Software Effort Estimation with Different Artificial Neural Network

Jagannath Singh
Department of Computer Sc. and Engineering
National Institute of Technology Rourkela
Rourkela, India
E-mail: jagannath.singh@gmail.com

Bibhudatta Sahoo
Department of Computer Sc. and Engineering
National Institute of Technology Rourkela
Rourkela, India
E-mail: bdsahu@nitrkl.ac.in

***Abstract*— Failures of software are mainly due to the faulty project management practices, which includes effort estimation. Continuous changing scenarios of software development technology makes effort estimation more challenging. Ability of ANN(Artificial Neural Network) to model a complex set of relationship between the dependent variable (effort) and the independent variables (cost drivers) makes it as a potential tool for estimation. This paper presents a performance analysis of different ANNs in effort estimation. We have simulated four types of ANN created by MATLAB10 NNTool using NASA dataset.**

Keywords- Effort Estimation, Artificial Neural Network, NNtool, MMRE

I. INTRODUCTION

Software estimates are the basis for project bidding, budgeting and planning. These are critical practices in the software industry, because poor budgeting and planning often has dramatic consequences. When budgets and plans are too pessimistic, business opportunities can be lost, while over-optimism may be followed by significant losses [1].

Software estimation can be modeled as the three stages, 1st stage involves *size estimation*, 2nd stage includes *effort estimation*, and *time estimation*, followed by the 3rd stage as *cost estimation*, and *staffing estimation*. Figure 1 shows the interaction between these modules in a typical software estimation process in Software Development Life Cycle.

According to the last research reported by the Brazilian Ministry of Science and Technology-MCT, in 2001, only 29% of the companies accomplished size estimation and 45.7% accomplished software effort estimate [2], so effort estimation has motivated considerable research in recent years.

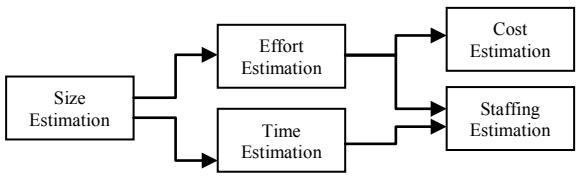

Figure. 1. Sequence of estimates in Software Development Life Cycle

Software effort estimation is the process of predicting the most realistic use of effort required to develop or maintain software. Effort estimates are used to calculate effort in person-months (PM) for the Software Development work elements of the Work Breakdown Structure (WBS).

Classifications of estimating methods based on that of Boehm are algorithmic, expert judgment, analogy, Parkinson, price-to–win, top-down and bottom-up. COCOMO and Function Point Analysis (FPA) comes under algorithmic method. Software effort estimation using artificial neural networks is grouped within the analogy based method.

Artificial Neural Network (ANN) is a massively parallel adaptive network of simple nonlinear computing elements called Neurons, which are intended to abstract and model some of the functionality of the human nervous system in an attempt to partially capture some of its computational strengths [3,12,13]. An artificial neural network comprises of eight basic components (*i)neurons, (ii)activation function, (iii)signal function, (iv)pattern of connectivity, (v)activity aggregation rule, (vi) activation rule, (vii) learning rule* and *(viii)environment* [10].

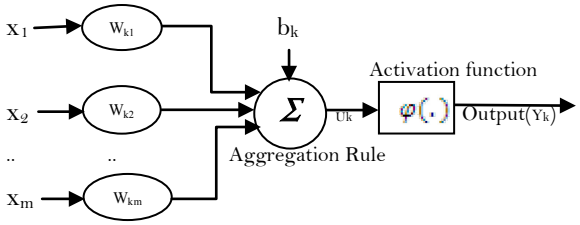

Figure. 2. Architecture of an artificial neuron

In mathematical notation, any *neuron-k* can be represented as follows:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \qquad \text{and} \qquad y_k = \varphi(u_k + b_k)$$

where $x_1, x_2, \ldots, x_m$ are the input signals , $w_{k1}, w_{k2}, \ldots, w_{km}$ are the synaptic weights of the corresponding neuron, u_k is the linear combiner output, b_k is the bias, $\varphi()$ is the activation function and y_k is the output signal of the neuron.

