Software Cost Estimation using Single Layer Artificial Neural Network

Shaina Arora[1], Nidhi Mishra[2]

[1]Assistant Professor (CSE) AIET, Jaipur, India
{shaina.arora22@gmail.com}
[2]Associate Professor(CSE),Poornima University,Jaipur,India
{nidhi.mishra@poorinma.edu.in}

Abstract— *The most challenging task of software project management is the cost estimation. Cost estimation is to accurately assess required assets and schedules for software improvement ventures and it includes a number of things under its wide umbrella, for example, estimation of the size of the software product to be produced, estimation of the effort required, and last but not the least estimating the cost of the project. The overall project life cycle is impacted by the accurate prediction of the software development cost. The COCOMO model makes employments of single layer feed forward neural system while being actualized and prepared to utilize the perceptron learning algorithm. To test and prepare the system the COCOMO dataset is actualized. This paper has the goal of creating the quantitative measure in not only the current model but also in our proposed model.*

Keywords— Software Cost Estimation, COCOMO, Artificial Neural Network, Feed Forward Neural Network, Magnitude Relative Error.

I. INTRODUCTION

Estimation of required assets and calendar should be possible through precise cost estimation. Talking about the parameters, the exactness of the product advancement and the precision of the administration choices both are interrelated. The exactness of the previous is dependent on the precision of the last mentioned in terms of relying that is the former will rely on the latter. There are number of parameters for example improvement time, effort estimation and group, and for the calculation of each one of the models is required. For estimation of software cost effort estimation technique used most popular COCOMO model. What is unique about COCOMO model is that it makes use of mathematical formula to analyze project cost effort estimation. The paper based on COCOMO model is making use of single layer neural network technique using perception algorithm.

II. COCOMO

The Constructive Cost Model or better known as the COCOMO model, first presented by Dr. Barry Boehm in 1981 surpassed all the software development practices that took place in those days. Software development techniques have been undergoing many changes and evolving since those days. The COCOMO model can be sub-divided in following models based on the type of application [12].

A. Basic COCOMO
Project Qualities details are not needed to implements parameterized equation of basic COCOMO model.

$$\text{Person Month}=a\,(KLOC)\,b \qquad (1)$$

$$\text{Development Time} =2.5*PM\,c \qquad (2)$$

Three modes of progress of projects are there, on which all the three parameters depend, namely a, b and c.

B. Intermediate COCOMO
According to basic COCOMO model there is no provision to do software development. To add the accuracy in basic COCOMO model there is 15 cost drivers provided by Boehm. Cost driver can be classified into

Table.1: Cost Drivers

Cost Drivers			
Product Attributes	Computer Attributes	Personnel Attributes	Project Attributes
RELY DATA CPLX	TIME STOR VIRT TURN	ACAP AEXP PCAP VEXP LEXP	MODP TOOLS SCED

1. Product attributes
   - Required software reliability or better known as RELY
   - Database size or better known as DATA
   - Product complexity or better known as CPLX
2. Computer attributes
   - Execution time constraint   or better known as TIME
   - Main storage constraint or better known as STOR
   - Virtual machine volatility   or better known as VIRT
   - Computer turnaround time or better known as TURN

3. Personnel attributes
- ➢ Analyst capability or better known as ACAP
- ➢ Application experience or better known as AEXP
- ➢ Programmer capability or better known as PCAP
- ➢ Virtual machine experience or better known as VEXP
- ➢ Programming language experience or better known as LEXP

4. Project attributes
- ➢ Modern programming practices or better known as MODP
- ➢ Use of software tools or better known as TOOLS
- ➢ Required development schedule or better known as SCED [12].

