ISSN:0975-9646

Software Development Effort Estimation Using Fuzzy Logic: A Review

Shifali [1], Naveen Bilandi[2]

[1]M.Tech (CSE) Student, DAV University Jalandhar
[2]Assistant Professor in dept. of CSE, DAV University Jalandhar

Abstract: Accurate effort prediction is an ongoing challenge to software engineers. It is an important task in the management of software projects. Effort estimation is the challenging and is an important area in the software project management research field. The development of software has always been characterized by the parameters that contain certain level of fuzziness. This requires some degree of uncertainty be introduced in the models, to make the model reliable. Fuzzy Logic techniques are used to tackle the uncertainty issues. Fuzzy Logics could produce better estimates provided that various parameters and factors pertaining to fuzzy logic are carefully set. The primary purpose of this paper is to estimate the software development effort using Fuzzy Logic Techniques in order to improve accuracies.

Keywords: Fuzzy Logic and Fuzzy Logic System (FLS), Software Effort Estimation, Estimation Models, Evaluation Techniques, COCOMO.

1. INTRODUCTION

Software effort estimation is the process of predicting the effort required to develop a software system based on incomplete, crude, uncertain or ambiguous inputs. It deals with the prediction of most probable cost and time to actualize the effort required development task. Development effort estimates are required by project managers for planning and for controlling the process of software development. Software development estimation techniques can be classified into three general categories:

(1) **Expert judgment:** It covers a wide range of estimation approach and it aims to derive estimates based on an experience of experts on similar projects.
(2) **Algorithmic models**: It attempts to represent the relationship between effort and one or more characteristic of a project; the main cost driver in a model is usually taken to be some notion of software size (e.g. the number of lines of source code).
(3) **Machine learning:** Fuzzy logic models are included in this category as well as neural networks, genetic programming, regression trees and case-based reasoning. Software cost and schedule estimation supports the planning and tracking of software projects.

The effort prediction aspect of software cost estimation is concerned with the prediction of the person-hour required to accomplish the tasks. Researchers have proposed so many models to be used for effort estimation. One of the main inputs to any effort software size e.g., line of code (LOC). As such, measuring/estimating the software size accurately and also as early as possible is of prime importance. A good size estimate can lead to a good effort estimate. The accuracy of parametric effort prediction models is improved via calibration to the target development environment. The calibration activity involves adjusting the parameters of the formulae. Examples of popular effort prediction models include Constructive Cost Model (COCOMO), Constructive Cost Model II (COCOMO II), and Software Life Cycle Management (SLIM). These models are centred on using the future software size as the major determinant of effort.

Several Algorithmic manual models are purposed for effort estimation. They are in the form of

$\text{Effort}= a*(\text{SIZE})^{b}$

Where ‘a’ and ‘b’ are empirically determined constants, Size is length of thee code in KLOC.

There are various effort estimation equations Based on KLOC Such As:

Halsted Equation:

$\text{Effort} =5.2(\text{KLOC})^{1.50}$ (1)

Bailey-Basil:

$\text{Effort} = 5.5 + 0.73(\text{KLOC})^{1.16}$ (2)

Doty:

$\text{Effort} = 5.288(\text{KLOC})^{1.047}$ (3)

Traditional Parametric model COCOMO (Constructive Cost Model) was given by Boehm, the most famous & popular algorithmic effort estimation model. In it the software projects are grouped into three classes of the projects

1) Organic
2) Semidetached
3) Embedded.

The relationship between these three classes in terms of time and size is shown in the graph as given below:

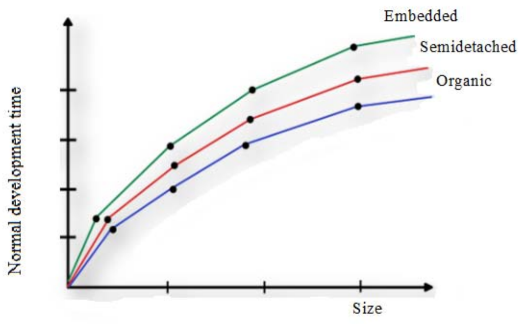

Figure 1: Development time versus size

The COCOMO Model General equation comes in the form of:

$\text{Effort} = a\,(\text{KLOC})^{b}$

The values of ‘a’ and ‘b’ depends upon the class of the project it belongs.

