GENERIC MODEL OF SOFTWARE COST ESTIMATION: A HYBRID APPROACH

Lalit V. Patil
Research Scholar,
Bharath University, Chennai
lalitvpatil@gmail.com

Rina M. Waghmode
M.E. I.T.
S.K.N.C.O.E., Pune
rinawaghmode18@gmail.com

S. D. Joshi
Prof. Dept of Comp.
BVDUCOE, Pune
sdj@live.in

V. Khanna
Dean, Dept. of I.T.
Bharath University, Chennai
khannakrishna@gmail.com

***Abstract*- Software companies ensure to complete the project within time and cost, for which good planning and thinking is required. Software project estimation is a form of problem solving which cannot be solved in a single piece of data by using some formulae. Decomposition of the problem helps in concentrating on smaller parts so that they are not missed. It aids in controlling and approximating the software risks which are commendably fixed and accurate. This paper represents an innovative idea which is the working of Principal Component Analysis (PCA) with Artificial Neural Network (ANN) by keeping the base of Constructive Cost Model II (COCOMO II) model. Feed forward ANN uses delta rule learning method to train the network. Training of ANN is based on PCA and COCOMO II sample dataset repository. PCA is a type of classification method which can filter multiple input values into a few certain values. It also helps in reducing the gap between actual and estimated effort. The test results from this hybrid model are compared with COCOMO II and ANN.**

Keywords—Hybrid; Principal Component Analysis (PCA); Artificial Neural Network (ANN); COCOMO II.

I. INTRODUCTION

When it comes to project management, cost estimation is considered as one of the challenging tasks. It helps in estimating resources, required time and cost for software development. Software cost estimation is a complex activity that requires knowledge of the number of parameters about the project for which the estimate is being constructed. The parameters include materials used for project development and a The parameters include materials used for project development and a deep knowledge of the market. The estimator relates the old project with the new project and estimates the cost deep knowledge of the market. The estimator relates the old project with the new project and estimates the cost. This cost is totally fluctuating as it does not denote the exact values. Software cost estimation is not single person oriented application. Any person with knowledge of software and experience can estimate the cost of the software.

The following parameters are considered in estimating the cost of software.

- People directly involved in implementation of the software.
- Project manager
- Developers
- Independent cost estimators team
- Experts in cost estimation

Software cost estimation methods are divided into two categories:

A. Algorithmic method

Algorithmic method, also known as the conventional method provides the mathematics and experimental equations to compute software cost. This is based on inputs and historical data, such as cost factors, scale factors, etc. Some of the popular algorithmic models are Source Line of Code (SLOC) [1], Function Point (FP) size estimation [2] and COCOMO II model [3].

B. Non-algorithmic method

A non-algorithmic model, was born in 1990's. It includes new approaches that are based on soft computing, such as analogy, expert judgment, neural networks and fuzzy logic. It is based on information of the previous projects because these methods perform the estimation by analysis of the chronological data and follow the human behavior [4].

The full paper is organized in sections which are listed as below: in Section III, after introductions and the related work problem statement determines the gap between actual and estimated costs. In section IV, we can briefly overview the proposed method which is based on algorithmic & non algorithmic methods. Section IV includes the potential significance of the proposed model. Section V includes results and discussion. Finally, conclusion and future scope explained in section VI and VII.

II. RELATED WORK

After more than fourty years of research, many software cost estimation methods are available, including algorithmic and non-algorithmic methods. In the initial stage, software cost estimation was done manually using simple thumb rules or estimating algorithms. The first automated software cost estimating tools were built in the early 1970's. In addition, software cost estimation tools allows the construction of customized estimating templates that are derived from actual projects. This can be used for estimating projects of similar sizes and kinds.

Software project failures have been a crucial subject in the last three decades. Software projects usually not fail during the implementation. Most of the project fails related to the planning and estimation steps. During the last decade, several studies have been done in terms of finding the reasons of the software project failures.

978-1-4799-2572-8/14/$31.00 ©2014 IEEE

In 2006, Galorath and Evans conducted an intensive search among 2100 internet sites. It revealed 5000 excuses for the software project failures. Among the observed reasons; insufficient requirements, lack of user involvement, premium support and resources, extraordinary expectations, poor planning, abrupt change in decisions in the early stages of the project and inaccurate estimations were the most prominent reasons.

