Survey on Different Machine Learning Techniques for Software Effort Estimation

Lekshmi R
Department of Computer Science & Engineering
SCT College of Engineering
Trivandrum, Kerala

Binu Rajan
Department of Computer Science & Engineering
SCT College of Engineering
Trivandrum, Kerala

ABSTRACT
Software development effort estimation is the process of predicting the effort required to develop or maintain software based on vague, incomplete or uncertain inputs. Accurate estimate of software development effort is required in the early stages of development life cycle for planning the development activities. Determination of software cost, allocation of resources, scheduling and monitoring of development activities are all dependent on the effort. Hence effort estimation is crucial for the control, quality and success of all software development projects. This paper provides an overview of the three general categories of estimation models namely; Expert Judgment based models, Algorithmic models and Non Algorithmic models. Moreover a comparison of different machine learning techniques, namely Fuzzy Logic, Artificial Neural Network, Case Based Reasoning and Fuzzy Neural Network is done in order to study which machine learning method is more suitable in which situation. Advantages and Disadvantages of these four machine learning techniques are identified as well as it was found that when applying these techniques to the COCOMO dataset the fuzzy logic and Fuzzy Neural Network showed better performance compared to other techniques.

General Terms
Software Effort Estimation, Project Planning.

Keywords
Software Effort Estimation, Development Effort, Estimation Techniques, Machine Learning.

1. INTRODUCTION
Software is nowadays used in almost all areas of human endeavour. A good programming knowledge is no longer sufficient to construct large software. Serious problems arise in the cost, timeliness, maintenance and quality of many software products. The main objective of software engineering is to solve these problems by producing good quality and maintainable software on time, within budget.

Accurate effort estimation has major impact on the management of software development. Underestimation of effort results in a situation where the developers will be forced to complete the product in a short time span which in turn results in a product which is not fully functional or tested. As a result the product may contain errors which will in turn increase the cost of maintenance. On the other hand, if the effort is overestimated, then more resources than the required number will be allocated to the project and this hampers the efficient resource utilization in an organization.

Several estimation models have been proposed and developed to provide accurate estimates and to decrease the estimation errors. Many of these models are parametric models which uses a formula that is parameterized from historical data for estimating the development effort. COCOMO, FUNCTION POINTS and SLIM are some of the examples for parametric models. A number of machine learning techniques were developed in the recent years for better effort prediction.

This paper provides an overview of the general categories of effort estimation models and different measures used to evaluate these models. A detailed study of different machine learning techniques namely, Fuzzy Logic, Artificial Neural Network, Case Based Reasoning and Fuzzy Neural Network is done and their advantages and disadvantages are identified. Moreover these techniques are compared based on their performance while using the COCOMO dataset.

2. BACKGROUND
2.1 Estimation Techniques
Software effort estimation techniques can be roughly classified into three categories:

2.1.1 Expert Judgement
These models are based on the experience of the experts, i.e, knowledge acquired from the implementation of past projects. Delphi Technique[3] and Work Breakdown Structure are the most common examples of expert judgment based models.

a) Advantages

- Experience from the past projects is utilized to assess the factors that influence the new projects.
- Useful for new programs for which no historical data is available.
- Experts can consider the impacts caused by new technologies and languages in the new project[1].

b) Disadvantages

- The method cannot be quantified.
- Experts may be biased.
- Factors used may vary based on the experts and hence is difficult to document it.

2.1.2 Algorithmic Models
Algorithmic estimation models use mathematical formulas that relate independent variables such as effort drivers do dependant variables such as effort/cost. These formulas are based on the historical data. One of the well known

algorithmic estimation model is the COCOMO model developed by Barry Boehm[2].

a) Advantages

- It is able to reproduce estimates as the formulas are based on past historical data.
- Helps reduce the responsibility of experts in estimation process.
- It is objective in nature and can be calibrated in order to take into account the previous experience.

b) Disadvantages

- Additional overhead may be needed for calibrating the system to the local circumstances.
- The effort drivers are vague at the starting stage and this will affect the accuracy of the estimates.
- Some factors and experience are difficult to quantify.

