Back Propagation Algorithm Based Model for Software Cost Estimation
Sheena Goyal, Sheilly Padda
Department of Computer Science and Engineering, Chandigarh Group of Colleges, Landran, Mohali, Punjab, India

DOI: 10.23956/ijarcsse/V7I6/0278

Abstract: Software Cost Estimation is a tough task as it includes various variations in every step of implementation which creates problems in measuring the KLOC (Line of Source Code in thousands) for cost estimation. Software industry becomes very vast from decades but software cost estimation is still a big problem for which software industry is still suffering. Our proposed model is for tuning parameters of COCOMO model software cost estimation using Particle Swarm Optimization (PSO). We will be using clustering methods to divide the data items into number of clusters. Once the data has been divided, it will be easier to implement Particle swarm optimization on each cluster. PSO is used for tuning the parameters of each cluster. Tuning stands for by continuously processing (here processing is applying PSO) in the data until the best value is found. The clusters and the tuned parameters will be trained on Neural Network by back propagation algorithm. Back propagation is done by calculating the equation once again using the better values found in our approach.

Keywords: COCOMO, Software Cost Estimation, Particle Swarm Optimization, KLOC, BPN- back propagation network, EAF- Effort Adjustment Factor.

I. INTRODUCTION
Cost estimation is typically measured regarding effort. For the software industry, correct cost estimation and avoiding loss of money due to the wrong cost estimation. Estimated judgment of the expenses on the resources and the effort required allocating the resources for an undertaking leads to calculate software cost estimation. The effort is the measure of time for one individual to work for a certain time on the project. The most widely recognized metric used is Person Months (PM’s) for an individual. It is essential to record the attributes of every individual in the advancement environment into light when contrasting the effort of two or more individuals undertaking in light of the fact that no two situations that are improved are same. Cost estimation can have a huge effect on the life cycle and calendar for a task. Cost estimation can likewise have a vital impact on resource allotment. It is essential for an software industry to distribute better resources like more experienced staff for improved activities.

II. BACKGROUND
This section includes the discussion of COCOMO model, Particle Swarm Optimization and Back propagation technique for Software Cost Estimation.

A. Constructive Cost Model (COCOMO)
COCOMO stands for Constructive Cost Model, it is a software cost estimation model that was first published in 1981 by Barry Boehm. It is an algorithmic approach to estimating the cost of a software project. By using COCOMO you can calculate the amount of effort and the time schedule for projects. From these calculations you can then find out how much staffing is required to complete a project on time. Effort can be calculated by the following equation:

Effort = a*(size)^b

Where a and b are the values depending upon the complexity of project.

B. Particle Swarm Optimization (PSO)
PSO is a stochastic global optimization method which is based on simulation of social behavior. As in GA and ES, PSO exploits a population of potential solutions to probe the search space. In contrast to the aforementioned methods in PSO no operators inspired by natural evolution are applied to extract a new generation of candidate solutions. Instead of mutation PSO relies on the exchange of information between individuals, called particles, of the population, called swarm. In effect, each particle adjusts its trajectory towards its own previous best position, and towards the best previous position attained by any member of its neighborhood. In the global variant of PSO, the whole swarm is considered as the neighborhood. Thus, global sharing of information takes place and particles profit from the discoveries and previous experience of all other companions during the search for promising regions of the landscape. To visualize the operation of the method consider the case of the single objective minimization case; promising regions in this case possess lower function values compared to others, visited previously [3].

Algorithm for PSO can be stated as:
PSO is used for optimization problem. Each single solution in the search space is a bird. We call it particle.
All the particles have
1. Fitness values, evaluated by fitness function
2. Velocities which direct the flying of the object

Particles fly through the problem space by following current optimization particles. PSO is initialized with a group of random particles (solutions) and searches for optima by updating generations. In each iteration, each particle is updated by following two “best” values. The first solution is the best solution it has achieved so far. The value is called “pbest”. Another best value that is tracked by PSO is the best value obtained by any particle in the population. This best value is a global best and called gbest. When a particle takes part of the population as its topological neighbors, the best value is a local best and is called lbest. After finding the two best values, the particle updates its velocity and positions with following equation (a) and (b).

