International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Review on Cost Estimation Prediction Using ANN

Anshul[1], Nitin Jain[2]

[1]M. Tech (CSE), Hindu College of Engineering, Sonipat, Haryana, India.

[2]Assistant Professor (CSE), Hindu College of Engineering, Sonipat, Haryana

Abstract: ***Cost prediction is the process of estimating the effort required to develop a software system. Cost prediction is an important part of software development. Now days, software is the most expensive component of computer system. This paper provides a general overview of software cost estimation methods including the recent advances in the respective field. Algorithmic models such as SLOC, function point size estimation and COCOM model .Non-Algorithmic models such as analogy and expert judgment. Machine learning methods such as artificial neural networks and their techniques and fuzzy logic. This paper highlights the cost estimation models that have been proposed and used successfully and their comparison and applications.***

Keywords: Algorithmic models, Cost Estimation Prediction, SLOC, Non-Algorithmic models, Machine learning models .

1. Introduction

Cost prediction is the process of estimating the effort required to develop a software system. Several indicators should be considered to estimate the software cost and effort. One of the most important indicators which should be noticed is the size of the project. The estimation of effort and cost depends on the accurate prediction of the size. Generally, the effort and cost estimations are difficult in the software projects because software projects are often not unique and there is no previous experience about them. Therefore, prediction becomes complicated. Accurate cost prediction is important because:

- It can help to prioritize and classify the development projects with respect to an overall business plan.
- It can be used to determine the kind of resources required and how well these resources will be used.
- It can be used to determine the impact of changes and support re-planning of project.
- Projects can be simple to manage and control when resources matches to the real needs.
- Customers expects the projected costs in the same line with the actual development costs.

Software cost prediction involves determination of effort, time and cost. Effort is measured in person- months of the programmers, analysts etc..This effort estimate can be converted into cost by multiplying the estimated effort required by the average salary per unit time of the staff involved. Accurate estimation of software development effort has major implications for the management of software growth. If management's estimate is very less, then the software development team will be under considerable pressure to complete the product on time, and hence the resulting software may not be tested and fully functional . Thus, the product may contain the remaining errors that need to be corrected during a later part of the software life cycle, in which the cost of corrective maintenance is more. On the other hand, if a manager's estimate is greater, then too many resources will be allowed to the project. ANN is one of the techniques which involve a series of steps for the computation of maintainability effort. As the traditional computers are not excellent to interact because of the immense parallelism, noised data, failure to adapt to certain circumstance, and fault tolerant, so ANN provides a better option for handling software quality.

2. Estimation Technique

There are many techniques or methods for the software cost estimation prediction. Cost estimation is the process of estimating the effort required to develop a software system. Cost prediction is an important part of software development. Cost estimation techniques are classified into various categories, but the information industry wants a simple and accurate estimation method for their work.

Cost Estimation Techniques three main categories:
1. Algorithmic,
2. Non-algorithmic
3. Machine Learning Methods.

1. Algorithmic Model
Algorithmic models (AM) "calibrate" prespecified formulas for estimating development effort from historical data. Mainly algorithmic models are statically or formula based models which takes historical cost information and which is based on the size of the software.

Some methods are:
a) Source Line of Code
SLOC is an estimation parameter that illustrates the number of all commands and data definition but it does not include instructions such as comments, blanks, and continuation lines. This parameter is usually used as an analogy based on an approach for the estimation.

After computing the SLOC for software, it is compared with other projects whose SLOC has been computed before, and the project size is estimated. SLOC is used to easily measure the size of the project. After finishing the project, all estimations made are compared with the actual ones.

Thousand Lines of Code (KSLOC) are used for estimation in large scale. Using this metric is common in many estimation methods. SLOC Measuring seems very difficult at the early stages of the project because of the lack of information about requirements.

Since SLOC is computed based on the instructions of languages, and comparing the size of software which uses different language is very difficult. But, SLOC is the basis of the estimation models in many complicated software estimation methods. SLOC is governed by the equation 1.1

$$S = \frac{{}^{S}L + {}^{4S}M + {}^{S}H}{6} \quad (1.1)$$

Where,

S_L as the lowest, S_H as the highest and S_M as the most probable size

b) Function Point Size Estimates

At first, Albrecht (1983) presented Function Point metric to measure the functionality of project. He believes function points offer several significant advantages over SLOC counts of size measurement.