2.1.3 Non Algorithmic Models

Non algorithmic models were introduced in the 1990's. The limitations of algorithmic models insisted the researchers to explore techniques which are based on soft computing. They delved into new approaches based on Artificial Intelligence techniques such as Artificial Neural Networks(ANN), Fuzzy Logic, and Genetic Algorithms in order to predict accurate estimates[4]. Neural Networks are capable of generalizing based on trained data set. Fuzzy logic provides powerful linguistic representation that has an innate capacity to model the uncertainty/vagueness of inputs and outputs.

2.2 Measures for Evaluating Estimation Models

Some of the common measures for evaluating the different estimation models are:

2.2.1 Sum Squared Error (SSE)

It is defined as

$$E = \left[\sum_{i=1}^{n} (P_i - A_i)^2\right] \quad (1)$$

Here Pi denotes the predicted value of data point i and Ai denotes the actual value of data point i. Total number of data points is given by n. The value of E ranges from 0 to infinity.

2.2.2 Mean Squared Error (MSE)

MSE is defined as:

$$E = \left[\frac{1}{n}\sum_{i=1}^{n} (P_i - A_i)^2\right] \quad (2)$$

Pi and Ai corresponds to the predicted and actual value of data point i respectively. Total number of data points is represented by n. Again, E ranges from zero to infinity.

2.2.3 Root Mean Squared Error (RMSE)

RMSE is defined as

$$E = \left(\sqrt{\frac{1}{n}\sum_{i=1}^{n} (P_i - A_i)^2}\right) \quad (3)$$

Pi, Ai and n are same as in MSE.

2.2.4 Mean Magnitude of Relative Error (MMRE)

MMRE measures the difference between actual and estimated effort relative to the actual effort[5]. It is defined as

$$MMRE = \frac{1}{n}\sum_{i=1}^{n} \frac{|(P_i - A_i)|}{A_i} \quad (4)$$

2.2.5 Relative Absolute Error (RAE)

The relative absolute error takes the total absolute error and normalizes it by dividing by the total absolute error of the simple predictor. The relative absolute error of individual data set j is defined as:

$$E_j = \frac{\sum_{i=1}^{n} |P_{ij} - A_i|}{\sum_{i=1}^{n} |A_i - A_m|} \quad (5)$$

where Pij = Predicted value by the individual data set j for data point i. Ai = Actual value for data point; n = Total number of data points; Am = Mean of all Ai.

2.2.6 Percentage of Prediction (PRED)

It is calculated from MRE. PRED(X) is defined as the ratio of data points with error less than or equal to X to the total number of data points.

$$PRED(X) = d/n \quad (6)$$

where d is the number of data points with MRE less than or equal to X and n is the total number of data points.

3. MACHINE LEARNING TECHNIQUES FOR EFFORT ESTIMATION

3.1 Fuzzy Logic

The idea behind fuzzy logic closely corresponds to human being's feeling and inference process. Concept of Fuzzy Logic was introduced by Lotfi Zadeh, a professor at the University of California[6]. Fuzzy logic is an approach by which data is processed by allowing partial set membership rather than crisp set membership.

Most commonly used fuzzy logic system comprise of three basic stages. The first stage is called the fuzzification step and is used to convert the crisp input values to fuzzy values based on linguistic variables and membership functions. In the second stage there is a rule base which consists of some IF-THEN rules and an inference engine that derives the output based on the rules from the rule base and the input fuzzy values. The final stage is the defuzzification step where the fuzzy output value is mapped to its corresponding crisp value by the use of membership functions.

3.1.1 Effort Estimation Using Fuzzy Logic

The methodology adopted in [7] by Iman Attarzadeh et al. combined the fuzzy logic with the COCOMO II model. The

17 effort drivers and 5 scale factors serve as input to this method. Since fuzzy logic is applied to estimate the effort, these effort drivers and scale factors are converted to fuzzy variables with values Very Low (VL), Low (L), Nominal (N), High (H) and Very High (VH). Two-sided Gaussian membership functions are utilized here. After this step, fuzzy rules based on these linguistic variables are formulated. In this method, more than 193 rules were formulated for all input variables. Some of the rules used in [7] are:

IF TOOL is Low TEHN effort is Low

IF PCAP is Very Low THEN effort is Very High

IF RESUE is Nominal THEN effort is Nominal

IF DATA is Very High THEN effort is Very High

Based on these fuzzy rules and fuzzy inputs, the inference engine will produce an output. As a final step, defuzzification is performed by using the Mean Of Maximum technique.

