An Adaptive Approach for Software Cost Estimation using Hybrid Model

Swati waghmode[1], Dr.kishor Kolhe[2]

PG Student [IT] , Dept. of Information Technology, MIT College of Engineering,Pune India[1]
Associate Professor, Dept. of Information Technology, MIT College of Engineering, Pune, India[2]

***Abstract*- Technology has become an indispensable part of everyone's life in today's competitive and globally connected world. One of the key and most important arm of continuously evolving technological framework is evolvement and innovations in areas of different software technologies, tools and packages that are either available or being made available on continual basis. While software development and configuration in one of the major contributors to the global economy, it's also remains one of the most complex and challenging business to be in as it doesn't always offer you tangible work-products to plan and estimate for Among different phases of software development process, software cost estimation and planning is the most important phase and when executed effectively and accurately, help us avoid costly delays and/or project scraps that are prevalent in this industry. There isn't a single formula based method in market that can guarantee you accurate estimation and it typically involves decomposition of larger project in smaller, manageable chunks and then estimating these individual pieces in relation to larger work product using facts and data from our previous experiences from similar implementations. In this model, we have proposed a variable reduction technique based on auto-associative clustering, and multilayer feed-forward the neural networks. The Kernel component analysis is log-linear regression functions calibrated with large data set with ordinary least squares. Based on the COCOMOII data set, we have showed that Kernel component analysis can improve the estimation model accuracy by shrinking the input variables into an equivalent pattern and removing irrelevant variable. We have showed that the models obtained by applying Kernel component analysis are more persistent, correct and reliable.**

Keywords- Software cost estimation, KPCA, and COCOMOII.

I. INTRODUCTION

The process of calculating the program, hard work, effort, size on the software solution, and entire cost regarding developing the program application is called software cost estimation. A cost estimation completed in the beginning of project can assist decide which usually functions might be involved inside of learning resource difficulties from the project. The risk of task is will increase when the most crucial functions are involved towards the end of your project. Hence, cost estimation could have a large effect on the lifestyle cycle along with timetable for only a task. Which is just about the most challenging work inside of managing along with preserving software package, during your improvement approach cost along with time evaluation performs a crucial part inside of software cost evaluation approach. Cost evaluation to get a usual software project begins through first scoping along with planning cycle on the project. Such a earlier estimation linked to entire charge and implementation is really important because this specific estimation (both cost along with time) is needed because feedback intended for primary natural affirmation along with checking on the tasks general improvement along with health verify. After completion of all of the work incorporated, these types of estimations utilized intended for project output review. Software cost estimation methods are divided in two categories:

A. Algorithmic Method:

Algorithmic models generally known as parametric models. These methods begin using a formula to calculate the charge estimation. The system is developed from models that is created by combining related price tag factors. In this method costs usually are analyzed using mathematical formulae inputs with metrics to produce an estimated output. This method uses the mathematical equations to achieve the application estimation. The specific equations use historical info or maybe theory.

B. Non-Algorithmic Method:

Within this method estimation process is done using the analysis of previous datasets. Non-algorithmic methods will not use any formula to calculate the application cost estimation. This method tends to make comparison between previous dataset and existence dataset. After considering the families of software cost estimation, we've proposed a novel idea of "Kernel Principle Component Analysis" (KPCA) that improves reliability and accuracy without having applying the exhaustive procedures. This paper is organized in sections as listed below: Section I & II provides introduction and relevant work. Section III provides a brief overview of proposed method that is based on algorithmic and non-algorithmic methods. A brief introduction to the concept of KPCA that helps in increasing accuracy of software cost estimation without application of any exhaustive procedures is discussed in the same section. Section IV provides a brief overview of significance of proposed method. Section V includes analytical results of COCOMOII model. Conclusion and future scope is explained in sections VI & VII

II. RELEVANT WORK

Software cost as well as effort appraisal plays an important role inside software task management. Research of all the available literature shows there are several computer software cost appraisal methods readily available including

gentle computing approaches. Following section presents the review of the work done:

