# Artificial Intelligence based smart Effort Estimation and Model Development

Syed Mohd Shahabuddin
Computer Science and Engineering
A.I.E.T., Lucknow (U.P.), India

Dr. Shafeeq Anmad
Computer Science and Engineering
A.I.E.T., Lucknow (U.P.), India

Manmohan
Computer Science and Engineering
A.I.E.T., Lucknow (U.P.), India

**Abstract-- In the present work artificial intelligence techniques like Artificial Neural Network (ANN), Adaptive Neuro Fuzzy Inference System (ANFIS), along with optimization technique like Simulated Annealing has been used for optimum model development for prediction of Development Effort Estimation using NASA project dataset, consisting of 63 software projects and further model assessment using Object Oriented software development approach. Next, a comparison of results based on SA approach with ANN, ANFIS and COCOMO models has been presented proving the estimation utility.**

## 1. Introduction:

Software has turn out to be significant to advancement in almost all areas of human life. The skill of programming only is no longer enough to make large programs. There are grave effort in the cost, timeliness, maintenance and quality of many software products. Software engineering has the aim of solving these issues by producing a good quality, maintainable software on time within budget. To achieve this objective, we have to focus in a disciplined manner on both the quality of the product and on the process used to develop the product. At the first conference on software engineering 1968, software engineering was defined as "The establishment and use of sound engineering principles in order to obtain economically developed software that is reliable and work efficiently on real machine" [4]. It is also define as "A discipline whose aim is the production of quality software, software that is delivered on time ,within budget and that satisfies its requirements". [5]. Both the definitions are popular and acceptable to majority. However due to increase in cost of maintaining software ,objective is now shifting to produce quality software that is maintainable ,delivered on time ,within budget, and also satisfies its requirements.

Software engineering [3] is concerned with all phases of software system production from the first stages of system specification through to maintaining the system when it's gone into use. As a discipline, software engineering has progressed very far in very short duration of your time, significantly compared to classical engineering field (like civil or electrical engineering). within the period of time of computing, not way more than fifty years ago, computerized systems were quite limited. Most of the programming was done by scientists making an attempt to resolve specific, comparatively small mathematical issues. Errors in those systems typically had solely "annoying" consequences to the mathematician who was making an attempt to find "the answer." now a days we regularly build monstrous systems, in terms of size and complexity. what's additionally notable is that the progression within the past fifty years of the visibility of the software from principally scientists and software developers to the general public of all ages. "Today, software system is functioning both explicitly and behind the scenes in almost all aspects of our lives, together with the vital systems that have an effect on our health and well-being." (Pfleeger 1998). Despite our speedy progress, the software industry is taken into account by several to be in an exceedingly crisis. Some forty years ago, the term "Software Crisis" emerged to describe the software industry's inability to provide customers with high quality products on schedule. "The average software development project overshoots its schedule by half; larger projects typically do worse. And, some three quarters of all big systems are "operating failures" that either don't perform as intended or don't seem to be used at all." (Gibbs, 1994) whereas the industry will celebrate that software touches nearly all aspects of our daily lives, we will all relate to software availability dates (such as computer games) as moving targets and to computers crashing or locking up. we've got several challenges we'd like to deal with as we tend to still progress into a a lot of mature engineering field, one that predictably produces high-quality softwares. The "systematic, disciplined, quantitative approach" is usually termed a software process model (in the general sense) or a software development method (in the precise sense). Specific software development processes contains a particular set of software development practices that are often performed by the software engineer in an exceedingly preset order

## 2. Related Work:

**Adanma C. Eberendu, (2014),** gave an overview of software cost estimation and answered the following questions: (1) what have people already done in area of cost estimation; (2) what are the types of software cost estimation model available; (3) how is the contingency allowance accounted for in software projects? Since none of the existing software cost estimation model can work accurately on its own, they proposed a hybrid model to bridge the gap and arrive at accurate and acceptable estimates.

**Shiyna Kumar, Vinay Chopra, (2013),** presented an overview of the different techniques currently available for software effort estimation in the software industry. Software effort estimation is a incredibly essential task in the software engineering field because the future of the project depends on the estimation report. The techniques discussed about algorithmic model, non algorithmic model and some soft computing technique. Finally it was concluded that it is not only the metrics which can be responsible for accurate estimation, but also it is how and when they are being used. So, one cannot say a specific technique is best fit for all the situations to give an accurate estimation.

**Prabhakar et. al., (2013),** they used Artificial Neural Network (ANN), and Support Vector Machine (SVM) learning techniques to analyze the results using China dataset for predicting software development effort. A similar study can be carried out to predict software effort using prediction models based on other machine learning algorithms such as Genetic Algorithms (GA) and Random Forest (RF) techniques. Cost benefit analysis of models may be carried out to determine whether a given effort prediction model would be economically viable.

**Srinivasa Rao T, et. al. (2013),** proposed software cost estimation based on PSO technique. Neural network technique was used to train the network for classification of values during testing phase. The porposed model could be applied to large datasets. It was also seen that the model could be very effective if it contained similar genres. However it was seen that PSO is a probabilistic model which could not generate exact values. But based on enough historical datasets the accuracy could be increased.