3.1.2 Validation

The method was validated using NASA dataset which consisted of 93 projects. The MMRE value was found to be equal to 0.36654 and PRED (25%) was obtained as 50%. In [Software Development Effort Estimation Using Soft Computing] similar method was employed and it was validated using the COCOMO dataset which consisted of 63 projects. Validation was done for three types of membership functions namely, triangular, trapezoidal and Gaussian. MMRE values were found to be 0.2603, 0.2237 and 0.1657 while using triangular, trapezoidal and Gaussian membership functions respectively. The higher PRED value and the lower MMRE values indicate a better accuracy in estimates.

3.1.3 Remark

This method produced more accurate estimates when compared to the normal COCOMO II model. It efficiently handles the vagueness/uncertainty characteristics of the effort drivers during the initial stage of development. Fuzzy logic offers a convenient way to generate a strong mapping between the input and output spaces. Determining correlation between the input variables as well as the input and output variables help in formulating efficient rules.

3.2 Artificial Neural Network

Artificial neural network (ANN) is a mathematical model inspired by biological neural networks (BNN). They are a collection of processing elements which corresponds to the neurons in the biological neural network. These processing elements are arranged in the form of layers. In single layer network there will be an input layer and an output layer and a single weight vector. These weights correspond to the synaptic strength of the neurons in the BNN. Multi layer networks consists of additional layers called hidden layers. ANN utilizes a learning process to learn the historical data and past experiences and based on its learning, the network will provide the estimates. The most widely used learning algorithm is the back propagation algorithm[13]. The artificial neurons calculate the weighted sum of the inputs and then apply an activation function on it to generate the appropriate output. During the training phase the difference between the predicted and target values are propagated back through the network for weight adjustments.

3.2.1 Effort Estimation by ANN

The method adopted in [8] by Iman Attarzadeh et al. is to structure the COCOMO II post architecture model using neural networks. There are 17 effort multipliers and 5 scale factors which are used as the input. Hence in order to estimate effort using ANN, 24 input nodes corresponding to the effort multipliers, scale factors and two biases are required.

The input nodes make use of identity activation functions. The structure also contains a hidden layer that utilizes a sigmoid activation function for processing the inputs. The output layer contains a single neuron whose output is the effort in person month.

The effort multiplier (EM) values are preprocessed to log(EM). The size is considered as a cofactor for the initial weights for scale factors. Weights associated with the effort multipliers and bias1 are denoted by pi for $1 \le i \le 17$ and those associated with scale factors and bias2 are denoted as qj for $1 \le j \le 5$. 'W' and 'b' denote the weights on the arc from hidden layer to the output layer respectively. The initial values of W & b are set as the offset of the values in the hidden units of the ANN. Bias1 is initialized to log(A) and Bias2 to 1.01. The weights pi and qj are initialized to 1. The computations in the hidden layer[7] are as follows:

$$f\left(p_0\, Bias1 + \sum_{1}^{17} p_i * \log(EM_i)\right) = sigmoid\left(Bias1 + \sum_{1}^{17} p_i * \log(EM_i)\right) = \frac{A * \prod_{i=1}^{17} EM_i}{1 + A * \prod_{i=1}^{17} EM_i} = \alpha \quad (7)$$

$$f\left((q_0 + \log(size))\right) * Bias2 + \sum_{j=1}^{5} (q_j + \log(size))(SF_j) = sigmoid\left(\log(size) * \left(Bias2 + \sum_{j=1}^{5} SF_j\right)\right) \quad (8)$$

$$= \frac{Size^{1.01 + \sum_{j=1}^{5} SF_j}}{1 + Size^{1.01 + \sum_{j=1}^{5} SF_j}} = \beta \quad (9)$$

