iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** IV **Month of publication:** April 2023

DOI: https://doi.org/10.22214/ijraset.2023.50822

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue IV Apr 2023- Available at www.ijraset.com

Software Effort Estimation Based on Use Case Reuse (Back Propagation)

Venkatesh A[1], Pavan[2], Santosh[3], Yugandhar G[4], Sunil Manoli[5]
School of Computer Science and Engineering, REVA University, Bengaluru-560064, Karnataka, India

Abstract: ***Project managers often use effort estimating strategies to manage the human resources of current or upcoming software projects. Prior to project implementation, cost, time, andpersonnel estimation are basically necessary. For every project of software, getting accuracy in Effort Estimation has always been difficult. In this study, the estimation of software development effort was determined using a back propagation model. This model's goal is to investigate the capabilities and potential uses of Utilizing artificial neural networks (ANN) as a tool for forecasting the effort required for software development. In order to estimate the software work, we are attempting to implement a machine learning technique in this research. Out of all machine learning methods, we are applying an algorithm based on Artificial Neural Networks that is Back propagation. The Desharnais dataset, a well-known publicly available dataset for estimation of software effort, is used to test the approach. The performance and accuracy of the tested model have been evaluated using three metrics: MMRE, MRE, and Pred (0.25). In the sections below that follows, I explain the algorithm and its results.***

Index Terms: Artificial Neural Network, Software Effort Estimation (SEE), Machine Learning, Back propagation.

I. INTRODUCTION

The action of estimating a time necessary to create software iscalled effort estimation. Estimating the work is a critical job in the software industry. To produce accurate estimates, manycomputational models have been developed. Initial estimateswithout a clear understanding of the needs are inaccurate, but asthe project advances, estimate accuracy increases. Choosing theright estimating technique is crucial as a result. Estimates of theeffort can be utilized as input in budgets, investment plans,iteration plans, and project plans evaluations, pricing methods,and bidding rounds. Since at least the 1960s, The problem ofsoftware development effort estimation has been addressed byresearchers and practitioners in the field of software. project. The biggest difficulty in project scheduling in the softwareindustry is deciding how much of the project's resources shouldgo toward the testing phase. It has been discovered that the testingphase often uses between 40% and 50% of the resources.

Estimating the specific amount of work that has to be put into the testing phase is quite difficult, though. As a result, the project planning is flawed. Inadequate testing of a project could cause thecompany to suffer severe losses. The study's primary focus has been on creating formal models for estimating software effort. Software effort estimation has been investigated using a range of methodologies, supporting vector regression (SVR) [4], radialbasis function (RBF) neural networks [1], bagging predictors [9], and more modern machine learning techniques, such as the COCOMO [12] and COCOMO [12]. Machine learning techniques create models using data from previous work., which are then used to predict how much work will go into upcoming ventures. The vast majority of techniques for calculating software effort only offer estimates [1][4][7][9][10]. However, in addition to the estimate, it would be important to include estimates accuracy measures [10]. As a result, an estimation technique would be able to give a range of accuracy for where the effort would fall.