"Author states Hybrid approach for improving accuracy of the software cost estimation so that estimated result is very close to the actual result using COCOMOII, Neural network, PCA technique. 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". L.V.Patil, R.M.Waghmode.et.al IEEE [1] "In this paper, a new regression model is created for software effort estimation based on use case point model. Furthermore, a Sugeno Fuzzy Inference System (FIS) approach is applied on this model to improve the estimation." Ali Bou Nassif, et al IEEE[2] "This paper proposed an effective fuzzy logic model for improving the COCOMO II to overcome the uncertainty of software attributes which resulted in producing more accurate estimation results. The aim of this research is implement the adaptive fuzzy logic model to improve the accuracy of software time and cost estimation. Iman Attarzadeh et.al. IEEE [3] "Author states non algorithmic method that is analogy based used for improving the effort estimation, in this method estimation is done by using historical data analogy-based effort estimation, i.e., the immediate neighbors of a project offer stable conclusions about that project". Ekrem Kocaguneli, et. al IEEE[4] "In this paper two methods with different technique used. In direct method size is measured in lines of code (LOC). In indirect method, size is represented as Function Points (FP).This technique improving the accuracy of software cost estimation model". Dr.N.Balaji et.al. [5]"This paper provides overview of existing software cost estimation models and techniques. Cost estimation models are basically of two types: algorithmic and non-algorithmic. This paper presents the pros and cons of algorithmic and non-algorithmic method". Sweta Kumari et al. [6]"In this paper, the author explores the use of Perceptron learning rule to implement COCOMO II for effort estimation, so that the estimated effort is more close to the actual effort. In this paper technique used COCOMOII, Neural network, Perceptron learning algo". Ridhika Sharma et al. [7]"This paper uses new fuzzy logic method for improving the accuracy of software cost estimation model. The result of this model are compared with COCOMOII model.In this paper Technique used-Fuzzy Logic Model, COCOMOII Model".Zia Uddin, et al. [8]. "In this paper the proposed neural networks model showed better software effort estimates as compared to traditional COCOMO." – Anupam kaushik, et al. [9]. "In this paper several existing methods for software cost estimation are represented and all existing methods for software cost estimation and comparing their features. It is useful for selecting the Special method for each project Estimation technique used: SLOC,Function point size estimates COCOMO,Analogy,Neural network."-Vahid, et al.[10]"This paper uses a new hybrid toolbox which is based on soft technique. Toolbox presenting a vital role it provide an efficient, flexible and user-friendly way of performing the effort estimation task".- Ch.VM K et al. [11]."This paper introduces novel model using fuzzy logic to estimate the effort required in software development. This model improving the accuracy of software cost estimation model which present the better accuracy as compared to other methods."-J. N. V. R. Swarupkumar, et al. [12]."It provides an overview of economic analysis techniques and their applicability to software engineering and management. It reviews the field of software cost estimation, including the major estimation techniques available, the state of the art in algorithmic cost models, and the outstanding research issues in software cost estimation."-Boehm B. W [19].

III. PROPOSED METHOD

Technology has become significant regions of organization improvement. A lot of the businesses rely upon technology such as computer systems & software. But also in another side organization likewise ponders the particular expenditure to be built on the software. A number of the businesses invest in brand new software whilst some of them acquire brand new software. In all of the these types of circumstances expenditure of your energy & money takes on a significant role so that it becomes necessary to help appraisal cost regarding software to be employed and also time period used for improvement.