The weights W and b are computed as[7]:

$$W = \frac{\beta}{2(1-\alpha)(1-\beta)} \quad (10)$$

$$b = \frac{\alpha}{2(1-\alpha)(1-\beta)} \quad (11)$$

The ANN output is calculated as[7]:

$$PM = W * \alpha + b * \beta = \frac{\alpha\beta}{(1-\alpha)(1-\beta)} = A.Size^{1.01 + \sum_{j=1}^{5} SF_j} * \prod_{i=1}^{17} EM_i \quad (12)$$

Based on the above computations forward and backward iterations are performed until the terminating condition is satisfied. Terminating condition can be either change in weights falls below a threshold value or until completion of a specific number of iterations. The training of the network is performed as below:

i. A training sample is selected and the input vectors are propagated through the network to get the output.

ii. Error is determined and the error gradients in all layers are computed.

iii. Weight changes are determined and updated.

iv. Above steps are repeated until stopping condition is reached.

3.2.2 Validation

Training and validation of the method was performed using the COCOMO and NASA datasets. The method produced an MMRE=0.457937192, which is less than the MMRE=0.502570573 which is produced by the original COCOMO II. This proves that the estimation accuracy of ANN-COCOMO II is much better than the COCOMO. The method also produced the PRED (25%) = 45.5% which is greater than the PRED (25%) = 37.5% produced by the original COCOMO II. It shows that the estimation accuracy of ANN-COCOMO II is much better when compared to the original COCOMO II.

3.2.3 Remark

Learning mechanism facilitates the network in learning from past experiences and outcomes. Also it is capable of modeling complex relationships between independent and dependent variables.

3.3 Case Based Reasoning(CBR)

Case based reasoning approach stores the past historical data in the form of cases in a knowledge base. To estimate the effort of a new development, a case is prepared for the new problem and then it is matched with the existing cases to find similar cases.

3.3.1 Effort Estimation Using CBR

A CBR cycle contains four main activities[9]:

i. Retrieve similar cases.

ii. Reuse the information from the retrieved case to solve the new case.

iii. Revise the proposed solution.

iv. Retain the experience from the new case for future use.

First, a new case is made based on the current problem to be solved. Based on this new case, similar case from the previous cases is retrieved. The solution of the retrieved case is reused for the new case. This proposed solution is tested for success through the revise process. Useful experience is retained by updating the case base with the newly learned case for future use.

The first step in estimation using CBR is to identify the project attributes that influence the effort. These attributes are used as the basis for finding similar cases from the case base. While using the COCOMO dataset, the 17 effort drivers can be used with possible adjustments. A number of parameters are to be considered while using the CBR technique. They are:

a. Similarity Measure: It measures the level of similarity between the cases. The most frequently used measure is the Euclidean distance. It can be weighted or unweighted Euclidean distance.

b. Scaling/Standardization: It represents the transformation of attribute values according to a defined rule such that all attributes are measured using the same unit.

c. Number of Analogies: It denotes the number of most similar cases that will be used for estimation.

d. Analogy Adaptation: It determines how to generate estimates based on the analogies. Some of the common methods are the mean of closest analogies, median of analogies and inverse rank weighted mean.

e. Feature Subset Selection: It involves determination of the optimum subset of features which provides the most accurate estimate.

3.3.2 Validation

Application of CBR on the COCOMO dataset reported an accuracy levels in the region of MMRE = 40 to 50% [10]. However, the system was only able to make predictions for 46 out of 63 projects.

3.3.3 Remark

The method used by CBR is similar to the analogical reasoning used by humans. The case base can be expanded easily. However the method involves more design decisions.

3.4 Fuzzy Neural Network(FNN)

The idea of combining neural network and fuzzy logic evolved during the early 90's. The fuzzy rule sets generated by the Fuzzy Neural Network (FNN) method was found to be more suitable than the rule set generated by expert judgment approach[11]. The FNN method can be utilized in situations where fuzzy logic or neural network cannot reach a satisfactory solution.

