International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 12, No.1 (Jul-2022)

https://dx.doi.org/10.12785/ijcds/120112

Analysis of Software Effort Estimation Based on Story Point and Lines of Code using Machine Learning

Amrita Sharma[1] **and Neha Chaudhary**[2]

[1]*Department of Computer Applications, Manipal University Jaipur, Jaipur, India*
[2]*Department of Computer Science Engineering, Manipal University Jaipur, Jaipur, India*

Received 14 Jun. 2021, Revised 3 May 2022, Accepted 15 Jun. 2022, Published 1 Jul. 2022

Abstract: Estimating the software work is a crucial job of persons participating in software project management. The difficulty in predicting effort is compounded by the fact that software development is always changing. Several techniques for estimating software development costs have been developed over the last three decades. There are a variety of cost estimation methodologies, algorithmic models, non-algorithmic models, and machine learning methods to choose from. To improve accuracy, machine learning approaches are combined with algorithmic or non-algorithmic models. Researchers in past worked on the effort and time estimation by using one type of development methodology in their work. Currently, the companies uses both agile and traditional techniques to software development. A comparison of agile and traditional development utilizing the neural network (NN) and genetic algorithm is presented in this research (GA). Estimation is performed on the Zia dataset and a github dataset using story points and lines of code, respectively. The smallest error and highest accuracy were attained utilizing machine learning approaches for projected effort values. The value of R2 based on story point is achieved using neural network and genetic algorithm is 0.97 and 0.96 respectively. On other hand, the value of R2 based on lines of code is achieved using neural network and genetic algorithm is 0.94 and 0.80 respectively. The mean magnitude relative error is used for comparison of proposed models with previous works. The dataset with the story point give best results followed by projects with lines of code.

Keywords: Software Effort estimation, story point, lines of code, machine learning.

1. Introduction

As the size and complexity of software grow, it becomes more difficult to build. One of the most significant processes in software development is software cost estimation. The software estimation consists of estimating the effort and time for developing the software. For efficient project planning, a realistic estimate is essential[1].

A. Software Effort

An effort is used to quantify workforce utilization in software engineering and is defined as the entire time spent by members of a development team to complete a task. The process of estimating how much effort will be necessary to develop or maintain a software program is known as effort estimation. This effort is generally quantified in terms of a person's work hours. In the early stages of the software development life cycle, effort estimating is used to help build project plans and budgets. A project manager or product owner can use this method to correctly anticipate expenses and allocate resources.

Although effort estimating can be used in a traditional software development process, it is most usually associated with Agile. The project owner is responsible for maintaining a backlog of project deliverables. They will calculate the time it will take to accomplish each item. They will look at user stories and story points rather than time or cost estimations.

B. Software Cost Estimation

Estimation techniques for software cost are classified into three categories, algorithmic models, non-algorithmic models, and machine learning models[2]. Algorithmic models uses some mathematical equations for estimation. COCOMO and its expansions, SEER-SEM, ESTIMAC, PUTNAM's model, and others are examples of algorithmic models [3]. The regression analysis is used to evaluate these models using past projects. Non-algorithmic models include expert judgment, estimation by analogy, top-down, bottom-up, etc. these models gives estimation by picking previous completed project and its parameters and comparing it with new project's parameters. Many studies have been conducted in this field to obtain reliable estimation results. Several decades of research have been conducted utilizing various machine learning algorithms to discover superior estimation. Machine learning models are created with the

E-mail address: amritasharma9468@gmail.com, chaudhary.neha@jaipur.manipal.edu

algorithmic or non-algorithmic models combined with machine learning techniques. Software is now built using a variety of development methodologies in software engineering. These techniques' estimations are dependent on a variety of input parameters. Different size estimate matrices are used in these development methodologies. Researchers have used a variety of machine learning approaches and evolutionary algorithms to improve the accuracy of effort estimation. In this publication, a comparison of the genetic algorithm and the artificial neural network is presented. In prior work, a genetic algorithm was employed to optimize the parameters for traditional project effort estimation. This approach is used to estimate effort in agile projects. In this paper, effort estimation using an artificial neural network is used for both agile and traditional projects.

The following is a breakdown of the work. Section 2 discusses the relevant work done for estimating software effort. The problem statement is explained in part three. Section 4 discusses the proposed work and details the strategies used. The results and validations are discussed in Section 5, and the results are analyzed in Section 6 based on story points and lines of code. The paper comes to a close with section 7.