2. RELATED TO WORK:

Muzaffar et al. [21] have mentioned the traditional approaches to be used for effort prediction like use of models derived from historical data, or the use of expert opinion to obtain an effectiveness and robustness. The work which is being carried out produce promising results with the use of Fuzzy Logic which deals with imprecision used to characterise the early phases of software development projects. This paper presents an empirical study on the prediction of the accuracy of Fuzzy Logic based effort prediction which is highly dependent on the system architecture, parameters and the training algorithm. It also includes the impact of the nature of training algorithm on the Fuzzy Logic System (FLS) based effort prediction. One thing is to notice in this paper is that conclusions made with regard to the outcome are based on the artificial datasets whose generation procedure was justified and meaningful. As the need to use real world datasets can produce better outcomes.

A.Ahmed et al. [22] have presented about the adaptive Fuzzy Logic framework for effort prediction as algorithmic effort prediction models are not very efficient to cope with the uncertainty and imprecision present in the software projects. So the training and adaptation algorithm used in the framework are able to handle the imprecision, explain the prediction through rules, offers transparency in the prediction system, and could adapt to the new environment when new data is available. This paper includes the transparent FL-based framework which allows contribution from experts, and is equipped with training and adaptation algorithm for development effort prediction. This paper demonstrates the capabilities of the framework through empirical validation carried out on artificial datasets and the COCOMO.

Martin et al. [19] have investigated the comparison between Fuzzy Logic Models (FLM) and Linear Regression Model (LRM) because the engineers have the less capability which is being provided by personal training, therefore they cannot support their teams to produce reliable results. This paper includes the evaluation criterion which is based on the magnitude of error relative to the estimate (MER) as well as to the mean of MER (MMER). In this paper, small programs are being developed by programmers. Among these programs, Fuzzy Logic Models were generated to estimate the effort. Verification and Validation of the models are made. The output thus generated by the Fuzzy Logic model and Linear Regression produce the similar predictive accuracy, so Fuzzy Logic Model can be used as an alternative to estimate effort.

Cuauhtémoc Lopez-Martin [23] has predicted that Fuzzy Models are used to estimate the effort of software projects starting from the short scale programs. So he included the new and changed (N&C) as well as reused code which were gathered from small programs developed by programmers using the practices of personal software process and the data act as a input for the Fuzzy Model to estimate Effort. The accuracy of this model was compared with the statistical regression model which involves a well defined procedure to compute the result. Whereas the Fuzzy Model is a heuristic model whose generation involves some stages that need the judgement of the people. In future, classifiers and associative memories can be used to estimate and predict the effort of small programs as well as large programs.

3. FUZZY LOGIC AND FUZZY LOGIC SYSTEM:

Fuzzy Logic is a kind of many valued logic derived from Fuzzy set theory that is concerned with reasoning which is approximate rather than fixed and exact. Fuzzy Logic was first proposed by Zadeh [2] in 1965. Fuzzy logic has been applied to many fields, from control theory to artificial intelligence. Fuzzy Logic explains system of mathematics that is used to model the inference structure that enables appropriate human reasoning capabilities. Fuzzy logic variables may have a truth value that ranges in degree between 0 and 1. For example, the result of a comparison between two things could be not "tall" or "short" but ".38 of tallness." Zadeh explains that "*As complexity rises, precise statements lose meaning and meaningful statements lose precision."*It provides a technique to deal with the imprecision that occur in the measurement process. Fuzzy Logic can handle both quantitative and qualitative information within one model. Furthermore, when linguistic variables are used, these degrees may be managed by specific functions such as "many", "low", "medium", "often", "few". On contrary, Fuzzy sets are the sets whose elements have degree of membership. In classical set theory, the membership of elements in a set is assessed in binary terms according to bivalent condition that is it describes crisp events, events that either do or do not occur.

