Prediction of software development effort estimation using neural networks

[1]T. M. Kiran Kumar, [2]Yashaswini T K

[1]Assistant Professor, [2]Project Student
Department of MCA
Siddaganga Institute of Technology, Tumkur 572103, India

***Abstract*- Failures of software are mainly due to the flawed project management practices, which include effort estimation. Continuous changing outlines of software development technology make effort estimation more challenging. Several methods are available in order to estimate the effort among which soft computing based method plays a prominent role. Software cost estimation deals with lot of uncertainty among all soft computing methods neural network is good in handling uncertainty. This paper proposes a BPNN to utilize improves effort estimation for Cocomo data set. In this paper MRE & MMRE are used as the evaluation criteria.**

***Keywords* - *E*ffort estimation, Neural networks, BPNN**

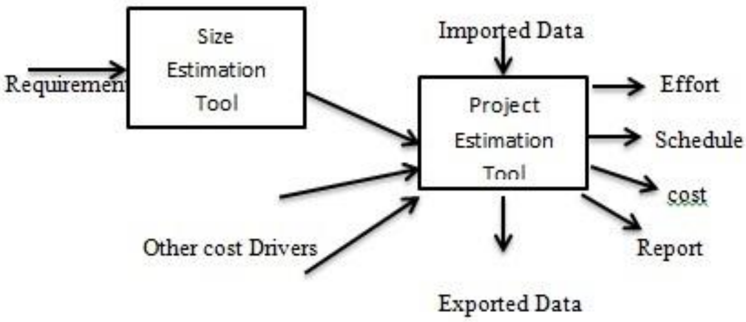
I. INTRODUCTION

An important objective of the software engineering group has been to develop useful models that are correctly estimating the software effort. Effort estimation is a process of forecast probable cost and development time to develop a software, process or product. Accurate effort estimation is important as over estimation may lead to loss of business and under estimation may lead to low quality of software which soon leads to software failure [1].

An imperative target of the product designing group has been to create helpful models that are precisely assessing the product exertion. Effort estimation is a procedure of anticipating plausible expense and advancement time to create a product, procedure or item. Exact exertion estimation is fundamental as over estimation may prompt loss of business and under estimation may prompt low nature of programming which eventually prompts programming disappointment. Exertion forecasts especially made at an early stage amid a task are useful for venture directors. There are heaps of existing techniques for exertion and cost estimation. Nonstop changing environment of programming improvement innovation attempt estimations all the more difficult.

The Software effort estimation techniques are for the most part sorted into algorithmic and non-algorithmic strategies. The algorithmic techniques are for the most part COCOMO, Function Points[2] and SLIM[3] Algorithmic techniques are likewise known as parametric techniques as they foresee programming improvement effort utilizing a settled numerical recipe that is parameterized from chronicled information. In any case, gauges at the preparatory phases of the task are hard to acquire since the essential source to gauge the exertion originates from the SRS record. Additionally, they experience issues in displaying the inalienable complex connections [4]. The constraints of algorithmic strategies prompt to look towards non-algorithmic techniques which are delicate registering based. These strategies have capacity to gain from past information and can display complex relationship between the reliant (exertion) and free variables.

In this paper we have analysed performance of different manufactured neural network models inserted in the COCOMO II to beat the imprecision and ambiguity of software attributes which results in creating better estimation results [5].



II. EFFORT ESTIMATION METHODS

The Survey reveals that different authors have computed different computational knowledge strategies on COCOMO dataset for effort estimation.

COCOMO II

The COCOMO II technique was created using COCOMO-81 model. The model was created by examining the changes in programming designing in the course of recent years reflecting these progressions.

Neural Networks for Software Effort Estimation:

Cocomo II provides two models

1. Early Design Model.
2. Post-Architecture Model.

Early Design Model: This model is used to make irregular estimates of a project's cost and duration before it is entire architecture is not resolved. It uses a small set of new Cost Drivers, and new estimating equations.

