EMPIRICAL BASED EFFORT ESTIMATION USING MACHINE LEARNING ALGORITHMS

V. Vignaraj Ananth[1], Dr.S. Srinivasan[2]

[1]Assistant Professor, [2]Associate Professor

[1]Department of Computer Science and Engineering, Thiagarajar College of Engineering, Madurai, India

[1]Department of Computer Science and Engineering, RMD Engineering College, Chennai, India

vignaraj112@gmail.com, ssn.cse@rmd.ac.in

Abstract

Estimation the budget is one of the major tasks in project management. There arises a need to more accurately estimate the required schedule and resources for the software projects. The software estimation process includes estimating the size of the software product, effort needed, development of project schedules, and estimating the overall budget of the project. To estimate the budget we need to consider effort, time and environment. Estimating the effort is the tedious process. Effort is represented as a function of size. In this paper, size is represented in term as Fuzzy number. A new model is approached in this paper by machine learning algorithms to estimate the effort required in the software process. The optimization of the effort parameters is achieved using the M5P technique to obtain better prediction accuracy. Furthermore, performance comparisons of the models obtained using the M5P technique with Random Forest technique are presented in order to highlight the performance achieved by each technique.

Keywords: Lines of Code (LOC), Function Point Analysis(FPA), Triangular Membership Function (TAMF), Trapezoidal Membership Function (TPMF), M5P technique, Random Forest technique.

1. Introduction

In this paper, we propose a new machine learning approach, and non-algorithmic problem to estimate of effort more appropriate in comparison with other Fuzzy models and algorithmic approaches like COCOMO Model, Doty Model, Halsted Model, Walston Felix Model, and Baili-Basili Model.

There are several techniques to estimate effort, such as, Estimation by Analogy, Top-Down approach, Bottom-Up approach [1]. A realistic approach to estimate the effort is algorithmic approach, which uses mathematical equations to estimate effort. The mathematical equations are mostly related to historical, research data and the inputs based on Lines of Code (LOC). COCOMO Model [2], which is an open model, to estimate the number of Person-Months required for the project. The Effort can be calculated by (1),

Effort = a* $(size)^b$, ----- (1)

Where a, b are empirical constants.

In indirect approach, The Function Point Analysis (FPA) begins by functional decomposition of the project and uses data functions and transactional functions to represent the functionality provided to the user. The data functions are Internal Logical File (ILF), External Interface File (EIF). The transactional functions are External Input (EI), External Output (EO) External Inquiry (EI). The FPA can be calculated by, FPA= $\sum\sum F_{ij} * Z_{ij}$, for j= 1 to 3 and i = 1 to 5, where Z_{ij} denotes count for component i at level (low, average or high) j, and F_{ij} corresponds to Function Points.

Yet another approach to estimate the effort is by using one of the machines learning technique, Fuzzy Logic. Fuzzy Logic is used to find fuzzy functional points and the corresponding result is defuzzified using its membership functions. Hence, the size estimation is in person-months.

2. Need for Effort Estimation

Effective monitoring and control of the software budget, to verify and improve accuracy of estimates is required. Success of an effort estimate method is not necessarily the accuracy of the initial estimates, but rather the rate at which estimates converge to the actual cost [3]. Many projects that have been developed over large distributed systems have 75 percent of the projects with over budget. 63 percent of the projects cost are more than the initial estimates. This algorithm provides the list of features that can be included and reduces the risk by scheduling costly tasks preliminary. Also, It gives more number of resources to the costly projects and assigns well experienced personnel to costly projects. Sometimes, it is called man-power loading as the required number of engineering and management personnel's are allocated to a project in a given amount of time.

2.1. Algorithmic models

In these approaches, The effort is estimated using direct algorithms like COCOMO, Doty Model, Halsted Model, Walston Felix Model, and Baili-Basili Model [4]. By specifying the lines of code, the effort can be calculated by using these metrics.

COCOMO Model
Effort= 3.2 * (kloc) $^{1.05}$
Doty Model
Effort=5.288*(kloc) $^{1.047}$
Halsted Model
Effort= 5.2 * (kloc) $^{1.50}$
Baili-Basili Model
Effort=5.5 + (0.73 *(kloc) $^{1.16}$)
Walston Felix Model
Effort= 5.2 * (kloc) $^{0.91}$

By using the above mentioned equations, effort estimation is carried out for the projects.

2.2. Function point analysis

Function points measure software size based on the functionality requested by and provided to the end user.

