An Effort Estimation Model for Software Development using Ensemble Learning

Abhishek Kumar
M.Tech Student,Department of CSE
Maharana Pratap College of Technology, Gwalior
(M.P.)

Unmukh Datta
H.O.D Department of Computer Science Engg.
Maharana Pratap College of Technology, Gwalior
(M.P.)

ABSTRACT
For a successful project development, it is important for any software organization that the project should be completed within time and budget, and the project should have requisite quality. This paper presents an Ensemble learning based Adaptive Neuro-Fuzzy Approach for Software Development Time Estimation. The concept behind this technique is based on ensemble learning methods. This technique combines multiple models into one model. The ensemble fits a new learner to the difference between the experiential response and the aggregated prediction of all learners which grown previously. In this paper, we describe a brief literature review of different techniques of software development time estimation along with a comparison of different techniques with our approach.

General Terms
Neuro Fuzzy, Software Time Estimation, Ensemble Learning

Keywords
Membership Function (MF), COCOMO, Adaptive Neuro Fuzzy Inference System (ANFIS), Neural Network, Fuzzy Logic, Prediction, MRE, MMRE, BRE, Development Time (DT),

1. INTRODUCTION
Accurate effort estimation is the state of art of software engineering, effort estimation of software is the preliminary phase between the client and the business enterprise. The relationship between the client and the business enterprise begins with the estimation of the software[1]. The overall software project can be defined by a number of attributes such as size and complexity. These attributes are the basic building block for effort estimation in software project. There are some multiplicative factors that determine the effort required to complete your software project such as cost drivers. There are different models for software development, so they define its own cost drivers such as Boehm's COCOMO defines many cost drivers based on product, computer, personnel, and project attributes and Albrecht's Function Point defines five main components system for function point analysis which is given in Fig 1. Alongside this enormous development, it is likewise understood the vitality of all these models in assessing the product advancement expenses and setting up the calendars for more rapidly and effortlessly in the foreseen situations[2]. Appraisals are completed at different phases of a product venture. Effort estimation is expecting more significance as a characteristic result of expanded outsourcing of programming advancement work[3]. In an outsourcing situation, effort estimation is required for the valuation of suppliers' recommendations. Potential builders considering an offer would need to examine the framework detail and produce assesses on which to base recommendations[4]. On account of in-house improvement, estimation is important to assess contending requests for the product and to allot the accessible assets to the most elevated work[5].

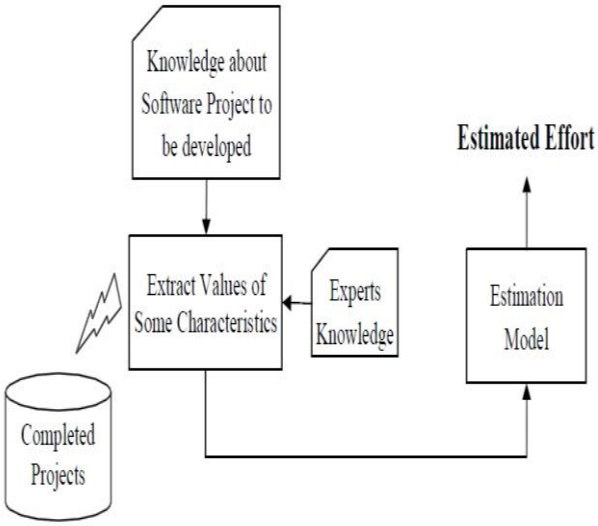

Figure.1 Software effort estimation process

Therefore, the outcome of software cost estimation can be listed as[6]:

- At every step estimation for effort is carried out during software project development.
- Budget for outsourced assignment is ready.
- Evaluation of different vendor's proposal is based on which an agreement with the selected vendor on the price to be paid is contracted
- A budget is allocated in the case of in-house development.
- A schedule for development process.

In this paper, we are going to accurately estimate the software effort with the help of Neuro Fuzzy approach and ensemble learning methods. A comparison with different types of neural network models based upon various parameters such as Magnitude of Relative Error (MRE), Mean Magnitude of Relative Error (MMRE), Balanced Relative Error (BRE) and Prediction (Pred).

