# An Ensemble Model for Software Development Cost Estimation

Mohammed Maher
*dept. of Software Engineering*
*University of Mosul*
Mosul, Iraq
maher@uomosul.edu.iq

Jamal Salahaldeen Alneamy
*dept. of Software Engineering*
*University of Mosul*
Mosul, Iraq
jamal_alneamy@uomosul.edu.iq

***Abstract*—Software cost estimation is a critical activity that must be fulfilled in order to ensure the successful completion of the software project. Conventional methods of software cost estimation suffer from subjectivity and overhead. To overcome these limitations in the traditional approaches, Machine learning-based techniques have been emerged as an effective solution for the problem of software cost estimation. A stacking ensemble model has been proposed in this study as an effective approach for software development cost estimation. The proposed model is composed of diverse base learners and a meta-learner which is the Random Forest. The model has been trained using the ISBSG dataset which is a high-quality, heterogenous, and well-maintained dataset. The hyperparameters of the model have been tuned using the PSO algorithm. The experiments revealed that the stacking ensemble model outperforms the single contributing model and that various data preprocessing techniques can have a considerable influence on prediction accuracy.**

***Keywords—Software Cost Estimation, Ensemble Model, Machine Learning, Random Forest, ISBSG***

## I. INTRODUCTION

Software cost estimation represents an essential activity in the project management process. The accuracy of the estimation has a high influence on the success of the project. Cost estimation enables the project manager to forecast the required resources to finish the project within time and according to the predefined set of specifications [1]. Overestimation and underestimation are the two main problems that face most software projects. The former may lead to business loss and low utilization of the allocated resources, and the latter may lead to overrun, delivering a product that does not conform to the specification, or total project failure [2]. According to the CHAOS Report 2015, 56% of the projects did not finish within the budget, 60% of the projects missed their deadlines, and 44% of the projects were not on target [3].

During the past century, many software cost estimation models and techniques have emerged. These methods can be generally classified into algorithmic and non-algorithmic techniques [4]. Most of the traditional methods lack accuracy due to subjectivity and bias. Furthermore, the parametric models require a great deal of overhead to get an initial estimation [5]. With the great success that machine learning has achieved in many fields and due to the limitation of the traditional cost estimation methods, much research has been conducted to estimate the cost of software development based on machine learning methods. Amongst these models are Support Vector Regression (SVR), Artificial Neural Networks ANN, K-Nearest Neighbor (KNN), Decision Tree, Random Forest, and Bayesian Network [6]. Another promising method that can increase the accuracy of the estimation is the ensemble of multiple machine-learning techniques into a single model. Ensemble methods can be classified into homogenous and heterogenous, a popular homogenous method is bagging and boosting which combine the results of single type base learners. On the other hand, the heterogeneous method combines the results of different types of base learners such as Random Forest and Leaner Regression (LR), a powerful heterogeneous method is the stacking [7].

The accuracy of the estimation that can be acquired from the different models depends greatly on the quality of the historical data of the software projects that they learn from. There are two main repositories of software engineering projects which can be used in building software cost estimation models. The first one is the Predictor Models in Software Engineering (PROMISE); The second one is the International Software Benchmarking Standards Group (ISBSG) [8]. The PROMISE dataset is publicly accessible; however, it has fewer number of projects in comparison with the ISBSG dataset, and the data of the PROMISE dataset are outdated [7]. On the contrary, the ISBSG dataset contains a larger number of projects with a huge number of features and attributes from several organizations worldwide, and it is continuously being updated and maintained [9].

In this research paper, a stacked ensemble model for software cost estimation has been built. The ensemble model combines the results of multiple learners that include: KNN, SVR, ANN, Bayesian Regressor, Leaner Regressor, and Random Forest. The results of the base learners have been combined using Random Forest as the meta-learner. The hyperparameters of the applied machine learning methods have been tuned using Particle Swarm Optimization (PSO). The ISBSG dataset has been used in building the model with 67% of the data for training and 33% for testing and validation. Various data preprocessing techniques have been applied to the dataset to increase the prediction accuracy of the model. For the evaluation of the performance of the prediction model, Mean Magnitude of Relative Error, Pred (25), and R-Squared have been used. The main objectives of this article can be summarized in the following points:

- Improve the estimation accuracy of software cost by building an ensemble model.
- Compare the performance of different machine learning models in software cost estimation.