Function point counting resources includes in User/analyst interviews, Requirements documents, Design documents, Data dictionaries, Use cases, User guides, Screen captures, and Actual software, Entity-relationship models and Semantic object models. Function Point represents a logical size such as LOC [5]. More complex functions contribute higher number of function points to logical size. Also, it uses data and transactional functions provided to the user.FPA is calculated by multipliers as shown in Fig. 1. Using these characteristics, they are summed to get an (2) "unadjusted function-point total (ufpt)".

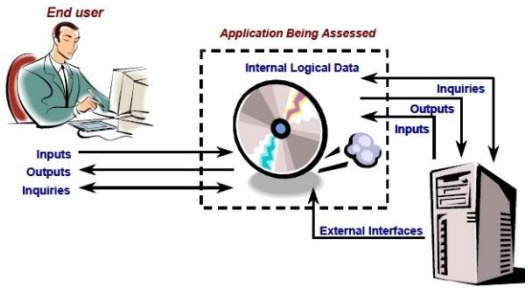

Fig. 1. FPA calculation

FPA=ufpt*[0.65+ (0.01*Σfi)] (2)

Where fi is the complexity adjustment factor varies from 0 to 14.

By using this metric, LOC can be calculated and Effort can be estimated for the project. Function point metrics are logical and comparable across projects, platforms, and languages.

Table 1. Program Characteristics

	Function Points		
Program Characteristics	**Low Complexity**	**Medium Complexity**	**High Complexity**
No. of Inputs	3	4	6
No. of Outputs	4	5	7
Inquiries	3	4	6
Logical Internal Files	7	10	15
External Interface Files	5	7	10

2.3. Fuzzy logic

In this section, we present the Fuzzy Logic approach. Fuzzy Logic finds the fuzzy functional points (i.e.) the fuzzy set is characterized by a membership function with each point in the fuzzy set being a real number in the interval [0,1] called degree or grade of membership and a mathematical tool for dealing with uncertainty. Here the fuzzy functional points are formed by normalizing all the FPA values to [0, 1] and the obtained result is defuzzified to get the functional points and hence, we estimate the effort in person- months [7].Thus membership in a set is found to be binary i.e., Whether, the element is a member of a set or not. It can be indicated as,

$\chi A(x) = \{ 1, x \in A$

$0, x \in! A\}$

Where $\chi A(x)$ is the membership of element x in set A and A is the entire set on the universe.

The Fuzzy Membership function might be Triangular, Trapezoidal, or Bell shaped Membership function. Each membership function uses certain metrics to estimate the effort.

i) Triangular Fuzzy Logic

A triangular fuzzy number (TAFN) is described by the attributes (α, m, β), where m is the modal value, α and β are the right and left boundary respectively [9].

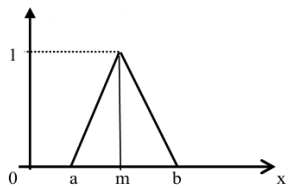

Fig. 2. Triangular Fuzzy Logic

The Triangular Membership Function (TAMF) (μ(x)) (3)for which is defined as:

$$\mu(x) = \begin{cases} 0 & , x \le \alpha \\ x - \alpha / m - \alpha & , \alpha \le x \le m \\ \beta - x / \beta - m & , m \le x \le \beta \\ 0 & , x \ge \beta \end{cases} \quad (3)$$

ii) Trapezoidal Fuzzy Logic

A trapezoidal fuzzy number (TPFN) is defined by its lower limit a, its upper limit d, and the lower and upper limits of its nucleus or Kernel b and c respectively.

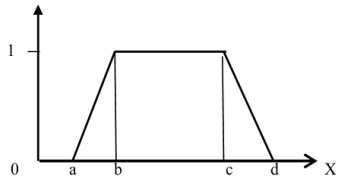

Fig. 3. Trapezoidal Fuzzy Logic

The Trapezoidal Membership Function (TPMF) (T(x)) for which is defined as (4):

$$T(x) = \begin{cases} 0 & , (x \le a) \text{ or } (x \ge d) \\ (x - a) / (b-a) & , x \ (a, b) \\ 1 & , x \ (b, c) \\ (d - x) / (d-c) & , x \ (c, d) \end{cases} \quad (4)$$

Based on the ratings the domain character values are fuzzified using TAMF and TPMF [9]. The value thus obtained is called membership function output, whose domain is specified, usually the set of real numbers, and whose range is the span of positive numbers in the closed interval [0, 1] (i.e.) Binary values. Each numerical value of the domain is assigned a specific value and 0 represents the smallest possible value of the membership function, while the largest possible value is 1.