2. RELATED WORK
Some of the models used for examining software effort estimation are:

(a) Algorithmic Models: From the examination of verifiable information, expenses are examined utilizing scientific formulae, connecting expenses or inputs with measurements

to deliver an expected yield[7]. These measurements are by and large qualities of the target framework and the usage environment called expense drivers. Different algorithmic forms; Linear models[8], Multiplicative models[9] and WalstonFelix[10].

(b) Expert Judgment: Expert judgment procedures include counselling, gathering of the specialists to utilize their experience and comprehension of the proposed undertaking project and evaluation of its cost[4]. This is requesting an assessment of assignment exertion from somebody who is proficient about either the application or the advancement environment. This strategy is frequently utilized when assessing the effort expected to change a current bit of programming. A standout amongst the most widely recognized techniques which work as indicated by this procedure is Delphi. Delphi orchestrates a particular meeting among the task specialists and tries to attain to the genuine data about the undertaking from their debates[5].

(c) Analogy Based: Relationship is characterized as "derivation that if two or more things concur with each other in a few regards, they will most likely concur in others". In software expense estimation approach a comparative, finished, task is recognized and its genuine exertion is utilized as the premise of the appraisal for the new venture[6]. It is infrequently utilized at the early phases of software advancement in light of such inalienable vulnerability and imprecision connected with property estimation. Similarity based thinking is regularly utilized in software effort estimation as an equivalent word for case based thinking (CBR) to portray the run of the mill case-based methodology where experience is held for future reference.

(d) Computational Intelligence Techniques: Computational knowledge is the investigation of versatile components to empower or encourage wise conduct in mind boggling and evolving situations. Thusly, computational insight joins false neural systems, transformative figuring, swarm Intelligence and fuzzy frameworks. Software cost estimation frameworks are extensive complex nonlinear stochastic systems[11]. Subsequently, it is elusive ideal gimmick weighting and undertaking determination in any cost estimation model. Computational Intelligence gives a plausible approach to acquire either ideal or imperfect arrangements. Computational Intelligence techniques can be adjusted to element changes in venture parameters. Software cost estimation moves can be made taking into account continuous datasets and verifiable thinking. Analysts have led a considerable measure of work for utilizations of computational insight in the field of software cost estimation.

3. NEURO FUZZY MODEL

A neuro fuzzy framework is a blend of neural system and fuzzy frameworks in such a path, to the point that neural system or neural system calculations are utilized to focus the parameters of the fuzzy framework. This implies that the primary proposition of neuro fuzzy methodology is to make or enhance a fuzzy framework naturally by a method for neural system strategies[12]. A much more vital viewpoint is that the framework ought to dependably be interpretable regarding fuzzy if-then rule, in light of the fact that it is in view of a fuzzy framework reflecting unclear information. The thought of a neuro fuzzy framework is to discover the parameters of a fuzzy framework by the method for taking in techniques got from neural network[13]. A typical way to apply a learning calculation to a fuzzy framework is to speak to it in an uncommon neural system like structural engineering. At that point a learning calculation, for example, back propagation is utilized to prepare the framework. Nonetheless, neural system learning calculations are generally inclination plummet strategies. This can't be connected straightforwardly to a fuzzy framework, in light of the fact that the capacities used to understand the induction methodology are normally not differentiable. Keeping in mind the end goal to understand the framework, we have to supplant the capacities utilized as a part of the fuzzy framework (like min and max) by differentiable capacities or don't utilize an inclination based neural learning calculation however a more qualified procedure[14]. Cutting edge neuro fuzzy frameworks are regularly spoken to as multilayer nourish forward neural system. A neuro fuzzy framework is a fuzzy framework that is prepared by learning calculation (normally) gotten from neural system theory[15]. The (heuristic) learning methodology works on neighbourhood data, and causes just nearby changes in the hidden fuzzy framework. The learning procedure is not information based, however information driven. Furthermore, a neural- fuzzy framework can simply be deciphered as an arrangement of fuzzy rules. It is conceivable both to make the framework out of preparing information starting with no outside help, and to instate it from earlier learning as fuzzy rules[16]. The learning methodology of a neural- fuzzy framework considers the semantic properties of the hidden fuzzy framework. It likewise approximates a n-dimensional (obscure) capacity that is in part given by the preparation information. The fuzzy guidelines encoded inside the framework speak to obscure specimens, and can be seen as unclear models of the preparation data[17].