After an ANN is created it must go through the process of learning or training. The process of modifying the weights in the connections between network layers with the

objective of achieving the expected output is called training a network. There are two approaches for training– supervised and unsupervised [12,13]. In supervised training, both the inputs and the outputs are provided. The network then processes the inputs, compares its resulting outputs against the desired outputs and error is calculated. In unsupervised training, the network is provided with inputs but not with desired outputs. The system itself must then decide what features it will use to group the input data [3].

Depending upon the architecture the ANN is of two types. A *feed-forward* ANN, is the architecture in which the network has no loops. But *feed-back* (recurrent) ANN is an architecture in which loops occurs in the network[12,13]. An ANN can be a single-layer perceptron or a multi-layer perceptron. In single layer perceptron consists of a single layer of output nodes, the inputs neurons are connected directly to the outputs neurons via a series of weights. But in multi layer perceptron an additional layer of neurons present between input and output layers. That layer is called *hidden* layer. Any number of hidden layers can be added in an ANN depending upon the problem domain and accuracy expected. In this paper we have used multiple layer feed forward ANN for simulation.

II. ANN IN EFFORT ESTIMATION

After the extensive research over last 25 years, still the software community faces challenges when it comes to effective resource prediction. Some techniques, including FPA, COCOMO model and original regression model, are not effective, because they are not suitable for all types of software. Therefore other techniques like machine learning, exploratory data analysis comes into existence [2].

Artificial Neural Network is used in effort estimation due to its ability to learn from previous data. It is also able to model complex relationships between the dependent (effort) and independent variables (cost drivers). In addition, it has the ability to generalize from the training data set thus enabling it to produce acceptable result for previously unseen data.

Most of the work in the application of neural network to effort estimation made use of feed-forward multi-layer Perceptron, Back-propagation algorithm and sigmoid function. However many researchers refuse to use them because of their shortcoming of being the "black boxes" that is, determining 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 complex real life problems [4]. The 7 steps for effort estimation using ANN can be summarized as follows:

Steps in effort estimation

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

Once the ANN is ready, simulation with the ANN can be conducted with the parameter of any new project, as show in fig.3 and it will output the estimated effort for that project.

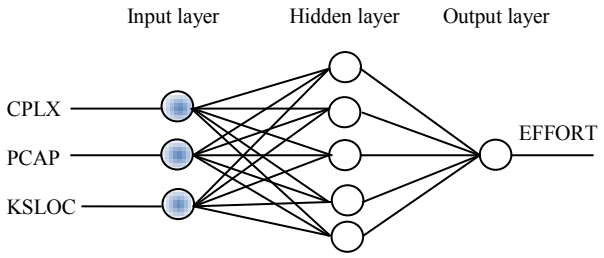

Figure. 3. Diagram of feed-forward multilayer ANN used in this paper

III. RELATED WORKS

Many researchers used their different ANN and different dataset, to predict the effort more correctly. G. E. Wittig, et al.[5] 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. In a paper by Ali Idri, et al. [4] uses COCOMO-81 dataset and three layered back-propagation ANN, applying 13 *cost drivers* as inputs and *development effort* taken as output. The ANN used are with 13 neurons in hidden layer and experimented for 300,000 iterations to find the average MRE = 1.50%.

F. Barcelos Tronto, et al.[2], 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 respectively.

Jaswinder Kaur, et al.[6] 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.

Roheet Bhatnagar, et al.[7] used MATLAB NN toolbox for effort prediction. He had used a dataset proposed by Lopez-Martin, which consists of 41 projects data. He has designed a 3-3-1 neural network, applied the Dhama Coupling (DC), McCabe Complexity (MC) and Lines of Code (LOC) as inputs. Development time was the only one output. The results of the experiment indicate that the percentage of error during training, validation and testing was between +14.05 to -25.60, +12.76 to -18.89 and +13.66 to -15.75 respectively.

K.K. Aggarwal , et al.[8] had investigated for finding the best training algorithm. Here ISBSG repository data was used on a 4-15-1 feed-forward ANN. Four inputs were taken-FP, FP standard, language and maximum team size. SLOC was the only output. The various training algorithm for ANN has been used and concluded that 'trainbr' is the best algorithm. 'traingd' was found to be the next best algorithm.