III. ARTIFICIAL NEURAL NETWORK

An Artificial Neural Network (ANN) is nonlinear information (signal) processing devices, which are built from interconnected elementary processing devices called neurons. An Artificial Neural Network (ANN) is an information-processing model that is stimulated by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unification to resolve specific tribulations [15].
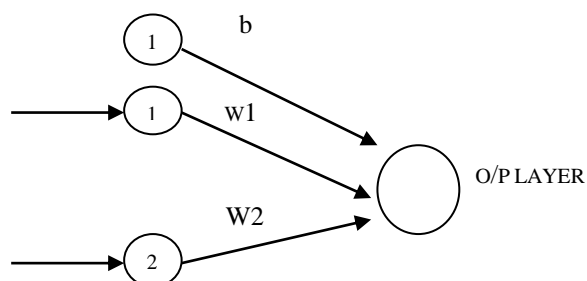

Fig.1: A Simple Artificial Neural Net

Figure 1 shows a simple artificial neural network with two inputs neurons (1, 2) and one output neuron (O). The interconnected weights are given by w1 and w2. There always exists a common bias of '1'. The input neurons are connected to the output neurons through weighted interconnections. This is a single layer network because it has only one layer of interconnections between the input and the output neurons. This network perceives the input signal received and performs the classification [15].

IV. RELATED WORK

Researchers in effort estimation models have developed multiple software's. Artificial neural network is capable for generating good information and modeling complex non-linear relationships. For the calculation of software effort estimation researchers all across the world have used the artificial neural network approach. Moreover, Boehm's COCOMO dataset is also used. N.Tadayon[9] reports the use of neural network with a back propagation. Anupma Kaushik[12] also research on multilayer neural network using perceptron learning algorithm. COCOMO [8] is the most effective and widely used software for effort estimation model, which arranges beneficial expert knowledge. For getting appropriate calculations COCOMO model is one of the most imperative tools, that produces capacity for developing effort estimation models with better analytical accuracy. In this paper, single layer feed forward neural network using perceptron learning algorithm and COCOMO data set have been used. Using this approach, an effort estimation model for software cost evaluation has been proposed.

V. PROPOSED NEURAL NETWORK MODEL

In proposed system Single Layer ANN used to calculate the software cost estimation. For this 16 input parameters are taken which includes 15 Effort Multiplier (EM) and one bias weight (b). First we calculate the net input output (sum of the product of each input neuron to their corresponding weight) after this applies identity activation function get estimated output.

A. Estimated Effort

The System is implemented with the help of single layer artificial neural network and trained using the perceptron leaning algorithm. The COCOMO dataset is used to train to test the network.

$$Sum = sum + EM_i * W_i$$
$$O_{est} = b + Sum;$$
$$MRE = ((oact - oest) / oact) * 100;$$
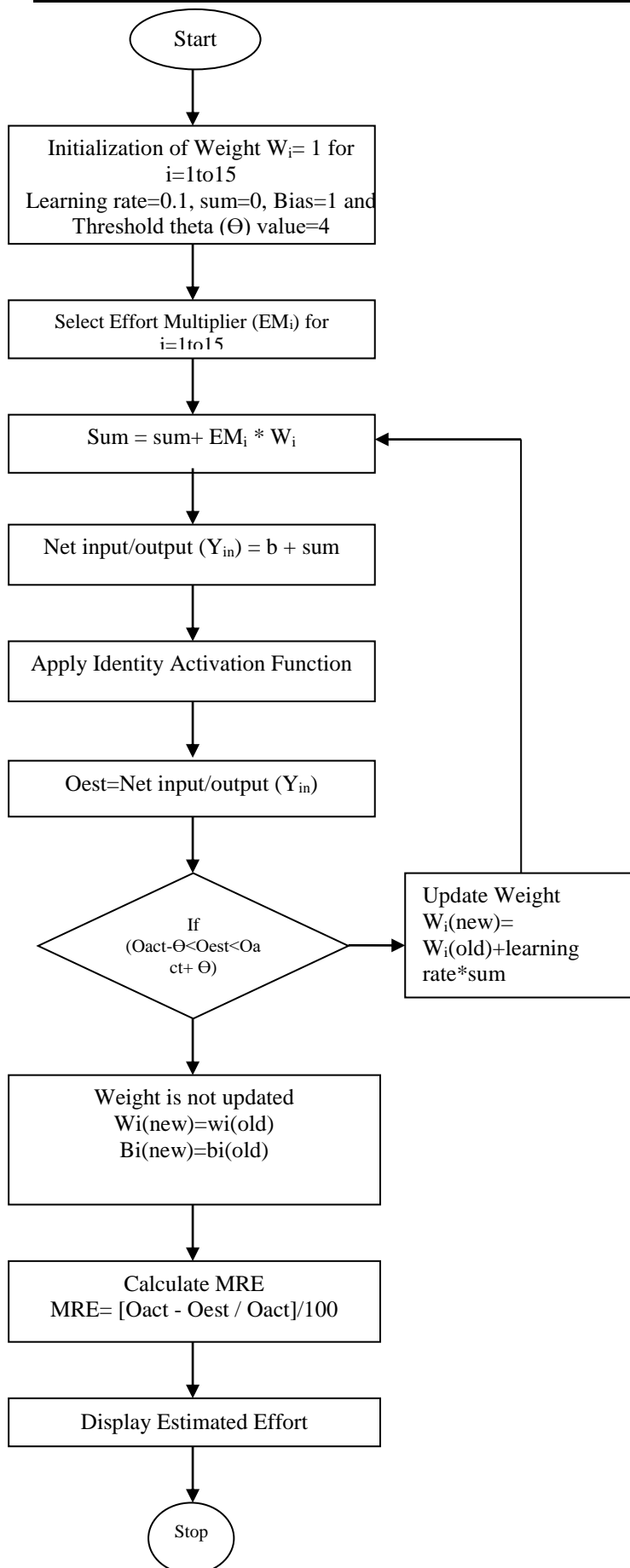
Here