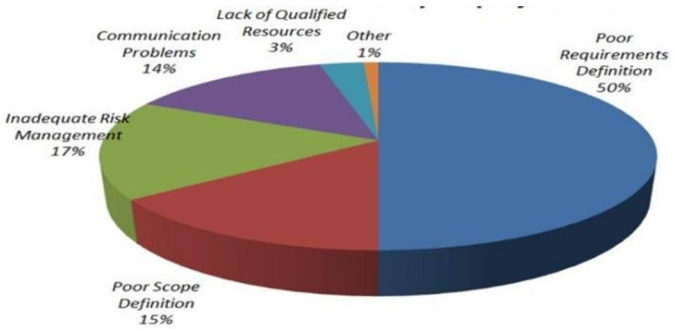


Figure 1.Outcome of Poor Project Management

II. LITERATURE SURVEY

A review of several recent studies on software project effort estimation is provided in this section. Numerous publications have lately been published in this area of current study. We concentrate our review in this section on a few significant article that used in machine learning to calculate software projects effort. Despite extensive research over the past 20 years, the software industry still faces significant challenges when it comes to accurate resource estimation. Authors have occasionally discussed different approaches to the problem. Jorgensen provided a detailed examination of numerous studies on the development effort. 10. Wavelet neural networks are used by K. Vinay Kumaret al.11 to estimate effort. An improved FLANN algorithm for software work prediction was presented by B. Tirimula Rao et al. For the purpose of forecasting development costs and timelines, Multilayer perceptron and back propagation are used by G. Witting and G. Finnie4 as learning methods. algorithms. N. Karunanitthi et al. shown how to use ann to gauge the dependability of software that makes use of several feed- forward and Jordan network techniques. The training method for ann with back propagation is used by N. Tadayon14. A training set of data and a validation set of data were created from the same data set, however this was not stated explicitly in the literature. Our goal in this vision is to use artificial neural networkapproaches to improve the five scaling variables for the COCOMO II estimate model as well as a number of dependent variables, including the cost drivers. It is difficult to comprehend the findings because neural networks, which act as "black boxes," have dispersed knowledge [5]. A neuro-fuzzy was suggested by Huang et al. to get past this restriction. Utilizing a cost model that is constructive and easy to understand to estimate software work [12]. A neuro-fuzzy technique, which has a great generalisation capacity and performs well with ambiguous and incorrect inputs, is the basis of this model. In simulations, the neuro-fuzzy model performed better for estimating software effort than the conventional COCOMO method [12].As we can see, there has been a lot of study on using decision tree-based models to improve SDEE accuracy. An algorithm for back-propagation learning is a type of artificial neural network. An ANN with a back-propagation algorithm is one of the most important learning algorithms in use right now. The issue of multiclass classification can also be solved with it. The hidden, output, and input layers make up an majority of ANNs. The nonlinear activation function argument, or sigmoid function, is specified by the weighted sum of the input neurons (Baareh et al., 2006). Let y1(p), y2(p),, yn(p) be the network's needed output and x1(p), x2(p),..., xn(p) be its inputs. P establishes the iteration number. El-Sayyad et al. (2015) provide the following illustration of the back-propagation neural network's function.

III. PROPOSED METHOD

The dataset is preprocessed so that effort may be calculated using a specific method and a limited number of equations. The estimated effort is then compared to the anticipated effort that was acquired by applying: Backpropagation, as well as the actual effort that was attained (Backpropagation). The created data mining models and method are then evaluated using MMRE, MRE and PRED Performance Evaluation Metrics (x).

A. Dataset (Desharnais)

Jean-Marc Desharnais produced this dataset in 1988 [4]. It is one of the earliest datasets for SDEE. It has so been utilized in numerous empirical research, including [10] [7] [9]. Dataset with 81 rows and 12 attributes which is of real software projects data from a software company called canadian make up this dataset. Based on their technological environments, these 81 projects have been divided into three subgroups: the conventional environment (46 projects), the "improved" traditional environment (25 projects), and the micro environment (10 projects). There are 12 features total in each project, nine of which (Team Experience, Manager Experience, Length(months), Entities, Transactions, Adjustment, Points Non Adjust, , Points Adjust, and Language(1,2,3)) are independent, and one of which (Effort) is dependent.

Sl.No.	Name of the attributes	Type	Description
1	Project	Numeric	Project ID which starts by 1 and ends by 81
2	TeamExp	Numeric	measured in years
3	ManagerExp	Numeric	measured in years
4	YearEnd	Numeric	Year the project ended
5	Length	Numeric	Duration of the project in months
6	Effort	Numeric	ActualEffort is measured in person-hours
7	Transactions	Numeric	Transactions is a count of basic logical transactions in the system
8	Entities	Numeric	Entities is the number of entities in the systems data model
9	PointsAdjust	Numeric	Size of the project measured in adjusted function points. This is calculated as: PointsAdjust= PointsNonAdjust x(0.65+ 0.01)x Envergure
10	PointsNonAdjust	Numeric	Size of the project measured in unadjusted function points. This is calculated as Transactions plus Entities
11	Language{1,2,3}	Categorical	Type of language used in the project expressed as 1, 2 or 3. The value "1" corresponds to "C", where the value"2" corresponds to "C++" and the value "3" corresponds to "Advanced JAVA"

Figure2. Attributes from the Dataset

B. Pre-processing of Data and Labelling

Preparing raw data to be used with a machine learning model is known as data pre-processing. In order to build a machine learning model, it is the first and most important stage. It is necessary to perform these actions in order to clean the data and prepare it for a machine learning model, which also improves the model's efficacy and accuracy.

There are ten attributes total in the dataset, one of which is effort,which is a dependent attribute. The remaining nine qualities are independent. First, the dataset is divided in half, with training data going into testing data in a ratio of 70: 30. The training datamake up 30% of the dataset, while testing data make up 70% of it.

