A FEATURE SELECTION BASED RELIEF ALGORITHM WITH FUZZY LOGIC FOR SOFTWARE EFFORT ESTIMATION

1Ravneet Preet Singh Bedi, 2Amardeep Singh
1,2Department of Computer Engineering, Punjabi University, Patiala – 147002, Punjab, India; 1bedirps2000@yahoo.com, 2amardeep_dhiman@yahoo.com

Abstract - Software effort estimation is a vital factor in any product industry. As programming gets developed in size and intricacy, it is extremely hard to precisely anticipate the cost of programming advancement. This was the difficulty in past years. The best entanglement of programming industry was the quick changing nature of programming advancement, which has made it hard to create parametric models that output high precision for programming improvement in all areas. This paper, proposes a novel technique to estimate software effort based on fuzzy logic (FL) along with relief algorithm. Relief algorithm is used to extract features. Mean Square Error and Accuracy are used as parameters to evaluate the results. Proposed technique is compared with various existing algorithms of software effort estimation and experimental results demonstrate that the proposed technique gives less error and hence provides better accuracy than the other existing techniques.

Keywords—Software effort estimation, Fuzzy logic, Relief algorithm, Fuzzy rules, Software development.

I. INTRODUCTION

Software effort estimation is the process of estimating the effort required to build software. Accurate effort estimation is the most vital factor in software industries. As overestimating the effort may threaten the consumers and underestimating the effort can cause breakdown of project. To overcome these issues, apart from the judgement of humans, researchers attempt to develop various approaches to accurately estimating software effort. These approaches can be categorized into two types:

- Algorithmic Model: These models consist of any mathematical equations. This kind of model is used when we have enough dataset for training of a model.
- Predicting Model: This type of model is used when we don’t have enough dataset for training of an algorithmic model, i.e., when we have sparse dataset for training then we use prediction model.

The steps for software effort estimation are:

Planning of software project includes cost estimation, size of product, required resources, required staff, and milestones. Below are the steps given to create estimate of the software. Greater accuracy is achieved by introducing this phase early in the SDLC process. This phase also helps developers to monitor the cost of the project and to schedule the factors influencing risk.

- Gather and Analyze Software Functional and Programmatic Requirements.
- Define the Work Elements and Procurements.
- Estimate Software Size.
- Estimate Software Effort.
- Schedule the Effort.
- Calculate the Cost.

II. LITERATURE SURVEY

Sadiq et al.[1] created two diverse linear regression models by utilizing fuzzy function point and non-fuzzy function point aiming to forecast the effort estimation of software. Authors also considers that whole project is organic by nature i.e., size of project is among 2 to 50 KLOC. Project manager can able to manage the cost and also ensured that quality is managed accurately after effort of software is obtained.

Nisar et al.[2] displayed an overview on Software Development Effort Estimation Using Fuzzy Logic. The point of this study is to break down the utilization of Fuzzy logic in the current models and to give in depth audit of programming and venture estimation systems existing in industry and writing, its qualities and shortcomings.

Martín et al.[3] portrayed an application whose outcomes are compared and of a multiple regression. A subset of 41 modules created from ten projects is utilized as information. Result demonstrates that the estimation of MMRE (a combination of Magnitude of Relative Error, MRE) applying fuzzy logic was somewhat higher than MMRE applying various regression; while the estimation of Pred(20) applying fuzzy logic was marginally higher than Pred(20) applying multiple regression. Additionally, six of 41 MRE were equivalent to zero (with no deviation) when fuzzy logic was connected (no comparative case was exhibited when multiple regression was connected).

Kushwaha et al.[4] proposed software cost estimation display on the basis of fuzzy logic. The fuzzy logic demonstrates fuzzifies the two sections of the COCOMO display i.e. normal exertion expectation and the exertion alteration factor. The investigation demonstrates that the execution of the FIS improved by expanding the quantity of
enrollment methods. Approval test was done on NASA 93 and COCOMO98 open database.

Reddy et al.[5] implemented programming development exertion forecast utilizing Fuzzy Triangular Membership Function and GBEll Membership Function and contrasted with COCOMO. A contextual analysis in light of the/ASA93 dataset contrasts the proposed fuzzy model and the Intermediate COCOMO. The outcomes were verified utilizing diverse methods like VAF, MARE, VARE, MMRE, Prediction and Mean BRE. It is verified that the Fuzzy Logic Model utilizing Triangular Membership Function gave preferred outcomes over alternate models.

