**(i)** All the inputs and outputs were fuzzified...
**(ii)** All possible combination of inputs were considered

## 4. LITERATURE REVIEW

Venkatachalam [11] investigated the application of ANN to cost estimation. Other latest works using neural networks (NN) in cost estimation are reported in Refs. [11,12]. ANN is able to generalize from trained data set. Over a known set of training data, an ANN learning algorithm constructs mappings that fit the data, and fits previously unseen data in a reasonable manner as well [6]. A marriage between NN and FL, Neurofuzzy, was introduced into cost estimation in Ref. [2]. A Neurofuzzy system can combine the linguistic attributes of a fuzzy system with the learning and modeling attributes of a NN. EC has also recently found its usefulness in software effort estimation. Burgess and Lefley [19] applied genetic programming (GP) to software effort estimation. EC simulates evolution on a computer.

Chulani et al. [12] applied Bayesian analysis in calibrating the 1998 version of COCOMO II model to 161 data-points. When compared with the 1997 calibration done using multiple regressions, the Bayesian approach was adjudged to perform better and more robust. Bayesian analysis was also used in the calibration of the 2000 version of COCOMO II by Boehm et al. [9]. The result of this was a higher predictive accuracy.

H.K.Verma et.al [13] Sharma present an enhanced fuzzy logic based framework for software development effort prediction. The intermediate COCOMO is extended in the proposed study by incorporating the concept of fuzziness into the measurements of size, mode of development for projects and the cost drivers contributing to the overall development effort. The said framework tolerates imprecision, incorporates experts knowledge, explains prediction rationale through rules, offers transparency in the prediction system, and could adapt to changing environments with the availability of new data. Ch. Satyananda Reddy et al. [14] proposed to use Gaussian Membership Function (GMF) for the cost drivers by studying the behaviour of COCOMO cost drivers.

Prasad Reddy P.V.G.D et al. [15] implemented a software development effort prediction model using Fuzzy Triangular Membership Function and GBell Membership Function and then is compared with COCOMO model.

Non-algorithmic techniques include Expert Judgment and Machine Learning approaches [16].Machine Learning approaches include techniques that are based on Fuzzy Logic, Regression Tress, Analogy, Rule Induction, Bayesian Belief Networks and Neural Networks [3].FL with its offerings of a powerful linguistic representation can represent imprecision in inputs and outputs, while providing a more expert knowledge-based approach to model building.
Attempts have been made to fuzzify some of the existing algorithmic models in order to handle uncertainties and imprecision problems in such models. The first realization of the fuzziness of several aspects of one of the best known

 

[9], most successful and widely used model for cost estimation, COCOMO, was that of Fei and Liu [13]. Fei and Liu observed that an accurate estimate of delivered source instruction (KDSI) cannot be made before starting the project. Therefore, it is unreasonable to assign a determinate number for it. Jack Ryder investigated the application of fuzzy modeling techniques to two of the most widely used models for effort prediction; COCOMO and the Function- Points models, respectively. Idri and Abran applied FL to the cost drivers of intermediate COCOMO model. The application of FL to represent the mode and size as input to COCOMO model was later presented by Musilek et al. In Ref. [12]. Musilek et al. presented a two-stage implementation called simple F-COCOMO model and augmented F-COCOMO model, respectively. Some relevant studies in application of soft computing methodologies towards software development effort estimation are discussed as follows. A study by Hodgkinson and Garratt claims that estimation by expert judgment was better than all regression-based models [2].

MacDonell et al. [17] explored an expert knowledge based application of FL to effort prediction. This particular research has evolved into the development of a tool, FULSOME, to assist project managers in making predictions. MacDonell also applied fuzzy modeling to software source code sizing in Ref. [9].

## 5. PROPOSED WORK

Various researchers [9-13] have used neural network models for effort/cost estimation. Neural networks have also been effectively used for software testing problems.[15-22].Various training algorithms can be used to train the network, but which one is the best suited for software engineering applications?

## 6. CONCLUSIONS

In this paper we analysis neural network using Bayesian Regularization training algorithm produces reduced condition numbers as compared to the fuzzy model having membership functions whose derivatives show discontinuities at some points. In fact it is at these points that the condition number increases. Therefore it is concluded from this work that the neural network model using Bayesian Regularization training algorithm is a more stable model than the fuzzy model having membership functions whose derivatives show discontinuities at some points for all applications and specifically to software engineering applications.

## 7. FUTURE WORK

By using neural networks can be used in Fuzzy Analogy to integrate a supervised learning approach by comparing the estimated and the actual values. However, there are some shortcomings that prevent neural networks from being well accepted in cost estimation modeling. The most important is that they are considered as a 'black boxes'. Consequently, it is not easy to understand and explain their process. To avoid this limitation, we have studied the interpretation of a cost estimation model based on a Backpropagation three-layer Perceptron network. This study has shown promising results by using fuzzy logic and the possibility theory in its estimation process, Fuzzy Analogy satisfies the two first criteria, i.e., the tolerance of imprecision when describing software project, and the uncertainty when estimating the development cost. The third criterion of an intelligent model that Fuzzy Analogy has not yet incorporated into its process is to learn from previous experiences. The learning criterion is required for any cost estimation model. Indeed, the software industry is continuously evolving: engineers use more and more high level programming languages, application generators, web-based technology, etc.