2. RELATED WORK

This section examines the effort and duration estimation work completed for projects using various development processes, which include a variety of techniques and approaches. The review is carried out for research projects that are based on various size matrices. Pospieszny et al. [3] suggested an ensemble model based on three machine learning techniques: support vector machine (SVM), Multilayer Perceptron (MLP), and Generalized Linear Models (GLM) (GLM). The presented methodologies are used to estimate software effort and duration using the ISBSG dataset. The author validates and compares their earlier work with the MMRE and PRED, finding that the SVM and ensemble model produces better results. For software cost, Rijwani et al. [4] used a multi-layered feed-forward neural network. The COCOMO II dataset was used to train this model using back-propagation methods. The COCOMO 81 dataset is transformed to the COCOMO II dataset using the Rosetta stone tool in this study. The MSE and MMRE are used to validate the estimation values. The four neural networks, general regression neural network (GRNN), multilayer perceptron (MLP), cascade correlation neural network (CCNN), and radial basis function neural network, have been compared by Nassif et al. [5]. (RBFNN). The MAR is used to determine the accuracy of neural networks. We found the neural network with the best estimation and significance in terms of model input. In this study, the CCNN outperformed the RBFNN in terms of prediction. Lopez-Martin [6] used the ISBSG dataset version 11 to create the dataset samples, which he then trained and tested using the MLP and RBFNN. The model was created using the dataset's appropriate independent variables. Only a few projects were able to be chosen to model the network by the writers. The residual analysis and the Friedman statistical test were used to evaluate the duration prediction, which was then compared to the simple statistical regression analysis. For projects written in third-generation languages, these two neural networks were more accurate in predicting software development time. Zahid et al. [7] established a software cost estimating framework that uses eight distinct data mining algorithms to obtain more accurate effort and duration estimates while excluding less useful aspects. Using the above-mentioned data mining techniques, a general input selection procedure was suggested in this article for five datasets from various sources. The median of the magnitude of the relative error and prediction methods were used to validate the findings. When compared to when no input selection technique was used, the results showed that the input selection procedure provided less MdMRE. Using the multilayer regression approach, Peter et al. [8] proposed the ATLM baseline model. The multilayer regression technique is useful for automating database transfer without having to tune any parameters. The effort for the three datasets, COCOMO 81, Desharnais, and OrgAll, was estimated using this baseline model. The LSD, MMRE, PRED, and RE were used to validate the estimation results. According to Federica and Alessio [9], the ATLM has not been thoroughly investigated. As a result, the author suggested a method for estimating effort based on linear programming (LP4EE). Experiments and robustness demonstrated the LP4EE approach to be superior to the ALTM [8]. The 10 separate datasets were used to implement the introduced method in [9]. The MAE and MdAE were used to compare the method to the ALTM and other state-of-the-art methodologies. Panda et al. [10] use four different neural networks to estimate the effort in agile development: the General Regression Neural Network (GRNN), the Probabilistic Neural Network (PNN), the Cascade-Correlation Neural Network (CCNN), and the Group Method of Data Handling (GMDH) Polynomial Neural Network. The user narrative point analysis took these neural networks into account. The neural network models were used to calculate the cost of software development. On a few common parameters, the neural networks were compared. Zia et al. [11] used an agile methodology dataset to estimate development time and cost based on story size, velocity, story complexity, friction, and dynamic factors, resulting in lower MMRE values for time and cost. By proposing the parameters of an estimating model for the agile methodology utilizing story points and velocity, Khuat and Le [12] presented a hybrid algorithm employing the PSO and ABC to obtain improved accuracy. The Zia dataset was utilized to suggest a model, which was then tested using the MMRE, MdMRE, PRED, R2, and MAR. For software effort estimation, Nassif et al. [13] employed a multilayer perceptron neural network. In this study, the size of the software in the use case point and the productivity of the team were used as inputs. The back-propagation approach was used to train the network, and the model's output was a software development effort. The mean relative error approach was used to validate the results. A total of 160 ISBSG projects, academics,

and small software development enterprises were used in this study. Several studies have employed productivity to calculate effort with the UCP, which is dependent on expert guesses. In several studies, productivity is combined with the size measure UCP to estimate effort. However, most productivity estimates are based on expert guesses, which can lead to inconsistency. To bridge this gap, Azzeh and Nassif [14] created a hybrid model that included the SVM and RBNN machine learning approaches. These methods were designed for the categorization and forecasting stages, as well as the model, built with the help of industry and student projects. The SA, MAE, MBRE, and MIBRE were used to evaluate this model, and the findings were superior to previous studies [13] [15] [16]. In their analogy-based estimation for software work estimation, Shahpar et al. [17] used the PSA-SA (Particle swarm optimization- Simulated annealing) approach for feature weighting. The performance of this technique was compared to that of the MMRE, MdMRE, and PRED on the Albrecht dataset. Nevena et al. [18] proposed a Taguchi-based artificial neural network with two alternative activation functions. Six separate datasets based on lines of code are used in this research. The clustering approach is used to apply the input values. The mean magnitude relative error was used to validate the results. By executing a minimal number of iterations, this work reduces the execution time.

3. Problem statement

To estimate the program effort using various size criteria, several machine learning and soft computing techniques were used. In the past, methods for measuring effort were employed for a single size metric, such as lines of code or story points. Any research paper that uses heterogeneous size measures to apply their proposed software estimating approach is not listed. The neural network and genetic algorithm are utilized in this study to assess the performance of both strategies for the two types of size metrics, story points and lines of code.

4. PROPOSED WORK

Based on story points and lines of code, this research proposes an artificial neural model to estimate software effort and timeline. The genetic algorithm suggested in the earlier study [19] is also improving the software effort estimation model for agile projects. Following that, a comparison is made to estimate work based on story points and lines of code.

A. Artificial Neural Network

The artificial neural network, like the real brain, is made up of interconnected basic processing devices called neurons. An input layer, a hidden layer, and an output layer make up a basic artificial neural network structure [20]. Various inputs are used as inputs to the neurons. Each input is multiplied by a connection weight (w) (n). The products are then passed to a function, which produces the output [21]. The neural network's ability to learn from prior data makes it suitable for estimating effort. Figure 1 depicts the basic structure of an artificial neural network.