The remaining of the research paper is organized as follows: section II presents related work and similar studies, the proposed model is explained in section III, section IV discusses the results obtained and the experiments performed, and conclusions and future work are in section V.

## II. RELATED WORK

Software cost estimation represents an active research field. Much research has been conducted in the past two decades aiming to improve the accuracy of the software cost

978-1-6654-5512-1/22/$31.00 ©2022 IEEE

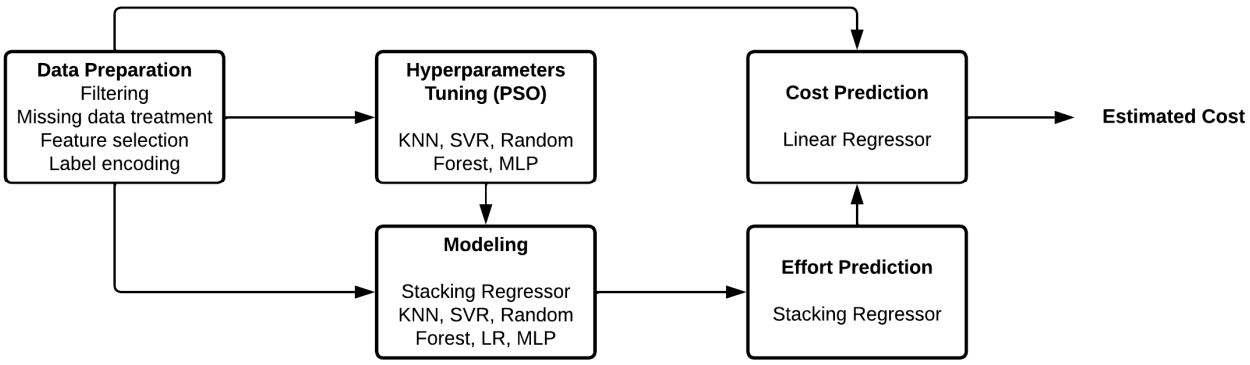


FIGURE 1. The Proposed Method for Software Cost Estimation

estimation process. Machine learning-based techniques have grown in popularity among researchers in the field of software cost and effort estimation due to their good generalization capability and fast learning speed. In this section, a review of recent research in the field of software cost estimation has been provided.

Fadhil et al. [10] have investigated using a dolphin algorithm-based technique to tune the coefficients of the Constructive Cost Model II (COCOMO II). They have constructed two models. The first one was built using the dolphin algorithm, and the second one was built using a hybridized version of both the dolphin algorithm and the bat algorithm. They have found that the estimates obtained using the dolphin algorithm have higher accuracy than the original COCOMO II model and the other optimization techniques such as the genetic algorithm. They have conducted their experiments using NASA-60 and NASA-93 datasets. The experiments revealed that the hybridized dolphin-bat algorithm is superior to other methods with 14.57 MMRE (Mean Magnitude of Relative Error) for the NASA-60 dataset, and 50.27 MMRE for the NASA-93 dataset.

Al Asheeri and Hammad [1] have experimented with 13 different machine-learning techniques to estimate the cost of development. They have carried out their experiments on two datasets Usp05-ft and Usp05. The random forest techniques outstand the rest of the techniques when applied to the first dataset, whereas the REPTree (Reduced Error Pruning Tree) outperformed random forest in the second dataset. ZeroR method had the lowest score among the other techniques in both datasets.

Carvalho et al. [11] have used the Desharnais dataset to compare the performance of Extreme Machine Learning (ELM) in software effort estimation with other machine learning techniques, which are KNN, Linear Regression, SVR, Multilayer Perceptron (MLP). They have used the Pearson correlation coefficient to select the most influential features on the project effort. Out of twelve attributes in the original dataset, five attributes were chosen. All the attributes have been normalized prior application of the machine learning methods. It was concluded that the ELM model performs better than the other investigated methods due to the high learning speed and the better generalization capability of the ELM model.

Palaniswamy and Venkatesan [7] implemented a stacked ensemble model for software effort estimation by using Linear Regression, Multilayer Perceptron, Random Forest, and Ada Boost Regressor as base learners and Support Vector Regression as meta learner. The ensemble model has been trained using the ISBSG dataset, which has been preprocessed by applying different missing data treatment techniques, and the most influential features have been selected according to the Pearson Correlation Coefficient. The researchers have carried out three experiments to set the hyperparameters of the ensemble model. In the first experiment, the hyperparameters have been selected randomly without tuning, in the second and third experiments, the hyperparameters of the ensemble model have been tuned using Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) respectively. The outcome of the experiments revealed that tuning the hyperparameters of the ensemble model using PSO can give a better result.