Kumar et al.[6] proposed a new model utilizing fuzzy logic with a specific end goal to assess the most essential variables of programming exertion estimation, for example, cost and time. Developers utilize MATLAB to decide the parameters of different cost estimation models. The execution of model is assessed on distributed programming project information. Examination of results from this model with existing models is appeared.

Verma et al.[7] expanded intermediate COCOMO in the proposed system by joining the idea of fuzziness into the estimations of size, method of improvement for ventures and the cost drivers adding to the general advancement exertion. The presented structure endures imprecision, fuses specialists learning, clarifies forecast method of reasoning through standards, offers straightforwardness in the expectation framework, and could adjust to changing situations with the accessibility of new information.

Sheta et al.[8] presented two new models for programming exertion estimation utilizing fuzzy logic. One model is created in view of the acclaimed Constructive Cost Model (COCOMO) and uses the Source Line of Code (SLOC) as info variable to gauge the Effort (E); while the second model use the Inputs, Outputs, Files, and User Inquiries to assess the Function Point (FP). The proposed fuzzy models demonstrate better estimation capacities contrasted with other detailed models in the writing and better help the venture administrator in processing the product required improvement exertion. The approval comes about are completed utilizing Albrecht informational index.

Malathi et al.[9] built up another way to deal with evaluation of programming exertion for or numerical information utilizing fuzzy approach. The current verifiable datasets, examined with fuzzy logic, deliver exact outcomes in contrast with the dataset provided with the existing systems.

Yadav et al.[10] reviewed the most well-known and broadly utilized exertion estimation methods utilizing fuzzy logic. The study demonstrates that fuzzy logic exertion estimation can be combined with different procedures, for example, neural system, Bayesian Network and Particle Swarm Optimization method.

III. PROPOSED TECHNIQUE
Soft computing is an area of research that deals with real life problems in a more effective way, thus providing more accurate results. This proposed work is based on using Fuzzy Logic (FL) based technique to predict efforts to be spent on a given software development project. Figure 1 shows the proposed model used for estimation based on FL. The fuzzy inference system that is proposed in this research work is based on Mamdani system. The model requires five input parameters viz. Complexity, Data, Tool, loc (lines of code) and skills. The choice of these five input parameters is inspired by the thought that there exist some unnecessary and skills. The choice of these five input parameters is inspired by the thought that there exist some unnecessary

<table>
<thead>
<tr>
<th>PROPOSED</th>
<th>TOOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple, Less, Medium, High, Very High</td>
<td></td>
</tr>
<tr>
<td>Free, Low, Average, High</td>
<td></td>
</tr>
<tr>
<td>Very Low, Low, Medium, High, Very High</td>
<td></td>
</tr>
<tr>
<td>Bare, Average, Very High</td>
<td></td>
</tr>
<tr>
<td>Novice, Average, Good, Expert</td>
<td></td>
</tr>
<tr>
<td>Low, Medium, High</td>
<td></td>
</tr>
</tbody>
</table>
The above model is capable of utilizing all three input factors and apply pre-defined fuzzy rule base to get an accurate prediction of software efforts. The results thus produced are compared with COCOMO II. COCOMO I & II lack the precision due to the reason that these models do not consider all input parameters especially the COCOMO I. The trouble with COCOMO II is that when applied to the records from Promise dataset, it tends to misinterpret both over as well as under. Whereas, the proposed model when applied to the same dataset produces results that are very much aligned with the actual results given with the records.

The input parameters as used in the model follow suitable membership functions in the corresponding Matlab implementation. Each linguistic variable for each input parameter follows the same type of membership function.

(i) **Complexity**: follows Gaussian membership function curve. The Gaussian function, named after Carl Friedrich Gauss, is a continuous function which approximates the exact binomial function.

\[
    f(x; \sigma, c) = e^{-\frac{(x-c)^2}{2\sigma^2}}
\]

Equation shows the Gaussian function for the above used membership function, where \(\sigma\) is the standard deviation and \(c\) is the position of the centre of the peak.