The artificial neural network creates a complex interaction between the dependent and independent factors for estimating software effort. It uses the training data set to generate acceptable answers for the unknown data set. Many researchers have presented many neural network models for handling complicated real-world problems in the past. We use an artificial neural network to estimate software development effort based on two size matrices in this paper. Using an artificial neural network, this research determines which size matrices provide the best estimation. Artificial neural networks were used on projects based on lines of code and story points. Figure 2 depicts a neural network model for estimating effort using story points. The architecture of an artificial neural network utilizing the agile development approach involves one input layer with two neurons representing the project's story point and velocity, one hidden layer with at least 10 neurons, and one output layer with one output representing development time. A neural network model of effort estimation for lines of code is presented in figure 3. Similarly, the design of a neural network developed traditionally has one input layer with one neuron, which represents the project's lines of code, one hidden layer, and one output layer, which represents the estimation results and represents the project's development time. A feed-forward back propagation network is what this network is. Other parameters in the prediction from story points and lines of code are the same, such as the Levenberg-Marquardt method for training, gradient descent with momentum weight for learning, and bias learning function for learning. The mean square error is the network's performance function.

B. Genetic algorithm

A genetic algorithm (GA) is a search method for finding a good solution in a multidimensional space that replicates the biological natural selection process. Currently, genetic algorithms outperform complete search techniques [22], [23], [24]. This approach has been applied with the COCOMO model in [19]. The approach was previously applied to improve the COCOMO model's four coefficients. For the agile approach, an estimation model introduced in [12] is used, in which the estimationn is done with the story point, velocity, and five parameters. The approach is used to improve the five parameters of the prior model in this paper. The basic components of a genetic algorithm are four. Chromosome, beginning population, operational functions, and fitness function are the components in consideration. The algorithm is showing in figure 4. The following are the steps of the parameter optimization algorithm [19]:

Step 1: For the five variables, the initial population is generated using real binary coding. Each individual variable has a decimal value ranging from 0 to 9. The starting population is Xi=1, 2,...10, where Xi is the population number.

Step 2: the objective function, showing in equation 1, is used for effort estimation using the story point using the

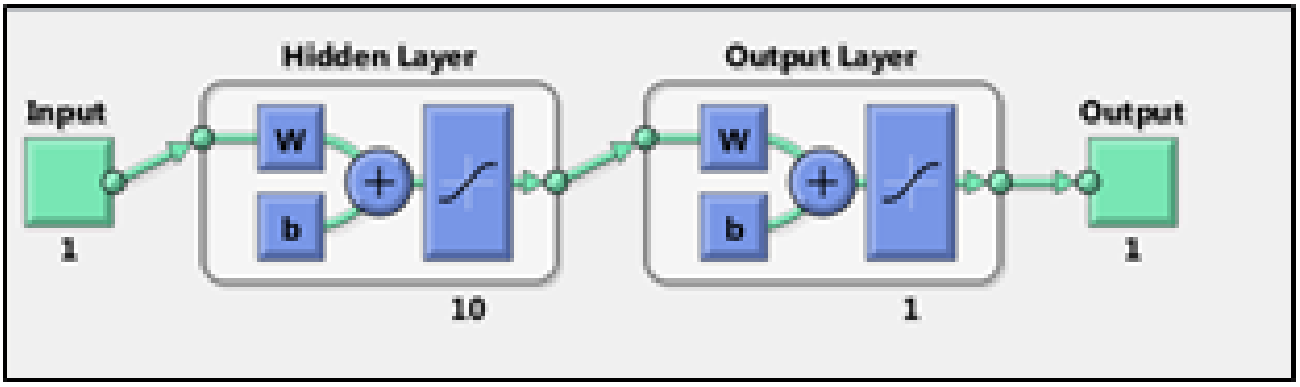

Figure 1. A simple structure of a neural network

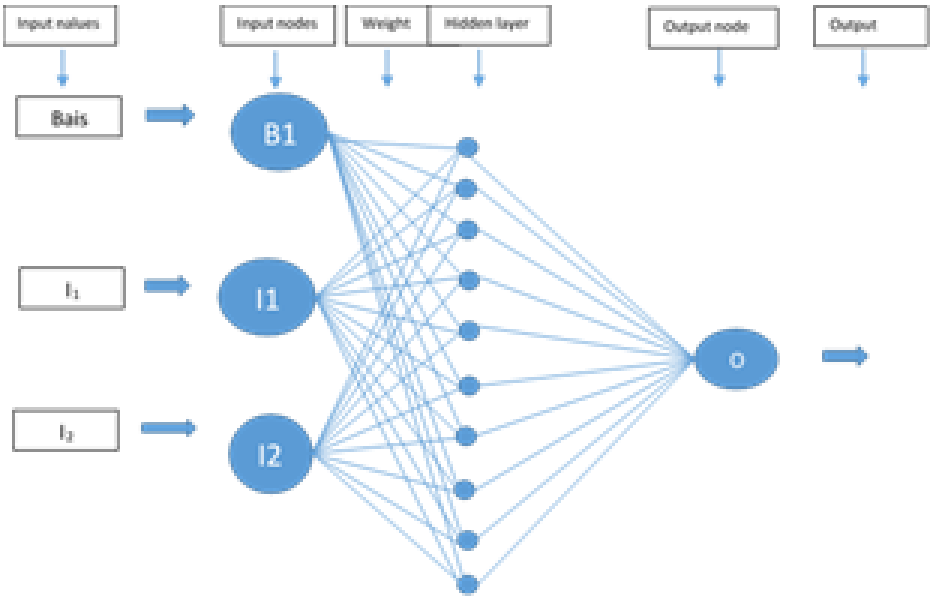

Figure 2. Neural Network Model For Effort Estimation Based on Story Point

individuals.

$$Effort = (A*SP)/(B*V)+C*In(SP)+D*In(V)+E \quad (1)$$

The effort is estimated for project j using individual i using equation 2.

$$Effort_{ij} = (X_i1*SP_j)/(X_i2*V_j)+X_i3*In(SP_j)+X_i4*In(V_j)+X_i5 \quad (2)$$

Step 3: Equation 3 is used to calculate an individual's fitness for a certain project.

$$fitness_{ij} = ((ActVal_j - EstVal_{ij}))/ActVal_j \quad (3)$$

Where ActValj and EstValij are the j project's actual effort and estimated effort with i individuals respectively.

