Software Effort Estimation for COCOMO-II Projects Using Artificial Neural Network

Kiran Kumar T.M[1*], Yashvanth Kumar K.P[2]

[1]*Assistant.Professor,* [2]*Project Student, Dept of MCA, Siddaganga Institute of Technology, Tumkur, India*

Abstract: **Software failures are mainly caused by defective projects management practices, including estimates of effort. Constant changing outlines of requirements and the technology software development make estimating efforts more complicated. Several methods are available to Estimate the effort of the soft computing-based method. The development effort needed for a project should be measured by software. It is important to estimate the construction effort required before any project is initially initiated. It is one of the greatest and most demanding tasks ever. The software cost estimate deals with a lot of uncertainty between all neural computing methods. In this paper we have used the historical COCOMO II data set projects using the artificial neural network technique to predict the effort estimation. We have used the mat lab tool for estimation. The experiment outputs suggest that the suggested model can provide better results and accurately forecast the software development effort.**

Keywords: **Artificial Neural Networks; Back Propagation; COCOMO II; Effort Estimation; MRE.**

I. INTRODUCTION

The main objective of any software industry is to develop the product on time within the budget which is relevant in practice. Effort evaluation is a mechanism by which development time and costs for software or product development can be estimated. Cost and time estimates for a variety of purposes, precisely is important. More than an estimate could hurt a company's financial loss.

Effort estimates made during the initial stages of project development can also be useful for project managers. However, at early stages, there is very little detail. Some algorithmic and non-algorithmic methods for estimation of effort already exist. Since the requirements are volatile and the complex environments the predictions of effort is still challenging task for any project managers in the software industry.

Correctness in estimating the software development effort required plays a critical role in the success of software project management a correct resource amount should be provided for a project. The calculated effort will be approximately equivalent to the true effort for a successful software evaluation method. A precise estimate allows managers to assign resources for all activities to plan and coordinate. Soft computing techniques are used here to predict the effort because of uncertainty in the prediction of effort is either individually or in combination as hybrid in a soft computation approach using many techniques such as the neural network, fuzzy logic, genetic algorithms etc. Now a day's estimation method using neural network is the interesting area for research compared to Theoretical estimation methods.

II. LITERATURE REVIEW

Estimating Software Effort is one of the most crucial responsibilities for all the people involved with Software Project Management. These days, a lot of competition is there in the software industry to make quality software in stipulated cost and time. Therefore, it is utterly important to accurately estimate effort in key phases of Software development. Regardless of the expanse of investigations over the past two decades, the software industry is still considerably challenging in case of effective resource estimation. Time to time, authors have discussed various methods for the same.

Copious literature is available on software effort estimation but only recent ones are cited here. Cuauhtémoc Lopez-Martin [1] presented a comparative analysis of fuzzy logic model (FLM) and linear regression model (LRM)for predicting the development effort of short programs based on two independent variables. The two inputs namely, new and changed code (N&C) and reused code are used as input parameters in estimating the development effort. The accuracy of the so developed model is compared with statistical regression model evaluation criteria were based on magnitude of error relative to the estimate(MER) as well as the mean magnitude of error relative (MMER). Twenty programs by seven programmers were used for the development of predictive models. This work has shown that FLM are slightly better in terms of prediction accuracy when compared with LRM when effort estimation is to be made for small programs. In most of the research, the data pertaining to the COCOMO 81 dataset has been used for developing the estimation models like fuzzy logic model, fuzzy regression and fuzzy neural network model have been developed by the researchers [2, 3, 4, 5]. These models differ slightly in terms of prediction accuracy and majorly in terms of number of input attributes.

Srichandan.[6], has opined that, artificial intelligence techniques such as neural network, fuzzy logic, genetic algorithm, case based reasoning etc. could provide a way for modeling. Tukutuku datasets and COCOMO datasets are used in this work. Radial basis function of neural network (RBFN) is developed for effort estimation. The model is evaluated using MRE and MMRE. Three estimation techniques have

been compared using function points. The models are artificial neural network, regression analysis and case-based reasoning. After performing the analysis various points were observed with respect to these models. Regression techniques appeared to give poor result as comparison to neural network and case-based reasoning. Case –based reasoning was the best method because of its expert judgment approach that provides the result in support of human judgment.