While testing the new instance, the effort attribute is removed from the dataset. The effort is then generated by the algorithm and verified against the actual effort values to ascertain the algorithm's correctness and the possible range of values for a particular set of project-related attributes.

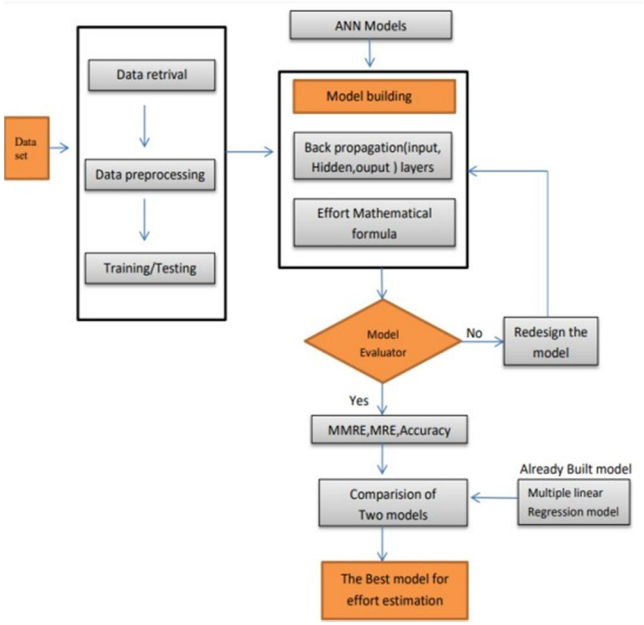


Figure3. Flowchart for Effort estimation process

C. Back Propagation Algorithm

Feedforward neural networks are trained using the popular back propagation algorithm. Instead of explicitly computing the gradient with respect to each individual weight as would be the case naively, Using the network weights as input, it calculates the gradient of the loss function and is incredibly effective. Due to their effectiveness, gradient methods—including the popular gradient descent and stochastic gradient descent—can be used to update weights and train multi-layer networks to minimize loss. When using back propagation algorithm, the loss function's gradient with respect to each weight is computed using the chain rule, layers by layers, and reiterating the previous iteration layer in order to prevent the computation of intermediate terms in the chain rule from being repeated.

Mathematical model

1) Equation 1 shows the results of the hidden layer's calculations:

$$(p) \quad = sigmoid[\sum_{i=1}^{n} xi(p)wij(p)\theta j$$

The quantity of input neurons is depicted by the variables which, n, j stands for number of the hidden layer, and wij, which stands for the estimated mapped weights between the variables Journal of Computer Science, 2019, Volume 15(3), Page 321.331, by Abdel Karim Baareh. A threshold value is a value that determineswhether data is considered to be valid the output layer is reached after passing through the hidden layer and the input layer.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue IV Apr 2023- Available at www.ijraset.com

2) Equation 2 shows how the sigmoid function was put into practice:

$$\frac{1}{1+e^{-x_j(p)}}$$

3) Equation 3 shows the calculated output of an output layers: $y_k(p)=sigmoid[[\sum_{j=1}x_{jk}(p)w_{jk}(p)-\theta j]$ where m is the quantity of input neurons in back propagation and k is the output layer

4) The Gradient of Errors obtained coming from the outputlayer is represented by Equation 4:
$\delta_k(p)=y_k(p)[1-y_k(p)]e_k(p)$

5) Equation 5 shows the weights determined by the ANN.:
$\Delta w_{jk}(p)=\alpha y_j(p)+\delta_k(p)$

6) Use Equation 6 to modify the ANN weights.
$w_{jk}(p+1)=w_{jk}(p)+\Delta w_{jk}(p)$

7) Equation 7 represents the hidden layer's estimated gradienterror.: $\delta_j(p)=y_j(p)[1-y_j(p)]\sum\delta_k(p)w_{jk}(p)$

8) Using Eq. 8, the weights are further computed..
$\Delta w_{ij}(p)=\alpha x_i(p)+\delta_j(p)$

9) The revised weights are represented in Equation 10:
$w_{ij}(p+1)=w_{ij}(p)+\Delta w_{ij}(p)$

They entirely overlap on other subspaces and only separate on the two-dimensional space bounded by B+ and B-. Finding this separation from the N-dimensional data is the objective.