Equation 4 is used to calculate an individual's fitness for all projects by taking the average of the value obtained in equation 3.

$$Fitness = Average(fitness_{ij}) \quad (4)$$

Step 4: The algorithm will come to an end after the maximum number of iterations have been completed, or after the population's best fitness has been identified.

Step 5: The parents are chosen using the roulette wheel technique of selection.

Step 6: A uniform crossover method is applied to create children from the chosen parents in a crossover. The genes are picked at random from the parents in this manner. The children are generated using Eqs. (5) and (6) after a random

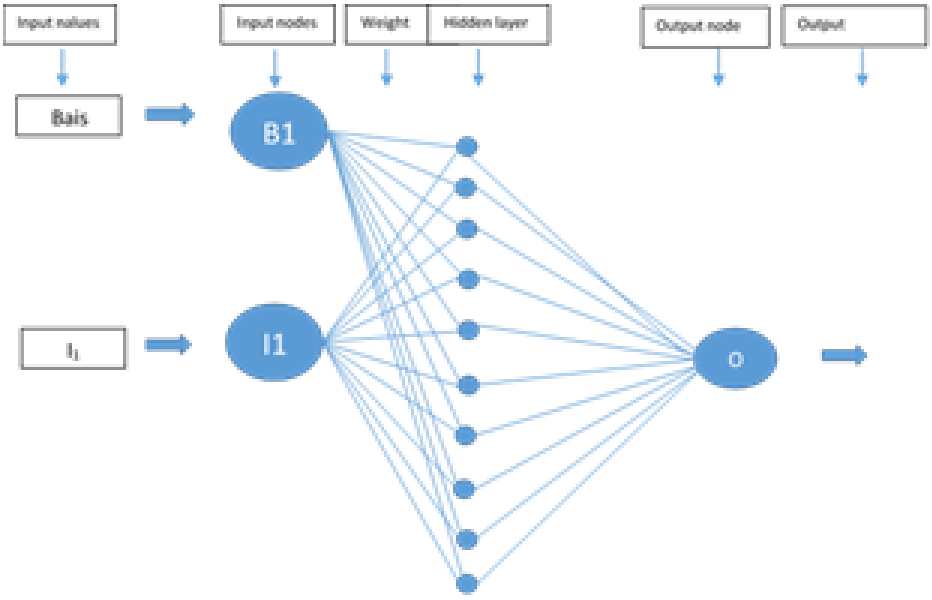

Figure 3. Neural Network Model For Effort Estimation Based on Lines of Code

chromosome is generated.

$$Child1 = a * p1 + (1 - a) * p2 \quad (5)$$

$$Child2 = a * p2 + (1 - a) * p1 \quad (6)$$

Where a is a random chromosome and p1 and p2 are the parents' chromosomes.

Step 7: The fitness of the merged population is assessed after offspring are merged in the initial population to produce new populations. The best outcomes are selected. Then, using the roulette wheel approach, new parents are chosen.

C. Data set

The proposed method is used to estimate development time by using both traditional and agile development methodologies. The number of lines of code and story points is used to estimate the length of time it will take to complete the project. The estimation with story points is based on the Zia dataset [12]. This dataset contains twenty-one projects from six software companies [11]. The agile methodology was used to create these projects. A dataset of nine projects is used in this work [19] for estimation with lines of code. This dataset contains nine python language projects, six of which are student projects and three of which are extracted from a GitHub repository.

D. Evaluation criteria

Some mathematical equations are used as evaluation criteria to see if the proposed model is operating well or not. The suggested model's performance is assessed using the mean magnitude relative error (MMRE) [25]. For comparing the proposed work to past work, the mean square error (MSE) and coefficient of determination (R2) [26] are applied. The following are the evaluation criteria applied in this work:

$$MMRE = ((ActVal - EstVal))/ActVal \quad (7)$$

$$MSE = (ActVal - EstVal)^2 \quad (8)$$

$$R^2 = 1 - (ActVal - EstVal)^2/(ActVal - (\hat{A}ctVal))^2 \quad (9)$$

Where ActVal denotes the actual value, EstVal denotes the estimated value, and ÂctVal denotes the actual value's mean.

E. Experimental details

Twenty-one Zia dataset projects were used to test the suggested approach of genetic algorithm for effort estimation necessary for developing agile projects. The genetic algorithm uses real binary encoding to build the initial population. The projects' effort is estimated with the help of specific individuals. The mean magnitude relative error is taken into account while determining fitness. The fitness of the algorithm is calculated using both the actual and estimated effort. With the roulette wheel selection approach, the best-fit individuals are chosen. Crossover and mutation are the two operative sets of a genetic algorithm. To obtain the child individuals, the crossover is applied to the parent individuals.