**ACKNOWLEDGEMENT**

The author is thankful to the Department of Computer Science and Engineering, Nims Institute of Engineering and Technology, NIMS University, Jaipur (Rajasthan) for providing the computing facilities to carry out this work. He is also thankful to his guide Dr. Prashant Johri and friend Naveen Bilandi.

## REFERENCES

[1] Anish Mittal, Kamal Parkash, Harish Mittal, "Software Cost Estimation using fuzzy logic", ACM SIGSOFT Software Engineering Notes, November 2010 Volume 35 Number 1.
[2] Robert W. Zmud, Chris F. Kemerer, "An Empirical Validation of Software Cost Estimation Models" Communication of the ACM Vol 30 No 5, May 1987.
[3] B.Boehm, Software Engineering Economics Englewood Cliffs, NJ, Prentice Hall, 1981.
[4] B. Boehm., Cost Models for Future Life Cycle Process: COCOMO2. Annals of Software Engineering. 1995
[5] Pankaj jalote, "An Integrated Approach for Software Engineering.", Third Edition. ISBN: 978-81-7319-702-4.
[6] A. Porter and **R.** Selby, "Empirically-guided software development using metric-based classification **trees,''** ***IEEE Software,*** vol. ***7,*** pp. 46-54, Mar. 1990.
[7] A. Porter and R. Selby, "Evaluating techniques for generating metricbased classification trees," ***J. Syst. Software,*** vol. 12, pp. 209-218, July 1990.
[8] L. H. Putnam, "A general empirical solution to the macro softwaresizing and estimating problem," ***IEEE Trans. Software Eng.,*** vol. 4, pp. 345-361, 1978.
[9] R. Selby and A. Porter, ''Learning from examples: Generation and evaluation of decision trees for software resource analysis," ***IEEE Trans. Software Eng.,*** vol. 14, pp. 1743-1757, 1988.

 

[10] S. Vicinanza, M. J. Prietulla, and T. Mukhopadhyay, "Case-based reasoning in software effort estimation," in ***Proc. 11th Int. Con$ Info.*** **SJsr.,** 1990, pp. 149-158.
[11] A.R. Venkatachalam, Software cost estimation using artificial neural networks, in: Proceedings of the International Joint Conference on Neural Networks, 1993, pp. 987–990.
[12] S. Chulani, B. Boehm, B. Steece, Calibrating software cost models using bayesian analysis, Technical Reports, USC-CSE-98-508, University of Southern California Center for Software Engineering, 1998.
[13] Harsh Kumar Verma, Vishal Sharma "Handling Imprecision in Inputs using Fuzzy Logicto Predict Effort in Software Development" , IEEE 2010
[14] Ch. Satyananda Reddy, KVSVN Raju " An Improved Fuzzy Approach for COCOMO's Effort Estimation using Gaussian Membership Function"" Journal Of Software, Vol. 4, No. 5, July 2009
[15] 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 And Applications.Vol.2.No.5.2011.
[16] C. Schofield, Non-algorithmic effort estimation techniques, Technical Reports, Department of Computing, Bournemouth University, England, TR98-01, March 1998.
[17] S.G. MacDonell, A.R. Gray, M.J. Calvert, FULSOME: a fuzzy logic modeling tool for software metricians, in: Proceedings of the 18th International Conference of the North American Fuzzy Information Processing Society—NAFIPS, IEEE Press, New York, 1999,pp. 263–267.
[18] S. Weiss and C. Kulikowski, ***Computer Systems that Learn.*** San Mateo, CA: Morgan Kaufmann, 199 1.
[19] B.W. Boehm, "*Software Engineering Economics*", Prentice-Hall, Englewood Cliffs, NJ, 1981.
[20] B.W. Boehm, E. Horowitz, R. Madachy, et al., "*Software Cost Estimation with COCOMO II*", Prentice-Hall, Upper Saddle River, NJ, USA, 2000.
[21] L. H. Putnam, "A general empirical solution to the macro sizing and estimating problem", *IEEE Transaction on Software Engineering*, vol. 4, no. 4, pp. 345–361, 1978.
[22] K. V. Kumar, V. Ravi, M. Carr, N. Raj Kiran, "Software development cost estimation using wavelet neural networks". *Journal of Systems and Software*, vol. 81, no. 11, pp. 1853–1867, November 2008.
[23] Y.S. Su, C.Y. Huang, "Neural-network-based approaches for software reliability estimation using dynamic weighted combinational models".
[24] *Journal of Systems and Software*, vol. 80, no. 4, pp. 606-615, April 2007.
[25] K.K. Shukla. "Neuro-genetic prediction of software development effort", *Information and Software Technology*, vol. 42, no. 10, pp 701–713, July 2000.

 