The recommended methodology is based on Algorithmic & Non-algorithmic methods for instance Function position size estimation, COCOMO &Artificial Sensory network. The combination of all these kind of methods assists in estimating cost in the software

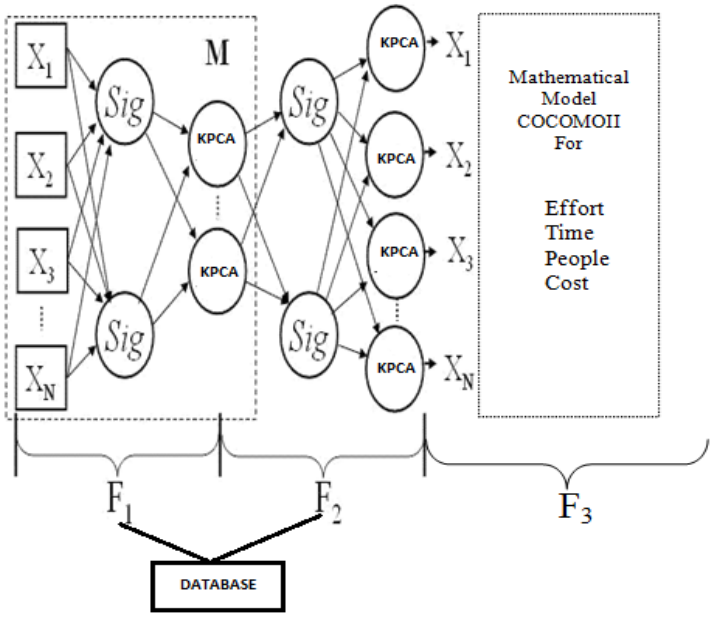

Figure 1: Proposed System Architecture Design for Software Cost Estimation

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

A. INPUT ACTUAL DATASET COLLECTED AS PER PROJECT SPECIFICATION

i) Size

It is essential part for software cost estimation. For predicting the size of the project function point estimation is better than SLOC. This method includes number of inputs, number of outputs, number of inquiries, number of logical files, number of interfaces by using this parameters we can find the projects complexity like simple, average, complex [6].

ii) Cost factors

Boehm introduced a set of 17 cost drivers in the Intermediate COCOMO that adds accuracy to the Basic COCOMO. The cost drivers are grouped in four categories:-

Product factors: It is formulated with factors of product such as product complexity, reliability etc.

Computer factors: It depends on Execution time constraint, main storage constraint etc.

Personnel factors: It depends upon the ability of the programmer/analyst in development by using experience & knowledge.

Project factors: It depends upon project features such as milestone, deliverables etc.

Above cost factors depends upon the rating values corresponding to real number known as effort multipliers (EM). Rating values having six levels: Very low, Low, Nominal, High, Very high, Extra high [17].

iii) 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) [6].

CLASSIFICATION USING KPCA

Why KPCA is better than PCA

1. PCA does not supporting mean for multilayer neural network.
2. Large dataset like 0.000009 assume as non-linear value PCA does not take a nonlinear space value. KPCA allows us to identify the kernel principal directions in which the data varies with large variance
3. PCA support explicit mapping that's why PCA work with single layer neural network. In practice, a huge data set leads to a huge K, and storing K may become a problem. One way to deal with this is to perform clustering on your huge dataset, and populate the kernel with the means of those clusters.
4. KPCA support implicit mapping that's why KPCA work with multilayer neural network.

Steps for calculating number of kernel principal component are given below [11].

To understand the utility of KPCA, particularly for clustering, observe that, while *N* points cannot in general be linearly separated in $d < N$dimensions, they can almost always be linearly separated in $d \geq N$dimensions. That is, given *N* points, $\mathbf{x}_i$, if we map them to an *N*-dimensional space with

$$\Phi(\mathbf{x}_i) \text{where} \Phi : \mathbb{R}^d \to \mathbb{R}^N.$$

in kernel PCA, a non-trivial, arbitrary Φfunction is 'chosen' that is never computed explicitly, allowing the possibility to use very high dimensionalΦ's if we never have to actually evaluate the data in that space. Since we generally try to avoid working in the Φ-space, which we will call the 'feature space', we can create the N-by-N kernel

$$K = k(\mathbf{x}, \mathbf{y}) = (\Phi(\mathbf{x}), \Phi(\mathbf{y})) = \Phi(\mathbf{x})^T \Phi(\mathbf{y})$$