Vahid et al. [1] Focused on all the existing methods for software cost estimation methods and comparing their features. It is useful for selecting the special method for each project. Around 1975, Allan Albrecht and his colleagues [2] in IBM White Plains have developed the original version of today's widespread function points metric. The FP measure is thought to be more useful than SLOC as a prediction of work effort because function points are relatively easily estimated from a statement of basic requirements for a program early in the development cycle. Boehm B. W [3] has referred to the current trends in software engineering economics. This project report provided an overview of economic analyses techniques. It surveyed the existing algorithmic and non algorithmis methods for software cost estimation. Attarzadeh et al. [4] used new fuzzy logic method for improving the accuracy of the software cost estimation model which presents better accuracy than other methods. Chiu et al. [5] has used analogy method that required one or more completed projects that are similar to the new project. According to the previous projects their cost and effort estimation is noticed, and the estimation of a new project is prepared. Jorgensen [6] has used expert-judgment method when there is the limitation in finding data and gathering requirements. Sikka et al. [11] has referred that Function Point Analysis (FPA) is used for determining the size of the software. It is also useful for estimating the efforts, duration and cost of the projects. Dr. Inz et al. [13] has presented the most commonly used and promising methods like PCA, fuzzy logic and neural networks along with cons of fuzzy logic. Anupama et al. [18] has intensified the accuracy of the COCOMO model by using ANN. Attarzadeh et al. [19] used a novel neural network constructive cost model for improving the accuracy of cost estimation.

III. PROBLEM STATEMENT

The existing methods require more effort, i.e. totals person and time due to which the cost of the software increases. The existing methods like SLOC COCOMO II, analogy, expert judgment, neural networks and fuzzy logic to show a substantial difference between the actual and estimated cost [5, 6,7,8].

By using the concept of PCA and ANN, the generic model of software cost estimation is designed to reduce the differences and to enhance the accuracy of COCOMO II model.

IV. PROPOSED METHOD

The proposed methodology is based on algorithmic & non-algorithmic methods such as a FP size estimate, COCOMO II & ANN. The combinations of all these methods help in estimating the cost of the software. Following is the objective of the proposed method:

1) The first objective is to choose accurate method for calculating the size as it plays a vital role in calculating cost and effort.
2) To improve the performance of existing methods we need to create a hybrid tool for software cost estimation that helps in a software development organization.

Refer Figure 1

Proposed system follows specific steps in which the flow is maintained. The detail of each stage is mentioned below.

A. Inputs

1) Size

Size is an essential part of software cost estimation. FP estimation is better than SLOC in predicting the size [9,10]. Number of inputs, number of outputs, the number of inquiries, number of logical files and the number of interfaces are used for computing the Unadjusted Function Point (UFP). It is evaluated by using these parameters and multiplying their count[11].

The following table represents the Technical Complexity Factor (TCF) regarding the size of the project.

TABLE I. TECHNICAL COMPLEXITY FACTORS

Fp_var1	Reliable backup	**Fp_var8**	Online updating of the master file
Fp_var2	Required data communications	**Fp_var9**	Inputs, outputs & DB complexity
Fp_var3	Distributed processing	**Fp_var10**	Internal processing complexity
Fp_var4	Critical performance	**Fp_var11**	Code reusability
Fp_var5	Operational environmental complexity	**Fp_var12**	Installation & configuration
Fp_var6	Online data entry	**Fp_var13**	Concurrent pipelined execution
Fp_var7	Multiple screens	**Fp_var14**	Ease of use

Calculate the TCF, FP and size using the following equations:

$$TCF = 0.65 + 0.01\left(\sum \mathrm{Fp}_{\_\mathrm{var}_i}(\mathrm{i} = 1 \text{ to } 14)\right) \quad (1)$$

$$FP = \mathrm{UFP} \times \mathrm{TCF} \quad (2)$$

$$SIZE(inKLOC) = (\mathrm{FP} \times \text{Select language})/1000 \quad (3)$$

2) Cost factors

Boehm introduced a set of seventeen cost drivers in the COCOMO II that adds accuracy to the COCOMO 81. The cost drivers are grouped in four categories [1]:

a) Product factors: It is formulated with factors of product such as Required Software Reliability (RELY), Data Base Size (DATA), Product Complexity (CPLX), Developed for Reusability (RUSE) and Documentation Match for Life-Cycle Needs (DOCU).

b) Computer factors: It depends on Execution Time Constraint (TIME), Main Storage Constraint (STOR), and Platform Volatility (PVOL).

c) Personnel factors: It depends upon the ability of the programmer/analyst in development by using experience & knowledge. It includes Analyst Capability (ACAP), Programmer Capability (PCAP), Personnel Continuity (PCON), Application Experience (AEXP), Platform Experience (PEXP), Language and tool experience (LTEX).

