Reusability in Software Effort Estimation model based on Artificial Neural Network for Predicting Effort in Software Development

Jyoti Mahajan[1], **Devanand**[2]

[1]Assistant Professor, Department of Computer Engineering,

Govt. College of Engg. & Tevhnology, Jammu

[2]Professor, Department of Computer Science & IT, University of Jammu.

Abstract: *Most of the recent research initiatives have focused on development of formal estimation models to improve estimation accuracy. Formal estimation models have been developed to measure lines of code or size of software projects. But, most of the models have failed to improve estimation accuracy. This paper focuses on the reusability in software development effort estimation in early stage of development based on ANN. Software reuse saves the software effort and improves productivity. This paper has proposed a new model called REBEE (Reusability Based Effort Estimation) for better effort estimation accuracy and reliability. This proposed model's accuracy is more improved with the help of ANN Enhanced Back Propagation algorithm. To evaluate the performance of the proposed model a set of projects compared with the existing COCOMO II model by MRE, MMRE and PRED for evaluation of software cost estimation. The final results show that the use of proposed REBEE model effort estimation is reliable, accurate and predictable in the early stage of development of project.*

Key words: *Effort Estimation, Software Reuse, COCOMO II, Artificial Neural Network*

1. Introduction

A survey on software effort estimation revealed that most of the projects were failed due to effort overrun and exceeding its original estimates. It also stated that 60-80 percent of software projects encounter effort overruns [1]. Effort overruns usually lead to cost overruns and missed project deadline. This would cause lack of productivity or loss of business. Software effort estimation is one of the most critical and complex task in software engineering, but it is an inevitable activity in the software development processes. Over the last three decades, a growing trend has been observed in using variety of software effort estimation models in diversified software development processes. Along with this tremendous growth, it is also realized that the essentiality of all these models in estimating the software development costs and preparing the schedules more quickly and easily in the anticipated environments.

Although a great amount of research time and money have been invested to improve the accuracy of the various estimation models. Due to the inherent uncertainty in software development projects such as complex and dynamic interaction factors, change of requirements, intrinsic software complexity, pressure on standardization and lack of software data, it is unrealistic to expect very accurate effort estimation of software development processes[2].

Software reuse has been given great importance in software development for decades. Software reuse has benefits such as reduced effort, improved productivity, decreased time-to-market and decreased cost. This research work addresses the significance of reusability in effort estimation and formulates new metrics for reusability to determine the reliable and accurate effort estimates.

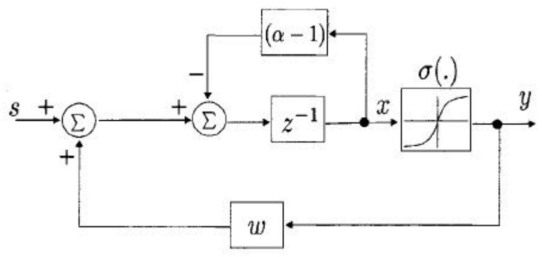
Predictable and reliable effort estimation is the challenging task in software project management. In research there are numbers of attempts to develop accurate cost estimation models based on various techniques. However, the evaluation of accuracy and reliability of the model shows its advantages and weaknesses. Selecting an appropriate model for a specific project is an issue in project management. The appropriate model which provides minimum relative error has to be considered as the best fit for effort estimation. In last decades, various methods for cost and effort estimation have been proposed in the following three categories:

- ✷ Expert Judgment (EJ)
- ✷ Algorithmic Models (AM)
- ✷ Machine Learning (ML)

Software reuse has been given great importance in software development for decades. Software reuse has benefits such as reduced effort, improved productivity, decreased time-to-market and decreased cost. This research work addresses the significance of reusability in effort estimation and formulates new metrics for reusability to determine the reliable and accurate effort estimates. This paper proposed a new model called REBEE. The accuracy of the proposed model has been improved with the help of ANN Enhanced Resilient Backpropagation (ERPROP) algorithm.