In this method estimation is done by following indicators:

- User Inputs
- User Outputs
- Logic files
- Inquiries
- Interfaces

A Complexity Degree which is between 1 and 3 is defined for each above indicator. 1, 2 and 3 represent simple, medium and complex degree respectively. Also, it is essential to define a weight for each indicator which can be between 3 and 15.

Steps in counting function points:

i. Counting the user functions. The raw function counts are arrived at by considering a linear combination of five basic software components: external inputs, external outputs, external inquiries, internal logic files, and external interfaces[2], each at one of three complexity stages: simple, average or complex .The number of function counts (FC) is the sum of these numbers which is weighted according to the complexity level .

ii. Adjusting for environmental processing complexity. The final function points is calculated by multiplying FC by an adjustment factor that is determined by considering 14 aspects of processing complexity. This adjustment factor allows the FC to be modified by at most 35% or -35%.

c. COCOMO Model

Constructive Cost Model (COCOMO) is widely used algorithmic software cost model [3] [4]. It include following three models:

Model 1 (Basic COCOMO Model)*:-* The basic COCOMO model computes software development effort and cost as a function of program size expressed in estimated lines of code (LOC) [5] [6].

The steps in this Model are:-

a. Obtain an initial estimate of the development effort from the estimate of thousands of delivered lines of source code (KLOC).

b. Determine a set of 15 multiple factors from different attributes of the project.

c. Adjust the effort estimate by multiplying the initial estimate with all the multiplying factors.

The initial estimate (also called nominal estimate) is determined by an equation of the form used in the static single variable models, where size is calculate in KLOC. The initial effort is calculate in person-month using the equation [7] of the type

$$EFFORT = a^* (KLOC)^b \quad (2)$$

Where, the value of constants a and b depends on the project type.

Model 2 (Intermediate COCOMO Model):- Intermediate COCOMO Model computes software development effort as a function of program size and set of —cost drivers that include subjective assessment of the products, hardware, personnel and project attributes.

Product attributes

a. Required software reliability
b. Size of application data base
c. Complexity of the product

Hardware attributes

a. Run-time performance constraints
b. Memory constraints
c. Volatility of the virtual machine environment
d. Required turnaround time

Personnel attributes

a. Analyst capability
b. Software engineer capability
c. Applications experience
d. Virtual machine experience
e. Programming language experience

Project attributes

a. Use of software tools
b. Application of software engineering methods
c. Required development schedule each of the 15 attributes is rated on a 6 point scale that ranges from very low to extra high . Based on the evaluation, an effort multiplier is determined from tables published by Boehm, and the result of all effort multipliers results is an *effort adjustment factor* (EAF).Usually, EAF values ranges from 0.9 to 1.4 [4].

The intermediate COCOMO model take the form:

$$EFFORT = a^* (KLOC)^b * EAF \quad (3)$$

Where, effort is calculated in person-months and *KLOC* is the estimated as number of delivered lines of code for the project.

Model 3 (Detailed COCOMO Model): The Model 3 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.

2. Non Algorithmic Model

Contrary to the Algorithmic methods, methods of this faction are based on analytical comparisons and inferences. For using the Non Algorithmic models some information about the previous projects which are similar the under estimate project is required and usually estimation process in these methods is done according to the analysis of the previous datasets.

Some methods are:

a) Analogy

Estimating by analogy means comparing the proposed project to previously completed similar project where the project development information id identified. Actual data from the previously complete projects are taken to estimate the planned project. This method is used either at system-level or at the component-level. Estimating by analogy is relatively simple and straightforward. Actually in some respects, it is an organized form of expert judgment since experts often search for analogous situations so as to inform their opinion.

The estimating steps of this method are:

i. Find out the characteristics of the proposed project.
ii. Select the most similar complete projects whose characteristics are stored in the historical based data.

