ISSN XXXX XXXX © 2017 IJESC

Research Article Volume 7 Issue No.5

Comparative Study of Software Effort Estimation using Different Algorithms: A Review Paper

Rishi Kishore[1], D.L.Gupta[2]
M.Tech Student[1], Assistant professor[2]
Department of computer Science &Engineering
Kamla Nehru Institute of Technology, Sultanpur, UP, India

Abstract:
Software effort estimation is the process which is studying during last 3 decades. Many of the researchers work on the estimation of effort using their own logic or ideas for estimate the accurate effort of software. Accurate estimation is a substantial factor in projects success and reducing the risks. In recent years, software effort estimation has received a considerable amount of attention from researchers and became a challenge for software industry. In the last two decades, many researchers and practitioners proposed statistical and machine learning-based models for software effort estimation. So that there are a number of algorithm are available for effort estimation and they are very near to the actual effort. This review paper compares the software effort estimation among three models that are COCOMO Model [2] [8], Genetic Algorithm and Neuro Fuzzy Model[11]. In this paper there is a Manhattan Distance, Mean Magnitude Relative Error and Root Mean Square evaluated at last and compare it with all models.

Keyword: COCOMO [2], Neuro Fuzzy Model[11], Genetic Algorithm[7][9]

I. INTRODUCTION

One of the aspect of foremost purpose of the software development organization is to develop models which are practically relevant. Another aspect is that the community considers as a prime objective is to estimate how accurately a model can estimate the effort incorporate in developing a software. Effort estimation is a process by which one can predict the development time and cost for developing a software process or a product. Estimated cost and time accurately is vital for couple of reasons. Over estimation may lead to harm in financial loss in an organization whereas under estimation may result in poor quality of software which eventually leads to failure of the software. Also Effort estimations done at an initial phases of development of a project may turn out to be helpful for project managers. But very less information is available at early stages. There are many already existing algorithmic and non-algorithmic methods for effort estimation. Dynamic environments in software development technology make effort predictions. Further exciting especially in early stages. So that it can be easy to find the effort estimation. This review paper is organized as follows: in Section 2, we describe software cost estimation in different models and algorithms. In Section 3, calculate the effort using different models. In Section 4, compare the estimated effort with table and graphs. And in Section 5, we present the conclusion and future work.

2. A. THE GENETIC ALGORITHM

Genetic Algorithms are search algorithms that are based on the concepts of natural selection and natural genetics. Genetic algorithm operates on chromosomes. Algorithm is started with a **set of solutions** (represented by **chromosomes**) called **population**. Solutions from one population are taken and used to form a new population. This is motivated by a hope, that the new population will be better than the old one. Solutions which are selected to form new solutions (**offspring**) are selected according to their fitness - the more suitable they are the more chances they have to reproduce. This is repeated until some condition (for example number of populations or improvement of the best solution) is satisfied. Some of the terms used in the GA should be familiar, so the explanations are given as follows.

a) Search Space

The space of all feasible solutions (it means objects among those the desired solution is) is called **search space** (also state space). Each point in the search space represent one feasible solution. Each feasible solution can be "marked" by its value or fitness for the problem. We are looking for our solution, which is one point (or more) among feasible solutions - that is one point in the search space.

Genetic algorithm pseudo code

```
1. [Start] Generate random population of $n$ chromosomes (suitable solutions for the problem).

2. [Fitness] Evaluate the fitness $f(x)$ of each chromosome $x$ in the population.

3. [New population] Create a new population by repeating following steps until the new population is complete.
    1. [Selection] Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected).
    2. [Crossover] With a crossover probability cross over the parents to form a new offspring (children). If no crossover was performed, offspring is an exact copy of parents.
```

3. **[Mutation]** With a mutation probability mutate new offspring at each locus (position in chromosome).

4. **[Accepting]** Place new offspring in a new population.

5. [Replace] Use new generated population for a further run of algorithm.

6. [Test] If the end condition is satisfied, **stop**, and return the best solution in current population.