2. Proposed Model - REBEE

The major goal of this proposed model is estimating more accuracy and reliable software effort with the help of software reusability concept. To implement ANN model, COREANN effort estimation Equation 1 should be transform from non linear model to linear model by applying natural logarithm on both sides. ANN is implemented with Enhanced RPROP.
A simple DNN *i* is as shown in Fig. 1.



A simple structure of the DNN is shown in Fig. 2.1 given below.

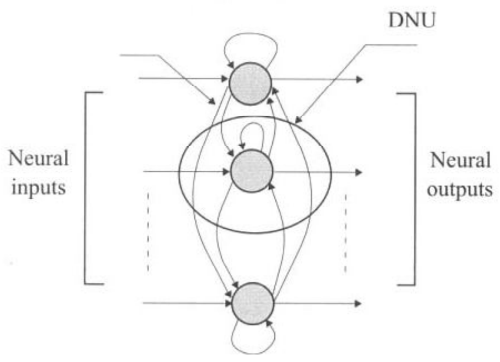


Fig. 2.2 A Simple DNN Structure

ln(PM) = ln(A) + 0.91 * ln(SIZE) + SF_1 * 0.01 * ln(SIZE) + …. + SF_5 * 0.01 * ln(SIZE) + ln(EM_1) + ln(EM_2) + ……… + ln(EM_{17}) ———— 1

[Linear Equation]

OP_{est} =WT_0 + WT_1 * IP_1 + WT_2 * IP_2 + …+ WT_6 * IP_6 + WT_7 * IP_7 +…+ WT_{23} * IP_{23} __________________ 2 [ANN Based Model For Effort Estimation]

Where

OP_{est} = ln(PM)

IP_1 = 0.91 * ln(SIZE)

IP_2 = SF_1 * ln(SIZE),………., IP_6 = SF_5 * ln(SIZE)

IP_7 = ln(EM_1),………,IP_{23} = ln(EM_{17})

WT_0 = ln(A)

WT_1 = 1,…………, WT_{23} = 1

IP_1 to IP_{23} => Inputs

OP_{est} => Output

WT_0 => Bias

WT_1 …… WT_{23} => Weights (Initial Value is 1)

Actual observed effort is compared with this estimated effort. The differences between these values are the error in the effort. It should be minimized.

3. Learning Algorithm

The basic principle of ERPROP is to eliminate the harmful influence of the size of the partial derivative on the weight step. Initially, the Enhanced RPROP algorithm is declared the following parameters :

(i) The increase factor value is $\eta^+ = 1.2$

(ii) The decrease factor value is $\eta^- = 0.5$

(iii) The initial update-value is “$_0$ = 0.1 (“$_{ij}$ = “$_0$)

(iv) The values of “$_{max}$ < 50 and “$_{min}$ > 1e^{-6}

```
for all weights and biases (i,j = 1,2,.......,N)
{
  if ($\frac{\partial E}{\partial w_{ij}}(t-1) * \frac{\partial E}{\partial w_{ij}}(t) > 0$) then
  {
    $\Delta_{ij}(t) = \text{minimum}(\Delta_{ij}(t-1) * \eta^+, \Delta_{max})$
    $\Delta w_{ij}(t) = -\text{sign}(\frac{\partial E}{\partial w_{ij}}(t)) * \Delta_{ij}(t)$
    $w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t)$
    $\frac{\partial E}{\partial w_{ij}}(t-1) = \frac{\partial E}{\partial w_{ij}}(t)$
    $\Delta w_{ij}(t-1) = \Delta w_{ij}(t)$
  }
  else if ($\frac{\partial E}{\partial w_{ij}}(t-1) * \frac{\partial E}{\partial w_{ij}}(t) < 0$) then
  {
    $\Delta_{ij}(t) = \text{maximum}(\Delta_{ij}(t-1) * \eta^-, \Delta_{min})$
    $\frac{\partial E}{\partial w_{ij}}(t) = 0$
  }
  else if ($\frac{\partial E}{\partial w_{ij}}(t-1) * \frac{\partial E}{\partial w_{ij}}(t) = 0$) then
  {
    $\Delta w_{ij}(t) = -\text{sign}(\frac{\partial E}{\partial w_{ij}}(t)) * \Delta_{ij}(t)$
    $w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t)$
    $\frac{\partial E}{\partial w_{ij}}(t-1) = \frac{\partial E}{\partial w_{ij}}(t)$
    $\Delta w_{ij}(t-1) = \Delta w_{ij}(t)$
  }
}
```