Ullah et al. [12] have proposed the Flower Pollination Algorithm (FPA) for optimizing the coefficient of the COCOMO II model. Experiments were performed using a standard Turkish industry dataset. The researchers have stated that optimizing the coefficients of the COCOMO II model using FPA outperforms the original COCOMO II model and the optimization using the Bat algorithm.

## III. Proposed Work

In this article, a stacked ensemble model for software cost estimation has been proposed. Stacking is a powerful technique that combines the results of diverse learners for higher generalization and improved prediction accuracy. The results of the different learners which represent the first-level learners of base learners are combined efficiently by the second-level learner of meta learner [13].

The ISBSG dataset has been used to train the model. The dataset has been split into 67% for training, and 33% for testing during tuning the hyperparameters of the model, half of the test data have been used for validation, and the other half for actual testing. The performance of the ensemble model has been evaluated using K-fold cross-validation. The KNN, SVR, Random Forest, Bayesian Regressor, Linear Regressor, and MLP has been applied in the first level as base learners. In the second level, the Random Forest has been used as a meta-learner to combine the results of the base learners. The PSO algorithm has been applied to tune the hyperparameters of the learners of the proposed model.

### *A. Dataset Selection*

The ISBSG Release 2021 R2 dataset has been selected for the training of the proposed model. The ISBSG dataset contains a large volume of completed software projects. It consists of 10,600 projects which have been submitted from more than 26 different countries. The major sectors of the

projects are communications, insurance government, banking, manufacturing, medical and health care, financial, electronics and computers, and the service industry. The ISBSG dataset can have multiple potential uses; It can be used for benchmarking projects or organizations, estimating software development effort, cost, or schedule, assisting in evaluating changing the development environment, and for software development, engineering, and academic research. In terms of qualitative and quantitative measures, the ISBSG exceeds the available public datasets which are outdated and contain fewer projects.

### *B. Data Preparation*

The ISBSG dataset contains 10,600 projects with 252 attributes. However, the dataset suffers from a large amount of missing data which can have a bad influence on the prediction model. To enhance the estimation accuracy of the prediction model, the dataset must be preprocessed using various missing data treatment techniques, filtering, and feature selection techniques.

The data of ISBSG have different levels of quality rating. In this study, projects which have Data Quality Rating and Unadjusted Function Point Rating of either 'A' or 'B' have been selected. In order to increase the estimation accuracy, projects which have been sized using techniques other than IFPUG 4+ (International Function Points Users Group version 4 and greater) or NESMA (Netherlands Software Metrics Users Association) have been discarded. Since the intention of this study is to estimate the cost of software development, only projects with a Resource Level of '1' have been selected and the rest have been discarded because they incur the cost and effort of other operations.

To handle the missing data, columns, and rows that have more than 30% missing data have been dropped out, the mean imputation technique has been applied to numeric attributes and the most frequent category imputation has been applied to categorical attributes. After handling the missing data, the categorical attributes have been labeled encoded. The Least Absolute Shrinkage and Selection Operator (LASSO) has been applied to remove features that have the least correlation with the target variable. Outliers with three standard deviations from the mean have been discarded. In the end, out of 10,600 projects, only 4,698 remained, and 15 attributes out of 252 attributes have been selected and the target variable is the Normalized Work Effort Level 1.

### *C. Model Construction*

The stacking regressor is a powerful machine-learning technique that combines the prediction of multiple base learners by applying a meta-learner. In this research six different machine learning methods have been used as base learners:

- **K-nearest neighbors (KNN)** algorithm is a supervised machine learning technique that can be used for both classification and regression. It tries to predict the value of the testing data by calculating the distance with the training points and selecting the K number of points that is close to the test data [14].
- **Support Vector Regression (SVR)** is a supervised machine-learning technique that is used for prediction. It shares the same principles with the Support Vector Machine (SVM) which is used in classification problems. The main principle in SVR is to find the best fit line which has the maximum number of points [15].
- **Random Forest** is an ensemble machine-learning technique that can be used for regression and classification problems. It consists of many decision trees; the outputs of these individual trees are aggregated to generate the final prediction [16].
- **Bayesian Regression** is a powerful machine learning technique that provides point estimates of the regression parameters with an entire distribution over these parameters. Bayes can provide better performance than other complicated techniques in many applications [17].
- **Linear Regression** is a linear approach to analyzing the relationship of a variable based on other variables. It is used to predict the value of the dependent variable given the values of the independent variables. Linear regression is a widely used technique in prediction [18].
- **Multilayer Perceptron (MLP)** is a class of feedforward artificial neural networks that can be used to solve linear and non-linear problems. It consists of three layers: the input layer, the hidden layer, and the output layer [19].