We note that $\Phi(\mathbf{x_i})^T \Phi(\mathbf{x})$denotes dot product, which is simply the elements of the kernel$K$. It seems all that's left is to calculate and normalize the$\mathbf{a_i}^k$, which can be done by solving the eigenvector equation

$$N\lambda \mathbf{a} = K\mathbf{a}$$

Where N is the number of data points in the set λ and$\mathbf{a}$ is the eigenvalues and eigenvectors of K. Then to normalize the eigenvectors$\mathbf{a}^k$'s, we require that

$$1 = (\mathbf{a}^k)^T \mathbf{a}^k$$

Care must be taken regarding the fact that, whether or not x has zero-mean in its original space, it is not guaranteed to be centered in the feature space (which we never compute explicitly). Since centered data is required to perform an effective kernel principal component analysis, we centralize K to becomeK'

$$K' = K - \mathbf{1_N}K - K\mathbf{1_N} + \mathbf{1_N}K\mathbf{1_N}$$

Where$\mathbf{1_N}$ denotes an N-by-N matrix for which each element takes value$1/N$. We use K' to perform the KPCA algorithm

More important data in the project estimation is collected in the sample data set which consists of many important factors for effort estimating such as software size, effort, productivity, development progress of project, project attributes, platform attributes, scale attribute, architecture etc. Then quantify the data before processing the data. The sample data set should be preprocessed, because some data might get missed, in the mean time for calculating the eigenvalue we should compute correlation coefficient matrix by using KPCA and input values and finally we

should determine the number of kernel principal components based on size, cost factors and scale factors.

B. ARTIFICIAL NEURAL NETWORK

Artificial Neural Network is used in cost estimation due to its ability to learn from existing dataset.
A basic neural network includes a number of inputs that are applied by some weights which are combined together to give an output. The steps used for cost estimation by using Delta feed-forward multi-layer ANN are summarized as follows [1]:

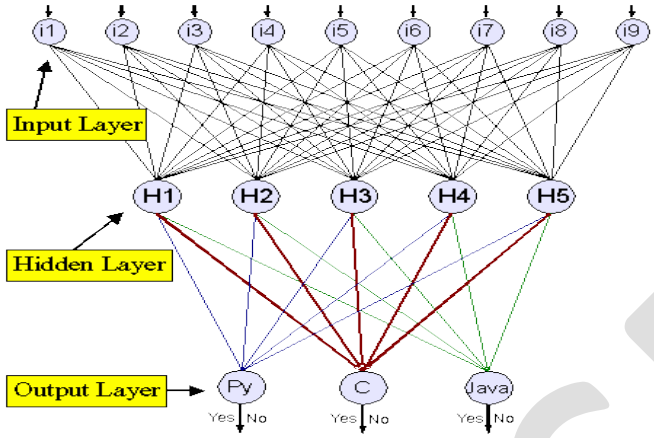

Figure 2: Architecture of Delta feed-forward neural network

Step1: The input layer receives input signal i.e. principal components and sends it to the hidden layer.
Step 2: It includes data training.
Training algorithm includes following steps:

a. Choose the training sample i.e. kernel principal components and train it with sample dataset in matrix form.
b. Determine error in hidden layer but there is less chances of error because KPCA provides exact eigenvalue.
c. If error occurs then update the neural network weights.
d. Repeat until the neural networks error is sufficiently small after an epoch is complete.

Step 3: Output layer sends size, effort multiplier and scale factor rating values to COCOMOII by using activation function as shown in figure 2.