**Syed Ali Abbas, et. al. (2012),** dealt with a detailed discussion about the models that were presented earlier in various studies. It contains many parametric and non parametric modeling techniques. Thus it clearly gives a detailed analysis of the pros and cons, similarities and differences amongst the models. From the analysis it was concluded that any approach based on rigid mathematical formula can not alone solve the issue of effort estimation. However, expert based techniques be of great help in this case and can be validated by the application of some parametric models and soft computing techniques.

## 3. Methodology:

As discussed earlier, FIS and ANN leads to the integration into ANFIS, which is a high level thinking tool, having learning capability [1, 11]. In the present work subtractive clustering has been used as a rule extraction methods for FIS identification. For this MATLAB toolbox has been used [12]. Here the initial parameters of the ANFIS are identified using the subtractive clustering method. The clustering radius is the most important parameter in the subtractive clustering algorithm and is optimally determined through a trial and error procedure. In the present case for each fuzzy set in the fuzzy system, Gaussian membership function has been used. Their numbers and fuzzy rules needed for the ANFIS development are determined through subtractive clustering rule extraction method. Here hybrid learning algorithm is used for determining the parameters of Gaussian membership function. The parameters used in the model for training ANFIS are given in Table 1 and the rule extraction method used are given in Table 2. Tables 3 summarizes the results of types and values of model parameters used for training ANFIS.

**Table 1: Parameters used in all the models for training ANFIS**

| | |
|---|---|
| Rule extraction method used | Subtractive clustering |
| Input MF type | Gaussian membership (‘gaussmf’) |
| Input partitioning | Variable |
| Output MF Type | Linear |
| Number of output MFs | One |
| Training algorithm | Hybrid learning |
| Training epoch number | 10 |
| Initial step size | 0.01 |

**Table 2: Rule extraction method used for training ANFIS**

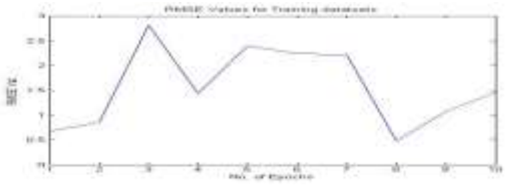
| Rule Extraction Method | Type |
|---|---|
| And method | ‘prod’ |
| Or method | ‘probor’ |
| Defuzzy method | ‘wtever’ |
| Implication method | ‘prod’ |
| Aggregation method | ‘max’ |

**Table 3 Values of parameters used for training ANFIS**

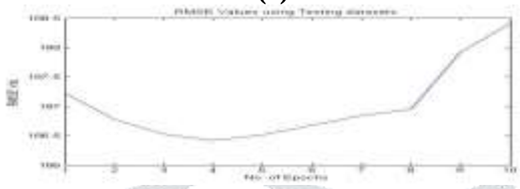
| No. of nodes | No. of linear parameters | No. of non-linear parameters | Total no. of prameters | No. of Training data pairs | No. of testing data pairs | No. of fuzzy rules |
|---|---|---|---|---|---|---|
| 1311 | 646 | 1216 | 1862 | 40 | 23 | 38 |

## 4. Results and discussions

In the present work the model so developed has been tested by ANFIS method. The performance of the best prediction model developed has been measured using performance measuring criteria, like RMSE, which are further compared both for training and testing datasets separately [45]. A separate plot for both training and testing datasets using RMSE criteria has been carried out, as shown below in Fig. 1 below and the corresponding range of values (minimum and maximum) are summarized in **table 4**.



**(a)**



**(b)**

**Fig. 1: Graphical plot of RMSE value variation during ANFIS training and testing of datasets**

**Table 4. Range of RMSE Val. during training and testing phase**

| | RMSE Value | |
|---|---|---|
| | Minimum | Maximum |
| **Training datasets** | 0.4824 | 2.8096 |
| **Testing datasets** | 186.41 | 188.41 |

Further Table 5 gives the RMSE values using both the COCOMO and ANFIS techniques.

**Table 5 Performance evaluation using RMSE criteria**

| | Using COCOMO | Using ANFIS |
|---|---|---|
| RMSE Val. | 532.2147 | 112.638 |

From the above analysis it can be inferred that ANFIS has performed better during training phase than testing phase but its overall RMSE value is 112.638. which shows a marked improvement than those calculated in COCOMO model i.e. 532.2147 ( given above in table 5). Thus, it is clear that proper selection of influential radius which affects the cluster results directly in ANFIS using substractive clustering rule extraction method , has resulted in reduction of RMSE and MAE both for training and testing data sets. Hence, it is seen that for small size training data, ANFIS has outperformed COCOMO model. In order to depict how well ANFIS has performed over COCOMO model, a comparative plot of actual effort versus predicted effort, both by COCOMO and ANFIS technique, has been shown in Fig. 4.3. From the graph it is seen that ANFIS model line almost closely follows the actual effort line than those of COCOMO. This again depicts the superiority of ANFIS technique over COCOMO model for effort estimation.