2.4. Defuzzification

Defuzzification refers to the concept of applying fuzzy to crisp the conversions. The fuzzy results generated cannot be used as such and hence it is necessary to convert the fuzzy quantities into crisp quantities for the estimation of effort. This can be achieved by using defuzzification process. The defuzzification has the capability to reduce a fuzzy to a crisp single-valued quantity or as a set, or converting to the form in which fuzzy quantity is present. Defuzzification can also be called as "round off" method. Defuzzification reduces the collection of membership function values in to a single sealer quantity. Defuzzification is the process of producing a quantifiable result in fuzzy logic, given fuzzy sets and corresponding membership degrees. It will have a number of metrics that transforms variables into a fuzzy result, that is, the result is described in terms of membership in fuzzy sets. The defuzzification is applied to the value that had been obtained from the fuzzification process. The fuzzified output has to be defuzzified into the real number such that it gives the effort that has been required for the cost estimation. It can be calculated by (5),

$$D(y) = \begin{cases} \mu(x)*w1 & , 0<c(x)\le 1 \\ \mu(x)*w1+(1-\mu(x))*w2 & , 1<c(x)\le 2 \\ \mu(x)*w2+(1-\mu(x))*w1 & , 2<c(x)\le 3.5 \\ \mu(x)*w2+(1-\mu(x))*w3 & , 3.5<c(x)\le 5 \\ \mu(x)*w3+(1-\mu(x))*w2 & , 5<c(x)\le 6.5 \\ \mu(x)*w3+(1-\mu(x))*w5 & , 6.5<c(x)\le 8 \end{cases} \quad (5)$$

Using this metrics, the fuzzified values are applied correspondingly to these metrics and are defuzzified to get a single-valued integer .Thereby effort is estimated.

3. Proposed work

3.1. Over fitting

In over fitting, a statistical model describes noise or random error instead of the underlying relationship [8]. Over fitting occurs when a model is excessively complex, such as having much more parameters relative to the number of observations. A model that has been over fit has poor predictive performance, as it overreacts to minor fluctuations in the training data.

3.2. Tree pruning

Pruning is a technique in machine learning that reduces the size of decision trees by removing sections of the tree which provide little power to classify instances. Pruning reduces the complexity of the final classifier, and hence improves predictive accuracy by the reduction of over fitting.

3.3. Tree smoothing

It aims to connect the sharp discontinuities between adjacent linear models at the leaves caused during pruning.

3.4. Weka Tool

Data mining is software to cluster or to do regression analysis on the datasets. It is free open source software. Algorithms discussed in this paper are executed via this tool.

3.5. Dataset

Dataset is divided into training set and test set data [6]. A training set is a set of data that is used to discover potentially predictive relationships. A test set is a set of data used to assess the strength and utility of a predictive relationship. Test and training sets are used in genetic programming and statistics, machine learning, intelligent systems.

3.6. M5P Algorithm

The M5 algorithm builds a regression trees by splitting the dataset recursively through tests on a single variable that reduce variance on the dependent (target) variable.

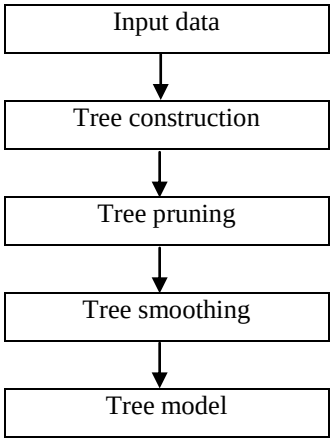

Fig. 4. Flow chart M5P Algorithm

3.7. Random Forest Algorithm

The Brieman's algorithm is popularly used to implement the RF technique [8]. To obtain an RF technique-based effort estimation model, the steps presented underneath are taken into consideration. These proposed steps help in constructing each tree, while using RF technique.

1) Let F be the number of trees in the forest and Dataset of D points (x1 , y1)(x2 , y2)....(xD , yD) is considered

2) Each tree of the forest should be grown as follows.

Steps from 3 to 9 should be repeated f number of times to create F number of trees

3) Let N be the no. of training cases, and M be the no. of variables in the classifier

4) To select training set for the tree ,a random sample of n cases, from the original data of all N accessible training cases is chosen. The rest is taken as test dataset.