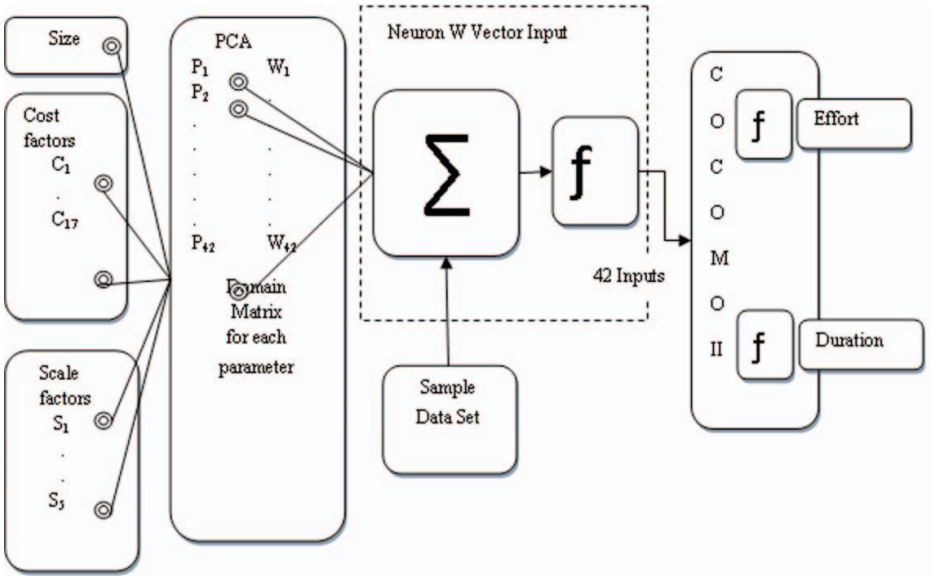

Figure 1: Proposed System Architecture Design for Software Cost Estimation

d) Project factors: It depends upon project features such as Use of Software Tools (TOOL), Multisite Development (SITE) and Required Development Schedule (SCED).

All the above cost factors depend upon the rating values corresponding to real numbers known as Effort Multipliers (EM). The rating value ranges are in the form of very low, low, nominal, high, very high, extra high.

3) Scale factors

COCOMO II depends on the five scale factors such as Precedentedness (PREC), Developing Flexibility (FLEX), Architecture / Risk Resolution (RESL), Team Cohesion (TEAM) and Process Maturity (PMAT) [1].

B. Classification using PCA

PCA was first introduced by Karl Pearson in 1901 [12,13,14, 15]. Taking into consideration the categories of software cost estimation methods a new concept of PCA is used which improves the accuracy, turnaround time and reduction in cost and effort.

Why the PCA is better than Fuzzy logic?

1) To find exact principal components rather than approximate like if the value is 0.99 consider it as 0.99 and not 1.00 in making it a round figure [16,17].
2) It identifies the principal directions in which the data varies with large variance.

Steps for calculating the number of principal components are given below:

1) Compute the correlation coefficient matrix using inputs.
2) Eigen value can be computed by using a correlation coefficient matrix.
3) By using an Eigen value number of principal components can be determined.

As discussed above input consists of many important factors like size, cost factors and scale factors. It should be pre-processed by using PCA. The ACCORD Dot NET framework is used to calculate the principal components. Methods and algorithms in machine learning, mathematics, statics, computer vision, computer audition and several other scientific computing techniques to the Dot NET environment is provided by the ACCORD Dot NET framework.

C. Artificial neural network

Why we choose ANN?

Artificial Neural Network is used in cost estimation due to its ability to learn from previous data.

The three factors that define ANN are as follows:

1) The interconnection pattern between different layers of neurons.
2) The learning process for updating the weights of the interconnections.
3) The activation function that converts a neuron's weighted input to its output activation.

The nodes in the network are divided into input layer; associated with weights which contain information about the input signals and output layer which are going through the network to some nodes in the hidden layer.

ANN Architecture:

Depending upon the architecture of the ANN it can be classified in two types namely; **feed-forward ANN** where the network has no loops and information moves in only one direction, i.e. forward from the input nodes, then through the hidden nodes to the output nodes. The second method is **feedback ANN** where loops occur in the network. An ANN can be a single-layer perceptron or a multi-layer

perceptron. Single-layer perceptron consists of a single layer of output nodes; the inputs neurons are connected directly to the output neurons via a series of weights, but in multi-layer perceptron an additional layer of neurons is present between input and output layers which is called as hidden layer. Number of hidden layers can be added in an ANN depending upon the problem domain and accuracy expected. This paper has used multiple layer **feed-forward ANN** for simulation [18,19,20].