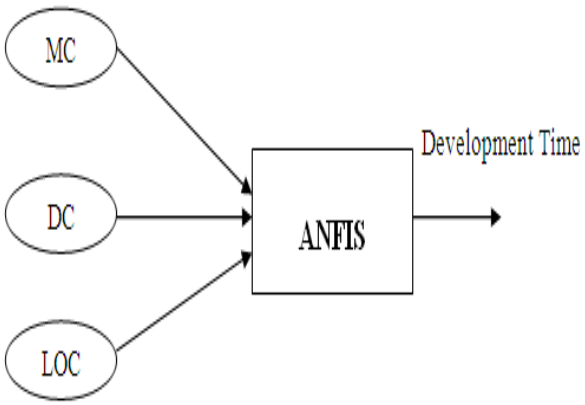

Figure.2 General Neuro-fuzzy models for software Development Time estimation.

4. ENSEMBLE LEARNING

Ensemble learning alludes to a gathering of routines that take in a target work via preparing various individual learners and consolidating their expectations. It utilizes numerous learning calculations to get preferred prescient execution over could be gotten from any of the constituent learning calculations[18]. In ensemble algorithms, bagging methods form structure a class of calculations which assemble a few occurrences of a discovery estimator on arbitrary subsets of the first preparing set and after that total their individual forecasts to structure a last expectation. These techniques are utilized as an approach to decrease the change of a base estimator (e.g., a choice tree), by bringing randomization into its development system and afterward making a troupe out of it. As a rule, sacking strategies constitute an exceptionally straightforward approach to enhance regarding a solitary model, without making it important to adjust the fundamental base

calculation. This gives an approach to decrease over fitting, stowing systems work best with solid and complex models (e.g., completely created choice trees), interestingly with boosting strategies which normally work best with powerless models (e.g., shallow choice trees).

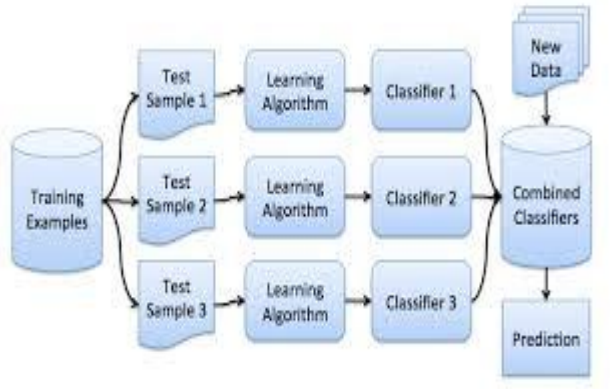

Figure.3 Bagging and Boosting

5. METHODOLOGY

The main idea behind ensemble learning is to combine multiple models into one. At every processed step, the ensemble fits a new learner to the difference between the experimental response and the aggregated prediction of all learners developed earlier. Ensemble fits to minimize mean-squared error during the process. As in ANFIS we are providing only one model, it was not possible to combine an arbitrary number of models into one. In this work we are using two general ensemble learning methods bagging and boosting. The fig.4 shows the proposed model for modeling the Bagging and LSBoost predictor.

The algorithm for the proposed work is as follows:

Step 1: Determine the all inputs of the model.

Step 2: Divide the data into two sets, one is Training data set, and test data set. The train data set contains 70% to 90% of all data and the left behind data are used for the test data set.

Step 3: Generate an ANFIS model by assigning the no. of membership functions and the type of membership function.

Step 4: Train this network by taking the value of epochs 100 and save this Fuzzy Inference System file (.fis).