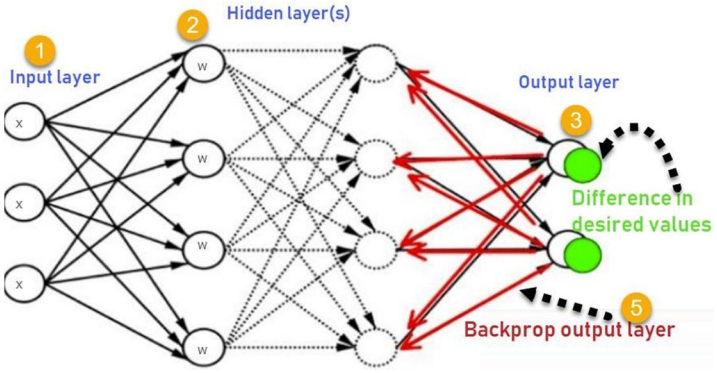


Figure4.Workflow Diagram of Back propagation Algorithm

Supervised learning is a technique used by neural networks to produce output vectors from input vectors on which the network is based. If the result does not match the created output vector, it compares the generated output to the favoured results, and creates an errors report. It is the weights then made adjust in accordance with an bug report to produce the results you want.

Algorithm of Back propagation

- Step1: First, X must enter through the pre connected path. Step2: Using true weights W, the input is simulated. Typically, weights are selected at random.
- Step3: Determine each neuron's output through the hidden layerfrom the input layer and out towards the output layer.
- Step4: Determine the errors in the layer of outputBack propagation Error(diff)= Actual Output - Desired Output
- Step5: Return between the hidden layer and the output layer and change the weights to lower the error.
- Step6: Continue the procedure until the intended result isobtained.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue IV Apr 2023- Available at www.ijraset.com

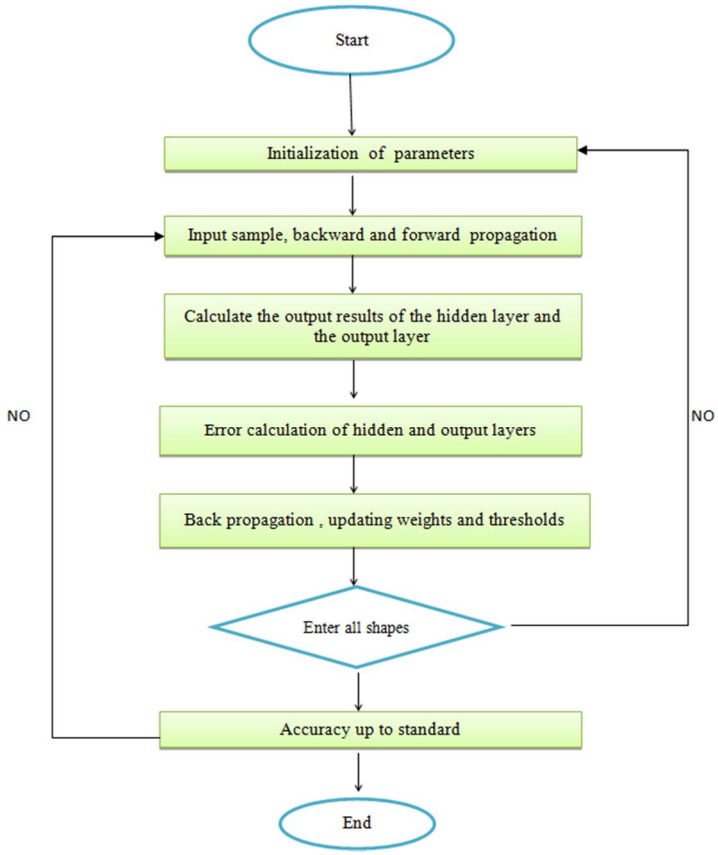


Figure5. Flow Chart of Back propagation

D. Multiple Linear Regression model

The statistical approach known as the use of multiple linear regression to forecast the result depends on the values of two or more other variables other variables. It's a development of multiple linear regression and is occasionally just referred to as multiple regression. The factors We employ independent or explanatory factors to forecast the value of the dependent variable, meanwhile the variable we are trying to forecast isreferred to as the dependent variable.

Formula for multiple linear regression model

$Yi=\beta0+\beta1xi1+\beta2xi2+….+\beta pxip+€$

Where as:

- The predicted or dependent variable in this case is yi.
- The value of y where x1 and x2 are both 0 is known as the y- intercept, or β0.
- Y's change in relation to a shift of one unit in either xi[1] or xi[2] is represented by the regression coefficients β1 and β2, respectively.
- With regard to each independent variable, βp represents the coefficient of slope.
- The model's random error (residual) term is ϵ.