The artificial neural network experiment was carried out for both agile and traditional projects. The dataset is divided into training, validation, and testing datasets,

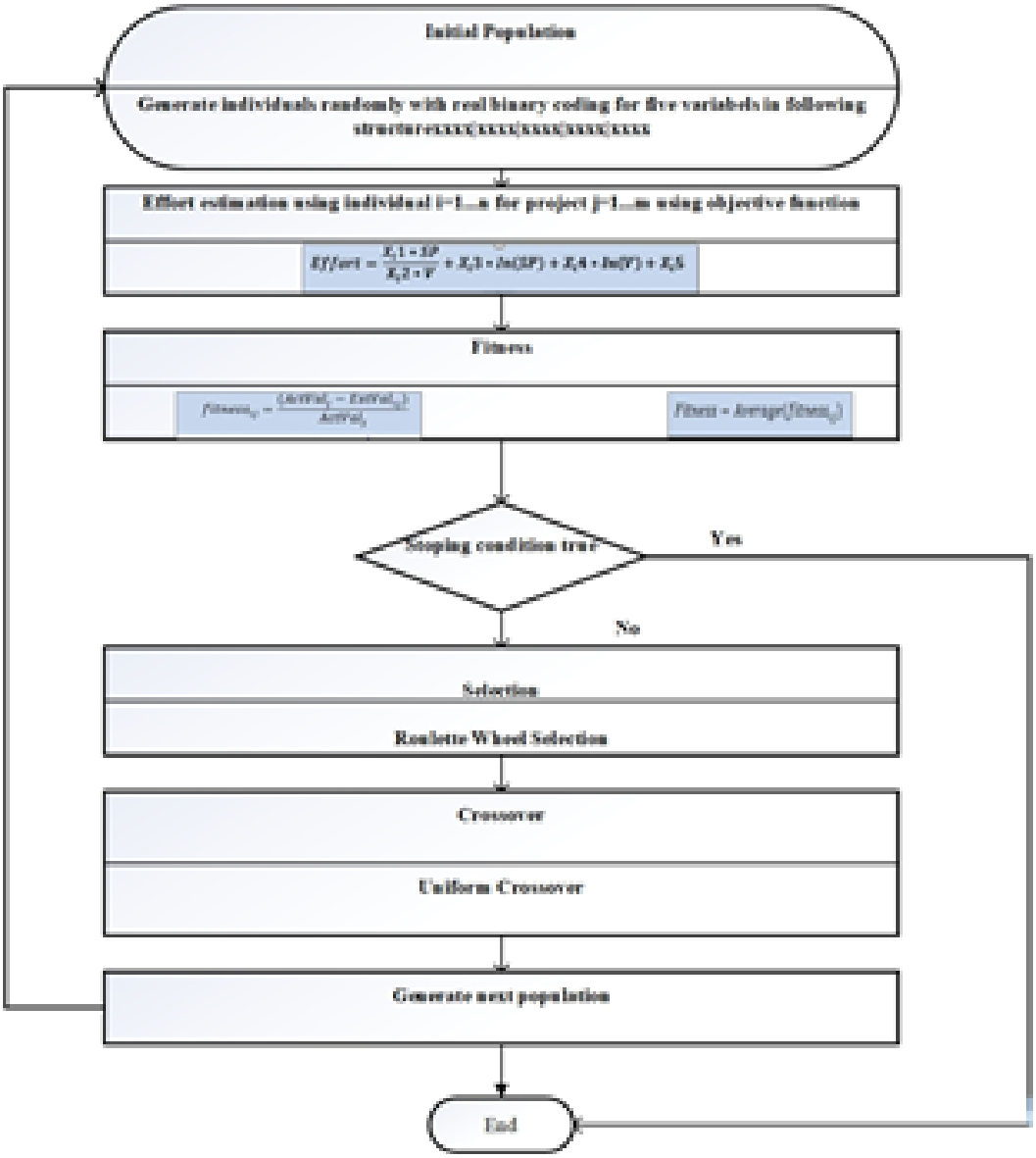

Figure 4. Neural Network Model For Effort Estimation Based on Lines of Code

respectively, to estimate software development costs using agile and traditional development methodologies. The data was divided at random by a neural network model. The data for agile is separated into 70 per cent, 15 per cent, and 15 per cent training, validation, and testing datasets, while the data for traditional projects is divided into 60 per cent, 20 per cent, and 20 per cent training, validation, and testing datasets. The back-propagation function trainlm is used to train the model. For agile and traditional approaches, the hidden layers are 4 and 2, respectively. The project's velocity and total user story points for agile approaches, and the kilo lines of code for traditional approaches, are the input parameters, and the output parameter is the project development time.

5. DISCUSSION

Matlab code and the MatLab neural network toolbox are used to conduct genetic algorithm and neural network studies. The five initial populations for the genetic algorithm have been defined. The genetic algorithm is used to optimize the parameters provided in [12] utilizing the twenty-one projects from the agile dataset. The child individuals are located using the uniform crossover. The algorithm's maximum iterations are 1000. Based on the fitness function, the best set of parameters is chosen. The MMRE, or mean magnitude relative error, is the fitness function. The best findings of parameters for agile effort estimation are used to select a set of parameters. The following are the model's optimum parameters as determined by the genetic algo-

rithm: A= 1.528102485 B=1.610603017 C=0.098278468 D=2.663927182 E=2.718403087

The agile model is used to estimate the required effort for agile development using a set of parameters. The work is compared to an existing model for estimating agile effort using the mean magnitude relative error. In comparison to the existing model of agile work estimating, we discovered that the value of error is the smallest. The error value from the current model was 30.76671, but it was 6.742869 from the agile model's optimized parameters. Figure 5 depicts the real effort, the existing model's value of effort, and the estimated value of effort from the genetic algorithm.