Step 5: Generate the Bagging and Boosting model by:

i. Setting the Number of Ensemble Members.

ii. Preparing the Weak Learners.

iii. Fit the ensembles (Bagging and Boosting).

iv. Save the fitted ensembles.

Step 6: Calculate the value of Development Time with these saved models i.e ANFIS, Bagging and Boosting file and the testing data set.

Step 7: Calculate and Compare the Value of MMRE and PRED.

At the step of ensemble learning, bagging nodes first train a set of models with particular data values, then tests data is predicted with each of the models in the set and at last the voting node detects the majority class. After which the Boosting nodes i.e. Learner and Predictor will apply the LSBoost algorithm to the data set together with the corresponding model.

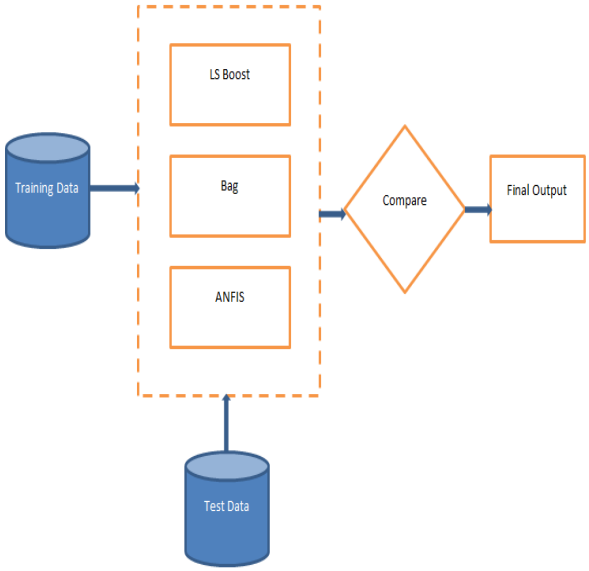

Figure.4 Final methodology

6. PERFORMANCE EVALUATION METRICS

The valuation consists of comparing the accuracy of the estimated effort with the actual effort. The following evaluation criterion has been used to assess and compare the performance of the proposed models.

MMRE: It is one of the common standards for the evaluation of cost estimation model i.e. magnitude of relative error (MRE), and mean magnitude of relative error (MMRE) .MRE is defined as follow:

$$MMRE = \frac{1}{n}\sum_{i=1}^{n}\frac{|Actual\ Time - Predicted\ Time|}{Actual\ Time} \times 100$$

PRED: Prediction (PRED) at level n is defined as the % of projects that have absolute relative error less than n. A model which gives higher PRED is better than that which gives lower PRED.

BRE: A model which gives lower BRE (Balanced Relative Error) is better than that which gives higher BRE.

$$BRE(\%) = \frac{|Estimated\ Time - Actual\ Time|}{Min(Estimated\ Time, Actual\ Time)} \times 100$$

7. RESULTS AND DISCUSSION

Our experiment consists in estimating the software development effort by using the neural networks approach on the Lopez-Martin et.al. Data set. They used the sets of system development projects, where the Development Time (DT), Dhama Coupling (DC), McCabe Complexity (MC) and the Lines of Code (LOC) metrics were used for 41 modules. The development time of each of the 41 modules were used including five phases: requirements understanding, algorithm design, coding, compiling and testing [6, 7]. Table I shows the different dataset used for carrying out experimentation. These models were trained with first 31 inputs from the standard dataset and later 10 inputs from the same dataset were used to

test the models. The effectiveness of neural network model over the regression analysis model has already been provided with the same dataset[8]. The simulation was done using the fuzzy logic toolbox, neural network toolbox and ANFIS toolbox in MATLAB.