Post-Architecture Model: The Post-Architecture model coating the actual development and maintenance of a software product.

Artificial Neural Network is old in effort estimation due to its capacity to learn from previous data. It is also able to model complex connection between the dependent (effort) and independent variables (cost drivers). In addition, it has the ability to derive from the training data set thus enabling it to produce acceptable result for previously invisible data. The goal of the Neural Network is to model the relationship between the input and output from the historic data so that it can be used produce the good estimate for the future projects. Neural Network is compared to regression models and sophisticated Neural Network is better than regression method for estimating effort [6].

III. NEURAL NETWORKS IN PREDICTION

BACK PROPOGATION:

The back propagation learning algorithm is one of the best widely used methods in neural network. The network associated with back-propagation learning algorithm is termed as back propagation network. While training a network a set of input-output combination is provided the algorithm provides a procedure for changing the weight in BPN that helps to classify the input output combination correctly.

The aim of the neural network is to train the network to achieve a balance between the net's capacity to respond and its understanding to give reasonable responses to the input that is similar but not identical to the one that is used in training. Back propagation algorithm modify from the other algorithm by the method of weight calculation during learning. The defect of Back propagation algorithm is that if the hidden layer increases the network become too complex.

IV.DATASET DESCRIPTION

CocomoII The COCOMO Dataset not new in the analysis and acceptance of the model is achieving from the historic projects of NASA. One set of dataset response of 63 projects and other has 93 projects. The datasets is of COCOMO II format. In our measures 93 projects are used for training and 63 projects are used for testing .Number of effort adjustment factor is increases by 5, now it becomes 22 as shown in table 1.

Table 1: Cocomo dataset

Cost Drivers	***Descriptions***
DATA	Database Size
CPLX	Product Complexity
TIME	Execution Time Constraints
STORE	Main Storage Constraints
RUSE	Requirement Reusability
DOCU	Documentation match to life cycle needs
PVOL	Platform Volatility
SCED	Scheduling Factors
RELY	Required Reliability
TOOL	Use Of Software Tools
APEX	Application Experience
ACAP	Analyst Capability
PCAP	Programmability Capability
PLEX	Platform Experience
LITE	Language and Tool Experience
PCON	Personnel continuity
SITE	Multisite Development

Effort adjustment factors used in intermediate Cocomo other than intermediate **Cocomo**

Table 2: Cocomo dataset

Scale Factor	***Description***
Precedentedness (PREC)	Reflects the previous experience of the organization
Development Flexibility (FLEX)	Reflects the degree of flexibility in the development process.
Risk Resolution (RESL)	Reflects the extent of risk Analysis carried out
Team Cohesion (TEAM)	Reflects how well the development team knows each other and work together
Process maturity(PMAT)	Reflects the process maturity of the organization

IV. RELATED WORK

G. E. Wittig, et al.[1] used a dataset of 15 commercial systems, and used feed-forward back-propagation multilayer neural network for their experiment. ANN used in this paper are with numbers of hidden layers varying from 1-6 , but found the best performance for only one hidden layer with sigmoid function. It has been observed that for smaller system the error was 1% and for larger systems error was 14.2% of the actual effort. Jaswinder Kaur, et al.[2] implemented a back-propagation ANN of 2-2-1 architecture on NASA dataset consist of 18 projects. Input was KDLOC and development methodology and effort was the output. He got result MMRE as 11.78. Many researchers used their different ANN and different dataset, to predict the effort more correctly. F. Barcelos Tronto, et al.[4], also used COCOMO-81 dataset, with only one input, i.e TOTKDSI (thousands of delivered source instructions). All the input data were normalized to [0, 1] range. Here a feed-forward multilayer back-propagation ANN was used with the 1-9-4-1 architecture.