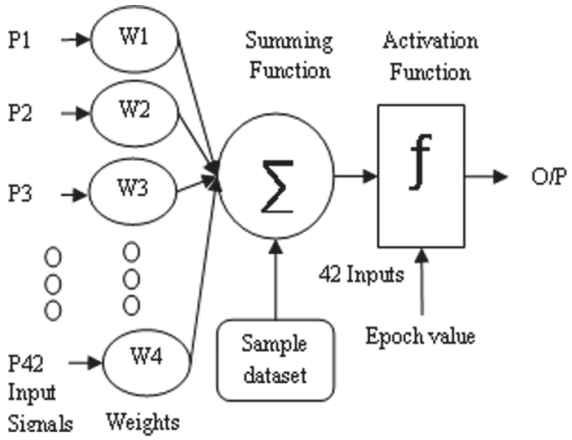

Figure 2: Architecture of feed-forward neural network using delta rule

A basic neural network consists of a number of inputs that are applied by some weights which are combined to give an output. Different neural network learning algorithms is used such as perceptron learning, delta rule learning, back propagation learning etc. We have used delta rule learning algorithm to train the neural network and solve the different problems. Delta rule learning algorithm uses sigmoid activation function where each neuron has a continuous activation function instead of a threshold activation function. Following step shows working of neural network:

Step 1: The input layer receives principal components as an input signal from PCA and sends it to the hidden layer.
Step 2: It includes data training.
Training algorithm includes the following steps:

1) Choose the training sample dataset and train it with principal components.
2) Determine the error in the hidden layer. There are fewer chances of error because PCA provides exact Eigenvalue.
3) If an error occurs ,then update the neural network weights.
4) Repeat until the neural network error is sufficiently small after an epoch is complete.

Step 3: Output layer sends size, effort multiplier and scale factor rating values to COCOMO II by using activation function.

D. Output using COCOMO II

The latest version of COCOMO is COCOMO II. Historical projects referred by COCOMO 81 dataset are 63, whereas COCOMO II dataset includes 161 historical projects [21]. The estimated effort in Person-Months (PM) for the COCOMO II is given as [22,23]:

$$Effort = A \times [SIZE]^{E} \times \prod_{i=1}^{17} EM_i \qquad (4)$$

$$WhereE = B + 0.01 \times \sum_{j=1}^{5} SF_j \; A = 2.94, B = 0.91$$

$$Time_{Months} = C \times (Effort)^{F} \qquad (5)$$

$$WhereF = D + 0.2 \times 0.01 \sum_{j=1}^{5} SF_j \; C = 3.67, D = 0.28$$

$$People = \frac{Effort}{Time} \qquad (6)$$

COCOMO II uses function point size estimation method for calculating the size of the software. It is composed of seventeen effort multipliers and five scale factors.

$$Cost = (Effort \times Payment_{Month}) \qquad (7)$$

V. ADVANTAGES OF THE PROPOSED MODEL

1) Identifying and increasing the poor productivity of existing methods.
2) Overcoming the issues of inaccurate code size.
3) To find the exact value rather than approximate and to provide sensitive result.
4) Improve accuracy over existing methods
5) To reduce the difference between actual and estimated costs.
6) To complete project within time, budget and resources.

VI. RESULTS AND DISCUSSION

Data on the five projects were gathered from an organization by distributing the questionnaire. According to the questionnaire, actual effort of the project was obtained [24]. From above equations, the effort of COCOMO II, ANN and Hybrid Model was derived which is as shown in TABLE II.