Gerhard witting [7] has mentioned the most effective requirement in terms of software is to find the best method for software effort estimation and Artificial Neural Network is considered as a good approach because of its capability to give good results in terms of large and complex software projects. BPN model developed in this work has shown encouraging results. A reported work by Heiat[8] has compared the prediction precision of two different kinds of neural networks they are multilayer perceptron and a radial basis function network with the accuracy of a statistical regression. The author has used three sets of data i.e., 1) the IBM DP Services Organization comprising 24 projects,2) the Kemerer dataset comprising 15 projects, and the 3) Hallmark dataset comprising 28 projects. Two experiments were conducted, for the first experiment he has pooled the projects from the Kemerer and the IBM datasets—which include third generation programming languages, whereas in the second experiment, projects from the Kemerer, the IBM, and the Hallmark datasets—which comprise both third generation and fourth generation programming languages. These three datasets were garnered from publications that happened before the 1988. The results from this study indicated that when a pooled third generation and fourth generation language dataset were used, the neural network produced enhanced performance over conventional statistical regression. The prediction accuracy is tested with MMRE. The training set for the models incorporated 32 projects for the first experiment and 60 for the second experiment, whereas seven projects were used for testing the models.

Oliveira [9] provided a relative study on support vector regression (SVR), a radial basis function neural network (RBFN) and linear regression. The results generated were based on set of 18 projects and it showed that SVR considerably outperforms RBFN and linear regression. The precision criterion was the MMRE. Vinayet al.[10] have proposed a wavelet neural network (WNN). The WNN is compared with a multilayer perceptron, a radial basis function network, multiple linear regressions, a dynamic evolving neuro-fuzzy inference system, and a support vector machine (SVM). The precision criterion was the Magnitude of Relative Error (MMRE). The datasets used from Canadian financial, and IBM data processing services (IBMDPS), consisting 24 and 37 software projects, respectively. Based on their results, WNN was found to be pragmatic and it also outperformed all the other techniques. De Barcelos et al [11] compared the accuracy of a feed forward multilayer perceptron neural network against statistical regression. The author has used the COCOMO dataset of 63 projects, for training and testing the models. The prediction accuracy evaluation criterion was the MMRE. This study was on hinged on the investigation of the behaviour of these two techniques when predicting variables as categorical variables are used. Results presented in this study indicated that these two techniques were competitive. C.Lopez-Martin et al[12] through their work have opined that the accuracy of a general regression neural network model is statistically equal or better than that obtained by a statistical regression model using data obtained from industrial environments. Each model was generated from a separate dataset obtained from the International Software Benchmarking Standards Group (ISBSG) software projects repository.

III. ANN ARCHITECTURE AND TRAINING

Artificial Neural Network (ANN) uses machine learning and patter recognition methodology for accurate estimates for software development effort. In this way, it provides accurate estimates of software. ANN increases effort estimation efficiency based on mean absolute error. It is discovered. ANN has the capacity to discover relationships between dependent and independent variables. It is also possible to learn from past records. ANN is the artificial neuron interconnection. ANNs include neurons (nodes) and synapses (weights) as two essential components of biological neural networks. This paper focuses mainly on how the effort can be predicted with the Artificial Neural Network

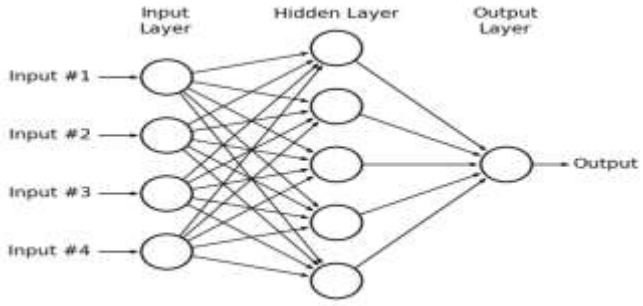

Fig1: Artificial Neural Networks

A. Data description

COCOMO II data projects collection for the estimation of effort with the ANN model was used in this paper. Effort in persons months is estimated using the mathematical formula in algorithmic cost estimation methods. Historical proof contributes to mathematical relationships and constants.