The performance in MMRE found was 420, where as that of COCOMO and FPA was 610 and 103 respective. The paper presented by TOSUN, et.al. [5], a novel method for assigning weights to features by taking their particular importance on cost in to analysis. Two weight assignment searching are implemented which are inspired by a widely used numerical technique called Principal Component Analysis (PCA). The paper by BURGESS and LEFLEY [7], calculate the potential of Genetic Programming (GP) in software effort estimation and comparison is made with the Linear LSR, ANNs etc.

The paper by Abbas Heiat, [8], measure the neural network approach and old regression analyses in terms of mean value percentage error. The balancing was done in terms of multilayer perceptron and radial basis function based neural network to that of regression tests which display that neural network gives the best results and improved performance in terms of effort estimation. A recent study by Jorgensen provides a detailed review of different studies on the software development effort [15].

Nasser Tadayon [16] has proposed a dynamic neural network that will initially use COCOMO II Model. COC OMO, however, has some limitations. It cannot forcefully deal with imprecise and uncertain information, and calibration of COCOMO is one of the most functional tasks that need to be done in order to get accurate estimations. So, there is always scope for developing effort estimation models with better guessing accuracy. In Ref[19]. The author has explained that one of the greatest challenges for the software industry is to select the best approach to compute the effort estimation cost of the software. Neural techniques have proved very effective in software effort estimation.

The performance of a neural network depends on its architecture and their parameter settings. There are many parameters dominate the architecture of the neural network including the number of layers, the number of nodes in each layer, the transfer function in each node, study algorithm parameters and the weights which determine the connectivity between nodes. Garbage selection of network patterns and learning rules may cause serious difficulties in network performance and training. The complication is to decide the number of layers and number of nodes in the layers and the research algorithm as well. However, the criterion is to select the minimum nodes which would not impair the network performance. The number of layers and nodes should be minimized to amplify the performance [21]. ANN(Artificial Neural Network) techniques to compute the presentation indices Mean Magnitude-Relative-Error (MMRE), Relative-Root-Square Error (RRSE), Correlation Coefficient (CC), Root-Mean Square-Error (RMSE), Mean-Square-Error (MSE).

By the above reference work what we came about is that there is no one good technique will be used for the Predicting Effort Estimation using neural networks.

V. EVALUATION CRITERIA

For evaluating the different software effort estimation models, the most widely accepted evaluation criteria are the mean magnitude of relative error (MMRE), Probability of a project having relative error less than 0.25,Root mean square of error, Mean and standard deviation of error.

$$\mathbf{MRE_i} = \frac{|actual\ \ effort - predicted\ \ effort\ |}{actual\ \ effort} \qquad (1)$$

The MRE value is calculated for each observation whose effort is predicted. The aggregation of MRE over multiple observations can be achieved through the mean MMRE.

$$\textbf{MMRE} = \frac{1}{N}\,{}_{i}\sum_{i}^{N} MRE \qquad (2)$$

$$\textbf{PRED}_{(25)} = \frac{\text{MRE} \leq 0.25}{\textbf{N}} \qquad (3)$$

Consider Y is the neural network output and T is the desired target. Then Root mean square error (RMSE) can be given by[1].

$$\textbf{RMSE}=\sqrt{(Y-T)2} \qquad (4)$$

VI. EXPERIMENT

DATA PREPARATION

We have used COCOMO dataset for this experiment. This dataset consists of 93 projects data. In this dataset there are 17 attributes.

ANN PREPARATION

In this experiment we have created different types of neural network and compare their performance. In that back-propagation neural networks and one recurrent neural network is used. MATLAB10 NN tool is used for this experiment.

Maximum of the work in the application of neural network to effort estimation made use of feed-forward multi-layer, Back-propagation algorithm. However many researchers refuse to use them because of their fault of being the black boxes that is, certain why an ANN makes a particular decision is a difficult task. But then also many different models of neural nets have been proposed for solving many elaborate real life problems [9].