TABLE I. DATASET AND ANN USED

Author	Learning Algorithm	Dataset	No. of Projects	No. of Inputs	ANN Configuration
I.F. Barcelos Tronto [2]	Back-propagation	COCOMO	63	1	[1-9-4-1]
G. E. Wittig[5]	Back-propagation	Commercial Systems	15	-	[23-4-1]
Ali Idri [4]	Back-propagation	COCOMO	63	13	[13-13-1]
Jaswinder Kaur[6]	Back-propagation	NASA	18	2	[2-2-1]
Mrinal Kanti Ghose[9]	Back-propagation	Lopez-Martin	41	3	[3-3-1]
A.R. Venkatachalam[14]	Back-propagation	COCOMO	63	22	[22-45-2]

Reviewing the extensive research in effort estimation using ANN, as show in table-I, it is found that many of the researchers are using Feed-forward Back-propagation ANN for their simulation. Multi-layer ANN can be used with any number of hidden layers, but ANN with only one or two hidden layers also gives good results. All the inputs applied to the ANN were normalized to [0,1] range. *In MATLAB10 NNtool , if we are constructing any ANN ,than it's inputs need not to be normalized.* Depending upon the type of software project dataset, different ANN training algorithm gives different results. Among all training algorithms 'trainlm' gives satisfactory results in all types of datasets.

Levenberg-Marquardt (trainlm) function is the fastest training function for small Neural networks. It require more memory and computation time, so it is less efficient for large network (with thousands of weights) [11]. In this study we have used "*trainlm*" algorithm for 4 different neural networks.

IV. PERFORMANCE CRITERIA

A. Mean Magnitude Relative Error(MMRE)

MMRE is frequently used to evaluate the performance of any estimation technique. It measures the percentage of the absolute values of the relative errors, averaged over the N items in the "Test" set and can be written as[6]:

$$MMRE = \frac{1}{N}\sum_{i=1}^{N} |(y_i - \hat{y}_i|/y_i$$

where y_i represents the i^{th} value of the actual effort and $\hat{y}_i$ is the estimated effort.

B. Root Mean Square Error(RMSE)

RMSSE is another frequently used performance criteria which measures the difference between values predicted by a model or estimator and the values actually observed from the thing being modeled or estimated. It is just the square root of the mean square error, as shown in equation given below [6]:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

C. Balance Relative Error(BRE)

BRE is another evaluation criteria for accuracy[9]:

$$BRE(\%) = 100 * |(y_i - \hat{y}_i|/\min\ (y_i, \hat{y}_i)$$

D. Pred(l)

Pred(l) is a proportion of a given level *l* in the accuracy[9]:

$$Pred(l) = k/N$$

Where N is the total number of observations and *k* is the number of observations with MRE less than or equal to l.

V. EXPERIMENT

Data Preparation

We have used NASA public dataset for this experiment. This dataset consists of 60 projects data. In this dataset 17 attributes were there, but we have considered only four attributes, as show in table-II. Here CPLX means product

complexity, PCAP means programmer capability and KLOC means thousand of source lines of code. The values for CPLX and PCAP were in fuzzy format, so for the experiment we have to convert it into numeric format as: 1- very high, 2- high, 3- normal, 4- extra high, 5- low.

ANN Preparation

In this experiment we have created four different types of neural network and compare their performance. Cascade, Elman and Feed-forward are three back-propagation neural networks and one recurrent neural network is used. MATLAB10 *NN tool* is used for this experiment. For all the neural networks 3-5-1 architecture is used, i.e. 3 neurons in input layer, 5 neurons in hidden layer and 1 neuron in output layer. Training algorithm used is 'trainlm'. For training the dataset is divided into three divisions-for training 48(80%), for validation 6(10%) and for testing 6(10%). Stopping criteria was set by number of epochs as 1000 and goal as 0.00.