Multiple linear regression model is a statistical tool that uses two or more independent variables to predict the outcome of a dependent variable. Analysts can use the technique to determine the total variance of the model as well as the proportional contributions of each independent variable.

This regression analysis model enables higher variance and accuracy when it comes to outcome prediction as well as analyzing the impact of each explanatory variable on the model's overall variance due to the many variables, which can be linear or nonlinear.

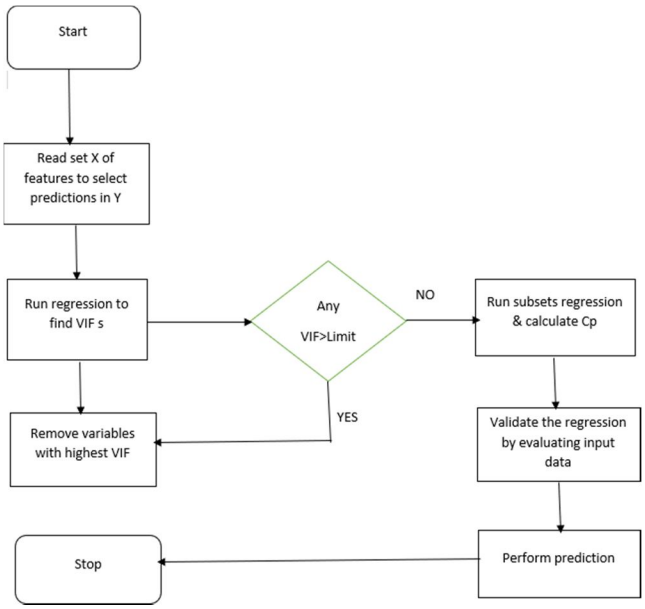


Figure 6. Flow Chart of Multiple Linear Regression model

When the Multiple linear regression model involves two or more independent variables and one dependent variable. is used to assess the connection between them. When seeking information,you can utilize the multiple linear regression model to:

1) How closely two or more independent variables are related. are positively correlated with one dependent variable (for example, the amount of rainfall, temperature, and Adding fertilizer impacts crop growth).

2) Value of the dependent variable in relation to a specific of the independent variables' values (For instance, the crop's anticipated yield at specific levels of temperature, rainfall, and fertilizer application).

3) Considering the most relevant attributes, the effort values are calculated. The predicted value of Effort is compared with the actual effort value for each of the output of the algorithmic model corresponding to its input. Each time the weight vector, is updated and the output is computed again and compared. The graph of predicted vs actual values

4) The estimated effort is mainly dependent on the attributes which are present in the dataset. Whenever we calculate the estimated effort we will compare it with the actual effort which is present in the dataset the difference is MRE here.

IV. RESULT

Firstly, only the attributes impacting the effort are considered and rest of the attributes are dropped based on the correlation matrix. The heatmap of the values of attributes are-

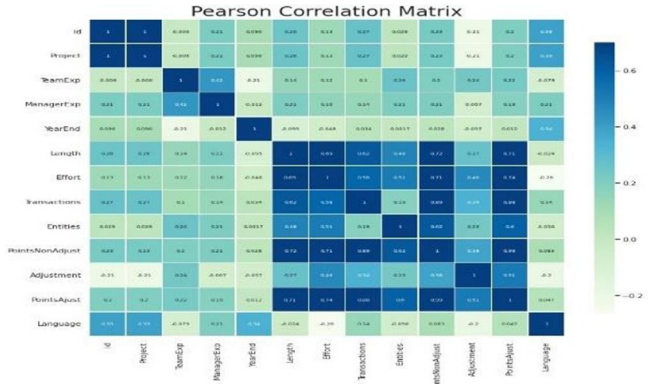


Figure 7: heatmap

The results revealed that the NFR feature "Language" is the most statistically significant attribute, and that removing it from the regression model's training leads in a 25 percent rise in MMRE.

The next statistically significant attribute was "Envergure," whichwas followed by the less statistically significant attributes "Team Exp" and "Manager Exp."
Software effort and the attribute "Team Exp" have a negative correlation, whereas "Manager Exp" has a positive correlation.
The findings also demonstrated that the MMRE error increases by100% when the quality standards attributes are disregarded, and prediction models are created exclusively using software size asan independent.