TABLE II. COMPARISON OF MODELS

Project	Actual Effort	COCOMO II	ANN	Hybrid
1	670	846.83	819.44	778.33
2	360	507.17	487.89	457.69
3	70	126.51	117.17	102.69
4	18	39.36	35.10	27.71
5	72	82.49	75.60	64.40

Figure 3 displays the varitey of projects which includes a variation of actual and estimated effort of COCOMO II, ANN and Hybrid model. In the Hybrid model, estimated effort is very close to the actual effort. Thus, Hybrid model helps in improving the accuracy of software cost estimation. Performance measure depends on the following equation [25,26,27]:

$$MRE = \frac{|ActualEffort - EstimatedEffort|}{ActualEffort} \qquad (8)$$

MRE (Magnitude of Relative Error)

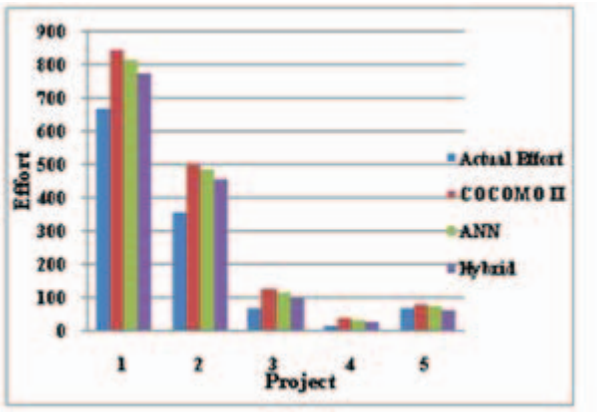

Figure 3: Comparison of Estimated Vs Actual Effort

MRE of COCOMO II, ANN and Hybrid model is calculated by using TABLE II and the equation (8) as shown in TABLE III.

TABLE III. MRE RESULTS

Project	MRE of COCOMO II	MRE of ANN	MRE of Hybrid
1	0.26	0.22	0.16
2	0.40	0.35	0.27
3	0.80	0.67	0.46
4	1.18	0.95	0.53
5	0.14	0.05	0.10

Hybrid model helps in minimizing the MRE which enhances the accuracy of COCOMO II and ANN model.

VII. CONCLUSION

Software resource estimation methods and models have a major impact on successful software engineering practice. They provide milestone budgets and schedules that help projects determine when they are making satisfactory progress and when they need corrective action. They help decision makers analyze software cost-schedule-value trade-offs and make decisions regarding investments, outsourcing and legacy software phase-outs. They help organizations prioritize investments in improving software productivity, quality and time to market. They are included as essential capabilities in virtually all major software capability maturity models, software engineering textbooks, and software engineering bodies of knowledge. There are large numbers of software available in the market to estimate the cost of the software. The implications of the results are based on the COCOMO II sample data set. The COCOMO II repository consists of more than 161 projects collected from different countries around the world. In the proposed model, PCA and ANN produces estimates that are more accurate than the ones provided by the same kind of algorithm without applying PCA and ANN. The important result is that the proposed technology increases the correctness of the estimates without worsening the variability, which improves the accuracy, turnaround time and performance of the system.

VIII. FUTURE SCOPE

Software development has become a crucial and important investment for many organizations. Now a day's Software engineering practitioners are becoming aware about accurately predicting the cost and quality of the software product under development. The Kernel Principal Component Analysis (KPCA) model exhibits a significantly different classification bias, a characteristic that makes it a valuable ensemble. The result confirms that accuracy is generally improved by the addition of the KPCA based model.

REFERENCES

[1] Vahid Khatibi, Dayang N.A.Jawawi, *"Software Cost Estimation Methods: A Review"*, Journal of Emerging Trends in Computing and Information Science, Vol.2, No.1, pp. 21-29, January 2011.

[2] Albrecht. A.J. and J. E. Gaffney, *"Software function, source lines of codes, and development effort prediction: a software science validation"*, IEEE Trans Software Engg., Vol.SE-9, No.6, pp.639-648, Nov.1983.

[3] Boehm B. W. *"Software Engineering Economics"*, IEEE Trans Software Engg., Vol.SE-10, No.1, pp.4-21, Jan.1984.

[4] Attarzadeh I. Siew Hock Ow, *"Improving the accuracy of software cost estimation model based on a new fuzzy logic model"*, World Applied science journal, 8(2), pp. 177-184, 2010.

[5] Chiu, N.H., Huang, S.J., *"The adjusted analogy-based software effort estimation based on similarity distances"*, Journal of Systems and Software, 80 (4), pp. 628–640, 2007.

[6] Jørgensen, M. *"Practical guidelines for expert-judgment-based software effort estimation"*, IEEE Software, Vol. 22(3), pp. 57-63, June 2005.

[7] Khaled Hamdan *"Practical Software Project Total Cost Estimation Methods"*, IEEE International Conference on Multimedia Computing and information Technology (MCIT), pp.5-8, IEEE March 2010.

[8] Y. F. Li, M. Xie, T. N. Goh. *"A Study of Genetic Algorithm for Project Selection for Analogy Based Software Cost Estimation"*. Industrial Engineering and Engineering Management IEEE international conference, pp.1256-1260, Dec.2007.