Project No.	Actual DT	Estimate DT using FFBP NN	EstimateDT USING Cascaded FFBP NN	Estimated DT using Layer Recurrent NN	Estimated DT using Trapezoidal MF using Neuro fuzzy	GBell	Boosting	Bagging
31	19	9.04	9.54	9.49	19.0000	18.99988	18.95345	18.95345
32	13	9	9.84	9	13.0000	12.9999	12.91972	12.91972
33	12	9	21.91	9	12.0000	12.00052	12.10164	12.10164
34	12	9	9	9	12.0000	7.581409	12.23131	12.23131
35	21	21.98	22	9.21	21.0000	15.67964	21.42979	21.42979
36	21	21.99	22	18.89	21.0000	204.0081	26.37097	26.37097
37	19	21.97	21.87	9.17	19.0000	19	19.00911	19.00911
38	18	21.99	9.71	19.19	18.0000	18.00004	18.00497	18.00497
39	24	9.31	9	9.12	16.6995	24.5	24.51646	24.51646
40	25	9.31	9	9.12	16.6995	24.5	24.51646	24.51646
41	18	9	9	9	16.6684	17.99998	17.98736	17.98736

Table 1. Summarizes the comparison of various neural network models, Neuro fuzzy models with Ensemble learning model using parameters.

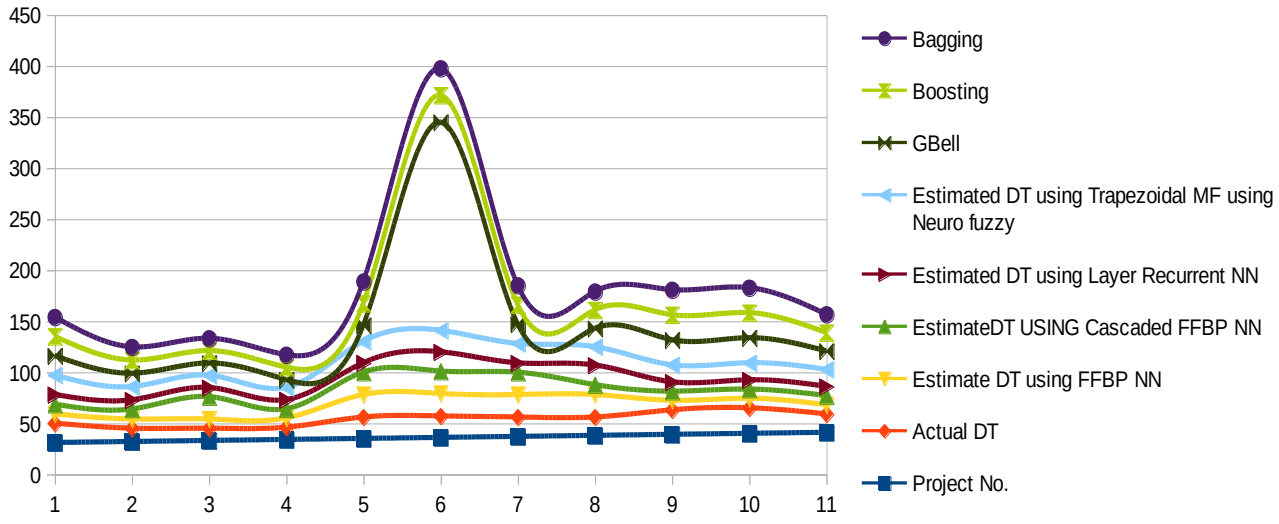

Figure.4 Comparison of various neural network models, Neuro fuzzy models with Ensemble learning model using parameters.

We obtained the Development Time (DT) for each of the trained neural network models with respect to other models and a comparative analysis was carried out which is shown in figure.4, based on the different standardized assessment criteria like MRE, MMRE, BRE and Pred(25).

8. 8. CONCLUSION

The paper proposed a new approach for estimating of software project development time using ensemble learning methods. In this paper, Adaptive Neuro Fuzzy Inference model is considered with bagging and boosting to predict the future values. It is observed that Neuro Fuzzy model using these function gives better results than all other models. It is also observed that ensemble learning techniques give better results for all the three parameters. In order to achieve more accurate estimation, the estimated values of several other techniques and combine their results may be useful. A comparison of different models for cost estimation is carried out.

9. ACKNOWLEDGMENTS

The authors are greatly indebted to the Department of Computer Science and Engineering,**,Maharana Pratap College of Technology, Gwalior (M.P.)**,for providing excellent lab facilities that make this work possible.