The process concerned to predict software price estimates can be broken down into the following steps according to

- Selection of analogies
- Assessing difference and similarities
- Assessment of analogy quality
- Consideration of any special cases
- Creating the estimate

b) Expert Judgment

Expert judgment techniques involve consulting with software cost estimation expert or a group of the experts to use their experience and understanding of the proposed project to arrive at an estimate of its cost. Generally speaking, a group consent technique, Delphi technique is the best technique to be used. The strength and weakness are complementary to the strengths and weaknesses of algorithmic method. To provide a sufficiently broad communication bandwidth for the experts to exchange the volume of information necessary to calibrate their estimates with those of the other experts, a wideband Delphi technique is introduced over standard Delphi technique.

The estimating steps using this method:

i. Coordinators present each expert with a specification and an estimation form.
ii. Coordinator calls a group meeting in which the experts discuss estimation issues with the coordinator and each other.
iii. Experts fill out forms anonymously.
iv. Coordinator prepares and distributes a summary of the estimation on an iteration form.
v. Coordinator calls a group meeting, specially focus on having the experts discuss points where their estimates varied widely.
vi. Experts fill out forms, again secretly, and steps 4 and 6 are repeated for as many rounds as appropriate.

The wideband Delphi Technique has subsequently been used in a number of studies and cost estimation activities. It has been highly efficacious in linking the free discuss advantages of the group meeting technique.

3. Machine learning Models

Most techniques about software cost estimation use statistical approach, which are not able to provide reason and strong results. Machine learning approaches could be appropriate at this filed because they can increase the accuracy of estimation by training rules of estimation and repeating the run cycles.

Some models are:

a) *Artificial Neural Networks*

ANNs posses' large number of highly interconnected processing elements called neurons, which generally operate in parallel and are configured in regular architectures. Each neuron connected with the other by a communication link and each connection link is associated with weights which contain information about the input signal. The neuron calculates the weighted sum of its inputs and generates an output if the sum exceeds a certain threshold. The process continues until one or more outputs are generated. These are estimation models that can be "trained" using historical data to produce ever better results by automatically adjusting their algorithmic parameter values to reduce the delta between known actual and model predictions.[8][9].

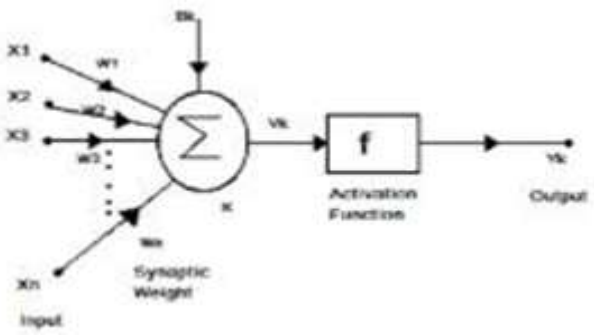

Figure I: Artificial Neural Network

It include following techniques:

i) Back Propagation Algorithm

This is the most widely used algorithm for supervised learning with multilayered feed forward networks. Once the network has been constructed, the model must be trained by giving it with a set of historical project information. Inputs and the related known actual values for project schedule. Then the model repeats on its training algorithm, automatically adjust the factors of its estimation functions until the model estimate and the actual values are within some pre-specified value. [10][11].

ii) Resilient Back Propagation

RPROP is used for performing supervised batch learning for multilayered feed forward Networks. The basic principle of RPROP is to eliminate the harmful influence of the size of the partial derivative on the weight step. RPROP changes the size of weight step ,directly by providing the concept of

resilient update values. So, the adaptation effort (AF) is not deteriorated by un-foreseeable gradient behavior [12].

iii) Gradient Descent Learning

This algorithm tries to minimize the error E between actual and desired output by adjusting the synaptic weights between the neurons by an amount proportional to the first derivative of the mean squared error with respect to the synaptic weight. So, if W_{ij} is the weight update of the link connecting the i^{th} and j^{th} neuron of the two neighboring layers, then Wij is defined as

$W_{ij} = n\ \partial E/\partial W_{ij}$

Where, n is the learning rate parameter and $\partial E/\partial Wij$ is the error gradient with reference to the weight W_{ij}.