3.4.1 Effort Estimation Using FNN

The software effort estimation using FNN includes four main activities: fuzzification, fuzzy rule base, fuzzy inference engine and defuzzification[12]. Fuzzification process converts the effort drivers into linguistic fuzzy values. Fuzzy rule base stores the fuzzy rule set. Artificial neural network is used to determine the weight of each fuzzy rule. These weights show the significance of each rule in determining the effort. The fuzzy inference engine will determine the result based on the input and the relevant fuzzy rules. Defuzzification will convert the fuzzy result into crisp value. The four activities are shown in Fig. 1. The membership functions used in the fuzzification and defuzzification process is determined using an unsupervised learning approach.

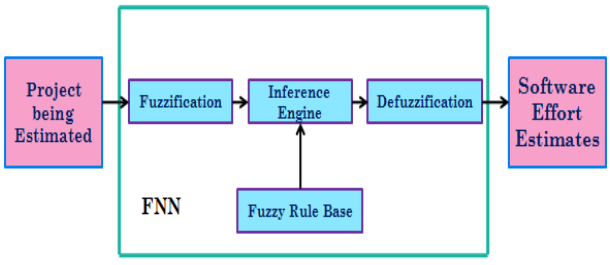

Figure 1: Effort Estimation using FNN[11]

The framework of FNN for effort estimation consists of four layers: Input layer, fuzzification layer, fuzzy rule layer and defuzzification layer as shown in Fig. 2.

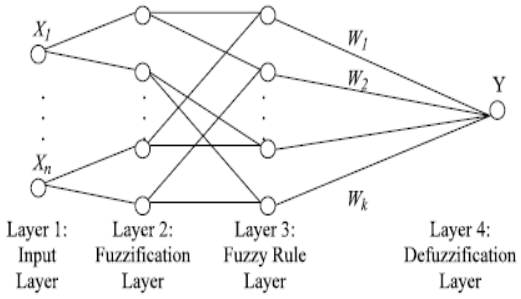

Figure 2: Framework of FNN[11]

Input Layer: The nodes in the input layer transmit the input values to the nodes in the next layer. The input values are the effort driver values which are the independent variables that affect the development effort.

Fuzzification Layer: Fuzzification of input values is performed to reduce the estimation errors caused due to the uncertainty and vagueness in the effort driver values. The input values from the nodes in the input layers are transformed to fuzzy linguistic values by the nodes in the fuzzification layer. A triangular membership function is adopted by this method. The k-means method is utilized to find the peak of the linguistic labels in the membership function. The number of clusters and the starting points required by the k-means method are generated using the SOM. Different effort drivers generate different membership functions depending on the clustering results.

Fuzzy Rule Layer: The nodes in this layer are called the rule nodes. Each node represents a fuzzy rule which is represented in the "if-then" form. The node multiplies the incoming values and transmits the product that represents the firing strength of that fuzzy rule. This layer performs a fuzzy 'AND' operation which generates the minimum value of all the incoming membership values as the output.

Defuzzification Layer: The nodes in this layer transform the fuzzy result into crisp values. Each fuzzy rule from the previous layer is associated with a weight in this layer. ri is the new firing strength of the fuzzy rule Ri from Layer 3. Wi is the weight in the fuzzy rule Ri : ri =Wi Ri .

The output node combines all the firing strengths with the corresponding singleton constituent. The gradient descent method is used here to adjust the weights of the fuzzy rules. Based on how significant a rule is for accurate estimation, the weights get increased or decreased.

The number of nodes in the input layer is based on the number of effort drivers used. The number of nodes in the fuzzification layer depends on the number of linguistic levels for each effort driver. The number of nodes in the fuzzy rule layer depends on the number of linguistic levels for effort and the defuzzification layer contains a single node.

3.4.2 Validation

For validation the COCOMO data set is divided into three pairs of training and test sets. Training is performed with all three pairs and the final result is formed by aggregating the individual results. The FNN method produced an MMRE value of 0.24, 0.22 and 0.21 for the dataset 1, dataset 2 and dataset 3 respectively. The Pred(.25) was found to be 0.86, 0.71 and 0.67 for the dataset 1, dataset 2 and dataset 3 respectively. The lower values of MMRE and high values of Pred indicate that the method provides better accuracy.