Fuzzy Logic System (FLS) are the system that has a direct relationship with fuzzy concepts (e.g. Fuzzy Sets and quantitative values) and Fuzzy Logic. The most popular Fuzzy Logic systems is classified into three types: pure Fuzzy Logic System, Takagi and Sugeno's Fuzzy System, and Fuzzy Logic System with fuzzifier and defuzzifier. The mostly used application is as crisp data is taken as input into Fuzzy Sets and produce crisp data as output, therefore last one technique is most widely used in which fuzzifier maps crisp inputs into fuzzy sets and the defuzzifier maps Fuzzy Sets into crisp outputs. This type was first proposed by Mamdani [3]. It consists of four main components:

Fuzzifier: It converts the crisp input into a fuzzy set. Membership Functions are used to graphically describe a situation.

Fuzzy Rule Base: It uses if-then rules.

Fuzzy Inference Engine: A collection of if -then rules stored in fuzzy rule base is Known as inference engine. It performs two operations i.e. aggregation and composition.

Defuzzification: It is the process that refers to the translation of fuzzy output into Crisp output.

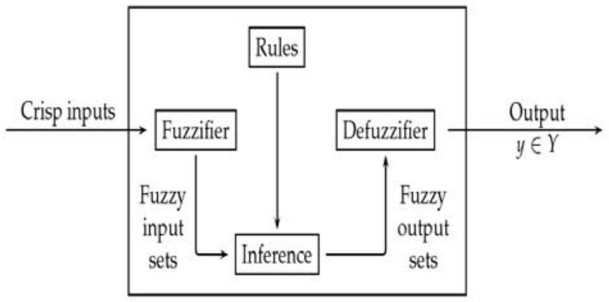

Figure 2: Block Diagram of Fuzzy Logic system with fuzzifier and defuzzifier

Disadvantages of a Fuzzy model could be that (1) it requires a lot of data, (2) the estimators must be familiar with the historical developed program, and (3) it is not useful programs much larger or smaller than the historical data.

4. Types Of Models For Effort Prediction:

Software effort estimation is one of the first step of software measurement. So this is the one of the most oldest and mature aspect of software metrics. It includes both types of models named as Algorithmic and Non-algorithmic estimation techniques.

4.1 *Algorithmic models:*

Boehm was the first researcher to look at software Engineering from an economic point of view. He introduces a cost estimation model, COCOMO-81 in 1981, after investigating a large set of data from TRW in the 1970s [4]. Putnam also introduced an early model known as SLIM in 1978 [5]. These both models are based on linear regression techniques, [6] using data from past projects. Both COCOMO and SLIM take number of lines of code (about which least is known very early in the project) as the major input to their models. Albrecht's function points measures the amount of functionality in a system as described by a specification [6]. A survey on these algorithmic models and other cost estimation approaches is presented by Boehm et al. [7].

Most models rely on accurate estimate of either size of software in terms of line of code (LOC), number of user screen, interfaces, complexity, etc. at a time when uncertainty is mostly present in the project [8].

COCOMO model, has failed to present suitable solutions that consider technological advancements [9]. The reason why algorithmic models have not proven to provide such solution is that, they are unable to capture the complex set of relationships (e.g. the effect of each variable in a model to the overall prediction is made using the model) that are evident in many software development environments [10]. They can be successful for a particular type of environment, but not comfort enough to adapt new environment. Their inability to handle categorical data (that is, data that are specified by a range of values) and most importantly lack of reasoning capabilities (that is, ability to draw conclusions or make judgments based on available data) contributed to the number of studies exploring non algorithmic methods (e.g. FL).