iv) Delta Rule

Delta rule is the special case of Gradient Descent Learning. Delta rule is also referred as the Widrow-Hoff Learning Rule. According to this learning rule the mechanism for weight modification during the training process acts in an appropriate way in order to minimize the difference between the desired output and the actual output produced by the processing elements. It is also called the Least Mean Square Learning Rule, because it is used to minimize the mean squared error of that difference**.**

v) Cascade Neural Network

Cascade NN is a feed-forward neural network [13] where the first layer will get signal from input values. Each subsequent layer will receive signal from the input and all previous layers. Cascade-correlation (CC) is an architecture, feed-forward, generative, supervised learning algorithm for artificial neural networks. Cascade- Correlation begins with a small network, then automatically repeats and add new hidden units one by one creating a multi-layer structure. Cascade-correlation performs better then the Back-propagation learning algorithm. Accuracy is high in Cascade-correlation because it has less error values [14].

b) Fuzzy Logic

All systems, which work based on the fuzzy logic try to simulate human behavior, reasoning and understanding. In many problems, where decision making is very difficult and conditions are hazy, fuzzy systems are proficient tool in such conditions. This technique always wires the facts that may be ignored.

There are following four stages in the fuzzy logic approach:
Stage 1: Fuzzification: For the linguistic terms, produce trapezoidal numbers.
Stage 2: By producing a new linguistic term, develop the complexity matrix.
Stage 3: To find the rate of productivity and the effort for the new linguistic terms.
Stage 4: Defuzzification: To find out the effort needed to complete a task and to compare the existing method.

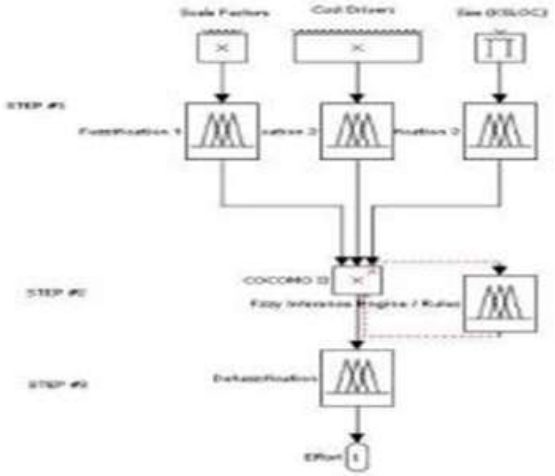

Figure 2: Fuzzy method example

For example in (Attarzadeh, Ow, 2010) COCOMO technique has been implemented by using fuzzy method. The Figure 2 displays all mentioned steps; In the first step fuzzification has been done by scale factors, cost drivers and size . In step 2, principals of COCOMO are considered and defuzzification is accomplished to gain the effort.

4. Comparison of Estimation Techniques

At this section according to the previous presented subjects, it is possible to compare mentioned estimation techniques based on advantages and disadvantages of them. This comparison could be useful for selecting an appropriate technique in a particular project. On the other hand, selecting the estimation technique is done based on capabilities of methods and state of the project. Table I shows a comparison of mentioned techniques for estimation. For performing comparison, the admired existing estimation techniques have been selected. According to the current comparison and which is based on the principal of the algorithmic and non algorithmic methods(described in previous sections); for using the non algorithmic methods it is necessary to have the enough information about the similar previous projects, because these methods perform the estimation by analysis of the historical data. Also, non algorithmic methods are relative simple to learn and understand because all of them are followed by the human behavior. On the other hand, Algorithmic methods are based on mathematics and some tentative equations. They are usually difficult to learn and they need to the much data about the current project state.

Table 1: Comparison of the existing methods

Technique	*Type*	*Advantages*	*Disadvantages*
COCOMO	Algorithmic	Clear results, very common	Much data is required, It ‘s not suitable for any project,
Expert Judgment	Non Algorithmic	Fast prediction, Adapt to especial projects	Its success depends on the expert, Usually is done partially.
Function Point	Algorithmic	Language free, Its results are better than SLOC	Mechanization is hard to do , quality of output are not considered

Analogy	Non Algorithmic	Works is based on the actual experiences, having special expert is not	A lots of information about previous projects is needed, In some cases there are no similar projects.
ANN	Non Algorithmic	It can derive precious information and regularities from large databases essentially.	No guidelines available for designing the NN topologies.
Fuzzy	Non Algorithmic	Training is not required, Flexibility	Hard to use, Maintaining the degree of meaningfulness is difficult

Application

Software effort estimation is very important task for software industry for successful completion of the project. Some important characteristics of NNs are that they exhibit mapping capabilities, their capability to generalize, processing in parallel and fault tolerance.[15] Neural networks have been successfully applied to a variety of real world tasks in industry, business and science. Applications include Accounting and finance, health and medicine, engineering [16] and manufacturing, marketing, bankruptcy prediction, image processing, handwriting recognition, speech recognition, product inspection and fault detection [17].