B. Our approach

The proposed enhanced neural network model is built using MATLAB software's Neural Network toolbox. Simulation and modelling are simplified with Matlab tools. The size of the input and output vector is decided, as mentioned earlier. The first tests will involve the arbitrary selection of numbers of processing elements. Neural network models with a hidden

layer and a different number of processing elements or neurons are created. One layer with non-linear triggering function, according to Whites Theorem, is sufficient to accurately map non-linear functional interaction.

Optimal network architecture was explored using the test and error approach to try and create a better model. In the present study, ANNs containing 2 to 20 nodes were taken into consideration to reduce the number of networks needing training and testing. Initially, the learning rate was reduced and slowly increased. During the training process, different permutations and combinations of these two factors have been used. When the average error i.e. tolerance level dropped below 0.99999, the goal error was set to stop during the training. Training is the method whereby an ANN 's weight is measured, using an iterative technique to eliminate an error or objective function predetermined like the MSE. Therefore, ANN training is basically a non-linear problem with least squares, which is solved by non-linear minimum square methods. The background propagation algorithm is used for training feed forward neural network architecture.

The following can be outlined in the seven steps for estimating using ANN:

1. *Data Collection*: Collect data for projects which have already been developed, such as method and other features.

2. *Division of data*: Divide data number into two factors – training and validating set. Divide data set

3. *ANN Design*: Create a neural network of neurons with the number of input layers, such as the number of project characteristics.

4. *Training*: first training for the neural network. Training:

5. *Validation*: The subsequent testing ends with validation set data validating the ANN.

6. Testing: Finally, the created ANN is tested through the feeding of test data.

7. *Mistake calculation*: ANN performance analysis. Should it be sufficient to stop, phase (3) again, make some adjustments to the network parameters and proceed

Table 1: Effort computation using ANN technique:

Project ID	Effort from the dataset	Computed effort with COCOMO	Computed effort with ANN
1	2040	1745	2032.12
3	243	233	240.12
11	218	184	194.34
18	11400	9344	11312.11
20	6400	4566	5813.2
26	387	312	371.34
27	134	90	124.56
50	176	154	173.12
52	145	112	132.13
54	20	8.5	12
55	18	10	17.87
58	958	723	952.12
60	57	20	48.13
61	77	45	72.78

IV. PERFORMANCE CRITERIA

The capability of the Algorithmic COCOMO II and projected estimate model are inspected and matched using following performance measures

A common criterion for the evaluation of cost estimation models namely magnitude of relative error (MRE) and the aggregation of MER over multiple observations through mean magnitude of relative error (MMER) have been used. MER is given by

$$MERi = \frac{|\text{actual effort}_i - \text{estimated effort}_i|}{\text{estimated effort}_i} \quad \text{---------- (1)}$$

And MMER is given by,

$$MMER = \left(\frac{1}{N}\right) \sum_{i=1}^{N} MER_i \quad \text{---------- (2)}$$

Here MER is ought to measure the error relative to the estimate. MMER is a measure of accuracy of an estimation technique. MMER is inversely proportional to accuracy. In several reported works, a MMER ≤ 0.25 has been considered to be acceptable.

Table 2: MRE values

Project id	MRE (%) with COCOMO	MRE (%) with ANN
1	14.46	0.38
3	4.11	1.18
11	15.59	10.85
18	18.035	77.09
20	28.65	9.16
26	19.37	4.04
27	32.83	7.04
50	12.50	12.50
52	22.75	8.87
54	57.5	40
55	44.44	0.72
58	24.53	24.53
60	64.91	15.56
61	41.55	5.48