4.2 *Non-algorithmic models:*

These models are soft computing based that were sought in 1990s. It is a kind of methodology centering within FL, artificial neural networks (ANN) and evolutionary computation (EC). These have capability to provide flexible information processing to handle the real life problems. FL with powerful representation can represent imprecision in inputs and outputs. This provides a more expert Knowledge-based approach to model building. A study by Hodgkinson and Garratt Proposed that expert judgement estimation was better than all regression-based models [9].

Some of the existing algorithmic models are fuzzified in order to handle the uncertainties and imprecision problems in such models.

FL has also offered itself as a useful tool to aid other techniques for software cost estimation like analogy. Similarity between projects is often used when estimating software effort by analogy. The nearest neighbour algorithm [10] is one of the approach use to derive similarity as input to the estimation process. This algorithm cannot handle projects attributes described by categorical variables

The ANN technique was used for cost estimation [11]. ANN is able to generalise from trained data set. Over a known set of training data, an ANN learning algorithm constructs mapping that fit the data, and fits previously unseen data in a reasonable manner [5].

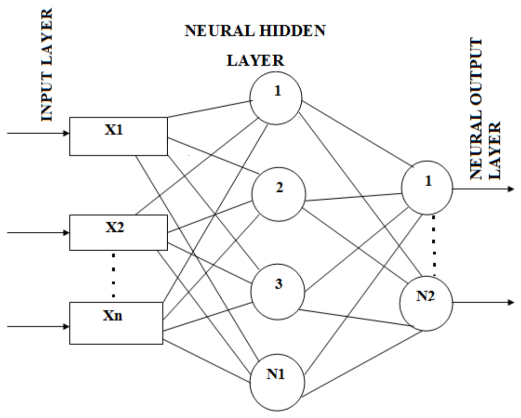

Figure 3: Typical Multilayered ANN Feed forward structure composed of three layers

Here it consists of networks of simple processing elements (called 'neurons') operating on their local data and communicating with other elements. Actually the design of ANNs was motivated by the structure of a real brain.

There exist two main types of training process: supervised and unsupervised training. Supervised training (e.g. multi-layer feed-forward (MLF) neural network) means, that neural network knows the desired output and adjusting of weight coefficients is done in such way, that the calculated and desired outputs are as close as possible. Unsupervised training means, that the desired output is not known.

Here it includes MLF neural networks trained with a back-propagation learning algorithm, are the most popular neural networks. A MLF neural network consists of neurons that

are ordered into layers (Fig. 3). The first layer is Called the input layer, the last layer is called the out- put layer, and the layers between are hidden layers.
EC has also recently found its usefulness in software effort estimation. It uses applied genetic programming (GP) to software effort estimation. EC simulates evolution on a computer.

5. Software Measurement:

As there is a wide range of software product size measures available but source lines of code (SLOC) is the most widely used among all the models and researchers use it to correlate effort [13]. There are two measures of source code size: physical source lines of code, and logical source lines of code. The physical source lines measure gives the size in terms of the physical length of the code as it appears when printed, but with comments, blank lines, and delimiters not counted [14]. It includes the independent variables of the models that are new and changed (N&C): integrated by added and modified code as well as reused code [15]. The added code is the SLOC added during current program, while the modified code is the SLOC changed in base program when modifying a previously developed program. The base program is the total LOC of the previous program and reused code is the LOC of program that is developed previously and is used without any modification. A coding standard should create a consistent set coding practice which is used as a criterion for the judgement of the produced code [15].Therefore, it is important to always use same coding and counting standards.

6. Evaluation Criteria:

The magnitude of relative error (MRE), (a common criterion for the evaluation of cost estimation models) does not always give the best prediction model, as it implies that the results and conclusions on prediction of models over the past 15-25 years are unreliable and may misguide the entire software engineering discipline[16], therefore a preferred accuracy criterion is the magnitude of error relative to the estimate(MER), because unlike MRE, measures the inaccuracy relative to the estimate[17]. Results of the mean MER (MMER) had better results than the mean (MMRE); that's why the mean MER (MMER) is used in the study.
The MER is defined as follows:

$$\mathbf{MER}\boldsymbol{i} = \frac{|\,\text{Actual Effort}i - \text{Estimated Effort}i\,|}{\text{Estimated Effort}i}$$