5. Conclusion

Finding the most important reason for the software project failures has been the subject of many researches in last decade. According to the results of several researches presented in this paper, the root cause for software project failures is inaccurate estimation in early stages of the project. So introducing and focusing on the estimation methods seems necessary for achieving to the accurate and reliable estimations. In current study most of the present estimation techniques have been illustrated analytically. As the software project managers are used to select the best estimation method based on the conditions and status of the project, describing and comprising of estimation techniques can be useful for decreasing of the project failures. There is no estimation method which can be present the best estimates in all various situations and each technique can be suitable in the specific project. It is necessary to understand the principals of each estimation method to select the best suited. Because the performance of every estimation method depends on several parameters such as complexity of the project, duration of the project, expertise of the staff, development method and so on. Some evaluation metrics and an actual estimation example have been presented in this paper just for describing the performance of an estimation method (for example COCOMO). A number of performance comparisons between neural and traditional estimation techniques have been made by many studies.

The work to be done in future is to study the new software cost estimation methods and models that can be help us to easily understand the software cost estimation process and Trying to improve the performance of the existing methods and introducing the new methods for estimation based on today's software project requirements .

References

[1] Pressman, Roger S "Software Engineering: APractioner's Approach", 6th Edn., McGraw-Hill New York, USA.,ISBN:13:9780073019338,2005.
[2] K.Ramesh and P.Karunanidhi "Literature Survey On Algorithmic And Non- Algorithmic Models For Software Development Effort Estimation", International Journal Of Engineering And Computer Science ISSN:2319-7242 Volume 2 Issue 3 March 2013 Page No. 623-632.
[3] R.K.D. Black, R. P. Curnow, R. Katz and M. D. Gray, BCS Software Production Data, Final Technical Report, RADC-TR-77-116, Boeing Computer Services, Inc., March 1977.
[4] B.W. Boehm, —Software Engineering Economics,! Prentice- Hall, Englewood Cliffs, NJ, USA, 1981.
[5] Chris F.K, —An Empirical Validation of Software Cost Estimation Models!, Management Of Computing -Communications of ACM, Vol: 30, No. 5, pp.416-429, , May 1987.
[6] Magne J and Martin S, — A Systematic Review of Software Development Cost Estimation Studies !, IEEE Transactions On Software Engineering, Vol. 33, No. 1, pp. 33-53, January 2007.
[7] K.Subba Rao and L.S.S Reddy "Software Cost Estimation in Multilayer Feed forward Network using Random Holdback Method, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 10, October 2013.
[8] Bogdan m wilamowsky," NN architectures and learning, "fellow member IEEE, USA.
[9] I.F Barcelos Tronto, J.D. Simoes da Silva, N. Sant'Anna," The artificial neural networks model for software effort estimation,"2006.
[10] Sebastian Seung," Multilayer perceptrons and back propagation Learning," 2002
[11] Mikael Boden," A guide to recurrent neural networks and back propagation," Halsted University, 2001
[12] Martin Riedmiller, "RPROP- Description and Implementation Details, Technical Report", University of Karlsruhe,1994.
[13] Vachik S.Dave, Kamlesh Dutta," Application of Feed-Forward Neural Network in Estimation of Software Effort," Intelligent Systems & Communication (ISDMISC) 2011.
[14] Anjana Bawa, Mrs.Rama Chawla," Experimental of Effort Estimation Using Artificial Neural Network."
[15] Girish kumar jha,"Artificial NNs and its applications."
[16] Rossana M. S. Cruz, Helton M. Peixoto and Rafael M. Magalhaes," Artificial Neural Networks and Efficient Optimization Techniques for Applications in Engineering."
[17] F. Farnood Ahmadia, M. J. Valadan Zoeja, H. Ebadia, M. Mokhtarzadea," the application of neural networks, image processing and cad-based environments facilities in automatic road extraction and vectorization from high resolution satellite images."