TABLE II. DATASET OF NASA PROJECTS

Project No.	CPLX	PCAP	KSLOC	EFFORT
1	1	1	70	278
2	2	1	227	1181
3	2	2	177.9	1248
4	2	3	115.8	480
5	2	3	29.5	120
6	2	3	19.7	60
7	2	3	66.6	300
8	2	3	5.5	18
9	2	3	10.4	50
10	2	3	14	60
11	2	3	16	114
12	2	3	6.5	42
13	2	3	13	60
14	2	3	8	42
15	2	2	90	450
16	2	3	15	90
17	2	2	38	210
18	3	2	10	48
19	2	3	161.1	815
20	2	3	48.5	239
21	2	3	32.6	170
22	2	3	12.8	62
23	2	3	15.4	70
24	2	3	16.3	82
25	2	3	35.5	192
26	2	3	25.9	117.6
27	2	3	24.6	117.6
28	2	3	7.7	31.2
29	2	3	9.7	25.2
30	2	3	2.2	8.4
31	2	3	3.5	10.8
32	2	3	8.2	36
33	2	3	66.6	352.8
34	2	1	150	324
35	2	3	100	360
36	2	2	100	215
37	2	1	100	360
38	2	2	15	48
39	2	3	32.5	60
40	2	2	31.5	60
41	2	2	6	24
42	2	3	11.3	36
43	2	1	20	72
44	2	2	20	48
45	2	2	7.5	72
46	2	2	302	2400
47	2	3	370	3240
48	2	3	219	2120
49	2	3	50	370
50	2	2	101	750
51	3	1	190	420
52	2	3	47.5	252
53	4	3	21	107
54	3	1	423	2300
55	3	2	79	400
56	5	3	284.7	973
57	5	3	282.1	1368
58	2	2	78	571.4
59	2	2	11.4	98.8
60	2	2	19.3	155

A. Cascade Neural Network

Cascade NN is a feed-forward neural network where the first layer will get signal from input. Each subsequent layer will receive signal from the input and all previous layers.

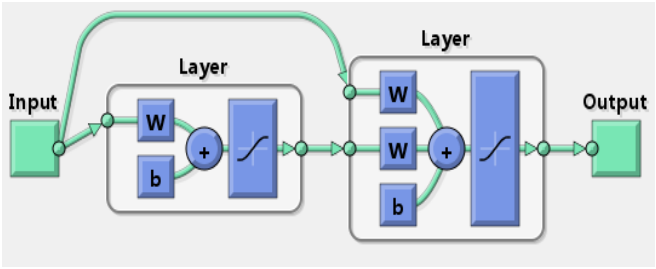

Figure 4. The Cascade ANN using MATLAB

B. Feed-Forward Neural Network

In a feed-forward neural network the first layer will get signal from input. Each subsequent layer will receive signal from its previous layer only.

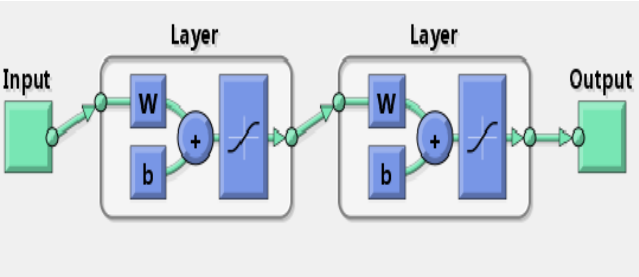

Figure 5. The Feed-Forward ANN using MATLAB

C. *Recurrent & Elman Neural Network*

Recurrent neural networks are the neural networks where the connections between the neurons form a directed cycle. Unlike Feed-forward neural network, it can use its internal memory for processing of the input.

Elman NN is a special type of recurrent neural network where an additional set of "context units" is connected with the input layer. These are also connected with the hidden layer with connection weight one.

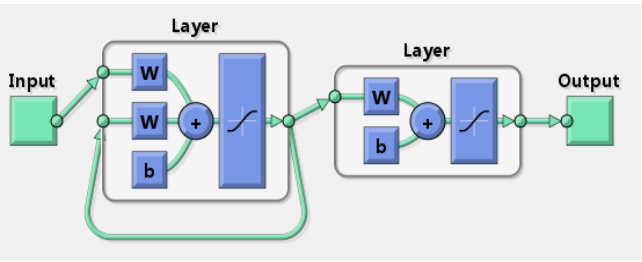

Figure 6. The Recurrent / Elman ANN designed with MATLAB

VI. SIMULATION AND RESULTS

All the networks were trained for 10 iterations and the most accurate results were considered. Table-III summarizes the development effort estimates as obtained with different neural networks. Fig.-7 and Table-IV presents a performance comparison of different neural networks.