5) A RF tree Tf is developed to the loaded data, by repeatedly rehashing the accompanying steps for every terminal node of the tree, till the minimum node size n min is arrived. Keeping in mind the end goal to make more randomness, distinctive dataset for each one trees is made.

6) The no. of input variables m is selected to ascertain the choice at a tree node. The value of m ought to be substantially short of what M.

7) For each tree node, m variables should be randomly chosen on which the decision at that node is based.

8) The best split focused around these m variables in the training set is calculated. The value of m ought to be held consistent throughout the development of the forest. Each tree should be fully grown and not pruned.

9) Then, the results of ensemble of trees T1,T2,...,Tf,....,TF are collected.

10) The input vector should be put down for each of the trees in the forest. In regression, it is the average of the individual tree predictions.

4. Various criterions for assessment of software effort estimation models

There are four important criterions for assessment of software effort estimation models.

1. VAF (Variance Accounted For) (%):

   $$\text{VAF (\%)} = \left(1 - \frac{Var(E-E)}{Var\ E}\right) * 100$$

2. Mean absolute Relative Error (%):

   $$\text{MAE (\%)} = \frac{\sum f}{\sum n} * 100$$

3. Variance Absolute Relative Error (%):

   $$\text{VAR (\%)} = \frac{\sum(\text{Re}-\text{mean})}{\sum f} * 100$$

4. Pred (n): Prediction at level n((Pred (n)):

   $$\text{VAR (\%)} = \frac{\sum(\text{Re}-\text{mean})}{\sum f} * 100$$

5. Result and Discussion

The performance of effort is predicted based on the MARE and Prediction analysis. The estimated effort of LOC is compared with the actual effort of LOC in the Fig. 5. The estimated effort of FP is compared with the actual effort of FP in the Fig. 6. The MARE of LOC and FP is compared in the Fig. 7. It has been clearly identified that Function point based estimation is better than the LOC estimation.

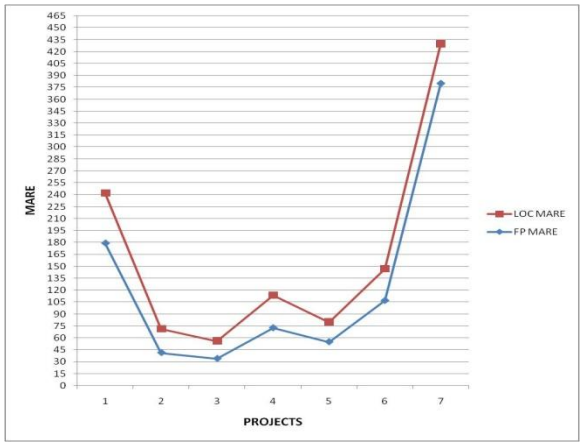

Fig.5. Variation between the actual and estimated effort using LOC

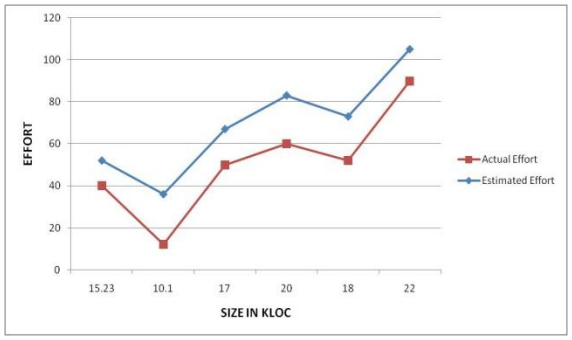

Fig.6. Variation between the actual and estimated effort using LOC in FP

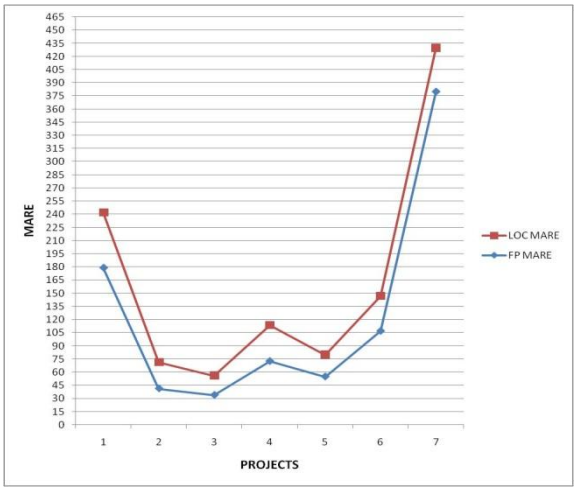

Fig.7. MARE analysis

The Tables 2 – 5 indicates the lines of code with the actual effort and the estimated effort using the COCOMO model. Both MARE and Prediction analysis has been applied to the direct and indirect approaches. The actual effort is the original effort and the estimated effort is the one which has been done in the estimation process using the COCOMO method.