Artificial neural networks are also used in traditional and agile projects. Using the twenty-one projects of the Zia dataset [11] and the nine projects of the dataset [19], the neural network is trained using the back-propagation approach. The datasets are separated into three groups: training, testing, and validation. After the network has been trained, the required effort for agile projects is achieved, and the development time for traditional projects is reduced.

Table 1 shows the values of needed effort derived from the genetic algorithm and neural network for agile projects. The comparison of the real effort, estimated effort, and effort from the existing model using the genetic algorithm is shown in figure 5, and the comparison of the real and estimated effort values using the neural network is shown in figure 6 for the agile dataset. Table 2 shows the values of the required effort for traditional projects using the genetic algorithm and neural network. Figure 7 showing the actual value and estimated values of required effort for the traditional projects using genetic algorithm and neural network.

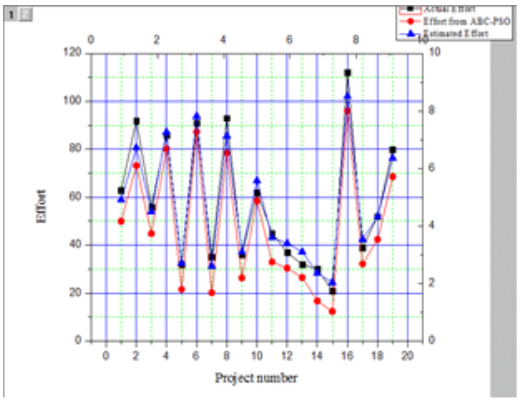

Figure 5. Actual, Estimated Effort and effort from existing parameters for agile projects using GA

6. ANALYSIS OF RESULTS

In this paper, an artificial neural network is used to provide an estimates model for both agile and traditional projects. The prior work's proposed evolutionary method is used for parameter optimization in order to evaluate

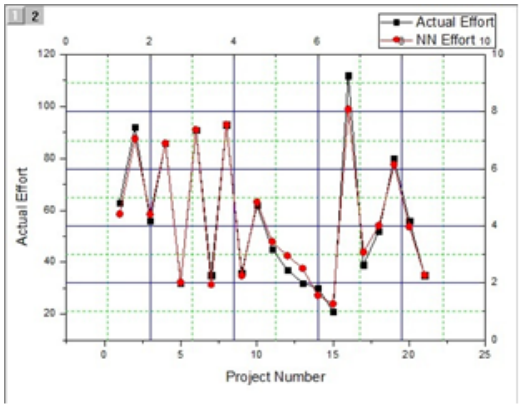

Figure 6. Actual Effort and estimated effort for agile projects by using NN

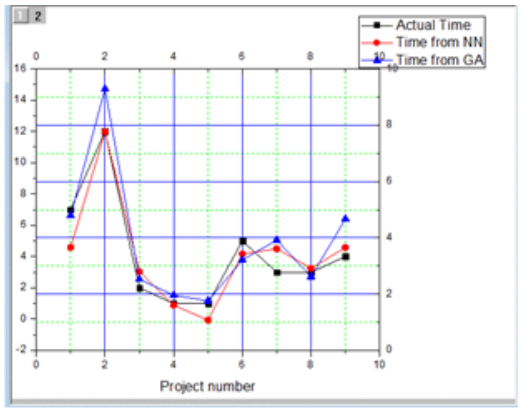

Figure 7. Actual development time, time from the NN and GA for the traditional projects

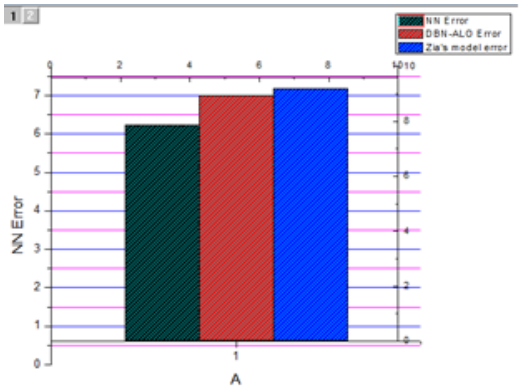

Figure 8. MMRE values of Effort from Neural Network and previous models for an agile dataset

TABLE I. Actual time and estimated time for agile dataset

Story point	velocity	Actual Effort	Estimated Effort from NN	Estimated Effort from GA
156	2.7	63	58.55264	58.90127
202	2.5	92	87.60839	80.6662
173	3.3	56	58.39326	54.05853
331	3.8	86	85.85413	87.15399
124	4.2	32	32.24533	32.59593
339	3.6	91	91.1036	93.79215
97	3.4	35	31.44829	31.39752
257	3	93	93.05162	85.50479
84	2.4	36	34.76436	37.12756
211	3.2	62	63.16225	66.85247
131	3.2	45	47.83794	43.11271
112	2.9	37	42.50776	40.794
101	2.9	32	37.59734	37.19078
74	2.9	30	27.17001	28.34406
62	2.9	21	23.97659	24.41054
289	2.8	112	98.91936	102.0788
113	2.8	39	43.87053	42.40129
141	2.8	52	54.16337	51.8985
213	2.8	80	77.57226	76.31322
137	2.7	56	53.68131	52.21915
91	2.7	35	35.06025	36.03734