The MER value is calculated for each observation *i* whose effort is predicted.

$$\mathbf{MMER} = (1/N)\sum_{i=1}^{N} MERi$$

Pred (*l*) is used as complementary criterion for counting the percentage of estimates that fall within less than *l* of actual values. The common value used for *l* is 25% and a prediction model is considered as acceptable when its accuracy level is 75% [18]. It is calculated as follows:
Pred (*l*) = K/N

Where N is the total number of projects, and K is the number of projects with a MER less than or equal to *l*. In this, we used 0.25 as a value for *l*.
In general, the accuracy of an estimation technique is proportional to the Pred (*l*) and inversely proportional to the MMER. As reference, for effort prediction models, an MMRE≤0.25 is considered as acceptable [19].

7. Various Applications Of Fuzzy Logic:

1. Fuzzy logic has been used in numerous applications such as facial pattern recognition, air conditioners, washing machines, vacuum cleaners, transmission systems, and knowledge-based systems for multiobjective optimization of power systems, systems, models for new product pricing or project risk assessment, medical diagnosis and treatment plans. It has been successfully used in numerous fields such as control systems engineering, image processing, power engineering, industrial automation, robotics, consumer electronics, and optimization.
2. The unique contribution of Fuzzy Logic is that it provides a practical approach to automate complex data and inference process that are usually performed by human experts with years of formal training.
3. In developing a weather forecast, expert consult data from various observations and models, each having a different level of relevance and reliability.
4. A Fuzzy Logic algorithm solution would first develop modules for analysing and performing quality control for various source of information. For e.g.: The standard image processing technique might be used to measure the local characteristics in radar or satellite data.
5. Statistical analysis might be employed to help determine data quality, or to determine conditional probabilities based on historical data.
6. Physical models such as trained neural networks and other empirical models might be used to relate raw sensor measurements to a quality of interest. For e.g.: Determining temperature profiles from radiometer measurements.

8. Conclusion And Future Scope:

This paper presents a review on the study of Fuzzy Logic system factors that are used for the software development effort prediction. Various models are being used like COCOMO and SLIM for effort estimation also includes various advantages of Fuzzy Logic to estimate effort. This paper basically includes three facts: (1) software effort estimation is one of the most critical activities in managing software projects, (2) Different kinds of factors like SLOC and various other factors which are to be used in evaluation criteria are applicable and compared with one another to produce more realistic and reliable results, because only single software estimation factor cannot be used for all the problems, (3) To produce good quality products, personal training of engineers is must to support their teams consistent. The comparisons are made between MER and MMER to estimating the effort, considering the dependability between MER and effort. Result showed that the Fuzzy Logic can predict the accurate results. So a Fuzzy Logic could be used as an alternative for estimating the development effort at personal level.

Future research involves that various techniques to be used like neural networks and the factors which are to be used to predict effort like SLOC and terms used in evaluation criteria are useful in other areas also. This can provide a new path for the researcher in their field, so that they can match their outputs or reports otherwise for the confirmation of similar outcomes.

REFERENCES:

[1]. Roheet Bhatnagar, Vandana Bhattacharjee, Mrinal Kanti Ghose, A proposed novel framework for early effort estimation using Fuzzy Logic techniques, Global Journal of computer science and technology (2010) 66-72.
[2]. L.A. Zadeh, Fuzzy sets, Information and Control 8 (1965) 338–353.
[3]. E.H. Mamdani, Applications of fuzzy algorithms for simple dynamic plant, Proceedings of the IEEE 121(12), 1974.
[4]. Z. Fei, X. Liu, and f-COCOMO: fuzzy constructive cost model in software engineering, Proceedings of the IEEE International Conference on Fuzzy Systems, IEEE Press, New York, 1992, pp. 331 -337.
[5]. K. Srinivasan, D. Fisher, Machine learning approaches to estimating software development effort, IEEE Transactions on Software Engineering 21 (2) (1995).
[6]. N.E. Fenton, S.L. Pfleeger, Software Metrics—A Rigorous and Practical Approach, second ed., PWS Publishing, Boston, MA, 1997.
[7]. B. Boehm, C. Abts, S. Chulani, Software development cost estimation approaches—a survey, Technical Reports, USC-CSE-2000-505,University of Southern California Center for Software Engineering,2000.
[8]. J. Ryder, Fuzzy modeling of software effort prediction, Proceedings of IEEE Information Technology Conference, Syracuse, NY, 1998.
[9]. A.C. Hodgkinson, P.W. Garratt, A neurofuzzy cost estimator, in: Proceedings of the Third International Conference on Software Engineering and Applications—SAE, 1999, pp. 401–406.
[10]. C. Schofield, Non-algorithmic effort estimation techniques, Technical Reports, Department of Computing, Bournemouth University, England, TR98-01, March 1998.
[11]. A.R. Venkatachalam, Software cost estimation using artificial neural networks, in: Proceedings of the International Joint Conference on Neural Networks, 1993, pp. 987–990.
[12]. A. Idri, A. Abran, T. Khoshgoftaar, and Fuzzy analogy: a new approach for software cost estimation, in: International Workshop on Software Measurement (IWSM˙ı01), Montreal, Quebec, Canada, 2001, August 28–29.
[13]. R. Jeffery, M. Ruhe, I. Wieczorek, A comparative study of two software development cost modeling techniques using multi organizational and company-specific data, Information and Software Technology 42 (2000) 1009–1016.
[14]. W. Humphrey, a Discipline for Software Engineering, Addison Wesley, 1995.
[15]. T. Foss, E. Stensrud, B. Kitchenham, I.Myrtveit, A simulation study of the model evaluation criterion MMRE, IEEE Transactions on Software Engineering 29 (11)(2003).
[16]. M. Jorgenson, D.I.K. Sjoberg, The impact of customer expectation on software development effort estimates, International Journal of Project Management 22 (2004) 317–325.
[17]. M. Jorgenson, U. Indahl, D. Sjoberg, Software effort estimation by analogy and regression toward the mean, The Journal of Systems and Software 68 (2003) 253–262.
[18]. M. Jorgensen, M. Shepperd, A systematic review of software development cost estimation studies, IEEE Transactions on Software Engineering 33 (1) (2007) 33–53.
[19]. Cuauhtemoc Lopez-Martin, Cornelio Yanez-Marquez, Agustin Gutierrez-Tornes, Predictive accuracy comparison of Fuzzy models for software development effort of small programs, the Journals of systems and software 81(2008) 949-960.
[20]. Noel Garcia-Diaz, Cuauhtemoc Lopez-Martin, Arturo Chavoya, A comparative study of two Fuzzy Logic Models for Software development effort estimation, Procedia Technology 7(2013) 305-314.
[21] . Zeeshan Muzaffar, Moataz A. Ahmed, Software development effort prediction: A study on the factors impacting the accuracy of Fuzzy Logic systems, Information and Software Technology 52(2010) 92-109.
[22]. Moataz A. Ahmed, Moshood Omolade Saliu, Jarallah AlGhamdi, Adaptive Fuzzy Logic-based framework for software effort prediction, Information and Software Technology 47(2005) 31-48.
[23]. Cuauthemoc Lopez-Martin, A Fuzzy Logic model for predicting the development effort of short scale programs based upon two independent variables, Applied Soft Computing 11(2011) 724-732.
[24]. Danial Svozil, Vladimir Kvasnicka, Jiri Pospichal, Introduction to multilayer feed foward neural networks, Chemometrics and Intelligent Laboratory Systems 39 (1997) 43-62.
[25]. John K. Williams, Cathy Kessinger, Jennifer Abernethy, Scott Ellis, Book Artificial intelligence methods in environmental sciences 2009, pp 347-377