Table 2. LOC based on algorithmic models with actual effort and the estimate effort

LOC	Actual effort	Estimated effort
48	1107.3	1465.83
50	84	145
39	72	112
164	246	510
200	130	625
40.5	82.5	160.7

Table 3. FPA with actual effort and the estimate effort

LOC in FP	Actual effort	Estimated effort
15.23	40	52
10.1	12	36
17	50	67
20	60	83
18	52	73
22	90	105

NASA Data set was applied for all algorithmic and non-algorithmic approaches, and results are shown below;

Table 4. Comparison table shows the error measures of RF-M5P, TAMF and TRMF

Project id	AEXP	MODP	TOOL	CPLX	KLOC	Actual Effort	TAMF	TRMF	RF and M5P Estimated Effort
1	0.91	1	1	0.85	25.9	117.6	128.3	125.6	123.9
2	0.91	1	1	0.85	24.6	117.6	125.9	123.7	120.3
3	0.91	1	1	0.85	7.7	31.2	36.5	38.5	38.1
4	0.91	1	1	0.85	8.2	36	40.56	44.6	41.7
5	0.91	1	1	0.85	9.7	25.2	32.7	31.3	29.8
6	0.91	1	1	0.85	2.2	8.4	14.3	12.5	11.6
7	0.91	1	1	0.85	3.5	10.8	15.2	16.4	15.1
8	0.91	1	1	0.85	66.6	340.5	359.4	354.2	351.7
9	0.91	0.75	0.75	0.85	7.5	72	79.6	75.6	77.3
10	0.91	1	1	0.85	20	72	82.5	80.1	79.6

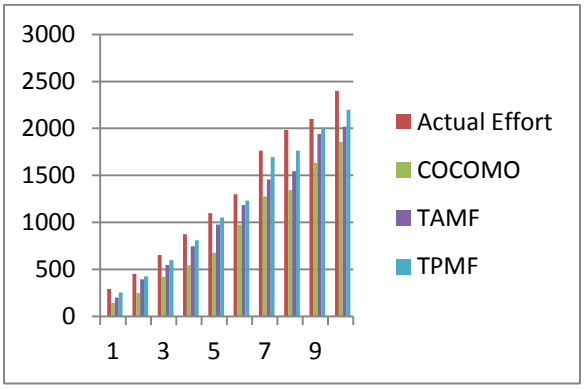

Fig. 8. Trapezoidal membership functions with relative error for NASA data set

The above graph shows the bar chart that represents comparative analysis of actual effort with that of the effort estimated using COCOMO, triangular and trapezoidal membership functions with relative error for NASA data set.