O_{est} is the estimated effort,

W_i effort multiplier weight,

MRE is magnitude relative error,

O_{act} actual effort

It describes about experimenting networks and calculating new set of weight.

B. Flow Chart of System Implementation

```
                    ┌─────────┐
                    │  Start  │
                    └─────────┘
                         │
   ┌─────────────────────────────────────────┐
   │ Initialization of Weight Wi= 1 for       │
   │              i=1to15                      │
   │ Learning rate=0.1, sum=0, Bias=1 and      │
   │    Threshold theta (Θ) value=4            │
   └─────────────────────────────────────────┘
                         │
   ┌─────────────────────────────────────────┐
   │ Select Effort Multiplier (EMi) for        │
   │              i=1to15                      │
   └─────────────────────────────────────────┘
                         │
   ┌─────────────────────────────────────────┐
   │         Sum = sum+ EMi * Wi               │◄──────┐
   └─────────────────────────────────────────┘       │
                         │                            │
   ┌─────────────────────────────────────────┐       │
   │      Net input/output (Yin) = b + sum     │       │
   └─────────────────────────────────────────┘       │
                         │                            │
   ┌─────────────────────────────────────────┐       │
   │      Apply Identity Activation Function   │       │
   └─────────────────────────────────────────┘       │
                         │                            │
   ┌─────────────────────────────────────────┐       │
   │        Oest=Net input/output (Yin)        │       │
   └─────────────────────────────────────────┘       │
                         │                            │
              ◆──────────────────◆      ┌──────────────────────┐
              │        If         │      │  Update Weight       │
              │ (Oact-Θ<Oest<Oact+Θ) ├──►│  Wi(new)=            │
              ◆──────────────────◆      │  Wi(old)+learning    │
                         │               │  rate*sum            │
                         │               └──────────────────────┘
   ┌─────────────────────────────────────────┐
   │          Weight is not updated            │
   │            Wi(new)=wi(old)                │
   │            Bi(new)=bi(old)                │
   └─────────────────────────────────────────┘
                         │
   ┌─────────────────────────────────────────┐
   │            Calculate MRE                  │
   │     MRE= [Oact - Oest / Oact]/100         │
   └─────────────────────────────────────────┘
                         │
   ┌─────────────────────────────────────────┐
   │        Display Estimated Effort           │
   └─────────────────────────────────────────┘
                         │
                    ┌─────────┐
                    │  Stop   │
                    └─────────┘
```

C. Steps Followed for Proposed Model

Step1. Initialize the bias(b=1), weights(W_i=1 for i=1to15),set learning rate (lrnrate=0.1) and Threshold theta (Θ) value=4.

Step2. Execute steps 3-9 until stopping condition is false.

Step3. Execute step 4-8 for each training pair.

Step4. Total Number of inputs 16 in which Total Effort Multiplier 15 and One Bias.

Step5. Calculate the response of each unit

a) Calculate Sum of total input using Sum = sum+ EM_i * W_i

b) Calculate Net input output using Net input/output (Y_{in}) = b + sum

Step6. Apply identity activation function and calculate estimated output i.e effort using Oest=Net input/output(Y_{in})

Step7. if (O_{act}-Θ<Oest<O_{act}+ Θ).