C. COCOMO II

COCOMOII is the latest version of COCOMO.COCOMO dataset includes 63 historical projects and COCOMOII dataset includes 161 historical projects. The estimated effort in person-months (PM) for the COCOMOII is given as:

$$Effort = A \times [SIZE]^{E} \times \prod_{i=1}^{17} EM_{i\,[1]}$$

$$Where\ E = B + 0.01 \times \sum_{j=1}^{5} SF_j \qquad A = 2.94,\ B = 0.91$$

$$Time = C \times (Effort)^{F} \quad Where$$

$$F = D + 0.2 \times 0.01 \times \sum_{j=1}^{5} SF_j \quad C = 3.67,\ D = 0.28\ [2]$$

$$People = Effort/Time\ [3]$$

COCOMOII uses function point size estimate for calculating the size of the software and composes of 17 Effort multipliers and 5 scale factors (SF) [4]

*Cost = Effort * Average salary per unit time + Other expenses [4]*

IV. RESULT AND DISCUSSION

Hybrid Model	PCA	Neural	COCOMOII	Category
14.77428	**14.77428**	**14.77428**	**14.77428**	ELOC
101.7767615	102.6995764	117.1765697	126.5100577	EFFORT
12.13184087	12.24184087	13.33184087	14.23184087	TIME
7.889226543	8.389226543	8.789226543	8.889226543	PEOPLE
26461.95799	26701.88986	30465.90812	32892.61499	COST

Table 1.Comparison of Existing model vs. Hybrid model using effort, time, people, cost

.Table 1 shows the Comparison of existing model vs. Hybrid model using effort, time, people, and cost which improves the accuracy of software cost estimation using kernel principal component analysis(KPCA).

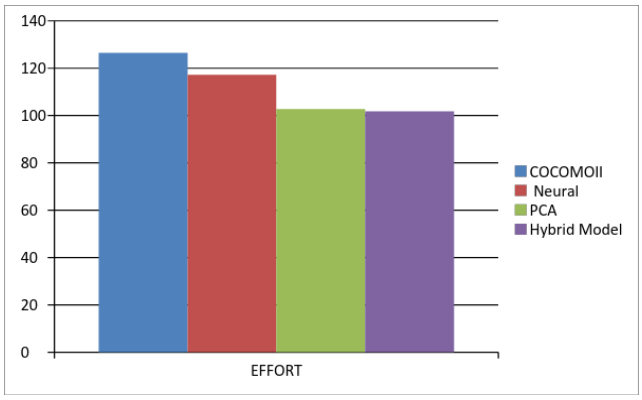

Fig.3 comparison of existing model effort vs. Hybrid model Effort

Figure 3 above shows comparative analysis of all four models w.r.t. effort. On X-axis, we have plotted comparative analysis all of four models in consideration w.r.t. effort values for each model. On the other hand i. e. on the Y-axis, we have plotted the value of effort for all four models (COCOMOII, Neural Network, PCA, and KPCA) which ranges from 0 to 140. The blue color shows the result of COCOMOII algorithm while purple color shows results of our proposed hybrid algorithm.

POTENTIAL SIGNIFICANCE OF THE PROPOSED MODEL

a) Support large domain space.

b) Learning using standard database: It uses standard rating values stored in sample data set which is provided by International Software Benchmarking Standard Group.

c) Very profound information is easily available.

V. CONCLUSION

By using proposed model the accuracy of cost estimation will be improved.Estimated cost can be very close to the actual cost.COCOMOII model used for calculating the effort,duration cost of the software.Neural network used for cost estimation due to its ability to learn from existing dataset.If KPCA is applied then it's estimates is more accurate than COCOMOII and Neural network.KPCA has the advantage that reconstructs the PCA using kernel for assuming very small or large nonlinear value to construct the cluster for reduction for better feature extraction rate. If KPCA is applied then its estimates is more accurate than existing models. KPCA calculate around 70% cost estimation compared to COCOMO model. KPCA calculate around 50% cost estimation compared to Neural Network. KPCA calculate around 15% cost estimation compared to PCA

VI. FUTURE SCOPE

Presently software program design experts are becoming aware about effectively forecasting the cost and excellent with the application. Software program development has turned into a essential and important investment decision for many organizations. We can do comparison based study on all SVM model as ICA, PCA, KPCA, IKPCA.