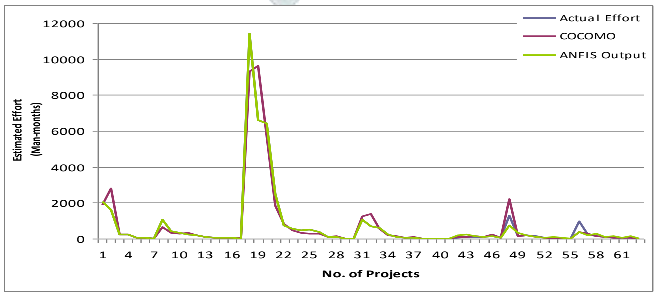
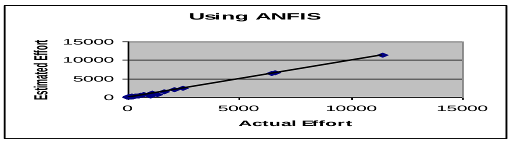
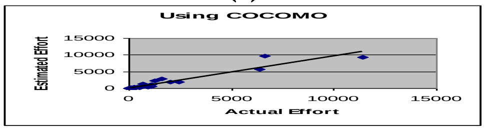


**Fig.2: Comparative plot of Actual Effort, COCOMO Effort and ANFIS Output**

Finally, **Figure 3(a) &(b)** shows the scatter plot of Actual Effort versus Estimated Effort using ANFIS and COCOMO models. The figures wisely demonstrate that (1) the model performance is in general accurate in case of ANFIS, where all data points roughly fall onto the line of agreement; (2) model using ANFIS is consistently superior to COCOMO



(a)



(b)

**Fig 3: Scatter Plot of Actual Effort Vs. ANFIS & Actual Effort Vs. COCOMO**

## 5. Conclusion

In the present study, subtractive clustering method has been used for identification of the initial parameters of ANFIS. From the analysis of the above results, given under heading Results and Discussions, it is seen that the Effort Estimation prediction model developed using ANFIS technique has been able to perform well over COCOMO Model.

## References:

[1] Adanma C. Eberendu, (2014), " Software Project Cost Estimation: Issues, Problems and Possible Solutions', IJESI, Volume 3 Issue 6, PP.38-43.
[2] Shiyna Kumar, Vinay Chopra,(2013), "A Review of Surveys on Estimating Software Development Effort", IJAIR Vol. 2 Issue 5, pp. 448-451.
[3] Srinivasa Rao T, et. al. (2013), "Predictive and Stochastic Approach for Software Effort Estimation", Int. J. of Software Engineering, IJSE Vol. 6 No. 1, pp. 97-115.
[4] Prabhakar et. al. , (2013), " Prediction of Software Effort Using Artificial Neural Network and Support Vector Machine", IJARSSE, Volume 3, Issue 3, pp. 40-46.
[5] Magne Jørgensen, et. al. (2012), "First Impressions in Software Development Effort Estimation:Easy to Create and Difficult to Neutralize", Proceedings of the EASE 2012 - Published by the IET, pp. 216-222.
[6] Syed Ali Abbas, et. al. (2012), " Cost Estimation: A Survey of Well- known Historic Cost Estimation Techniques", VOL. 3, NO. 4, April 2012 ISSN 2079-8407 Journal of Emerging Trends in Computing and Information Sciences, pp. 612-636.
[7] S.k. mohanty & A.k. bisoi, (2012), "Software Effort Estimation Approaches – A Review", International Journal of Internet Computing ISSN No: 2231 – 6965, VOL- 1, ISS- 3 2012, pp. 82-88.
[8] Sanjay Kumar Dubey, et.al., (2012), "Comparison Study and Review on Object- Oriented Metrics", Global Journal of Computer Science and Technology Volume 12 Issue 7 Version 1.0.
[9] Mohammad Saber Iraji, et.al., (2012), " Object Oriented Software Effort Estimate with Adaptive Neuro Fuzzy use Case Size Point (ANFUSP)", I.J. Intelligent Systems and Applications, 6, 14-24.
[10] Syed Ali Abbas, et. al. (2012), " Cost Estimation: A Survey of Well- known Historic Cost Estimation Techniques", Journal of Emerging Trends in Computing and Information Sciences, VOL. 3, NO. 4,, pp. 612-636.
[11] Tirimula Rao Benala, (2012), "Computational Intelligence in Software Cost Estimation: An Emerging Paradigm", ACM SIGSOFT Software Engineering Notes Page 1, Volume 37, Number 3, pp. 1-7.
[12] Carolyn Mair, (2011), "Human Judgement and Software Metrics: Vision for the Future", ICSE '11, ACM, pp. 81-84.
[13] Jovan Zivadinovic, et. al. (2011), "Methods Of Effort Estimation In Software Engineering", International Symposium Engineering Management And Competitiveness 2011 (EMC2011), pp.417-422.