TABLE II. Effort values for traditional projects using neural network and Genetic algorithm

KLOC	Actual time	Estimated time from NN	Estimated time from GA
16.675	7	4.58746	6.622391
44.144	12	11.97919	14.70587
5.206	2	3.027065	2.551075
2.78	1	0.89231	1.525638
2.015	1	-0.07298	1.171965
8.392	5	4.162545	3.772673
12	3	4.501828	5.0573783
5.6	3	3.248832	2.7082396
16	4	4.582043	6.4018978

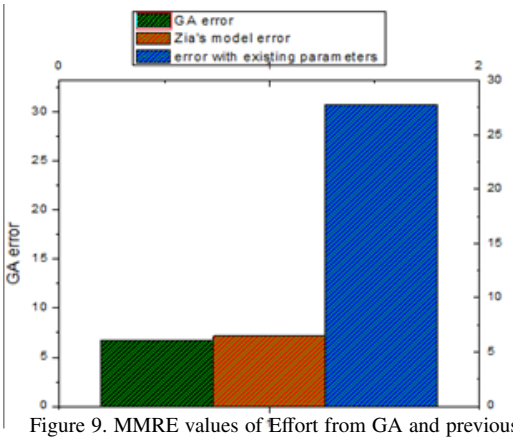

Figure 9. MMRE values of Effort from GA and previous models for Agile Dataset

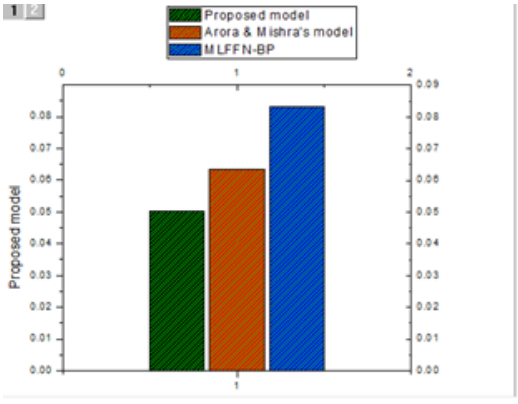

Figure 10. MMRE values of Neural Network model and previous models for traditional projects

the effort for agile projects. Projects with a size matrix of lines of code and story points are examined for use with the suggested paradigm. The mean magnitude relative error is used to evaluate estimating for agile and traditional projects (MMRE). The presented work is also compared to earlier work in order to validate it. The MMRE is used to make the comparison. Figure 8 shows the error between the neural network and the DBN-ALO [25] and Zia's regression model [11] using the agile dataset. The error between the genetic algorithm and Zia's regression [11] model and ABC-PSO [12] is shown in Figure 9. The error between the neural network and Arora Mishra's model [21] and MLFFN-BP [4] is shown in Figure 10. The mean magnitude relative error, mean square error, and R2, or coefficient of determination, are used to compare traditional and agile techniques. The comparison for agile projects is shown in table III, whereas the comparison for traditional projects is shown in table IV. For the agile methodology, the artificial neural network model yielded the lowest MMRE and highest R2 values, while the traditional methodology yielded the higher MMRE than agile. In comparison to traditional projects, the genetic algorithm produced better outcomes in terms of MMRE, MSE, and R2 for agile projects, as proposed in [19].

7. CONCLUSION AND FUTURE WORK

The artificial neural network is used in this study to forecast the effort and length of project development using the back-propagation approach using two types of development methodologies: traditional and agile. We acquire the estimated effort values for agile projects and the expected development time values for traditional programs from this work. In comparison to prior research studies, the results of the work demonstrate a significant improvement in estimated values using datasets of various categories (based on lines of code and user story).

Effort estimation for the agile dataset is also done using the genetic algorithm to optimize the parameters of the estimation model provided in [12]. Our prior work on the genetic method was used to standard projects with size matrix lines of code. This approach has already been used to optimize the parameters of Boehm's basic COCOMO model. The results are more precise with the modified parameters.

The artificial neural network is used in this project. In the future, we will combine various machine learning approaches with feature selection for the other development methodologies to improve estimation accuracy.

REFERENCES

[1] S. S. Ch and S. P. Singh, "Software cost estimation techniques using soft computing," in *International Peer Reviewed Refereed Journal*, vol. 2, no. II, 2015, pp. 42–52.

[2] A. Saeed, W. H. Butt, F. Kazmi, and M. Arif, "Survey of software development effort estimation techniques," in *Proceedings of the 2018 7th International Conference on Software and Computer Applications*, 2018, pp. 82–86.

[3] P. Pospieszny, B. Czarnacka-Chrobot, and A. Kobylinski, "An effective approach for software project effort and duration estimation with machine learning algorithms," *Journal of Systems and Software*, vol. 137, pp. 184–196, 2018.

[4] P. Rijwani and S. Jain, "Enhanced software effort estimation using multi layered feed forward artificial neural network technique," *Procedia Computer Science*, vol. 89, pp. 307–312, 2016.

[5] A. B. Nassif, M. Azzeh, L. F. Capretz, and D. Ho, "Neural network models for software development effort estimation: a comparative study," *Neural Computing and Applications*, vol. 27, no. 8, pp. 2369–2381, 2016.

[6] C. López-Martín and A. Abran, "Neural networks for predicting the duration of new software projects," *Journal of Systems and Software*, vol. 101, pp. 127–135, 2015.