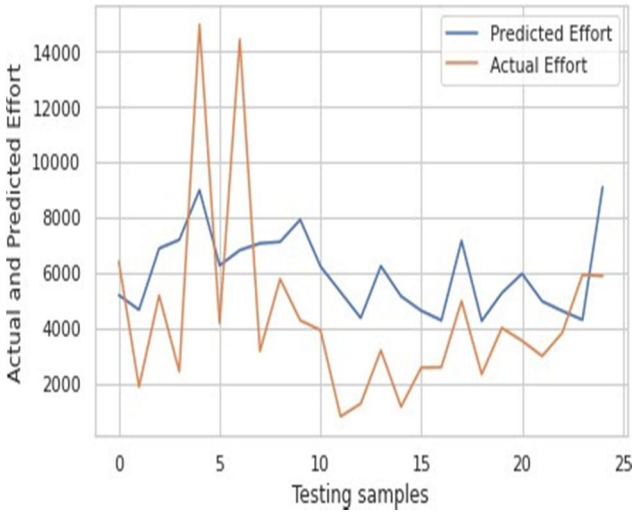


Figure8.Training and Validation Accuracy Graph

Figure5. Performance Accuracy

A. Accuracy of the trained Model

Accuracy plays an important role to determine whether the trained model will be suitable or whether it provide us with desired outputs or not. After splitting the datasets training 70% and testing 30% of datasets accuracy is obtained.
Figure9 shows the training and validation accuracy graph obtained by my proposed scheme. According to the observations, validation accuracy occurs to be less than training accuracy. This can be improved by reshuffling the validation set. The model should be trained more for validation samples as the datasets in that case are still rising and falling for some epochs.

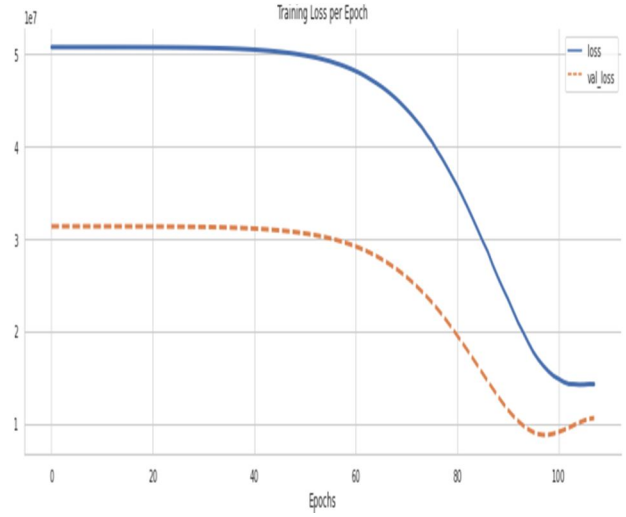


Figure9.Training and Validation Loss Graph

In Figure 7, Representation of training and validation loss graph is shown. It is observed that there is higher validation loss than training loss, it represents that my model is overfitting. It learns superstition which means that patterns in my datasets of training data but in reality, it's not the case and hence it is not true for validation data.

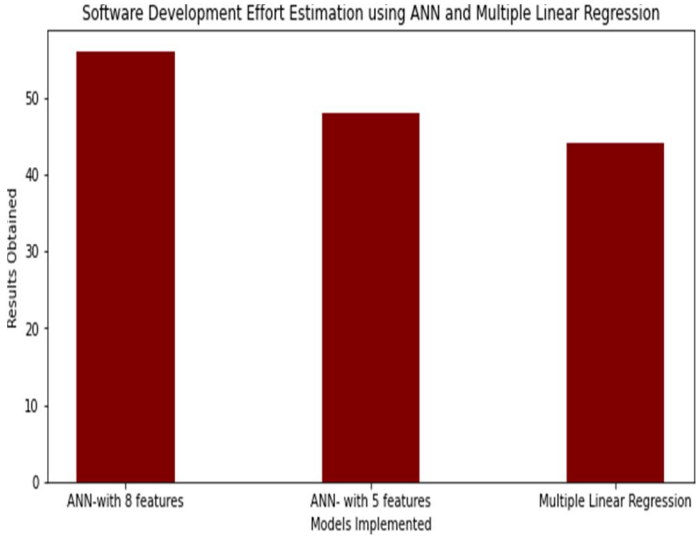


Figure10. Comparative results

The MMRE and MDMRE accuracy of the trained model by thevalidated datasets are-

Performance Evaluation Metrics	No.of Features	Obtained Results
Mean Magnitude Relative Error (MMRE)	8	[0.18]
Median of Magnitude of Relative Errors(MdMRE)	['TeamExp', 'ManagerExp', 'Length', 'Transactions']	[0.16]
Overall Prediction Accuracy	['Entities', 'PointsNonAdjust', 'Adjustment', 'PointsAjust']	[84.8]