\[ v_i = v_i + c_1 \times \text{rand}() \times (p_{best} - \text{present}) + c_2 \times \text{rand}() \times (g_{best} - \text{present}) \]  

(a) 

\[ \text{present} = \text{present} + v_i \]  

(b)

\( v_i \) is the particle velocity, \( \text{present} \) is the current particle (solution). \( p_{best} \) and \( g_{best} \) are defined as stated before, \( \text{rand}() \) is a random number between (0,1). \( c_1, c_2 \) are learning factors. Usually \( c_1 = c_2 = 2 \).

C. Back propagation Algorithm

The back propagation learning algorithm is one of the most widely used methods in neural network. The network associated with back-propagation learning algorithm is called as back propagation network. While training a network a set of input-output pair is provided the algorithm provides a procedure for changing the weight in BPN that helps to classify the input output pair correctly. Gradient descent method of weight updating is used.

![Architecture of a Back propagation Network](image)

The aim of the neural network is to train the network to achieve a balance between the ability of net to respond and its ability to give reasonable responses to the input that is similar but not identical to the one that is used in training. Back propagation algorithm differs from the other algorithm by the method of weight calculation during learning. The drawback of Back propagation algorithm is that if the hidden layer increases the network become too complex [11].

III. PROPOSED METHODOLOGY

A. Methodology

a) Divide the data items into clusters using k-means clustering.

K-means clustering follows the following steps:
Suppose that we have n sample feature vectors \( x_1, x_2, ..., x_n \) all from the same class, and we know that they fall into k compact clusters, \( k < n \). Let \( m_i \) be the mean of the vectors in cluster i. If the clusters are well separated, we can use a minimum-distance classifier to separate them. That is, we can say that \( x \) is in cluster \( i \) if \( \| x - m_i \| \) is the minimum of all the k distances. This suggests the following procedure for finding the k means:

- Make initial guesses for the means \( m_1, m_2, ..., m_k \)
- Until there are no changes in any mean
  - Use the estimated means to classify the samples into clusters
  - For i from 1 to k
    - Replace \( m_i \) with the mean of all of the samples for cluster i
  - end_for
- end_until

b) Use PSO for parameter tuning of each cluster.

c) Use back propagation algorithm to train the tuned parameters. Algorithm for back propagation follows the following steps:
initialize system weights do for each preparing sample ex forecast = neural-net-output(network, ex) genuine = teacher-output(ex) figure slip (expectation - real) at the yield units figure for all weights from concealed layer to yield layer figure for all weights from information layer to shrouded layer upgrade system weights until all samples grouped accurately or an alternate halting rule fulfilled give back where its due
d) Compare results.
B. Proposed Model
We considered COCOMO model for tuning parameters. The proposed model is
\[ \text{Effort} = a^{*}(\text{Size})^{b} + \text{EAF} + c \]
Where a, b are the cost parameters and c is a bias factor. For tuning the parameters, above given methodology is used for software cost estimation.

IV. IMPLEMENTATION
This section describes the experimental part of work. In order to conduct the study and for affectingly establish the model, a dataset from COCOMO model with 21 projects are used. In this implementation, we are tuning a, b and c parameters using above given methodology in “MATLAB” language. The performance measures considered with equations (a) and (b). Using this method, we have obtained following figures:

Figure 2: Shows the comparison between original measured effort and effort measured by tuning a parameter.

Figure 3: Shows the comparison between original measured effort and effort measured by tuning b parameter.
V. RESULTS AND DISCUSSION

We have considered the dataset in which size is less than 25 KLOC which means small projects are considered. In our project, variations between original and the outcome of our proposed model can be seen clearly which shows less linearity effort estimation using particle swarm optimization and back propagation algorithm. Cost estimation is typically measured regarding effort which leads to some linearity in software cost estimation.

VI. CONCLUSION

In our proposed model, we are trying to improve the previous model by removing the non-linearity of the data items which came while using COCOMO model. We are using clustering techniques as well as back propagation neural network algorithm as drivers for the improvement. Results are shown above.

REFERENCES