(ii) The membership functions used for **data** are triangular membership functions. The curve of these membership functions look like triangle. These also have beginning, ending and one sharp peak.

\[
    f(x; b, c) = \begin{cases} 
    0, & x \leq a \\
    \frac{x - a}{b - a}, & a \leq x \leq b \\
    \frac{c - x}{c - b}, & b \leq x \leq c \\
    0, & c \leq x 
    \end{cases}
\]

Equation shows a sample triangular function in Matlab. In this function, \(a\) and \(c\) represent the feet of the curve and \(b\) marks the position of the centre of the peak.

(iii) The membership function for **Tool** parameter uses generalized bell-shaped membership function. The special aspect about this function is that in this case, the function plot curve is flat from top. It looks like a plateau.

\[
f(x; a, b, c) = \frac{1}{1 + \left| \frac{x-c}{a} \right|^{2b}}
\]

Equation shows the function definition of generalized bell-shape function. In this case, \(c\) marks the position of the centre of the peak, whereas, \(b\) is usually positive.

(iv) Input parameter **Loc** behaves on the pattern of difference between two sigmoidal functions. Sigmoidal function is a mathematical function having “S” shaped curve and disigf chooses the difference between two such sigmoidal functions and plots a curve for the same.

\[
S(t) = \frac{1}{1 + e^{-\tau}}
\]

Equation shows a sigmoidal function.

\[
f(x; a, c) = \frac{1}{1 + e^{-a(x-c)}}
\]

(v) The last input parameter, namely **Skills** uses membership function exactly same as that of “Data”. The membership function is named triangular membership function. The membership function used Estimated is once again triangular membership function. It is written as trinmf in Matlab.

The fuzzy rules are defined as follows:

1. If (Complexity is Simple) and (Data is Free) and (Tool is low) and (Loc is Bare) and (Skills is Avg) then (Estimated is low) (1)
2. If (Complexity is Less) and (Data is Low) and (Tool is Medium) and (Loc is Average) and (Skills is Good) then (Estimated is High) (1)
3. If (Complexity is Medium) and (Data is Average) and (Tool is High) and (Loc is VeryHigh) and (Skills is Expert) then (Estimated is High) (1)
4. If (Complexity is High) and (Data is High) and (Tool is VeryHigh) and (Loc is Average) and (Skills is Good) then (Estimated is High) (1)

IV. EXPERIMENTAL RESULTS

This section shows the results of the proposed technique. Mean Square Error (MSE) and Accuracy are used to evaluate the results.

**Parameters Evaluation**

- **Mean Square Error**: Mean square error (MSE), also known as mean square deviation calculates the square of average errors, i.e., the deviation among the estimator and that is estimated.
The results shown above the error rate evaluated using fuzzy inference system. Then the error rate is evaluated which comes out to be 0.4721. Comparing this error rate with the existing classification techniques explained in the previous year report, the error rate of fuzzy inference system is reduced as in the existing techniques the error rate was 3.7087 for Linear Regression Classifier Results, 5.4907 for Multilayer Perceptron (Neural Network), 3.5045 for Bagging Classifier and 6.1516 for Decision Tree Classifier.

| Table 3: Comparison of Accuracy of Various Techniques |
|-------------------------|-------------------------|
| Technique               | Error Rate              |
| Fuzzy Inference System  | 99.5279                 |
| Linear Regression Classifier | 96.2913               |
| Multilayer Perceptron   | 94.5093                 |
| Bagging Classifier      | 96.4955                 |
| Decision Tree Classifier| 93.8484                 |

The above table shows comparison of accuracy of the proposed technique with other techniques. Comparing the accuracy with the existing classification techniques explained in the previous year report, the accuracy of fuzzy inference system is better as in the existing techniques the accuracy was 96.2913 for Linear Regression Classifier Results, 94.5093 for Multilayer Perceptron (Neural Network), 96.4955 for Bagging Classifier and 93.8484 for Decision Tree Classifier.

V. CONCLUSION

The success of a software depends upon the accurate and precise estimation of the software effort before developing the software. Estimation of the cost is most difficult task in the software industry. Various models for software effort estimation are developed in past. This research paper introduces a technique based on fuzzy logic and relief algorithm to estimate software development effort. Experimental results demonstrate that the proposed technique outperforms the existing techniques for various parameters.

REFERENCES