The 7 steps for effort estimation using ANN can be outline as follows:

1. Data Collection: Collect data for already developed projects like method used, and other characteristics.
2. Division of data set: Divide the number of data into two factors – training set & validation set.
3. ANN Design: Construct the neural network with number of neurons in input layers like as the number of characteristics of the project.
4. Training: Grain the training set first to train the neural network.
5. Validation: Later training is over then validates the ANN with the validation set data.
6. Testing: Lastly test the created ANN by feeding test dataset.
7. Error calculation: Analysis the performance of the ANN. If satisfactory then stop, else again go to step (3),make some changes to the network parameters and proceed.

VII. RESULT

Comparison results of BPN for training is given below in Table3. A model which gives lower MMRE is better than the model which gives higher MMRE. A model which gives higher PRED (25) is better than the model which gives lower PRED (25). Similarly the model which gives lower RMSE is better than the model which gives higher RMSE. The model which mean and standard deviation nearest to Zero is better than the models which gives mean and standard deviation far away from zero.

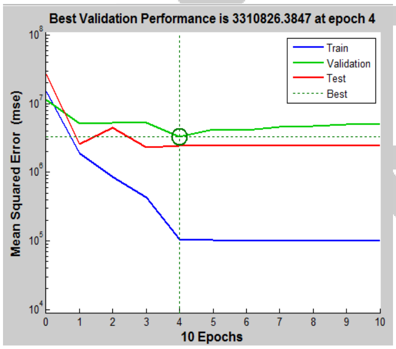


Fig1: Validation Performance

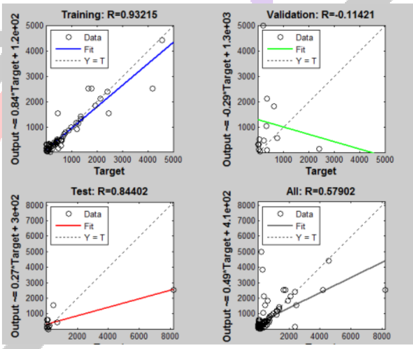


Fig2: Training Regression

Table3: Results of Training for BPN

Performance Measures	BPN		
	Min	Max	Mean
MMRE	0.0371	0.2098	0.0789
PRED(25)	72.712	89.002	79.806
RMSE	0.3449	2.0168	0.6122
Std.Dev	0.3267	0.9876	0.0006

VIII. CONCLUSION

Different methods of neural network have been used to calculate effort estimation. Each and every technique focuses on providing best software effort estimation. In our paper we propose neural network is a good approaching estimating development effort. It was suggested for complex and computationally large projects it's better to use neural network approach. But there is a need to examine accuracy of methods which mostly required in software effort estimation. We analysed that neuron based models have better estimation capability and hence can be used to calculate software effort estimation of all kinds of project.