Step 8 Weights are not updated. Go to step 10

Step9. Else Weights and Bias are updated

W_i(new)= W_i(old)+learning rate*sum

b(new)=b(old)+learning rate*Oact

Step9. Repeat step 5 to 7.

Step10. Calculate Magnitude Relative Error (MRE)

MRE = ((O_{act} - Oest) / O_{act})*100

Step 11: Stop.

VI. EVALUATION CRITERIA AND RESULTS

In this area, we depict the procedure utilized for figuring endeavors and the outcomes get when actualizing proposed neural system model to the COCOMO information set [14]. COCOMO information set is open source cost evaluating apparatus which comprises of 63 undertakings. The tool does the logical appraisal among the exactness of the assessed exertion with the genuine exertion. For examining programming exertion estimation we have calculated error using Magnitude of Relative Error (MRE) which is **Characterized as :-**

MRE= (actual effort-estimated effort/actual effort)*100 (3)
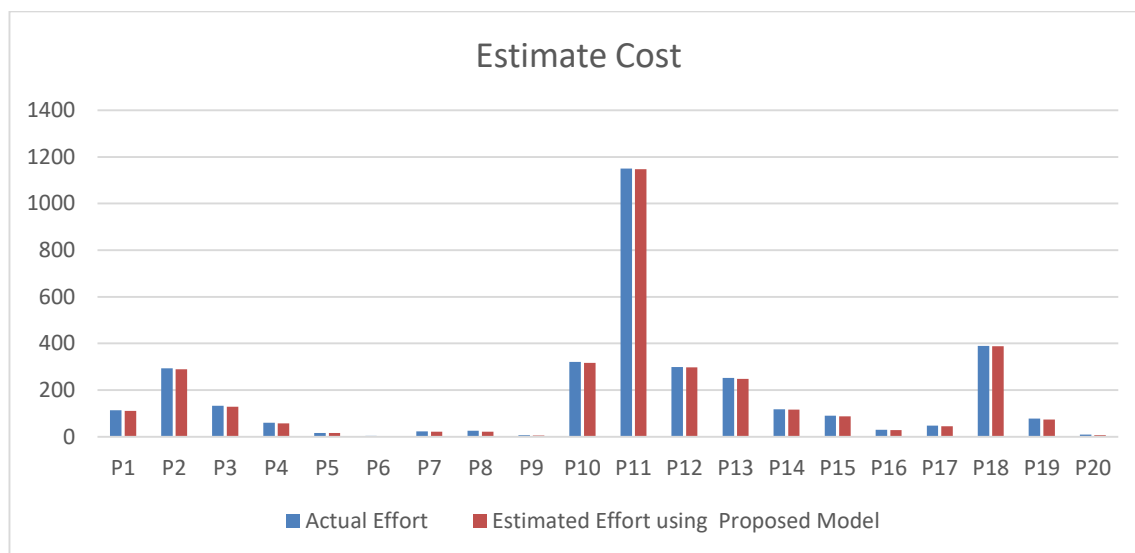
20 experimental values have been shown in table 2 which were tested. Actual effort of the model has been compared with these values. The comparison reflects us about the efficiency of our network. Table 3 contains the estimated effort, actual effort and Mean Magnitude of Relative Error (MRE) values for 20 experimented projects.

Table.2: Assessment of Calculated Effort

Project No.	Actual Effort	Estimated Effort using Proposed Model
P1	113	110.5
P2	293	289.7
P3	132	128.3
P4	60	56.6
P5	16	15.6
P6	04	0.8
P7	22	21.06
P8	25	21.1
P9	6.1	4.8
P10	320	316.5
P11	1150	1147.2
P12	299	297.58
P13	252	248.4
P14	118	116.1
P15	90	87.2
P16	30	28.8
P17	48	44.8
P18	390	387.8
P19	77	73.7
P20	9.4	6.4

Table.3: Assessment of MRE (Magnitude Relative Error)