[7] Z. H. Wani, K. J. Giri, and R. Bashir, "A generic data mining model for software cost estimation based on novel input selection procedure," *International Journal of Information Retrieval Research (IJIRR)*, vol. 9, no. 1, pp. 16–32, 2019.

[8] P. A. Whigham, C. A. Owen, and S. G. Macdonell, "A baseline model for software effort estimation," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 24, no. 3, pp. 1–11, 2015.

[9] F. Sarro and A. Petrozziello, "Linear programming as a baseline for software effort estimation," *ACM transactions on software engineering and methodology (TOSEM)*, vol. 27, no. 3, pp. 1–28, 2018.

[10] A. Panda, S. M. Satapathy, and S. K. Rath, "Empirical validation of neural network models for agile software effort estimation based on story points," *Procedia Computer Science*, vol. 57, pp. 772–781, 2015.

[11] S. K. T. Ziauddin and S. Zia, "An effort estimation model for agile software development," *Advances in computer science and its applications (ACSA)*, vol. 2, no. 1, pp. 314–324, 2012.

[12] T. T. Khuat and M. H. Le, "A novel hybrid abc-pso algorithm for effort estimation of software projects using agile methodologies," *Journal of Intelligent Systems*, vol. 27, no. 3, pp. 489–506, 2018.

[13] A. B. Nassif, D. Ho, and L. F. Capretz, "Towards an early software estimation using log-linear regression and a multilayer perceptron model," *Journal of Systems and Software*, vol. 86, no. 1, pp. 144–160, 2013.

[14] M. Azzeh and A. B. Nassif, "A hybrid model for estimating software project effort from use case points," *Applied Soft Computing*, vol. 49, pp. 981–989, 2016.

[15] G. Karner, "Resource estimation for objectory projects," *Objective Systems SF AB*, vol. 17, pp. 1–9, 1993.

[16] G. Schneider and J. P. Winters, *Applying use cases: a practical guide*. Pearson Education, 2001.

[17] Z. Shahpar, V. Khatibi, and A. Khatibi Bardsiri, "Hybrid pso-sa approach for feature weighting in analogy-based software project effort estimation," *Journal of AI and Data Mining*, vol. 9, no. 3, pp. 329–340, 2021.

[18] N. Rankovic, D. Rankovic, M. Ivanovic, and L. Lazic, "Improved effort and cost estimation model using artificial neural networks and

TABLE III. Error values for projects based on SP using NN and GA

MMRE	MSE	R2	
Error from NN for Agile	6.220651	17.03561	0.973897
Error from GA for Agile	6.742859	21.46326	0.967112

TABLE IV. Error values for projects based on LOC using NN and GA

MMRE	MSE	R2	
Error from NN for traditional projects	19.88667	0.600919	0.944562
Error from GA for traditional projects [22]	24.03720	2.185249	0.798399

taguchi method with different activation functions," *Entropy*, vol. 23, no. 7, p. 854, 2021.

[19] A. Sharma and N. Chaudhary, "Software cost estimation for python projects using genetic algorithm," in *International Conference on Communication and Intelligent Systems*. Springer, 2019, pp. 137–148.

[20] R. Bhatnagar, V. Bhattacharjee, and M. K. Ghose, "Software development effort estimation–neural network vs. regression modeling approach," *International Journal of Engineering Science and Technology*, vol. 2, no. 7, pp. 2950–2956, 2010.

[21] S. Arora and N. Mishra, "Software cost estimation using artificial neural network," in *Soft Computing: Theories and Applications*. Springer, 2018, pp. 51–58.

[22] J. Murillo-Morera, C. Castro-Herrera, J. Arroyo, and R. Fuentes-Fernández, "An automated defect prediction framework using genetic algorithms: A validation of empirical studies," *Inteligencia Artificial*, vol. 19, no. 57, pp. 114–137, 2016.

[23] ——, "An empirical validation of learning schemes using an automated genetic defect prediction framework," in *Ibero-American Conference on Artificial Intelligence*. Springer, 2016, pp. 222–234.

[24] J. Murillo-Morera, C. Quesada-López, C. Castro-Herrera, and M. Jenkins, "A genetic algorithm based framework for software effort prediction," *Journal of software engineering research and development*, vol. 5, no. 1, pp. 1–33, 2017.

[25] A. Kaushik, D. K. Tayal, and K. Yadav, "A comparative analysis on effort estimation for agile and non-agile software projects using dbn-alo," *Arabian Journal for Science and Engineering*, vol. 45, no. 4, pp. 2605–2618, 2020.

[26] A. Sharma and N. Chaudhary, "Linear regression model for agile software development effort estimation," in *2020 5th IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*. IEEE, 2020, pp. 1–4.

Amrita Sharma Amrita sharma is a research scholar. She has done Master Of Computer Applications from Sikkim Manipal University. She has done B.Sc. degree in Information Technology from the Sikkim Manipal University. Now she is doing PhD from Manipal University Jaipur, India. Her research area are software cost estimation, software testing, and machine learning. She has presented and published her research work in the scopus indexed international conferences.

Neha Chaudhary Neha Chaudhary is Associate Professor at the manipal university Jaipur, India. She has done her PhD from the Gautam Buddha Technical University, Delhi. She has published so many papers in scopus index journals, book chapters, and conference proceedings. Her research area are software testing, software estimation, machine learning and artificial intelligence.