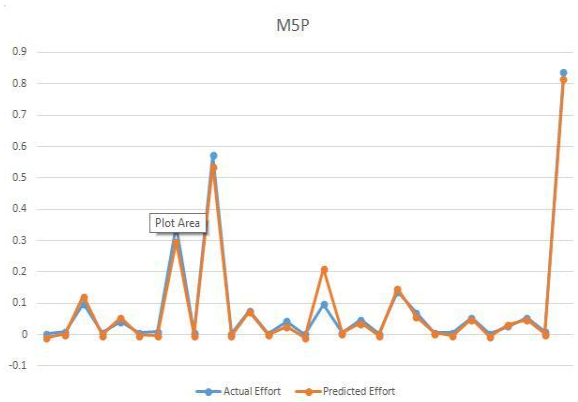

Fig. 9. M5P Actual Effort vs. Predicted Effort Chart

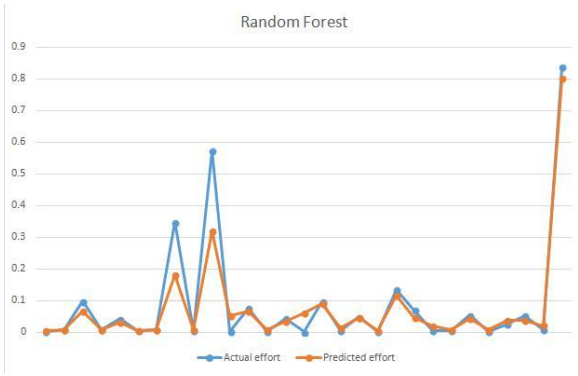

Fig. 10. Random Forest Actual Effort vs. Predicted Effort Chart

6. Conclusion

In this paper, we have proposed a new approach to estimate the software project effort. This approach is based on fuzzy logic. In fuzzy logic approach data is represented by fuzzy sets. In this investigation it is projected to characterize the size of the project using Triangular Membership Function which gives superior transition from one interval to another. A new fuzzy effort estimation model is proposed by using trapezoidal function to deal with the size and to generate fuzzy Membership Function and rules. After analyzing the results attained by applying COCOMO, triangular and trapezoidal Membership Function models, it is observed that the effort estimation of the proposed model gives more precise results than the other models. The effort estimated by means of fuzzifying size using M5P and RF Function yields better estimate which is closer to the actual effort..

Reference

1. M. Boraso, C. Montangero, and H. Sedehi, "Software cost estimation: An experimental study of model performances", tech. rep., 1996.
2. O. Benediktsson, D. Dalcher, K. Reed, and M. Woodman, "COCOMO based effort estimation for iterative and incremental software development", Software Quality Journal, vol. 11, pp. 265-281, 2003.
3. T. Menzies, D. Port, Z. Chen, J. Hihn, and S. Stukes, "Validation Methods for calibrating software effort models", ICSE '05:Proceedings of the 27th international conference on Software engineering, (New York, NY, USA),,pp.587-595, ACM Press, 2005.
4. Boehm, B., Abts, C., Brown, A. W., Chulani, S., Clark, B.K., Horowitz,E., Madachy, R., Reifer, D. J., Steece,B.Software cost estimation with COCOMO II. Prentice-Hall,Upper Saddle River, NJ, February 2000.
5. IFPUG. Function Point Counting Practices Manual: Release 4.0. International Function Point Users Group, Princeton Junction, NJ, 1994.
6. Alaa f. sheta," Estimation of the COCOMO Model Parameters Using Genetic Algorithm for NASA Software Projects", Journal of Computer Science ,2(2):118-123,2006
7. Ali Idri, alain Abran and Laila Kijri, "COCOMO cost modeling using Fuzzy Logic", International conference on Fuzzy Theory and technology At-lantic, 7New Jersy, March 2000.
8. Karel Dejaeger, Wouter Verbeke, David Martens, Bart Baesens, "Data Mining Techniques for Software Effort Estimation: A Comparative Study", IEEE Transactions on Software Engineering Vol 30 N0.2 March 2012.
9. Anish Mittal, Kamal Parkash, Harish Mittal,"Software Cost Estimation Using Fuzzy Logic", ACM SIGSOFT Software Engineering, Nov. 2010, Vol. 35, No.1.
10. Shashank Mouli Satapathy , Barada Prasanna Acharya, Santanu Kumar Rath : "Early stage software effort estimation using random forest technique based on use case points", research article , The Institute of Engineering and Technology Jounal.
11. Brandon Heung a, Hung Chak Hob, Jin Zhang a, Anders Knudbyc, Chuck E. Bulmer d, Margaret G. Schmidt : "An overview and comparison of machine-learning techniques for classification

purposes in digital soil mapping “, Geoderma, 265 (2016), pp. 62-77.

12. Cuauhtemoc Lopez-Martina, Alain Abran : “Neural networks for predicting the duration of new software projects “, The Journal of Systems and Software, 101 (2015) , pp.127–135.

13. Aditi Panda, Shashank Mouli Satapathy, Santanu Kumar Rath : “Empirical Validation of Neural Network Models for Agile Software Effort Estimation based on Story Points“, Procedia Computer Science, 57 (2015), pp. 772 – 781.

14. Wen Zhang , Ye Yang , Qing Wang : “Using Bayesian regression and EM algorithm with missing handling for software effort prediction”, Information and Software Technology, 58 (2015), pp. 58–70.

15. Mohammad Azzeha , Ali Bou Nassifb, “A hybrid model for estimating software project effort from the Use Case Points “,Applied Soft Computing (2016).

16. N.Shivakumar,V.Vignaraj Ananth,” Software cost estimation using Function Point with Non-algorithmic Approach”, Global Journal of Computer Science and Technology, Vol 13, No 8-C (2013): Global Journal of Computer Science and Technology.