[9] Z. Zia, A. Rashid and K. uz Zaman, "*Software cost estimation for component-based fourth-generation-language software applications"*, Institution of Engineering and Technology (IET) Software, Vol. 5, Iss. 1, pp.103– 110, Feb 2011.

[10] Marcio Rodrigo Braz, Silvia Regina Vergilio, *" Software Effort Estimation Based on Use Cases"*, Computer Software and Applications Conference, Vol. 1, pp. 221-228, Sept. 2006.

[11] Sikka G., Kaur A., Uddin M., *"Estimating function points: Using machine learning and regression models"*, Education Technology and Computer (ICETC), Vol.3, pp. V3-52-V3-56, June 2010.

[12] Bo Cheng, Xuejun Yu *"The Selection of Agile Development's Effort Estimation Factors based on Principal Component Analysis"*, International Conference on Information and Computer Applications (ICICA), Vol.24, pp. 112-118, 2012.

[13] Dr. Inż. Małgorzata Syczewska, dr inż. Piotr Wąsiewicz. "*Contemporary techniques to manage of databases in gait analysis"*, Available: baztech.icm.edu.pl, 2009.

[14] N. Pessel, J-F. Balmat, J. Bonnal, *"Neuronal Principal Component Analysis for the Diagnosis of a Non Linear System"*, Control and Automation MED, pp.1-6, June 2007.

[15] Max Welling *"Kernel Principal Components Analysis"*, Advances in neural information processing systems 15, Vol.15, pp. 70-72, 2005.

[16] Justin Wong Danny Ho Luiz Fernando Capretz *"An Investigation of Using Neuro-Fuzzy with Software Size Estimation"*. IEEE ICSE'09 Workshop, pp.51-58, May 2009.

[17] Xishi Huang, Luiz F. Capretz, Jing Ren *"A Neuro-Fuzzy Model for Software Cost Estimation"*, Third International Conference on Quality Software (QSIC'03), pp. 126-133, Nov. 2003.

[18] Anupama Kaushik, Ashish Chauhan, Deepak Mittal, Sachin Gupta, **"***COCOMO Estimates Using Neural Networks",* I.J. Intelligent Systems and Applications (IJISA), Vol.4, No.9, pp. 22-28, August 2012.
[19] Attarzadeh I. Siew Hock Ow, *"Proposing a New Software Cost Estimation Model Based on Artificial Neural Networks"*, IEEE International Conference on Computer Engineering and Technology (ICCET), Vol.3, pp.V3-487 - V3-491, April 2010.
[20] Salvatore A. Sarcia', Giovanni Cantone and Victor R. Basili, *"*Adopting Curvilinear Component Analysis to Improve Software Cost Estimation Accuracy Model, Application Strategy, and an Experimental Verification*",* EASE08, pp.1-10, 2008.
[21] Yongchang Ren, Xiaoji Chen, Tao Xing, Xuguang Chai, *"Research on Software Maintenance Cost of Influence Factor Analysis and Estimation Method",* Intelligent System and Applications (ISA), pp.1-4, May 2011.
[22] Maged A. Yahya, Rodina Ahmad, Sai Peck Lee, *"Effects of Software Process Maturity on COCOMO II's Effort Estimation from CMMI Perspective",* Research, Innovation and Vision for the Future (RIVF) IEEE International Conference, pp. 255-262, July 2008.
[23] Musilek A. *"On the Sensitivity of COCOMO II Software Cost Estimation Model"*, Eighth IEEE Symposium on Software Metrics (METRICS.02), pp. 13-20, 2002.
[24] Kemerer C. *"An empirical validation of software cost estimation models"*, Communications of the ACM, Vol.30, No.5, pp. 416-429, May 1987.
[25] Jacky Wai Keung, Barbara A. Kitchenham and David Ross Jeffery *"Analogy-X: Providing Statistical Inference to Analogy-Based Software Cost Estimation",* IEEE Trans Software Engg., Vol.34, No.4, pp.471 - 484, July/Aug. 2008.
[26] Shepperd M. *"Estimating Software Project Effort Using Analogies"* IEEE Transactions On Software Engineering, Vol. 23, No. 12, pp. 736-743, Nov. 1997.
[27] K. Srinivasan and D. Fisher, *"Machine Learning Approaches to Estimating Software Development Effort"*, IEEE Transactions on Software Engineering, Vol.21, No.2, pp. 126-137, Feb.1995.