10. REFERENCES

[1] M. Chemuturi, “Software Estimation Best Practices, Tools & Techniques: A Complete Guide for Software Projects Estimator”, available at

:http://books.google.co.in/books?id=IwEOB2Mfzx0C&pg=PA1 &source=gbs _toc_r &cad=4#v=onepage&q&f=false, 2009.

[2] S Basha and P. Dhavachelvan, "Analysis of Empirical Software Effort EstimationModels", International Journal of Computer Science and Information Security (IJCSIS), Vol. 7, No. 3, 2010.

[3] B. Hughes and M. Cotterell, "Software Project Management", Tata McGraw-Hill,2006.

[4] B. W. Boehm, "Software Engineering Economics", Prentice-Hall, Englewood Cliffs, NJ, USA, 1981.

[5] T. Gruschke, "Empirical Studies of Software Cost Estimation: Training of Effort Estimation Uncertainty Assessment Skills", 11th IEEE International Software Metrics Symposium, IEEE, 2005.

[6] C. C. Kung and J. Y. Su, "Affine Takagi-Sugeno fuzzy modeling algorithm by Fuzzy c-regression models clustering with a novel cluster validity criterion", IETControl Theory Appl., pp. 1255 – 1265, 2007.

[7] V. Khatibi, Dayang and N. A. Jawawi, " Software Cost Estimation Methods: AReview", Journal of Emerging Trends in Computing and Information Sciences, CIS Journal, Vol. 2, no. 1, ISSN 2079-8407, 2011.

[8] N. Sharma1, A. Bajpai and M. R. Litoriya, "A Comparison of Software Cost Estimation Methods: A Survey", The International Journal of Computer Science and Applications (TIJCSA), Vol.1, no. 3, ISSN – 2278 – 1080, May 2012.

[9] J. Keung, "Software Development Cost Estimation Using Analogy: A Review", Australian Software Engineering conference, IEEE, 2009, DOI:10.1109/ASWEC.2009.32, 1530-0803/09.

[10] T. R. Benala, S. Dehuri and R. Mall, "Computational Intelligence in Software Cost Estimation: An Emerging Paradigm", ACM SIGSOFT Software Engineering NotesPage, Vol. 37, no.3, 2012, DOI: 10.1145/180921.2180932.

[11] J.S. Pahariya, V. Ravi and M. Carr, "Software Cost Estimation using Computational Intelligence Techniques", World Congress on Nature & Biologically Inspired Computing(NaBIC 2009)978-1-4244-5612-3/09/2009 IEEE, 2009.

[12] Mrinal Kanti Ghose, Roheet Bhatnagar and Vandana Bhattacharjee. "Comparing Some Neural Network Models forSoftware Development Effort Prediction", IEEE 2011

[13] Venus Marza, Amin Seyyedi, and Luiz Fernando Capretz, "Estimating Development Time of Software Projects Using a Neuro Fuzzy Approach", World Academy of Science, Engineering and Technology 22-2008.

[14] Vachik S. Dave Kamlesh Dutta, Neural Network based Software Effort Estimation & Evaluation criterion MMRE, International Conference on Computer & Communication Technology (ICCCT)-2011.

[15] Cuauhtémoc López Martín, Software Development Effort Estimation Using Fuzzy Logic: A Case Study, Proceedings of the Sixth Mexican International Conference on Computer Science (ENC'05), 0-7695-2454-0/05 $20.00 © IEEE 2005.

[16] Moataz A. Ahmed, Moshood Omolade Saliuand Jarallah Al Ghamdi, "Adaptive fuzzy logic-based framework for software development effort prediction", Information and Software Technology 47- 2005.

[17] C.J. Burgess, M. Lefley, Can genetic programming improve software effort estimation? A comparative evaluation, Information and Software Technology 43 (2001) 863–873.

[18] Anish Mittal,Kamal Parkash,Harish Mittal " Software Cost Estimation Using Fuzzy Logic", ACM SIGSOFT Software Engineering Notes Page 1 November 2010 Volume 35 Number 1.