7. [Loop] Go to step **2.**

b) Crossover
Crossover selects genes from parent chromosomes and creates a new offspring. The simplest way how to do this is to choose randomly some crossover point and everything before this point copy from a first parent and then everything after a crossover point copy from the second parent.
Crossover can then look like this (| is the crossover point):

Table 1

Chromosome 1	11011 | 00100110110
Chromosome 2	11011 | 11000011110
Offspring 1	11011 | 11000011110
Offspring 2	11011 | 00100110110

There are other ways how to make crossover, for example we can choose more crossover points. Crossover can be rather complicated and vary depends on encoding of chromosome. Specific crossover made for a specific problem can improve performance of the genetic algorithm.

c) Mutation
After a crossover is performed, mutation take place. This is to prevent falling all solutions in population into a local optimum of solved problem. Mutation changes randomly the new offspring. For binary encoding we can switch a few randomly chosen bits from 1 to 0 or from 0 to 1. Mutation can then be following:

Table 2

Original offspring 1	1101111000011110
Original offspring 2	1101100100110110
Mutated offspring 1	1100111000011110
Mutated offspring 2	1101101100110110

The mutation depends on the encoding as well as the crossover. For example when we are encoding permutations, mutation could be exchanging two genes. After running this algorithm[15] ,the new value of a and b i.e. 2.022817 and 0.897183 found respectively for the basic COCOMO model.

B. COCOMO (Constructive cost model)
The constructive cost model was developed by Barry W.Boehm in the late 1970s and published in Boehm's 1981 book *Software Engineering Economics* as a model for estimating effort, cost, and schedule for software projects. It drew on a study of 63 projects at TRW Aerospace where Boehm was Director of Software Research and Technology.
COCOMO consists of a hierarchy of three increasingly detailed and accurate forms.

1 The Basic *COCOMO* Model
1. The Intermediate *COCOMO* Model
2. The Detailed *COCOMO* Model

1 The basic COCOMO model is a single-valued, static model that computes software development effort (and cost) as a function of program size expressed in estimated thousand delivered source instructions (KDSI). The basis COCOMO is also called COCOMO'81 model.

2 The intermediate COCOMO model computes software development effort as a function of program size and a set of fifteen "cost drivers" that include subjective assessments of product, hardware, personnel, and project attributes.

3 The advanced or detailed COCOMO'81 model incorporates all characteristics of the intermediate version with an assessment of the cost driver's impact on each step (analysis, design, etc.) of the software engineering process.

The Basic COCOMO computes software development effort (and cost) as a function of program size. Program size is expressed in estimated thousands of source lines of code (SLOC, KLOC). COCOMO applies to three classes of software projects:

The general formula of the basic COCOMO model is:

$$\mathbf{E = a \times (S)^{b}}$$

Where **E** represents effort in person-months, **S** is the size of the software product in KLOC (Kilo Lines Of Code) and, **a** and **b** are values dependent on the given table 3.

Table 3

Basic COCOMO	**E=a (Size)^{b}**
organic	**a** = 2.4, **b** = 1.05
semi-detached	**a** = 3.0, **b** = 1.12
embedded	**a** = 3.6, **b** = 1.20

These basic COCOMO model are based on the size of KLOC, i.e. organic model is used for small size of software product, semi-detached model is used for medium size software product and embedded model is used for large size of software product. In the updated model i.e. COCOMO II, another component EAF (Effort Adjustment Factor) is used for computing the effort more accurate.

C. Neuro Fuzzy Model
Neuro Fuzzy Model is also used for solving the software effort estimation problem, the neurons plays a vital role in the Neuro Fuzzy Model[17]. There are two basic types of Neuro-fuzzy systems: Mamdani Neuro-fuzzy systems and Takagi-Sugeno Neuro-fuzzy system.

The Sugeno based Neuro-fuzzy model has the following advantages over Mamdani Model:
(a) Expressing a clearer concept of inference process.
(b) Less number of fuzzy rules
(c) Faster learning
(d) Less memory space.

In this Neuro-fuzzy system[11] the Sugeno Based Fuzzy Inference System is designed with the initialization of the Membership Function of the different attributes and linear Membership Function for the output and deducing the fuzzy rules from the data. Then the Sugeno Based FIS is trained with the neural Network using the hybrid training algorithm. In the forward pass, Backpropagation learning algorithm and in the backward pass Least Mean Square Error (LMS) learning algorithm is used to update the non-linear and linear parameters of the Neuro-fuzzy system respectively.

According to the inference rule base concept, two Takagi-Sugeno if-then rules are as follows:

Rule 1: If x is A1 and y is B1, then f1=P1x+q1y+r1.
Rule 2: If x is A2 and y is B2, then f2=P2x+q2y+r2.