Project No.	Actual Effort	Our Proposed Model	MRE using our Proposed Model (%)
P1	113	110.5	2.7
P2	293	289.7	1.1
P3	132	128.3	2.8
P4	60	56.6	5.8
P5	16	15.6	2.7
P6	04	0.8	79
P7	22	21.06	3.6
P8	25	21.1	15.6
P9	6.1	4.8	40.9
P10	320	316.5	1.1
P11	1150	1147.2	0.2
P12	299	297.58	1.2
P13	252	248.4	1.4
P14	118	116.1	3.3
P15	90	87.2	2.6
P16	30	28.8	3.9
P17	48	44.8	6.7
P18	390	387.8	0.8
P19	77	73.7	4.3
P20	9.4	6.4	31.9

Fig.2: Graphical Representation of Calculated Effort

Table 2, Table 3 and Figure 2 shows that the described neural network model gives the most proficient effort estimation results as compared to other models.

VII. CONCULSION

Having a dependable and precise estimate of software development has never been an easy task and this is where has always lied the problem for many scholarly and industrial conglomerates since ages. Talking about anticipating the future programming shows how a cost estimation model is built based on single layer artificial neural network. The neural network that is used to estimate the software improvement effort is single layer feed forward network with identity activation function. Accurate value is attained through neural network. In future, for software cost estimation we will put our focus on neuro fuzzy approach.

ACKNOWLEDGMENT

REFERENCES

[1] Reddy Ch., KVSVN Raju.: A Concise Neural Network Model for Estimating Software Effort International Journal of Recent Trends in Engineering, Issue. 1, Volume 1 (2009).

[2] Bayindir, Colak R., Sagiroglu I., Kahraman S., H.T.:Application of Adaptive Artificial Neural Network Method to Model the Excitation Currents of Synchronous Motors, 11th International Conference on in Machine Learning and Applications (ICMLA) , vol 2, pp.498-502, (2012).

[3] Kumari S.: Performance Analysis of the Software Cost Estimation Methods A Review, International Journal of Advanced Research in Computer Science and Software Engineering, Issue 7,vol 3,(2013).

[4] Sharma T..: Statistical Analysis of various models of Software Cost Estimation International Journal of Engineering Research and Applications (IJERA),Issue 3,vol 2,pp.683-685, (2012)

[5] Kaushik A., Soni A.K, Soni R.: A Simple Neural Network Approach to Software Cost Estimation, Global Journal of Computer Science and Technology Neural & Artificial Intelligence, Issue 1, vol 13, Version 1.0,(2013)

[6] Hamza H., Kamel, Shams A., K.: Software Effort Estimation Using Artificial Neural Networks A Survey of the Current Practices, Tenth International Conference on Information Technology: New Generations (ITNG), pp.731-733, 15-17, (2013).

[7] Krenke A., Bešter J., Kos A..: Introduction to the Artificial Neural Networks, Methodological Advances and Biomedical Applications,(2011)

[8] Sharma T.: A Comparative study of COCOMO II and Putnam models of Software Cost Estimation, International Journal of Scientific & Engineering Research , Issue 11,vol 2, (2011).

[9] Tadayon N.: Neural Network Approach for Software Cost Estimation, IEEE International Conference on Information Technology: Coding and Computing (ITCC'05), vol 2 ,pp 815-818,4-6,(2005).

[10] Mukherjee S.: Optimization of Project Effort Estimate Using Neural Network IEEE International Conference on Advanced Communication Control and Computing Technologies, pp 406-410,(2014).

[11] Jodpimai, P., Lursinsap P., C.: Estimating Software Effort with Minimum Features Using Neural Functional Approximation in International Conference on Computational Science and Its Applications (ICCSA), pp.266-273, 23-26 (2010)

[12] Kaushik A., Chauhan A., Mittal D.,: COCOMO Estimates Using Neural Networks, International Journal if Intelligent Systems and Applications (IJISA), vol 4, No.9, (2012)

[13] Ghose, M.K., Bhatnagar, Bhattacharjee R., V.: Comparing some neural network models for software development effort prediction, National Conference on Emerging Trends and Applications in Computer Science (NCETACS), pp.1-4, 4-5 (2011).

[14] http://promise.site.uottawa.ca/SERepository/datasets/cocomo81.arff

[15] Andrej Krenke, Janez Bešter, Andrej Kos, "Introduction to the Artificial Neural Networks", Methodological Advances and Biomedical Applications, April 11, 2011.