REFERENCES

[1] G.E. Wittig and G.R Finnic, ―Using Artificial Neural Networks and Function Points to estimate 4GL software development effort‖, AJIS,1994, page:87-94.
[2] Jaswinder Kaur, Satwinder Singh, Dr. Karanjeet Singh Kahlon, Pourush Bassi, 2010. Neural Network- A Novel Technique for Software Effort Estimation.
[3]Sheta, A., Rine, D., & Ayesh, A. (2008, June). Development of software effort and schedule estimation models using soft computing techniques. In *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on* (pp. 1283-1289). IEEE.
[4] Ali Idri and Taghi M. Khoshgoftaar& Alain Abran,‖Can Neural Networks be easily Interpreted in Software Cost Estimation‖, IEEE Transaction, 2002, page:1162-1167.
[5] Venkatachalam, Software Cost Estimation Using Artificial Neural Networks, *International Joint Conference on Neural Networks*, Nogoya,IEEE, 1993.
[6] A.Tosun, B. Turhan and A.B. Bener, "Weighting Heuristics for Analogy- based Effort Estimation Models," Expert Systems with Applications, vol. 36, pp.10325-10333, 2009.
[7] C.J. Burgess and M.Lefley, "Can Genetics Programming improves Software Effort Estimation? A Comparative Evaluation," Information and Software Technology, vol.43, pp.863-873, 2001.
[8] Abbas Heiat, Comparison of Artificial Neural Network and Regression Models for Estimating Software Development Effort, *Information and Software Technology*, vol.44, pp.911-922, 2002.
[9] Prabhakar, Maitreyeea Dutta**,** 2013. Prediction of Software Effort Using Artificial Neural Network and Support Vector Machine.(2013).
[10] Clark, B., Chulani, S. and Boehm, B. (1998), "Calibrating the COCOMO II Post Architecture Model," International Conference on Software Engineering, Apr.1998.
[11] I.F. Barcelos Tronto, J.D. Simoes da Silva, N. Sant. Anna , ―Comparison of Artificial Neural Network and Regression Models in Software Effort Estimation‖, INPE ePrint, Vol.1, 2006.
[12] Putnam, L. H. (1978). A general empirical solution to the macro software sizing and estimating problem. *IEEE transactions on Software Engineering*, *4*(4), 345-361.
[13] Lippmann, Richard P. "An introduction to computing with neural nets." ASSP Magazine, IEEE 4.2 (1987): 4-22.
[14] S.Kanmani, J. Kathiravan, S. Senthil Kumar and M. Shanmugam, "Neural Networks Based Effort Estimation using Class Points for OO Systems",IEEE proceedings of the International Conference on Computing: Theory and Applications(ICCTA'2007).
[15] Jorgerson, M., "Experience with accuracy of software maintenance task effort prediction models," IEEE Transactions on Software Engineering, Volume 21 (8), 674–681, 1995.
[16] Nasser Tadayon, "Neural Network Approach for Software Cost Estimation", IEEE proceedings of the International Conference on Information Technology: Coding and Computing(ITCC '2005).
[17] S.N. Sivanandam, S.N. Deepa, Principles of Soft Computing, Wiley India (2007).
[18] B.W. Boehm, "Software Engineering Economics," *IEEE Trans Software Eng.*, vol. 10, no. 1, pp. 4-21, 1984.
[20] T. Mukhopadhyay, S.S. Vicinanza, and M.J. Prietula, "Examining the Feasibility of a Case-Based Reasoning Model for Software Effort Estimation," *MIS Quarterly*, vol. 16, pp. 155-171, June, 1992.
[19] Y. Miyazaki et al., "Method to Estimate Parameter Values in Software Prediction Models," *Information and Software Technology*, vol. 33, no. 3, pp. 239-243, 1991.
[20] Suresh Nageswaran, "Test Effort Estimation Using Use Case Points", Quality Week , San Francisco, California, USA, June 2001.
[22] Vahid Khatibi, Dayang N. A. Jawawi, "Software Cost Estimation Methods: A Review", Journal of Emerging Trends in Computing and Information Sciences, ISSN 2079-8407, Volume 2 No.1, pg. no 21-29,2010-11.
[23] Hareton Leung, Zhang Fan, "Software Cost Estimation", Article 2001.
[24] Bogdan Stepien, "Software Development Cost Estimation Meth- ods and Research Trends",Computer Science, Vol.5,2003.
[25] Jin Yongqin, Li Jun, Lin Jianming, Chen Qingzhang, "Software Project Cost Estimation Based On Groupware", World Congress on Software Engineering, IEEE, 2009.
[26] Chen Qingzhang, Fang Shuojin, Wang Wenfu, "Development of the Decision Support System for Software Project Cost Estimation", World Congress on Software Engineering,IEEE, 2009.
[27] Y. F. Li, M. Xie, T. N. Goh, "A Study of Genetic Algorithm for Project Selection for Analogy Based Software Cost Estimation, IEEE, 2007.