Figure 11. Training and Validation Accuracy

V. CONCLUSION

This system designed is based on various neural networktechniques that have been used to effort estimation. Every technique aims to provide the most accurate software effort estimation. In this research, we suggest that a good method for calculating software development effort is the back propagation algorithm, which is an ANN model. It was advised to adopt the back propagation strategy, which will quickly propagate errors, for complex and computationally intensive tasks, where the outcomes of the multilinear regression were contrasted however, it is necessary to assess the approaches' correctness since they are mostly needed in software work estimation. We found that neuron-based models are more accurate estimators and can therefore be utilized to determine software effort estimation for alltypes of projects.

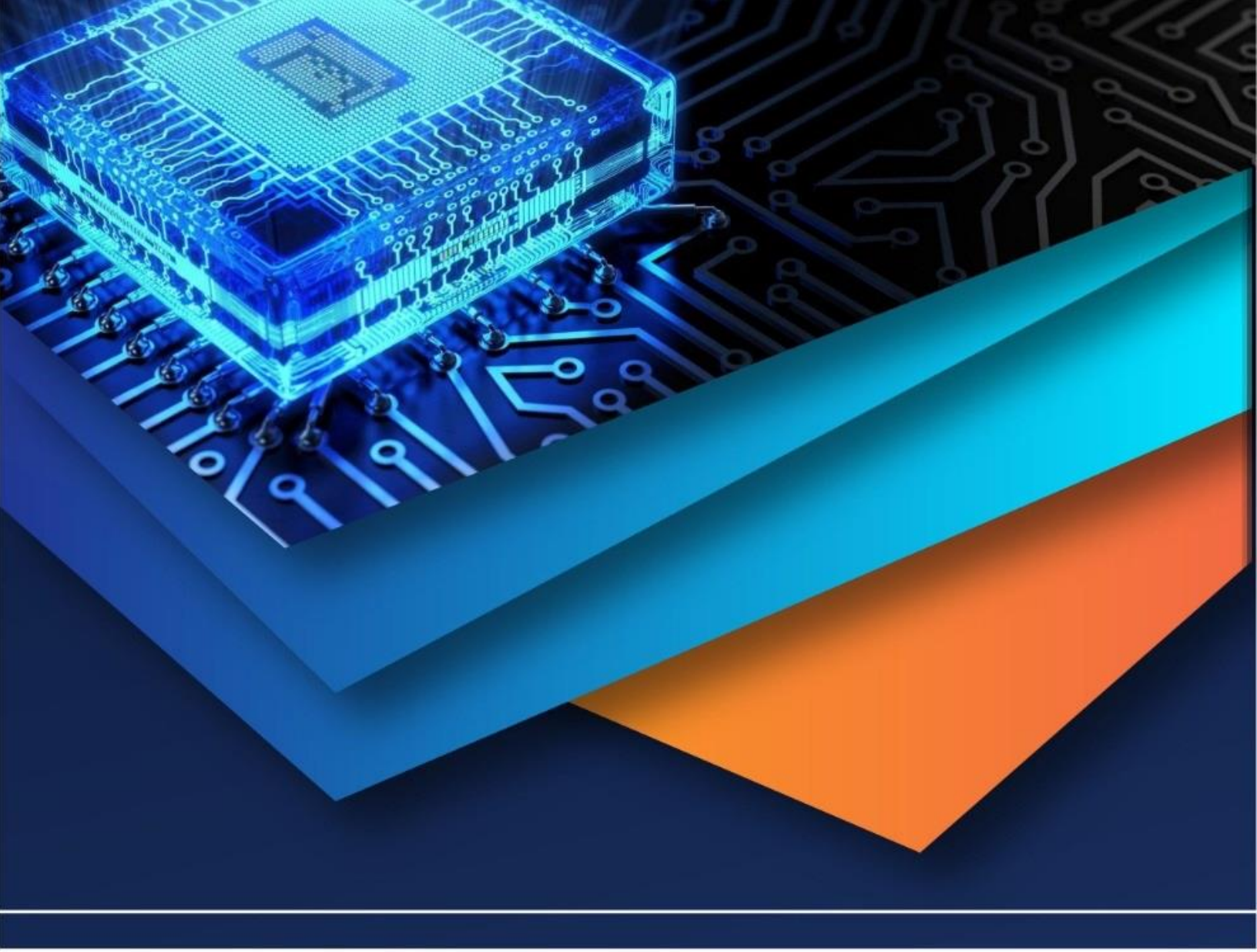
VI. FUTURE ENHANCEMENT

Future editions of this model will incorporate the suggested model along with some tuning strategies to deal with values that are accompanied by confusing and unclear facts. expanding it to astrategy that is likely to be a future study area Future synthetic data generated using well-known approaches will be used to train and evaluate the suggested model, resulting in increased accuracy over current methods. By fine-tuning hyper parameters and using optimization techniques, the performance of the model can be improved. This will result in a successful software development project using estimating approaches, therefore in the future, the estimation of the effort in development of software projects can be done by using other important attributes.

REFERENCES

[1] S.G MacDonell, “A comparison of modeling techniques for software development effort prediction” ,in :Proceedings of the International Conference on Neural Information Processing and Intelligent Information Systems ,Dunedin, New Zealand, Springer, Berlin, 1997, pp. 869–872.
[2] K. Strike, K. El-Emam, “Software cost estimation with incomplete data”, IEEE Transactions on Software Engineering, 27(10) 2001.
[3] A. C. Hodgkinson, and P. W. Garratt, “A neuro fuzzy costestimator”, in: Proceedings of the Third International Conference onSoftware Engineering and Applications—SAE 1999, pp. 401–
[4] C. Kirsopp, M. J. Shepperd, and J. Hart, “Search heuristics, casebased reasoning and software project effort prediction",Genetic and Evolutionary Computation Conference (GECCO), New York,AAAI, 2002.