REFERENCE

[1] Lalit Patil, RinaWaghmode, S.D.Joshi,.*"Generic Model of Software Cost Estimation : A Hybrid Approach", IEEE (IACC),*978-1-4799-2572-8/14, *2014*

[2] Ekrem Kocaguneli, Tim Menzies, Ayse Basar Bener, and Jacky W. Keung," *Exploiting the Essential Assumptions of Analogy-Based Effort Estimation*" IEEE Transactions on Software Engineering, VOL. 38 NO. 2, P 425-438, 2012

[3] Iman Attarzadeh, Siew Hock Ow,"*Improving Estimation Accuracy of the COCOMO II Using an Adaptive Fuzzy Logic Model",* IEEE, Taipei, Taiwan, PP-2458-2464, June 27, 2011

[4] Ali Bou Nassif and Luiz Fernando Capretz, Danny Ho" *Estimating Software Effort Based on Use Case Point Model Using Sugeno Fuzzy Inference System"* IEEE,PP 393-398,2011

[5] Dr.N.Balaji,Year2013,"*SoftwareCostEstimation using Function Point with Non Algorithmic Approach",*Global Journal of Computer Science and Technology Software & Data Engineering (USA),Volume 13 Issue 8 Version 1.0,Online ISSN: 0975-4172 & Print ISSN: 0975-4350

[6] SwetaKumari,July2013"*Performance Analysis of the Software Cost Estimation Methods:Review",* IJARCSSE,,Volume3,Issue7,ISSN:2277 128X

[7] Radhika Sharma, June 2013, "*COCOMO II Implementation Using Perceptron Learning Rule",* International Journal of Engineering Research & Technology (IJERT) ISSN: 2278-0181 Vol. 2 Issue 6.

[8] Ziauddin, March, 2013, "*A Fuzzy Logic Based Software cost Estimation Models*",International Journal of Software Engineering. Vol. 7, No. 2,

[9] Anupam Kaushik,August2011, *"COCOMO Estimates Using Neural Networks"*I.J. Intelligent Systems and Applications, 2012, 9, 22-28 Published Online in MECS.

[10] Vahid Khatibi, 2011,*"Software Cost Estimation Methods: A Review",* Journal of Emerging Trends in Computing and Information Sciences, Malaysia, Volume 2 No. 1 ISSN 2079-8407, 2011

[11] Hari, October 2011 "*SEEPC: A Toolbox for Software Effort Estimation using Soft Computing Techniques "*International Journal of Computer Applications (0975 – 8887) Volume 31

[12] JNVR Swarupkumar,Dec 2011,"*Fuzzy logic for Software Effort Estimation Using Polynomial Regression as Firing Interval"*,International journal of software engineering and its application

[13] Ch.Satyananda Reddy,January2010,"An Neural Network Model for Software Effort Estimation" Int.J.of Software Engineering, IJSE Vol.3 No.1

[14] Ch. Satyananda Reddy," May 2009, "*A Concise Neural Network Model for Estimating Software Effort*"," International Journal of Recent Trends in Engineering, Issue. 1, Vol. 1,

[15] Sikka.Kaur, 2010 "Estimating function points: Using machine learning and regression models",(ICETC), 2nd International Conference.

[16] Iman Attarzadeh 2012" *Proposing an Enhanced Artificial Neural Network Prediction Model to Improve Accuracy in Software Effort Estimation*" Fourth International Conference on Computational Intelligence,

[17] Max Welling,2003,"*Kernel Principal Components Analysis"*,Department of Computer Science, University of Toronto, 10 King's College Road,Toronto, M5S 3G5 Canada

[18] Z.Zia,2011"*Software cost estimation for component based fourth-generation- language software applications"*, I T Software, vol. 5, no. 1, (2011), pp. 103-110.

[19] BoehmB.W.,1981*"SoftwareEngineering Economics",* Englewood Cliffs, NJ,Prentice-Hall, 1981.