3.4.3 Remark

The accuracy of estimation is improved by the use of artificial neural network to assign weights for the fuzzy rules based on their significance and fuzzy logic for handling the uncertainty of effort drivers.

4. COMPARISON

The four machine learning techniques can be compared by verifying their MMRE value while using the COCOMO dataset. The MMRE value for each technique is given in the table below. From the observed values, it is clear that the Fuzzy Neural Network is a better approach when compared to the other methods. The Fuzzy logic can also be used for better accurate if one can select the best membership function.

Table 1: Comparison of four techniques

TECHNIQUE	MMRE
ANN	0.46
FUZZY LOGIC	0.26 – 0.16
FNN	0.21 – 0.24
CBR	0.40 – 0.50

5. CONCLUSION

Through this paper a comparison of four different machine learning techniques for effort estimation was done. The fuzzy logic method which resembles the human beings inference process, efficiently handles the uncertainty of the effort drivers during the initial stages. Artificial Neural Network is able to learn the past experiences through the learning mechanism of the net. The Case Based Reasoning method stores the historical data in the form of cases in a knowledge base. The knowledge base can be easily expanded. The fourth technique is the Fuzzy Neural Network which is a combination of the fuzzy logic and the artificial neural network. The accuracy of estimation is improved by the use of artificial neural network to assign weights for the fuzzy rules based on their significance and fuzzy logic for handling the uncertainty of effort drivers. A comparison of the MMRE value of these techniques when applied to the COCOMO dataset shows that Fuzzy Neural Network has the lowest MMRE values and hence is the better technique among the four. However, if appropriate membership functions are used Fuzzy logic can also give a low MMRE value.

6. REFERENCES

[1] Juan J. Cuadrado-Gallegoa, b, Pablo Rodríguez-Soriaa, Borja Martín-Herreraa, "Analogies and differences between Machine Learning and Expert based Software Project Effort Estimation", IEEE, 2010.

[2] W.B. Boehm, Chris. A. Abts, "Winsor brown, Sunita. Chulani," Bradford k. Clark, Ellis. Horowitz, Ray. Madachy, Donald J. Reifer and Bert. Steece. Software Cost Estimation with COCOMO II. Englewood Cliffs, NJ, USA: Prentice-Hall.2007.

[3] Helmer, "The use of the technique Delphi in problems of educational innovation. The RAND Corporation, 1966.

[4] M. Ruchika and J. Ankita, "Software Effort Prediction using Statistical Machine Learning Methods," (IJACSA) International Journal of Advanced Computer Science and Applications, vol. 2,no.1, 2011.

[5] K.Srinivasan and D. Fisher, "Machine Learning Approaches to Estimating Software Development Eddort", IEEE Transactions on Software Engineering, vol.21, Feb.1995.

[6] E. Cox, "Fuzzy Fundamentals", IEEE Spectrum, October 1992, pp. 58-61.

[7] Iman Attarzadeh and Siew Hock Ow, "A novel algorithmic cost estimation model based on soft computing technique," Journal of Computer Science 6 (2): 117-125, 2010.

[8] Iman Attarzadeh, Amin Mehranzadeh, Ali Barati, "Proposing an Enhanced Artificial Neural Network Prediction Model to Improve the Accuracy in Software Effort Estimation", IEEE, 2012.

[9] Aarmodt, A. and E. Plaza, 'Case-based reasoning: foundational issues, methodical variations and system approaches', AI Communications, 1994.

[10] Sarah Jane Delany and Padraig Cunningham, "The Application of Case-Based Reasoning to Early Software Project Cost Estimation and Risk Assesment", TCD-CS-200-10.

[11] X. Huang, J. Ren and L.F. Capretz. A Neuro-Fuzzy Tool for Software Estimation. Proceedings of the 20th IEEE International Conference on Software Maintenance, p. 520 2004

[12] Sun-Jen Huang, Nan-Hsing Chiu, "Applying fuzzy neural network to estimate software development effort", Springer, 2007.

[13] J.Kaur, S. Singh, K. S, Kahlon, and P. Bassi, "Neural Network-A Novel Technique for Software Effort estimation," *International Journal of Computer Theory and Engineering*, vol. 2, 2010.