TABLE III. ESTIMATED EFFORT BY DIFFERENT ANN

Project No.	Actual Effort	Cascade Forward ANN	Elman ANN	Feed Forward ANN	Recurrent ANN
51	420	403.6	779.9	465.1	436.8
52	252	278.1	336.6	194.6	190.7
53	107	142.3	48.8	118.0	95.8
54	2300	2303.8	2338.4	2313.4	2303.8
55	400	390.6	229.9	363.7	279.7
56	973	1019.9	1100.0	1159.0	1198.9
57	1368	992.5	1077.4	1137.3	1173.3
58	571.4	405.5	342.8	242.6	393.8
59	98.8	109.2	25.5	54.0	84.2
60	155	127.9	53.7	78.0	111.2

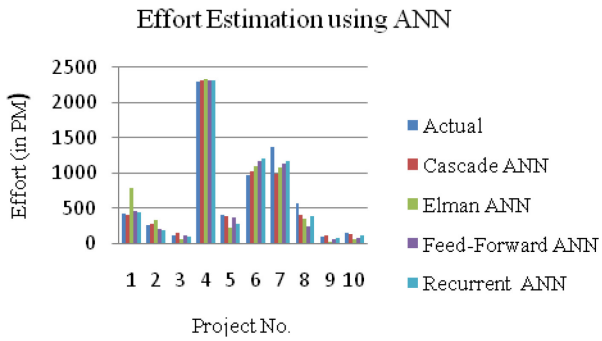

Figure 7. Effort Estimation Bar-chart

TABLE IV. COMPARISON OF RESULTS

Performance Criteria	ANN used			
	Cascade Forward ANN	*Elman ANN*	*Feed Forward ANN*	*Recurrent ANN*
MMRE %	4.98	43.16	5.35	6.91
RMSE	131.85	184.10	145.19	118.85
Mean BRE %	0.16	0.89	0.41	0.23
Pred(0.20) %	70	20	50	60

Feed-forward NN gives the next best results and Elman NN gives the worst result No estimation method is full-proof or hundred percent accurate. We have tried to explore one such method using artificial neural network.

VII. CONCLUSION

Estimation is one of the crucial tasks in software project management. This simulation with NASA dataset has been carried out using Matlab 10 NN tool box. All for ANN are trained using "*trainlm*" algorithm. The results from our simulation shows that Cascade feed- forward neural network give the best performance, among the four ANN. We have experimented with four attributes of the NASA public dataset and further investigation can be done with other attributes.

REFERENCES

[1] Stein Grimstad, Magne Jorgensen, Kjetil Molokken-Ostvold ,"Software effort estimation terminology: The tower of Babe"l, Elsevier, 2005.

[2] 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.

[3] Simon Haykin, "Neural Networks: A Comprehensive Foundation", Second Edition, Prentice Hall, 1998.

[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] 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.

[6] JaswinderKaur, Satwinder Singh, Dr. Karanjeet Singh Kahlon, PourushBassi,"Neural Network-A Novel Technique for Software Effort Estimation", International Journal of Computer Theory and Engineering, Vol. 2, No. 1 February, 2010, page:17-19.

[7] Roheet Bhatnagar, Vandana Bhattacharjee and Mrinal Kanti Ghose, "Software Development Effort Estimation –Neural Network Vs. Regression Modeling Approach", International Journal of Engineering Science and Technology,Vol. 2(7), 2010,page: 2950-2956.

[8] K.K. Aggarwal, Yogesh Singh, Pravin Chandra and Manimala Puri, "Evaluation of various training algorithms in a neural network model for software engineering applications" , ACM SIGSOFT Software Engineering , July 2005, Volume 30 Number 4 , page: 1-4.

[9] Mrinal Kanti Ghose, Roheet Bhatnagar and Vandana Bhattacharjee , "Comparing Some Neural Network Models for Software Development Effort Prediction" , IEEE, 2011.

[10] Satish Kumar, "Neural Networks: A Classroom Approach", Tata McGraw-Hill, 2004.

[11] Howard Demuth and Mark Beale, "Neural Network Toolbox-For use with MATLAB",User's Guide,Version-4,Page-5.28.

[12] N.K.Bose and P.Liang, "Neural Network Fundamentals with Graphs, Algorithms and Applications", Tata McGraw Hill Edition,1998.

[13] B. Yegnanarayana, "Artificial Neural Networks", Prentice Hall of India, 2003.