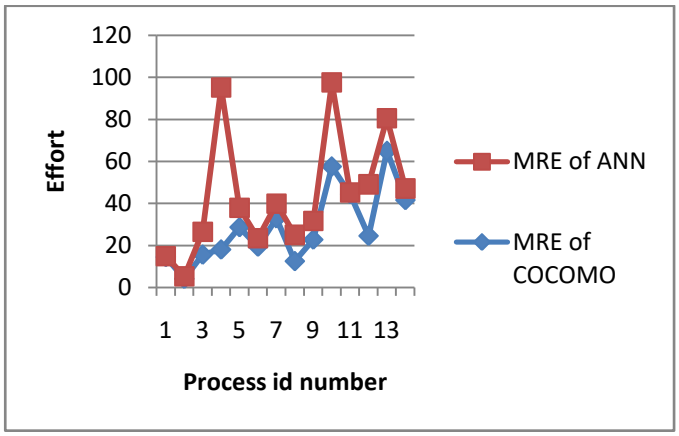

Fig2: Comparative chart showing MRE with COCOMO and MRE with ANN.

V. RESULTS AND DISCUSSIONS

Table 1 presents a relative chart of approximate and expected effort for 14 project values randomly selected using COCOMO and the proposed ANN methodology. Further table 2 presents both the COCOMO and the proposed ANN model with the magnitude of associated error (MRE). The number in the number shown. 2 note explicitly that the relative error has been greatly reduced, and the proposed model is therefore ideally suited to the calculation of effort. The primary results obtained can be simulated for anticipating the accurate software development effort.

VI. CONCLUSIONS AND FUTURE WORK

With the minimum number of layers and nodes that increase network performance, our proposed model maps current COCOMOII to ANN. In order to predict the development effort, the proposed ANN is the Multilayer Flow Network with 23 inputs and a hidden layer, which uses the back propagation algorithm.

The COCOMOII dataset was employed for the training, testing and validation of the network, and the neural network model provided significantly better estimates of effort than the estimate made by the popular COCOMOII algorithm model. Another big improvement with the use of neural networking model is the inclusion, in a common architecture, of expert knowledge and traditional models that can be used extensively for cost estimation in software

REFERENCES

[1]. C. Lopez-Martin, Applying a general regression neural network for predicting development effort of short-scale programs, Neural Comput. Appl. 20 (2011), pp.389–401.

[2]. Krishnakumar Pillai and VS Sukumaran Nair. A model for software development effort and cost estimation. Software Engineering, IEEE Transactions on, 23(8):pp.485-497, 1997.

[3]. Allan J Albrecht. Measuring application development productivity. In Proceedings of the Joint SHARE/GUIDE/IBM Application Development Symposium, volume 10, pp. 83-92, 1979.

[4]. IFPUG IFPUG. Function point counting practices manual, release 4.2. International Function Point Users Group, USA-IFPUG, Mequon, Wisconsin, USA, 2004.

[5]. Barry W Boehm et al. Software engineering economics, volume 197. Prentice-hall Englewood Cliffs (NJ), 1981.

[6]. S. Srichandan, A new approach of software effort estimation using radial basis function neural networks, Int. J. Adv. Comput.Theory Eng. 1 (2012),pp.2319–2526.

[7]. Gavin R. Finnie, Gerhard E. Wittig, Jean-Marc Desharnais, A comparison of software effort estimation techniques: Using function points with neural networks, case-based reasoning and regression models, J. Syst. Softw. 39 (3) (1997) pp.281–289, [Online], Available: http://dx.doi.org/10.1016/S0164-1212(97)00055-1.63

[8]. Heiat, A.: Comparison of artificial neural network and regression models for estimating software development effort. Journal of Information and Software Technology, 44(15), (2002), pp.911-922.

[9]. Oliveira ALI “Estimation of software project effort with support vector regression.”Neuro computing, Elsevier 69(13-15), (2005),pp.1749–1753.

[10]. Vinay KK, Ravi V, Carr M, Raj KN “Software development cost estimation using wavelet neural networks”. Journal of System Software, Elsevier 81, (2007), pp.1853–1867.

[11]. De Barcelos Tronto, I.F., Simoes da Silva, J.D., and Sant’Anna, N.: An investigation of artificial neural networks based prediction systems in software project management. Journal of Systems and Software, 81(3), (2008), pp. 356-367.

[12]. Lopez-Martin C., Isaza C., Chavoya A.: Software development effort prediction of industrial projects applying a general regression neural network. Journal of Empirical Software Engineering, 17(6), (2012), pp. 738-756.