4. Performance Measures

Company database containing 20 projects is used to test the proposed COREANN model. The following evaluation criterion is used to assess and compare the performance of the proposed model with existing COCOMO II Model.

A common criterion for the evaluation of cost estimation model is the magnitude of relative error (MRE), and mean magnitude of relative error (MMRE). MRE is defined as

$$MRE = \frac{|ActualEffort - PredictedEffort|}{ActualEffort} * 100 \quad \text{———} \; 3$$

And Mean Magnitude of Relative Error (MMRE) for N projects is defined as [11]

$$MMRE = \frac{1}{N}\sum_{i=1}^{N} MRE_i \quad \text{———} \; 4$$

Next to calculate the PRED (p) value. If lower MRE & MMRE and higher PRED(25), the software estimation model effort is more accurate and predictable than other models .

$$PRED(p) = \frac{K}{N} * 100 \quad \text{———} \; 5$$

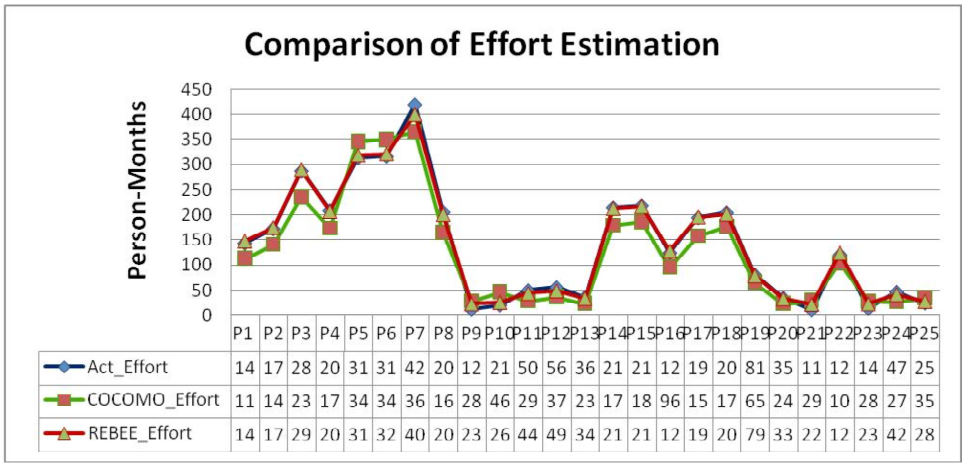
where K is the number of projects where MRE is less than or equal to p (normally p value is 25%).

5. Results

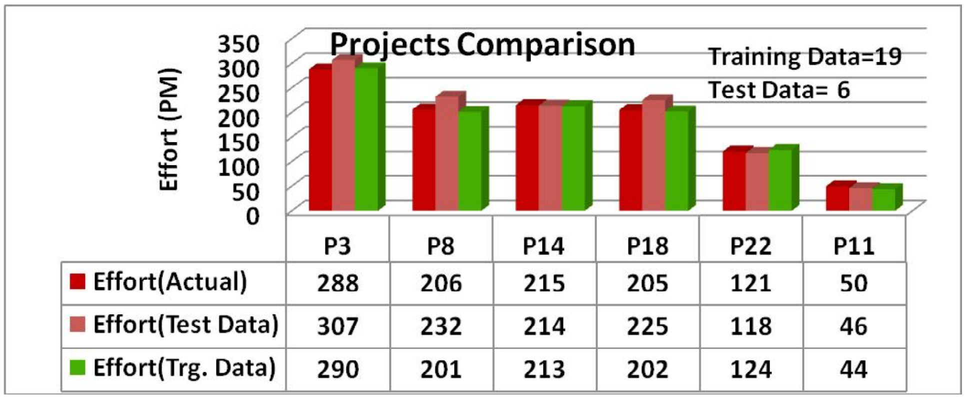
Out of the 25 project dataset, to forecast an effort of the proposed model. The estimated effort is comparing with existing COCOMO II and Actual effort of the project. This results are shown as Table – 1 and comparison graph also provided as below:

Table – 1 shows that the result for Effort Comparison of the proposed model with existing COCOMO II.

Project_Id	Actual_Effort	COCOMO_Effort	REBEE_Effort
P1	144	112.6	148
P2	173	141	175
P3	288	235	290
P4	209	173	207
P5	314	345.7	319
P6	317	349	322
P7	420	365	400
P8	206	165	201
P9	12	28	23
P10	21	46	26
P11	50	29	44
P12	56	37	49
P13	36	23	34
P14	215	178.5	213
P15	219	185.6	217
P16	125	95.6	128
P17	195	158	196
P18	205	176	202
P19	81	65	79
P20	35	23.5	33
P21	11	29	22
P22	121	104	124
P23	14	28	23
P24	47	27	42
P25	25	35	28



	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25
Act_Effort	14	17	28	20	31	31	42	20	12	21	50	56	36	21	21	12	19	20	81	35	11	12	14	47	25
COCOMO_Effort	11	14	23	17	34	34	36	16	28	46	29	37	23	17	18	96	15	17	65	24	29	10	28	27	35
REBEE_Effort	14	17	29	20	31	32	40	20	23	26	44	49	34	21	21	12	19	20	79	33	22	12	23	42	28



	P3	P8	P14	P18	P22	P11
🟥 Effort(Actual)	288	206	215	205	121	50
🟥 Effort(Test Data)	307	232	214	225	118	46
🟩 Effort(Trg. Data)	290	201	213	202	124	44

6. Conclusion and Future work

In software engineering, it is extremely difficult to select appropriate model for estimation effort estimation due to the availability of number of models. Software reuse has become a major factor in development. Hence, effort estimation for reuse must accurate for the successful project execution. This paper primarily concentrated on the computation of accurate effort with software reusability as the main focus. While comparing performance results of REBEE and COCOMO II, it clearly shows that the proposed REBEE works better than COCOMO II. In the future we would like to further investigate the performance of REBEE over different project types and study its responses.

7. References

[1] K. Molokken-Ostvold and M. Jorgensen, "A Review of Surveys on Software Effort Estimation," Proc. 2003 ACM-IEEE Intternational Symposium on Empirical Software Eng, pp. 220-230, 2003.

[2] Saleem Basha and Dhavachelvan P, "Analysis of Empirical Software Effort Estimation Models", International Journal of Computer Science and Information Security, Vol. 7, No. 3, 2010

[3] Hughes, R.T., 'Expert judgement as an estimating method', *Information and Software Technology,* 38(2), pp 67-75, 1996.

[4] Rush, C., and Roy, R. (2001). Expert Judgment in cost estimating: Modelling the reasoning process. Concurrent Engineering: Research and Applications (CERA) Journal, 9(4), December 2001.

[5] M. Jørgensen, "A Review of Studies on Expert Estimation of Software Development Effort," J. Systems and Software, vol. 70, nos. 1–2, pp. 37–60, 2004.

[6] Putnam, Lawrence H. (1978). "A General Empirical Solution to the Macro Software Sizing and Estimating Problem". IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. SE-4, NO. 4, pp 345-361, 1978

[7] B. Boehm, B. Clark, E. Horowitz, C. Westland, R. Madachy, and R. Selby, "Cost Models for Future Software Life Cycle Processes: COCOMO 2. 0," Annals of Software Engineering: Special Volume on Software Process and Product Measurement, Science Publishers, vol. 1, pp. 45-60, 1995.