On the basis of this Neuro Fuzzy System, the existing estimated effort using the MATLAB Tool is compare with the other models in table 5.

3. EVALUATION CRITERIA AND DATA SET

In this review paper, the effort is estimated using different algorithm and calculate the estimated effort for software development. After that the MD (Manhattan Distance), MMRE (Mean Magnitude Relative Error) and RMS(Root Mean Square) is calculated using the following equations.

$$\text{MD} = \sum_{i=1}^{n} (|\text{Effort} - \text{Estimated Effort}|) \qquad (8)$$

$$\text{MMRE} = \frac{1}{n} \sum_{i=1}^{n} \frac{|\text{Effort}_i - \text{Estimated Effort}_i|}{\text{Effort}_i} \qquad (9)$$

$$\text{RMS} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (|\text{Effort} - \text{Estimated Effort}|)} \qquad (10)$$

The evaluation is based on the well known NASA Dataset of 18 projects[15]. The dataset are given in the table 4 as follows.

Table 4

Project No.	KLOC	Methodology (ME)	Actual Effort
1	90.2	30	115.8
2	46.2	20	96.0
3	46.5	19	79.0
4	54.5	20	90.8
5	31.1	35	39.6
6	67.5	29	98.4
7	12.8	36	18.9
8	10.5	34	10.3
9	21.5	31	28.5
10	3.1	26	7.0
11	4.2	19	9.0
12	7.8	31	7.3
13	2.1	28	5.0
14	5.0	29	8.4
15	78.6	35	98.7
16	9.7	27	15.6
17	12.5	27	23.9
18	100.8	34	138.3

4. EFFORT AND PERFORMANCE COMPARISON

On the basis of above algorithms, we evaluate the software development effort using the given NASA dataset and compare them with the actual effort. The comparison of efforts is given in the table 5. According to the values getting from the evaluation we plot the graph using MatLab tool.

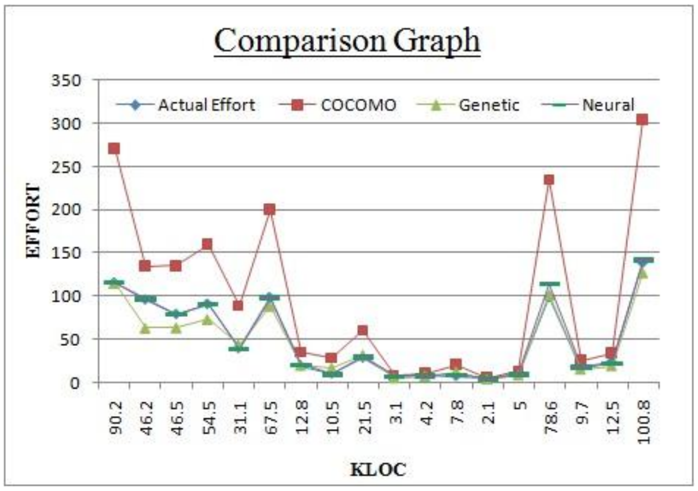
Table 5

Project no.	Actual effort	Basic COCOMO effort	Genetic algorithm effort	Neuro - fuzzy Model
1	115.8	96.91239	114.851	115.7
2	96.0	28.02283	63.01516	96.069
3	79.0	28.21393	63.38216	78.903
4	90.8	55.12036	73.08395	90.813
5	39.6	18.49423	44.18103	38.875
6	98.4	70.04356	88.54761	98.447
7	18.9	7.281292	19.92171	20.789
8	10.3	5.914074	16.67824	9.9815
9	28.5	12.55158	31.72472	29.343
10	7.0	1.642736	5.582104	7.3786
11	9.0	2.259694	7.330359	8.6934
12	7.3	4.328499	12.77405	8.9732
13	5.0	1.09136	3.93592	4.6705
14	8.4	2.713666	8.571574	10.129
15	98.7	83.0656	101.5073	113.81
16	15.6	5.441871	15.53358	18.126
17	23.9	7.102209	19.50229	22.647
18	138.3	109.7549	126.89	141.6

We also evaluate the value of Manhattan Distance (MD), Mean Magnitude Relative Error (MMRE) and Root Mean Square (RMS), using the equation 8, 9, 10 which are given as above.