The hyperparameters for the aforementioned algorithms have been tuned using Particle Swarm Optimization (PSO) which is a bio-inspired optimization algorithm. It tries to find the optimal solution by iteratively searching in the solution space [20]. The Random Forest was used as a meta-learner to compute the final prediction for the stacking ensemble model with 5-fold cross-validation.

### *D. Evaluation Criteria*

Software cost prediction models need to be evaluated in terms of estimation accuracy in order to measure model performance and gain knowledge of whether the given estimates are accurate or not. There are several available methods that can be used in performance evaluation. In this study we adapted the commonly used performance evaluation metrics which are the Mean Magnitude of Relative Error (MMRE), Percentage of Predictions (Pred(p)), and R-Squared (R2):

- **The mean Magnitude of Relative Error (MMRE)** is defined as the average of the difference between the actual value and the estimated value divided by the actual value.

$$MMRE = \frac{1}{n}\sum_{i=1}^{n} MRE \quad (1)$$

$$MRE = \left|\frac{Ei - \hat{E}i}{Ei}\right| \quad (2)$$

Where $n$ is the total number of projects, $Ei$ is the actual effort of $ith$ project, and $\hat{E}$ is the estimated effort. A lower value of MMRE indicates a low level of estimation error [21].

- **Pred(p)** is the percentage of prediction which measures the percentage of the estimated values that are within (p) percentage of the actual value [22].

$$PRED(p) = \frac{1}{n}\sum_{i=1}^{n} \begin{cases} 1, & MRE \leq p \\ 0, & otherwise \end{cases} \quad (3)$$

Where $n$ is the total number of projects, $p$ is the percentage of prediction which equals 25, and $MRE$ is the Magnitude of Relative Error as defined in equation (2).

- **R-Squared** is defined as the proportion of variance in the dependent variable that is predicated from the independent variable. R2 value falls within ranges between 0 and 1. The closer R2 is to 1 the better.

$$R2 = 1 - \frac{RSS}{TSS} \quad (4)$$

Where RSS is the residual sum of squares and TSS is the total sum of squares [23].

## IV. Results and Discussion

This section presents a discussion of the results obtained through the experiments on the stacking ensemble model using the ISBSG dataset. Each of the hyperparameters for the base learners as shown in Table I have been optimized using the PSO algorithm. The best-obtained values for the hyperparameters are shown in Table I.

TABLE I. HYPERPARAMETERS OF BASE LEARNERS

| *Method* | *Hyperparameter* | *Value* | | |
|---|---|---|---|---|
| | | *Min* | *Max* | *Best* |
| KNN | Neighbors | 1 | 4698 | 10 |
| SVR | Epsilon | 0.1 | 10 | 6.53 |
| Random Forest | Estimators | 10 | 1000 | 782 |
| Bayesian Regressor | Iterations | 300 | 3000 | 767 |
| MLP | Hidden layer size | 100 | 1000 | 736 |

The preprocessed dataset which contains 4,698 projects, each consists of 15 attributes: Data Quality Rating, Year of Project, Industry Sector, Organization Type, Application Group, Application Type, Language Type, Count Approach, Functional Size, Relative Size, Adjusted Function Points, Normalized Level 1 PDR (UFP), Project Elapsed Time, Project Activity Scope, Recording Method have been used in the software cost estimation process that has been performed in two stages because most of the projects in the dataset have been included without their cost. In the first stage, the software development effort has been estimated with the target variable as the Normalized Work Effort Level 1.

TABLE II. PERFORMANCE OF LEARNERS

| *Method* | *Performance Evaluation* | | |
|---|---|---|---|
| | *MMRE* | *R2* | *Pred25* |
| KNN | 0.77 | 0.53 | 36% |
| SVR | 1.65 | -0.11 | 19% |
| Random Forest | 0.04 | 0.98 | 95% |
| Bayesian Regressor | 0.92 | 0.68 | 49% |
| Linear Regression | 0.93 | 0.68 | 49% |
| MLP | 0.28 | 0.98 | 88% |
| Stacked Ensemble | 0.03 | 0.99 | 98% |

In the second stage, the software cost has been estimated with the target variable as the Total project cost depending on the estimated effort using Linear Regression which gives great results due to the high correlation between cost and effort. Table II shows the results obtained using the different base learners and the Random Forest as the meta-learner.