[8] Boehm, B. COCOMO II Model Definition Manual. Center for Software Engineering, University of Southern California. 1997.

[9] Chao-Jung Hsu, Nancy Urbina Rodas, Chin-Yu Huang and Kuan-Li Peng "A Study of Improving the Accuracy of Software Effort Estimation Using Linearly Weighted Combinations", 34th Annual IEEE Computer Software & Application Conference Workshops, 2010

[10] K. Srinivasan and D. Fisher, "Machine learning approaches to estimating software development effort," IEEE Transactions on Software Engineering, vol. 21, pp. 126-137, 1995.

[11] A. R. Venkatachalam, "Software Cost Estimation Using Artificial Neural Networks," Presented at 1993 International Joint Conference on Neural Networks, Nagoya, Japan, 1993.

[12] Balda, D,. M. and D. A. Gustafson, "Cost Estimation Models for the Reuse and Prototype Software Development Life-Cycles", ACM Sigsoft Software Engineering Notes, 15 (3), pp. 4250, 1990.

[13] A.Windsor Brown url: http://csse.usc.edu/publications/TECHRPTS/KBSA_Tech_Report/app5.pdf, 1999

[14] Barry Boehm, A.Winsor Brown, http://csse.usc.edu/publications/TECHRPTS/KBSA_Tech_Report/volumeI.pdf, 1999

[15] Sunita Chulani, Barry Boehm "Modeling Software Defect Introduction Removal: COQUALMO (COnstructive QUALity MOdel)", Technical Report USC-CSE-99-510, 1998

[16] C.Abst, B.Boehm, E.Clark. "COCOTS: A COTS Software Integration Lifecycle Cost Model - Model Overview and Preliminary Data Collection Findings", Technical report USC-CSE-2000-501, USC Center for Software Engineering, 2000.

[17] Heiat A, "Comparison of artificial neural network and regression models for estimating software development effort," Journal of Information and Software Technology, Volume 44, Issue 15, Pages 911-922, 2002.

[18] Wittig, G., Finnie, G., “Estimating software development effort with connectionist models”, Information and Software Technology, 39 (7), 469–476, 1997

[19] Karunanithi, N., D. Whitely, Y. K. Malaiya, “Using neural networks in reliability prediction”, IEEE Software, pp. 53-59, 1992.

[20] Tadayon, N., “Neural network approach for software cost estimation,” International Conference on Information Technology: Coding and Computing (ITCC 2005), Volume: 2, on page(s): 815- 818, 2005.

[21] Dawson, C.W., “A neural network approach to software projects effort estimation,” Transaction: Information and Communication Technologies, Volume 16, pages 9, 1996.

[22] Lionel C. Briand, Sandro Morasca, and Victor R. Basili, “An Operational Process for Goal-Driven Definition of Measures”, IEEE Transactions On Software Engineering, Vol. 28, No. 12, 2002.

[23] V. Basili, “Software Modeling and Measurement: The Goal/Question/ Metric Paradigm” University of Maryland, Department of Computer Science, Tech. Rep. CS-TR-2956, 1992.

[24] L. Briand, K. El Emam, S. Morasca, “Theoretical and Empirical Validation of Software Product Measures”, Technical Report ISERN-95-03, Fraunhofer Institute for Experimental Software Engineering, Germany, 1995.

[25] Murat Ayyýldýz, Oya Kalýpsýz, and Sýrma Yavuz, “A Metric-Set and Model Suggestion for Better Software Project Cost Estimation”, World Academy of Science, Engineering and Technology, 2006.

[26] G. Poels, G. Dedene, DISTANCE: A Framework for Software Measure Construction, Reserch Report 9937, Dep. of Applied Economics, Katholieke Universiteit Leuven, 1999.

[27] Mitat Uysal, “Estimation of the Effort Component of the Software Projects Using Simulated Annealing Algorithm”, World Academy of Science, Engineering and Technology ,2008.