Table 6

Model	MD	MMRE	RMS
COCOMO	346.0527	0.475664	25.88145
GA	119.7405	0.178024	10.56776
Neuro Fuzzy	30.7074	.064588	3.787555



5. CONCLUSION

This review paper proposes the software effort estimation based of three model, first The COCOMO Model, second the Genetic Algorithm and the third is Neuro Fuzzy Model. Every of the model using different methods for evaluating the effort. The COCOMO Model uses the Boehm's COCOMO formula for finding the software effort. Similarly second and third model uses some tools used in MatLab for optimize the component for finding the better result in the form of effort. At last on the basis of the result obtain from the table 5, we compare all models and graph is plotted to show the performance of all model. In this review paper, Neuro Fuzzy Model is founded the best model to evaluate the close result in comparison with the other given effort in the models.

6. REFERENCES

[1] K. Molokken and M. Jorgensen, A Review of Software Surveys on Software Effort Estimation, In *ISESE 2003. Proceedings International Symposium on Empirical Software Engineering*, IEEE, pp. 223–230, (2003).

[2] B. W. Boehm, *et al.*, Software Engineering Economics, Prentice-hall Englewood Cliffs (NJ), vol. 197, (1981).

[3] A.Sharma and D. S. Kushwaha, *Estimation of Software Development Effort from Requirements Based Complexity*, Procedia Technology, vol. 4, pp. 716–722, (2012).

[4] A.Sharma, M. Vardhan and D. S. Kushwaha, A Versatile Approach for the Estimation of Software Development Effort Based on srs Document, *International Journal of Software Engineering and Knowledge Engineering*, vol. 24(01), pp. 1–42, (2014).

[5] Gall, Innina, O. Burceva and S. Parsutins, The Optimization of Cocomo Model Coefficients Using Genetic Algorithms, (2012).

[6] K. Choudhary, "GA based Optimization of Software Development effort estimation", International Journal of Computer Science and Technology ,vol.1 (1), pp. 38-40,2010.

[7] A.S.Vishali, "COCOMO model Coefficients Optimization Using GA and ACO", International Journal of Advanced Research in Computer Science and Software Engineering ,vol. 4 (1), PP.771-777,2014.

[8] L. Oliveira, "GA-based method for feature selection and parameters optimization for machine learning regression applied to software effort estimation", information and Software Technology, vol.52,PP.1155-1166, 2010.

[9] O.Galilina, "The Optimization of COCOMO Model Coefficients Using Genetic Algorithms", Information Technology and Management Science ,pp. 45-51,2012.

[10] Urvashi Rahul Saxena, 2S.P Singh," Software Effort Estimation Using Neuro-Fuzzy Approach", *Assistant Professor, Department of Computer Science and Engineering, JSS Academy of Technical Education, Noida, India Assistant Professor, Computer Science Department, Birla Institute of Technology, Noida, India.*

[11] V. Khatibi, and D. Jawawi, " *Software Cost Estimation Methods: A Review"*, Journal of Emerging Trends in Computing and Information Sciences ,vol 2 ,pp. 21-29,2011.

[12] S. M. Sabbagh Jafari, F. Ziaaddini," *Optimization of Software Cost Estimation using Harmony Search Algorithm"* 1st Conference on Swarm Intelligence and Evolutionary Computation (CSIEC2016), Higher Education Complex of Bam, Iran, 2016.

[13] K.Choudhary, "GA based Optimization of Software Development effort estimation", International Journal of Computer Science and Technology ,vol.1 (1),pp. 38-40,2010.

[14] V. Khatibi, and D. Jawawi, " Software Cost Estimation Methods: A Review", Journal of Emerging Trends in Computing and Information Sciences ,vol 2 ,pp. 21-29,2011.

[15] Rohit Kumar Sachan, Ayush Nigam, Avinash Singh, Sharad Singh, Manjeet Choudhary, Avinash Tiwari and Dharmender Singh Kushwaha," *Optimizing Basic COCOMO Model using Simplified Genetic Algorithm"* Twelfth International Multi-Conference on Information Processing-2016 (IMCIP-2016).

[16] Anupama Kaushik, Assistant Professor, Dept. of IT, Maharaja Surajmal Institute of Technology, GGSIP University, Delhi, India," *COCOMO Estimates Using Neural Networks", I.J. Intelligent Systems and Applications*, 2012, 9, 22-28.