In order to obtain the cost estimation, the effort estimation obtained through the stacked ensemble model has been used as input to the cost estimation model together with the Functional Size and the Relative Size. The cost estimation model has been built with Linear Regression using the Functional Size, Relative Size, and Normalized Work Effort Level 1 as the independent variables and the Total project cost as the target variable. The data which have been used in building the cost estimation model has been

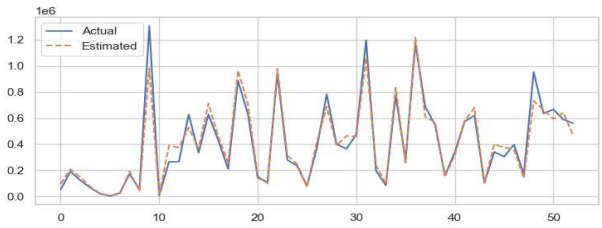


FIGURE 2. Actual vs. Estimated Cost

filtered according to the Cost currency, only projects with Euro as currency have been selected.

As shown in Figure 2, the proposed model has given predictions very close to the actual effort. The proposed model has been compared to other models in the literature as shown in Table III. It can be seen from the performance measurer that the proposed model has outperformed other models with a minimum MMRE value equal to 0.14. In terms of R-squared and Pred25, the proposed model has obtained 0.93 and 90% values respectively.

TABLE III. COMPARING THE PROPOSED MODEL WITH OTHER MODELS

| *Method* | *Year* | *MMRE* |
|---|---|---|
| Proposed Ensemble Model | 2022 | 0.14 |
| Whale–crow optimization [24] | 2019 | 0.24 |
| Gray Wolf Algorithm [25] | 2021 | 0.40 |
| Dolphin Bat Algorithm [10] | 2020 | 14.5 |
| PSO Optimized Fuzzy Model [26] | 2020 | 21.7 |
| Flower Pollination [12] | 2019 | 34.1 |

## V. Conclusions

Accurate software development cost estimation is an essential activity in project management. It can have a huge impact on the project's success or failure. Machine learning-based approaches have achieved great improvement in the prediction accuracy in compression with traditional methods. In this study, a stacked ensemble model using Random

Forest as a meta-learner of diverse base learners has been built. As it can be concluded from the results obtained, the proposed model outperformed other models in the surveyed literature.

Data preprocessing such as filtering, feature selection, and missing data treatment techniques are very crucial and have a huge impact on the prediction accuracy of the model. According to the experiments, the best-performed model was the stacking regressor using Random Forest with MMRE equal to 0.03, R2 equal to 0.99, and Pred25 of 98%, and the worst-performing model was SVR with MMRE of 1.65, R2 of -0.11, and Pred25 of only 19%.

It is recommended to use the ISBSG dataset and avoid publicly available datasets as they are outdated, no longer maintained, and contain fewer data. For future work, other machine learning techniques can be used in building the ensemble model, and other optimization techniques can be used in tuning the hyperparameters of the model.

## ACKNOWLEDGMENT

The authors would like to thank the ISBSG organization for providing a generous discount on the dataset which without it this work could not be completed.

## REFERENCES

[1] M. M. Al Asheeri and M. Hammad, "Machine Learning Models for Software Cost Estimation Estimation," 2019.

[2] Y. Mahmood, N. Kama, A. Azmi and M. Ali, "Improving Estimation Accuracy Prediction of Software Development Effort: A Proposed Ensemble Model," *The 2nd International Conference on Electrical, Communication and Computer Engineering (ICECCE),* 2020.

[3] "CHAOS REPORT," The Standish Group International, Inc., 2015.

[4] N. RANKOVIC, D. RANKOVIC, M. IVANOVIC and L. LAZIC, "A New Approach to Software Effort Estimation Using Different Artificial Neural Network Architectures and Taguchi Orthogonal Arrays," vol. 9, 2021.

[5] A. BaniMustafa, "Predicting Software Effort Estimation Using Machine Learning," 2018.

[6] F. Akhbardeh and H. Reza, "A Survey of Machine Learning Approach to Software Cost Estimation," *IEEE International Conference on Electro Information Technology (EIT),* pp. 405-408, 2021.

[7] S. K. Palaniswamy and R. Venkatesan, "Hyperparameters tuning of ensemble model for software effort estimation," *Journal of Ambient Intelligence and Humanized Computing,* 2020.

[8] L. Cheikhi and A. Abran, "PROMISE and ISBSG Software Engineering Data Repositories: A Survey," *Joint Conference of the 23nd International Workshop on Software Measurement (IWSM) and the Eighth International Conference on Software Process and Product Measurement (Mensura),* 2013.

[9] A. K. Bardsiri, "An Intelligent Model to Predict the Development Time and Budget of Software Projects," *International Journal of Nonlinear Analysis and Applications (IJNAA ),* vol. 2, no. 11, pp. 85-102, 2020.

[10] A. A. Fadhil, R. G. Alsarraj and A. M. Altaie, "Software Cost Estimation Based on Dolphin Algorithm," *IEEE Access,* vol. 8, pp. 75279-75287, 2020.

[11] H. D. PEREIRA DE CARVALHO, R. FAGUNDES and W. SANTOS, "Extreme Learning Machine Applied to Software Development Effort Estimation," vol. 9, 2021.

[12] A. Ullah, B. Wang, J. Sheng, J. Long, M. Asim and F. Riaz , "A Novel Technique of Software Cost Estimation Using Flower Pollination Algorithm," 2019.

[13] P. Varshini, A. Kumari and V. Varadarajan, "Estimating Software Development Efforts Using a Random Forest-Based Stacked Ensemble Approach," *Electronics,* vol. 10, 2021.

[14] M. J. Madari, H. Bahrami and M. Niazi, "Improve Software Effort Estimation using Weighted KNN in New Feature Space," *INTERNATIONAL JOURNAL FOR RESEARCH & DEVELOPMENT IN TECHNOLOGY,* vol. 11, no. 4, pp. 421-426, 2019.

[15] A. G. Floriano, C. L. Martín, C. Y. Márquez and A. Abrand, "Support vector regression for predicting software enhancement effort," *Information and Software Technology,* vol. 98, pp. 99-109, 2018.

[16] Z. Abdelali, H. Mustapha and N. Abdelwahed, "Investigating the use of random forest in software effort estimation," 2018.

[17] W. Zhang, Y. Yang and Q. Wang, "Using Bayesian Regression and EM algorithm with missing handling for software effort prediction," *Information and Software Technology,* 2015.

[18] W. Rhmann, B. Pandey and G. A. Ansari, "Software effort estimation using ensemble of hybrid search-based algorithms based onmetaheuristic algorithms," *Innovations in Systems and Software Engineering,* 2021.

[19] I. M. Keshta, "Software Cost Estimation Approaches: A Survey," *Journal of Software Engineering and Applications,* vol. 10, pp. 824-842, 2017.

[20] Z. Shahpar, V. K. Bardsiri and A. K. Bardsiri, "Polynomial analogy-based software development effort estimation using combined particle swarm optimization and simulated annealing," *Concurrency and Computation Practice and Experience,* vol. 33, no. 10, 2021.

[21] R. K. Sachan, A. Nigam, A. Singh, S. Singh, M. Choudhary, A. Tiwari and D. S. Kushwaha, "Optimizing Basic COCOMO Model using Simplified Genetic Algorithm," 2016.

[22] M. K. Mahadev and D. G. Gowrishankar, "Estimation of Effort in Software Projects using Genetic Programming," 2020.

[23] F. B. Ahmad and L. M. Ibrahim, "Software Development Effort Estimation Techniques Using Long Short Term Memory," *2022 International Conference on Computer Science and Software Engineering CSASE,* pp. 182-187, 2022.

[24] S. W. AHMAD and G. R. BAMNOTE, "Whale–crow optimization (WCO)-based Optimal Regression model for Software Cost Estimation," *Sādhanā ,* vol. 44, no. 94, 2019.

[25] C. A. u. Hassan and M. S. Khan, "An Effective Nature Inspired Approach for the Estimation of Software Development Cost," *16th International Conference on Emerging Technologies (ICET),* 2021.

[26] S. Chhabra and H. Singh, "Optimizing Design of Fuzzy Model for Software Cost Estimation Using Particle Swarm Optimization Algorithm," *International Journal of Computational Intelligence and Applications,* vol. 19, no. 1, 2020.