[5] X. Huang, D. Ho, J. Ren and L. F. Capretz, "Improving the COCOMO model using a neuro-fuzzy approach". Applied Soft Computing, vol. 7, pp, 29-40, January 2007.

[6] P. L. Braga, A. L. I. Oliveira, G. H. T. Ribeiro and S. R. L. Meira. "Bagging predictors for estimation of software project effort". In: IEEE/INNS International Joint Conference on Neural Networks, IJCNN'2007, Orlando Florida.
[7] R. Babuska ,Fuzzy Modeling for Control, Kluwer AcademicPublishers, Dordrecht, 1999
[8] A. A. Porter and R. W. Selby. Evaluating techniques for generating metric-based classification trees. J. Syst. Softw., 12(3):209–218, July1990
[9] Krishnamoorthy Srinivasan and Douglas Fisher. Machine learning approaches to estimating software development effort. IEEE Trans.Softw. Eng., 21(2):126–137, February 1995.
[10] Sutton S Informed projection Proceedings of the 2018 InternationalConference on Software and System Process, (76-85).
[11] Tanveer B, Vollmer A and Braun S A hybrid methodology for effort estimation in Agile development Proceedings of the 2018 International Conference on Software and System Process.
[12] Boehm B Software cost estimation meets software diversity Proceedings of the 39th International Conference on Software Engineering Companion. A. A. Porter and R. W. Selby. Evaluating techniques for generating metric-based classification trees. J. Syst. Softw., 12(3):209–218, July1990.
[13] Barry W. Boehm, " Software Engineering Economics", prentice– Hall , Inc., Englewood Cliffs, New Jersey 07632, 1981, ISBN0 – 13 –822122 – 7.[7]
[14] S. H. Lee, H. Goëau, P. Bonnet, and A. Joly, "New perspectives on plant disease characterization based on deep learning," Comput. Electron.Agricult., vol. 170, Mar. 2020, Art. no. 105220.
[15] Z.-W. Hu, H. Yang, J.-M. Huang, and Q.-Q. Xie, "Fine-grained tomato disease recognition based on attention residual mechanism," J. South China Agricult. Univ., vol. 40, no. 6, pp. 124132, Jul. 2019.
[16] Rahul Kumar Yadav,Dr. S. Niranjan,Software Effort Estimation Using Fuzzy Logic: A Review, International Journal of EngineeringResearch & Technology (IJERT) Vol. 2 Issue 5, pg.1377-1384, May2013. ISSN: 2278-0181.
[17] ImanAttarzadeh and Siew Hock Ow, Software Development Effort Estimation based on a New Fuzzy Logic Model,International Journal of Computer Theory and Engineering, Vol 1. No. 4 October2009.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY