Journal of Artificial Intelligence and Data Mining (JAIDM), Vol. 11, No. 1, 2023, 39-51.

Shahrood University of Technology

Journal of Artificial Intelligence and Data Mining (JAIDM)
Journal homepage: http://jad.shahroodut.ac.ir

Research paper

FEEM: A Flexible Model based on Artificial Intelligence for Software Effort Estimation

Amin Moradbeiky*

Department of Computer Engineering, Zabol Branch, Islamic Azad University, Zabol, Iran.

Article Info

Article History:
Received 13 September 2021
Revised 24 October 2021
Accepted 04 January 2022

DOI:10.22044/jadm.2022.11161.2265

Keywords:
Development Effort Estimation, Lightning Search Algorithm, Neural Networks, Software Project.

**Corresponding author: ampayam@yahoo.com (A. Moradbeiky).*

Abstract

Managing software projects due to its intangible nature is full of challenges when predicting the effort required for development. Accordingly, there exist many studies with the attempt to devise the models to estimate the efforts necessary in developing software. According to the literature, the accuracy of the estimator models or methods can be improved by the correct application of data filtering or feature weighting techniques. Numerous models have also been proposed based on the machine learning methods for data modeling. In this work, we propose a new model consisting of the data filtering and feature weighting techniques in order to improve the estimation accuracy in the final step of data modeling. The model proposed in this work consists of three layers. The tools and techniques in the first and second layers of the proposed model select the most effective features, and weight the features with the help of LSA (Lightning Search Algorithm). By combining LSA and an artificial neural network in the third layer of the model, an estimator model is developed from the first and second layers, significantly improving the final estimation accuracy. The upper layers of this model filter out and analyze the data of the lower layers. This arrangement significantly increases the accuracy of the final estimation. Three datasets of real projects are used in order to evaluate the accuracy of the proposed model, and the results obtained are compared with those obtained from different methods. The results are compared based on the performance criteria, indicating that the proposed model effectively improves the estimation accuracy.

1. Introduction

The required time and effort for software development are two important parameters in any software project. Prediction of the required effort, to achieve the schedule and other objectives of the project, is one of the important duties of the project managers. Therefore, the right decisions of the software project managers and the accurate effort estimation play an important role in the success of these projects. To this end, the project managers must be able to identify the effective parameters on the effort of the project, the relationship between them, and their relation effect on effort. Usually the project managers use their own experience to estimate the effort of the projects. However, the extreme dynamics of the software project environment make unreliable the obtained estimates from this method. Therefore, the project managers require tools and methods to enable them to estimate the required effort of project development accurately. Consequently, this issue is considered as a major challenge for the researchers and the practitioners in the software industry. The estimation methods are divided to the algorithmic and non-algorithmic categories [1], the first is based on the

mathematical and the second on the heuristic and metaheuristic methods.

In both methods, usually a model is devised for a number of tools and algorithms proposed to estimate the software development effort. In each one of these models, the tools and algorithms are combined in their unique sense to provide a precise estimate of the software development effort. Some of these tools or algorithms are applied as an intermediate tool to increase the accuracy of the method. The following studies support this claim in increasing the accuracy of the neural network:

- The neural networks are one of the most commonly adopted methods in AI; The Elman's neural network has been applied in [2] for the software development effort estimation.
- Rankovic *et al*. [3] have proposed four new models based on the artificial neural network, utilizing five datasets to test them.
- Kumar *et al*. [4] have used neural networks to deep learning in software effort estimation.
- The dilation-erosion-linear perceptron was introduced in 2012. This method is applied in many articles for prediction. This method will not be sufficient if there exists complexity of input/output. Araujo *et al*. [5] have optimized the structure of this perceptron using the descending gradient in the learning process, and have used it in software effort estimation.
- A combination of satin bowerbird optimization algorithm (SBO) and the neuro-fuzzy (ANFIS) has been applied to increase the accuracy in predicting the software error [6].

A number of researchers seek to increase the accuracy of the Analogy-Based Estimation (ABE) method through different tools or use ABE as a tool to increase the accuracy of the other tools.

- The ABE method has been commonly used for software effort estimation by the researchers. The differential equation (DE) algorithm has been applied in the similarity function to weight the features, named Differential evolution in Analogy-Based Estimation (DABE) [7], in order to improve the efficiency of this method.
- There exists no exact definition on the project similarity. A similarity region has been identified by [8] for feature selection in similar projects through the Case-Based Reasoning (CBR) concept.
- One of the algorithms combined through the ABE methods is the genetic algorithm [9].
- Application of the particle swarm optimization (PSO) algorithm to increase ABE precision [10] and a hybrid model from PSO and simulated annealing algorithm to improve ABE performance [11] has been proposed.

The fuzzy logic-based tools and technique combination with other methods have been used in some studies for performance and accuracy improvement.

- The estimation model (EM) proposed by [12] is to divide the projects into the categories with similar distribution parameters, followed by adopting the fuzzy method used in estimation and applied from the firefly algorithm in the rule-base system for selection.
- The effective parameters on the estimation have been proposed by [13], where an attempt has been made to increase precision through the fuzzy method.
- A combination of two algorithmic and non-algorithmic methods COCOMO and NEURO-Fuzzy has been applied in [14], where the accuracy of the estimation increased by sending the outputs of NEURO-Fuzzy to the COCOMOII.
- Idri *et al*. have assessed the effect of the missing data (MD) techniques on ABE and fuzzy-analogy [15].
- Usually the fuzzy logic is applied in solving the error prediction problem since it can perform with incomplete data, while the main problem is the great volume of rules that slows the decision-making process. An attempt has been made by [16] to reduce this volume by applying the fuzzy controllers instead of the fuzzy logic.
- Karimi and Gandomani have used a combination of differential evolution algorithm and fuzzy-neural network for software development effort estimation modeling [17].
- Chhabra and Singh have used the optimizing design of fuzzy model for software effort estimation using the PSO algorithm [18].

Some researchers only use the tools based on the algorithmic methods.

- COCOMO has been proposed by [19], COCOMOII has been proposed by [20], SLIM has been proposed by [21], function point analysis has been proposed by [22], and the Dotti model has been proposed by [23].

- The regression-based methods are the linear regression methods [24, 25], non-linear regression methods [25], and tree regression methods [26, 27].

The artificial intelligence algorithms can improve the efficiency of the formulated methods by searching for the appropriate configuration for these methods. This approach has been followed in some articles.

- The updated K-modes clustering basic algorithms have been applied in effort prediction. In the model proposed by [28], the Beysian belief network is constructed from of the COCOMO model, where the intervals are of fuzzy numbers, and then the PSO algorithm and genetic algorithm (GA) are combined to improve the software effort estimation.
- The machine learning algorithms are commonly applied in problem estimation. Two different types of Support Vector Machine (SVMs) have been applied by [29] to predict effort and compare with the other methods like neural networks and decision tree. Various feature selection methods have also been used for performance optimization of the machine learning-based methods [30].
- The meta-heuristic algorithms are commonly applied by the researchers in many cases. A hybrid meta-heuristic algorithm consisting of Cuckoo Optimization Algorithm (COA), harmonic Search [31], and DE algorithm has been applied to optimize the COCOMO parameters [32], and to improve the software effort estimation.

Some studies emphasize identifying the key project features and their relationship with the software development effort. There has been an emphasis on identifying the interrelated features influencing the software development effort [33]. The features influencing effort have been identified by a neural network [34]. The PSO algorithm [35] and the Bayesian technique [36] have been used to identify the features influencing the software development effort.

Table 1. Various methods in software effort estimation.

Study	Year	Dataset	Evaluation method	Method	Ref. No.
1	2019	21 Projects (1 dataset)	MMRE, Pred, MSE	ANN	2
2	2021	COCOMO, NASA, Kemerer	MAE, Pred, MMRE	ANN	3
3	2017	ISBSG, Albrecht, Kemerer	MMRE, Pred	ANFIS	6
4	2007	CF, DPS	MMRE, Pred, MdMRE	ABE	9
5	2012	CF, DPS, ISBSG	MMRE, Pred	PSO, ABE	10
6	2019	Desharnais, COCOMO	MMRE, Pred	Firefly algorithm	12
7	2019	4 Projects (1 dataset)	MMRE, VAF	Fuzzy	13
8	2018	COCOMO	MMRE	Neuro fuzzy	14
9	2021	Kemerer, Albrecht	MMRE, Pred	ANFIS	17
10	2020	COCOMO	MMRE, Pred	PSO, Fuzzy	18
11	2016	COCOMO	MMRE	Bayesian network	28
12	2018	ISBSG	MAE	SVR	29
13	2017	COCOMO	MMRE	Cuckoo search	31
14	2018	COCOMO	MMRE, Pred, MAE	DE	32
15	2020	ISBSG, Desharnais	MMRE, Pred	ACO, ABE	44

A novel model is proposed in this work by analyzing the models previously presented in the literature. The data preparation tools have been proposed in some studies in order to improve the estimation accuracy. Some studies have emphasized on the different effectiveness of various project features on the software development effort, and attempts have made to propose a model to exactly estimate the effort considering the project features and the effectiveness of different features. The effectiveness has been defined as a coefficient in the literature. Data modeling by the machine learning methods has also been performed in some studies. Accordingly, in the proposed model, various separately used techniques and tools in the literature for improving the estimation accuracy were adopted in a model with separate layers. Each layer in this model increases the accuracy of the next layer. Simply speaking, the output of each layer in this model is the input to the next layer, improving the final performance of the proposed model.

Section 2 discusses the ABE method used for estimating the software development effort. Section 3 introduces the criteria for calculating the accuracy of the proposed model. Section 4 introduces the proposed model. Section 5 discusses a cross-validation method for evaluating the stability of the proposed model results. Section 6 introduces three datasets of real projects used for testing the proposed model. Section 7 introduces the techniques compared with the proposed model. Section 8 presents the test results of the proposed model. The model results are analyzed in Section 9.

2. Analogy-based Estimation (ABE) Method

The estimation methods are of the two algorithms and non-algorithm. Since the first methods are not appropriate to be adopted in the dynamic environment of software projects, the second methods are applied in this context, making ABE one of the most applicable methods. The ABE method is adopted in the unspecified value estimation of single feature (i.e. effort or cost) of one project. The steps of this method are described in the following sub-sections.

2.1. Similarity function

The similarity of projects through studying the features with certain value(s) is determined through this function. For this purpose, the following Euclidean, Eq. 3 and Manhattan Eq. 4 similarity determination methods, are applied. The project features include both the digit and non-digit groups. With respect to the digit features, in both the methods, the space of digit features is estimated for the project's difference estimation. With respect to the non-digit features, the level of difference is set at 0 or 1. These methods differ in the digit feature value space estimation context, where the P and P' statements constitute the study projects and f_i and f'_i constitute the i^{th} feature of the P and P' projects, respectively. The result obtained reveals the similarity level between the two projects.

$$\delta = 0.0001$$

$$sim(p,p') = \frac{1}{\left[\sqrt{\sum_{i=1}^{n} w_i \, Dis(f_i, f_i')} + \delta\right]} \tag{1}$$

$$Dis(f_i, f_i') = \begin{cases} (f_i - f_i')^2 & \text{if } f_i \text{ and } f_i' \text{ are numerical or ordinal} \\ 0 & \text{if } f_i \text{ and } f_i' \text{ are nominal and } f_i = f_i' \\ 1 & \text{if } f_i \text{ and } f_i' \text{ are nominal and } f_i \neq f_i' \end{cases}$$

$$\delta = 0.0001$$

$$sim(p,p') = \frac{1}{\left[\sum_{i=1}^{n} w_i \, Dis(f_i, f_i') + \delta\right]} \tag{2}$$

$$Dis(f_i, f_i') = \begin{cases} |f_i - f_i'| & \text{if } f_i \text{ and } f_i' \text{ are numerical or ordinal} \\ 0 & \text{if } f_i \text{ and } f_i' \text{ are nominal and } f_i = f_i' \\ 1 & \text{if } f_i \text{ and } f_i' \text{ are nominal and } f_i \neq f_i' \end{cases}$$

2.2. Solution function

This function is applied in the effort estimation of one project according to the effort of k projects with more similarities.

$$C_p = \sum_{k=1}^{K} \frac{Sim(p, p_k)}{\sum_{i=1}^{K} Sim(p, p_i)} C_{p_k} \tag{3}$$

where P is the project, the effort value of which is intended to be estimated. Symbol P_i is the i^{th} project of K more similar project. The symbol C_{Pi} is a certain value to be estimated from the j^{th} more similar project.

2.3. Best value of K

The K value is applied in the effort estimations with a high accuracy. An appropriate K value mostly depends on the studied projects. If the difference in the studied projects is slightly high, the K value accuracy reduces because the effective projects manifest more differences at the final stage of estimation. If the studied projects are too close to one another, the low value of K results prevents the study of similar projects. The existence of these projects in the final stage is of a positive influence on the results' accuracy. This accuracy is due to a reduction in the noise rate during the estimation process. Consequently, no constant value of K can be considered, and thus it is better for K to be determined in its dynamic sense.

According to the above-mentioned points, no constant value of K can be considered. Therefore, it is better for K to be determined in its dynamic sense.

3. Equations for Estimation Error Calculation

In this section, the utilized equations are evaluated for the accuracy of the proposed model, and comparison with the other methods is made. These equations are commonly used for the accuracy evaluation by the researchers in the field. The results of equations are displayed as diagrams for a better accuracy evaluation and comparison. The utilized equations are presented in Eqs. (4 to 8), relative error (RE), magnitude of relative error (MRE), median magnitude of relative error (MdMRE), prediction percentage (PRED), and mean of absolute error (MAE).

$$RE = \frac{Estimate - Actual}{Actual} \tag{4}$$

$$MRE = \frac{|Estimate - Actual|}{Actual} \tag{5}$$

$$MdMRE = Median(MRE) \tag{6}$$

$$PRED(X) = \frac{A}{N} \tag{7}$$

$$MAE = \frac{1}{N} \sum_{1}^{n} |Estimate - Actual| \tag{8}$$

4. Proposed Model

This paper presents a new model called Flexible Effort Estimation Model (FEEM). FEEM is composed of two sections: training and test. The training section of this model consists of three layers, each responsible to refine the data and

enhance the precision of estimation. The model of layer 1 is shown in Figure 1, where the best features are selected based on the feature selection and ABE method with several iteration. At every one of the iterations, a subset of features is selected, and the MdMRE error value is calculated for that set of feature. The iteration continues until the whole set of selected features end. What is obtained here is a set of the best features with the highest effect on software development effort estimation, which is applied to send as an input to the next layer.

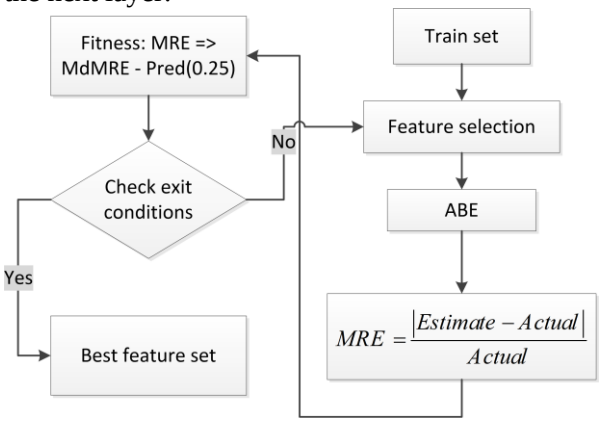

Figure 1. A flowchart of training section model, layer 1.

The layer 2 model is shown in Figure 2, which undergoes training through the selected features as its input. This model is iterated for many times through the LSA algorithm, and at each iteration, the LSA algorithm suggests an appropriate setting for ABE. The ABE method processes the projects and estimates them based on the settings suggested by the LSA algorithm. This process runs until the estimation error reaches a specific threshold or the iterations are ended. Finally, the best setting for ABE is the result obtained through implementing the model of this layer. The obtained settings are applied as the input for layer 3.

The second layer of the proposed model includes a hybrid model of the ABE and LSA methods. The ABE method searches for the most similar projects with the target project to estimate software development effort based on the features adaptation. The ABE method uses the LSA algorithm to increase the estimation accuracy. The LSA algorithm tries to propose the most appropriate configuration for the ABE method, and helps it to provide a more accurate estimation. The configuration proposed by the LSA algorithm differs based on the project conditions and its features. On the other hand, the first layer helps increase the second layer's accuracy by processing the input data to the first layer. Simply speaking, a higher quality data enters the second layer with the help of the first layer. The estimate obtained from the second layer is not the final estimation. In the third layer, a model is developed for the estimator of first and second layers and based on their input and output. This layer leads to an improved final estimated accuracy.

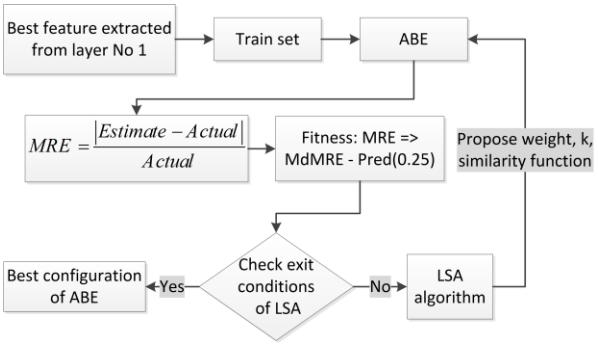

Figure 2. A flowchart of training section model, layer 2.

The layer 3 model is shown in Figure 3 that undergoes the predicting proper estimation error based on a project's data. In order to estimate the prediction error, the Artificial Neural Network (ANN) is applied. The proper configuration of ABE obtained in layer 2 is received as its input, and the best features are obtained in layer 1. This layer's model is iterated through the LSA algorithm, where at each iteration, the LSA algorithm proposes proper values for b and w of ANN. Here, ANN predicts the estimation error for each project, and ABE estimates the effort. The resulting values are applied in Eq. 9 for estimation.

$$E_{Final}[i] = \left| E_{ABE}[i] - (Error[i] \times Th) \right| \quad (9)$$

where *i* is the number of projects, E_{ABE} is the estimation from the ABE method, Error is the error proposed by ANN, and the *Th* coefficient is the percentage of effect of suggested error on the value of estimation. The result of this equation is the value of the final estimation.

After the final estimation of each project is run, the resulting value is applied in Eqs. 5, 6, 7, and 8. Consequently, the estimation error is calculated based on the settings suggested by the LSA algorithm. The obtained estimation error is returned to the LSA algorithm as a feedback, and this process goes on until the error resulting from estimation does not reach a specific threshold or iteration of the LSA algorithm ends. This layer will provide the best settings for estimation of the prediction error through ANN. These proposed settings reduce the final estimation error.

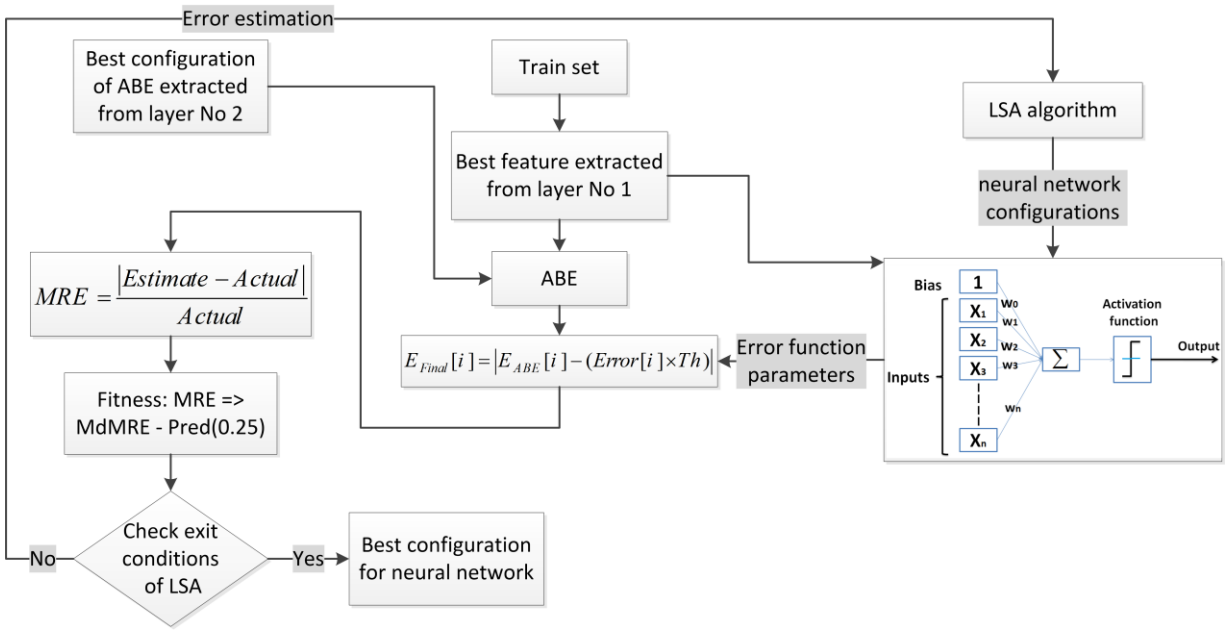

Figure 3. A flowchart of training section model, layer 3.

The test section flowchart is shown in Figure 4, where the set of test projects is estimated through the settings proposed in layer 2 for ABE and the features specified in layer 1 based on the estimation error predicted by layer 3. The MdMRE and PRED values resulting from running this stage are considered as the estimation errors.

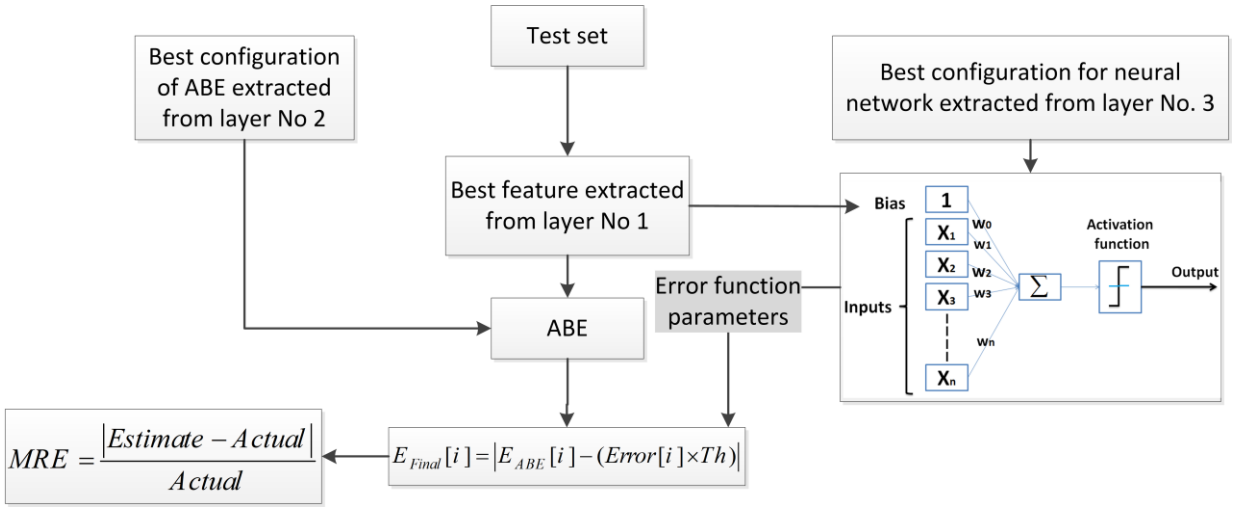

Figure 4. Test section model flowchart.

5. Validation Method

Based on the proposed model, the projects must be divided into the two groups of training and test. The arrangement of projects in dividing process effects on the accuracy of the proposed model [37]. For sustainability provement of the models, different evaluation methods including 3 fold, 10 fold, etc. can be used. Each one of these methods provides a specific arrangement for the projects. Based on the performed study [37], leave-one-out (LOO) is the best method for evaluation, and its achieved accuracy is independent from arrangement of projects. In this work, the LOO method is adopted.

6. Introducing Datasets

In the testing stage of the proposed model, the dataset of real projects are utilized. These datasets have been applied by many researchers. The details of the data analysis of these datasets are tabulated in Table 2.

Table 2. Datasets

Name	Number of sample	Number of features	Mean (effort)
COCOMO	63	17	683
Desharnais	77	10	4795
Maxwell	62	26	8223

The desharnais dataset consists of 81 real software projects. This dataset has been collected from the Canadian software houses. The projects in the

desharnais dataset are described by 11 features. In this dataset, one of the features named 'Cost' is dependent and ten other features named 'TeamExp', 'ManagerExp', 'YearEnd', 'Duration', 'Transactions', 'Entities', 'AdjFP', 'AdjFactor', 'RawFP', and 'Dev.Env', which are independent. In this work, only 77 projects of this dataset are used for tests since the other 4 projects have defective data.

The Maxwell dataset contains data on 62 real software projects. There is one dependent feature called 'effort' and 25 independent features indexed from 1 to 25 in this dataset.

The cocomo dataset contains data on 63 real software projects. The independent features are 'rely', 'data', 'cplx', 'time', 'stor', 'virt', 'turn', 'acap', 'aexp', 'pcap', 'vexp', 'lexp', 'modp', 'tool', 'sced', and 'loc', also 'actual' is the only dependent feature in this dataset.

7. Techniques

This proposed model is compared with the following methods for evaluating the accuracy:

- Ordinary Least Squares (OLS): this method is based on the regression and the best line of regression.
- Robust Regression (ROR): ROR uses regression for estimation. This method utilizes weighting to increase the estimation accuracy in the unusual data [38].
- Multivariate Adaptive Regression Splines (MARS): MARS is a non-linear and non-parametric regression method, indicative of some interesting features like the ease in interpretation and the ability to model complex non-linear correlations with a rapid output [39].
- Classification and Regression Tree (CART): One of the commonly used methods for data classification is the CART method. The CART method adopts decision tree for data classification [40].
- M5: The M5 method utilizes the modeling technique for data estimation, and the developed model has a tree structure. This method separately computes a linear regression for each leaf in the developed tree model [41].
- Multi-Layered Perceptron (MLP): Neural network is a non-linear modeling technique. MLP-based neural network has been applied by many researches. This method is based on a network of neurons in an input layer, one or more hidden layers, and an output layer [42].
- Case-Based learning reasoning (CBR): The CBR operator searches for the most similar sample to the sample we intend to estimate. The similarity of the samples is calculated through this method. In this method, K determines the number of most similar samples that must be used for data estimation [43].

8. Testing Datasets

The objective of testing this model is to evaluate the degree of its precision. The tests are run on the introduced datasets. The results here are displayed and analyzed through separate datasets. The precision of this model in these tests is calculated and displayed through the criteria, and the equations are introduced in Section 3.

8.1. Desharnais dataset test

Tde desharnais dataset test is selected as the first, the specifications of which are presented in Section 6-2. The MRE value obtained from implementing this proposed model is expressed in Figure 5. The MdMRE value for this test is 0.22, and the PRED value is 0.51.

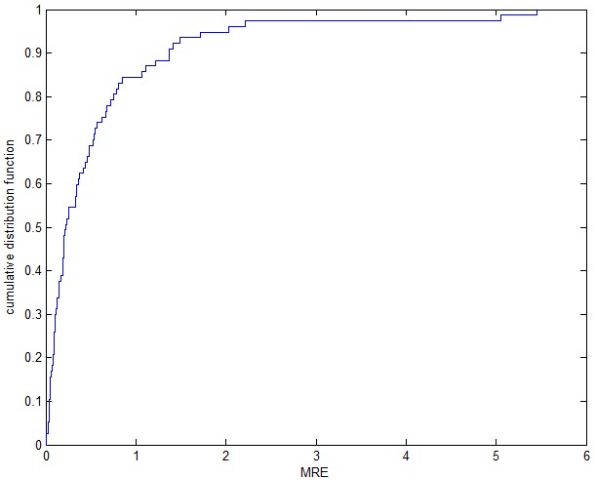

Figure 5. MRE error frequency distribution with FEEM model for desharnais dataset.

The frequency distribution diagram of the MRE error is graphed in Figure 5, where the percentages of distribution of different values of MRE error are exposed. The horizontal axis of this graph indicates the MRE quantity. The vertical axis of this graph represents the percentage of the projects with a specific MRE quantity.

As observed in Figure 5, a high percentage of errors falls within a range less than 0.5. The higher slope of this diagram in one area signifies a higher percentage of error distribution within that specified range. As the graph moves toward bigger errors, its slope becomes less, even reaches zero, indicating fewer projects with low estimation accuracy.

8.2. COCOMO dataset test

The specifications of this dataset are presented in Section 6-1. The MRE value obtained from implementing this proposed model is expressed in Figure 6. The MdMRE value for this test is 0.53, and the PRED value is 0.19.

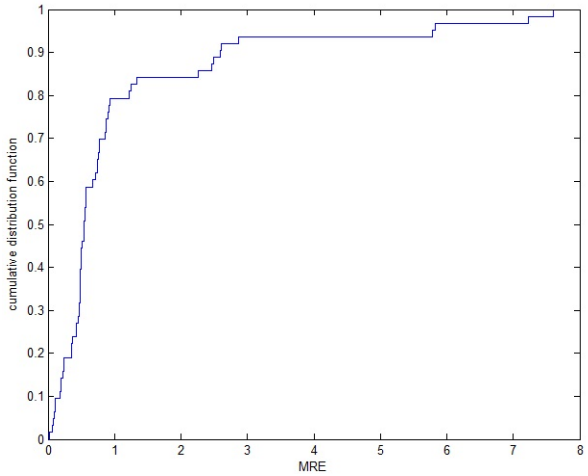

Figure 6. MRE error frequency distribution with FEEM model for COCOMO dataset.

The frequency distribution diagram of the MRE error for the COCOMO dataset is shown in Figure 6, where the horizontal axis represents the MRE quantity, and the vertical axis represents the percentage of the projects with a specific MRE quantity. As observed here, a high percentage of errors falls within a range less than 1. This is obvious from Figure 6. The higher slope areas of the diagram contain small errors, and as the diagram moves towards bigger errors, its slope becomes less, and even reaches zero, indicating fewer projects with a low estimation accuracy.

8.3. Maxwell dataset test

The next test is run on the Maxwell dataset, specifications of which are given in Section 6-3. The MRE value obtained from implementing this proposed model is expressed in Figure 7.

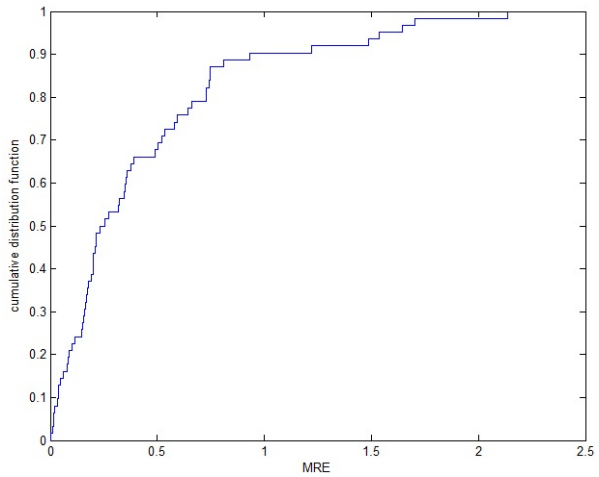

Figure 7. MRE error frequency distribution with FEEM model for Maxwell dataset.

The MdMRE value for this test is 0.24, and the PRED value is 0.5. The frequency distribution diagram of the MRE error for the Maxwell dataset is drawn in Figure 7, where the horizontal axis represents the MRE quantity, and the vertical axis represents the percentage of the projects with a specific MRE quantity. As observed here, about 70% of the projects are estimated with less than 0.5 error, and the big errors are of a small distribution.

9. Analysis and Comparison of Results

For an accuracy comparison of the proposed model with the other methods, numerous tests are performed. These tests are performed on the same test conditions with the proposed model. The results of the tests for comparison are shown in Tables 3 and 4. In these tables, the results of the tests are comparable with each other. The results obtained indicate the high precision of the FEEM model. Moreover, the PRED value of the FEEM model reveals a high precision estimation rate in this model.

This precision is due to the fact that the refining filters are separated, which, in turn, increase the data precision. In the estimation research works, in the cases where the data refining methods are applied, they join the estimation process, which leads to many problems. Combining the refinement and estimation processes lead to problems for the model leading to a less precision in the results. In the early tests run, the refinement and estimation are run in a simultaneous manner, making the results hardly precise. However, when the stages are defined and implemented in separate layers, the precision of the results faces drastic changes.

Another reason for the high precision of the results here is the predictive contribution of the estimation error obtained through Eq. 9. This method is very effective in normalizing and reducing MRE for the projects with a high estimation error percentage. Applying Eq. 9 and layer 3 leads to a considerable decline in the upper limit of MRE with a drastic decline in an acceptable range.

In the running tests on the FEEM model, identifying the proper sequence of layer placement is one of the most important items to be tested. The proposed sequence of layers is obtained as a result of running different tests and the layers' movement. The results of various tests confirm the sequence of layers.

The MdMRE and PRED criteria of the proposed model are compared with the other methods, as shown in Figsures 8, 9, and 10, respectively.

Table 3. Comparison of MdMRE criterion in datasets.

Method	Desharnais	Maxwell	COCOMO
CART	0.35	0.45	0.77
CBR K = 1	0.45	0.59	0.85
CBR K = 2	0.42	0.55	0.76
CBR K = 3	0.42	0.44	0.78
CBR K = 4	0.38	0.52	0.78
LSSVM	0.41	0.45	1.33
M5'	0.39	0.49	0.71
MARS	0.57	0.48	3.70
MLP	0.54	0.56	0.87
OLS	0.53	0.48	4.06
ROR	0.49	0.59	0.98
PSO + ABE [10]	0.40	0.47	0.75
ACO + ABE [44]	0.36	0.48	0.75
RF [45]	0.39	0.32	1.86
FEEM	0.22	0.24	0.53

Table 4. Comparison of Pred (0.25) criterion in datasets.

Method	Desharnais	Maxwell	COCOMO
CART	0.27	0.32	0.12
CBR K = 1	0.25	0.25	0.07
CBR K = 2	0.29	0.22	0.17
CBR K = 3	0.29	0.29	0.07
CBR K = 4	0.31	0.24	0.06
LSSVM	0.24	0.29	0.09
M5'	0.29	0.22	0.17
MARS	0.23	0.29	0.07
MLP	0.24	0.20	0.19
OLS	0.27	0.24	0.12
ROR	0.36	0.29	0.19
PSO + ABE [10]	0.40	0.29	0.09
ACO + ABE [44]	0.36	0.32	0.09
RF [45]	0.36	0.40	0.12
FEEM	0.51	0.5	0.19

The results of this comparison for the COCOMO dataset are shown in Fig. 8, where the MdMRE value of this proposed model is lower than that of the MdMRE value of all methods. The PRED value of this proposed model is greater than that of the PRED values of all methods. The Desharnais dataset is assessed in Figure 9, where the PRED value of this proposed model is greater than that of its MdMRE value. This difference reflects the high accuracy of the estimates provided by this model. The results of this comparison are shown for the Maxwell dataset, Figure 10, where the PRED value of this proposed model is greater than its MdMRE value. This difference reflects the high accuracy of the estimates provided by this model. The difference in the accuracy of this model with the other methods is based on the PRED and MdMRE criteria, Figure 10.

Another comparison is made based on the MAE benchmark in order to better assess the accuracy of this proposed model. This criterion represents the mean error of estimation in the projects, Figure 11, whereas the observed FEEM model is more accurate than all its counterparts. The accuracy of this model, according to the MAE criterion, is about 70% higher than its counterparts. The other methods, even close to this model, in one of the datasets, are not able to repeat their own estimation accuracy in other datasets. This point reflects the ability of this model to be adaptive in the project conditions.

In order to determine the FEEM overall performance, Wilcoxon, a statistical test, is executed, which would confirm the superiority of this model. The Wilcoxon test specifies the difference between two data samples, and the difference is determined by the P-value parameter. Based on this method, two samples of data are statistically different when the p-value quantity is less than 0.05. In this article, the P-value quantity of different methods is compared with the FEEM method. The P-value of each one of the assessed methods in comparison to the FEEM model is tabulated in Table 5. The P-value quantity of the Wilcoxon test in all methods and all datasets is less than 0.05. The results of this test confirm the statistical significance of this model.

Table 5. P-values obtained from Wilcoxon test.

Method	Desharnais	Maxwell	COCOMO
CART	0.0381	0.0347	0.0356
CBR K = 1	0.0215	0.0012	0.0015
CBR K = 2	0.0461	0.0119	0.0473
CBR K = 3	0.0283	0.0303	0.0125
CBR K = 4	0.0493	0.0071	0.0088
LS-SVM	0.0434	0.0138	0.00076
M5'	0.026	0.01	0.0338
MARS	0.0039	4.80E-05	4.38E-08
MLP	0.00078	0.00037	0.0342
OLS	0.0017	0.0219	3,51E-04
ROR	0.0313	0.000964	0.0166
PSO + ABE [10]	0.0391	0.0383	0.0368
ACO + ABE [44]	0.041	0.0389	0.037
RF [45]	0.048	0.05	2.3853e-005

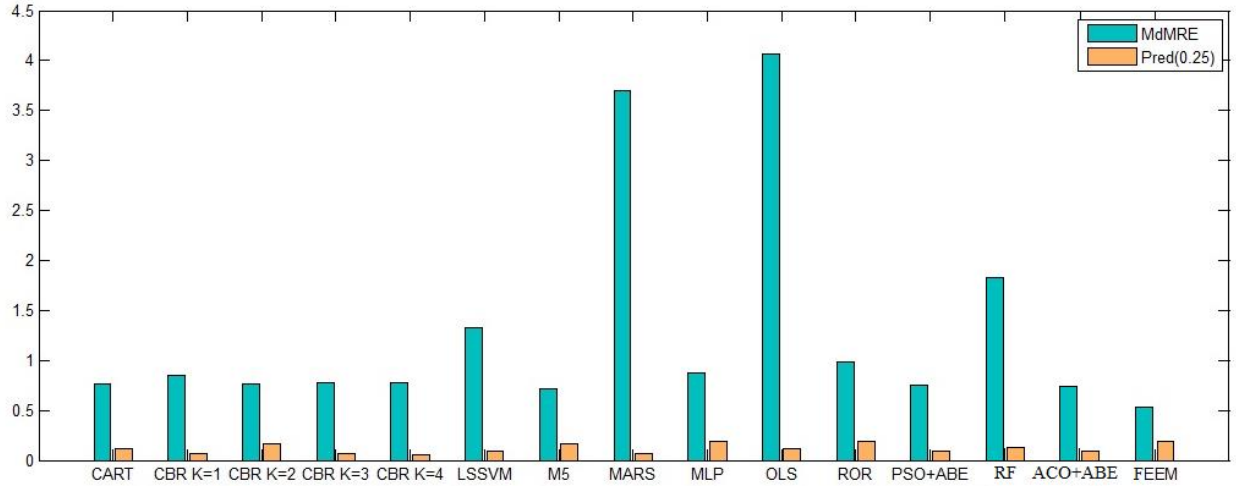

Figure 8. Comparing methods on cocomo dataset.

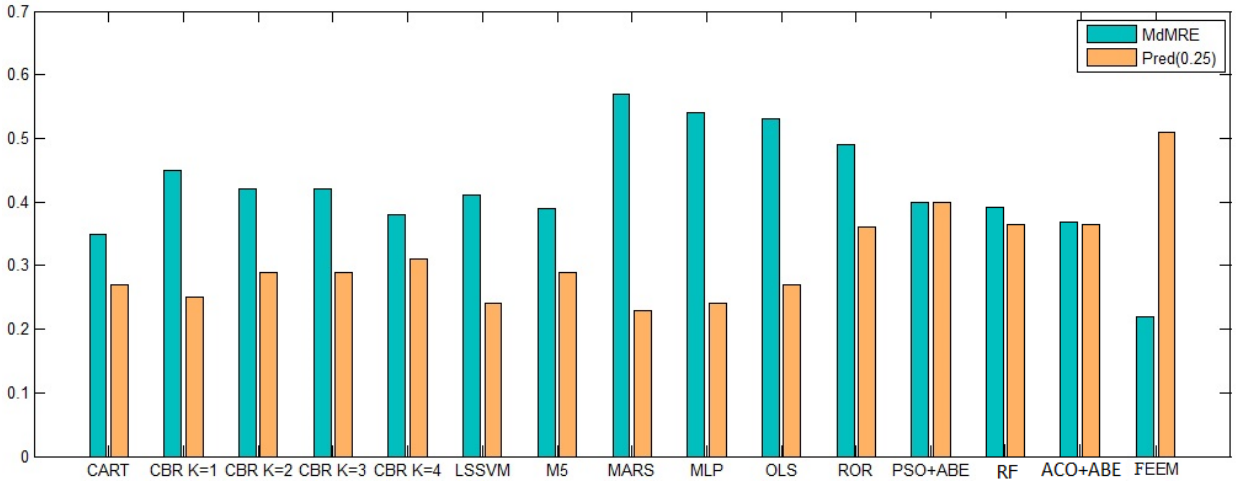

Figure 9. Comparing methods on Desharnise dataset.

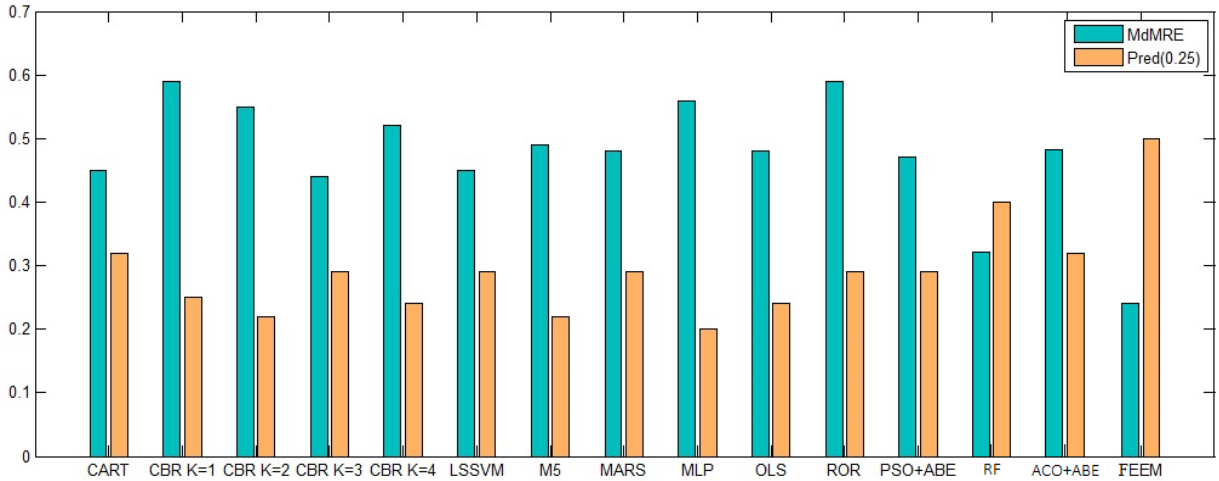

Figure 10. Comparing methods on Maxwell dataset.

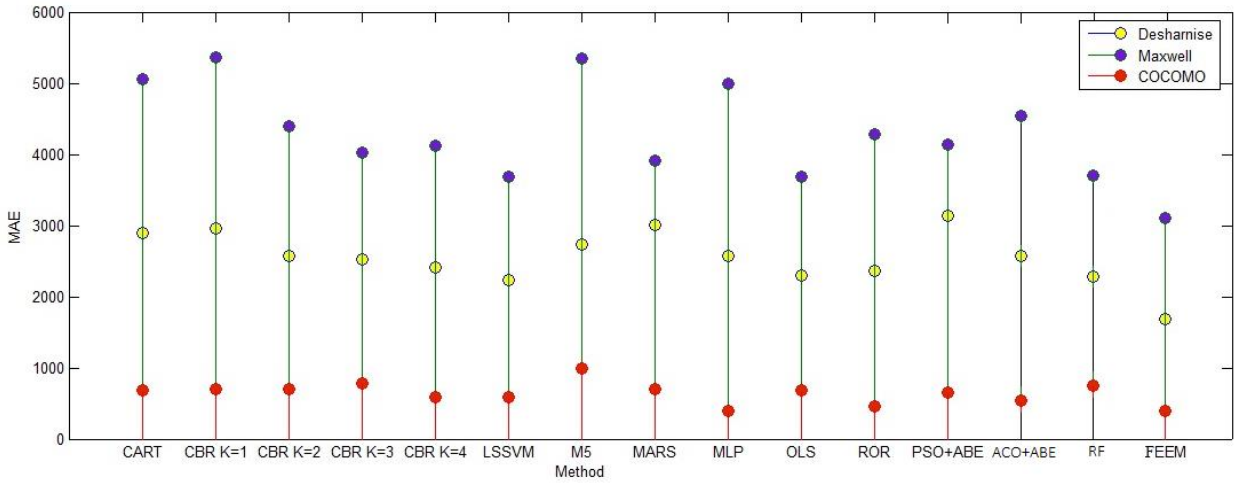

Figure 11. Comparison of MAE criterion in datasets.

10. Conclusion

According to the literature, the use of data processing methods, identification methods of effective features, and identification of types of relationships between the project features on the software project effort or data modeling increase the estimation accuracy. Moreover, the correct application of heuristic algorithms for the configuring methods and tools plays a key role in the increased efficiency. A novelty of this work is to present the sub-models with the above objectives for identifying the features and their effective relationships, exactly configuring the data modeling techniques, and estimating by the LSA algorithm based on the feature similarity. The other novelty is to propose a model consisting of three layers in which the sub-models are organized in their layers in order to improve their accuracy. The first layer of the proposed model acts on the project features. In the second layer, the ABE method, an estimation method based on the feature similarity, is configured using the LSA algorithm. The accuracy of the second layer is increased using the analysis results of the first layer on the project features. Combining the neural network and LSA, an estimator model of the first and second layers is developed in the third layer based on its outputs and inputs in order to increase the final estimation accuracy. Testing each layer slightly increased the estimation accuracy but properly organizing all these layers significantly increased the final estimation accuracy. Using the heuristic algorithm in this model improved the flexibility of the layers and their consistency with the project conditions. This model was tested, and its precise results were displayed. Precision of the results here suggests that many models presented by the researchers so far can become more precise if re-designed based on the procedures presented here. Here, a new method was applied to increase the accuracy of the estimation model. In addition to data modeling to estimate the effort, a separate modeling was performed to estimate the model error. The error modeling was made in layer 3. The result of this model indicates the final value of the estimation. A separate error modeling is contributive in reducing the error estimation. The approaches proposed in this work can enhance the precision and effectiveness of the future methods.

References

[1] A. K. Bardsiri and S. M. Hashemi, "Software effort estimation: a survey of well-known approaches," *International Journal of Computer Science Engineering (IJCSE)*, vol. 3, pp. 46-50, 2014.

[2] S. Bilgaiyan, S. Mishra, and M. Das, "Effort estimation in agile software development using experimental validation of neural network models," *International Journal of Information Technology*, vol. 11, pp. 569-573, 2019.

[3] D. Rankovic, N. Rankovic, M. Ivanovic, and L. Lazic, "Convergence rate of Artificial Neural Networks for estimation in software development projects," *Information and Software Technology*, p. 106627, 2021.

[4] P. S. Kumar, H. S. Behera, A. Kumari, J. Nayak, and B. Naik, "Advancement from neural networks to deep learning in software effort estimation: Perspective of two decades," *Computer Science Review*, vol. 38, p. 100288, 2020.

[5] R. D. A. Araujo, A. L. Oliveira, and S. Meira, "A class of hybrid multi-layer perceptrons for software development effort estimation problems," *Expert Systems with Applications*, vol. 90, pp. 1-12, 2017.

[6] S. H. S. Moosavi and V. K. Bardsiri, "Satin bowerbird optimizer: A new optimization algorithm to optimize ANFIS for software development effort

estimation," *Engineering Applications of Artificial Intelligence*, vol. 60, pp. 1-15, 2017.

[7] T. R. Benala and R. Mall, "DABE: differential evolution in analogy-based software development effort estimation," *Swarm and Evolutionary Computation*, vol. 38, pp. 158-172, 2018.

[8] Q. Liu, J. Xiao, and H. Zhu, "Feature selection for software effort estimation with localized neighborhood mutual information," *Cluster computing*, vol. 22, pp. 6953-6961, 2019.

[9] N.-H. Chiu and S.-J. Huang, "The adjusted analogy-based software effort estimation based on similarity distances," *Journal of Systems and Software*, vol. 80, pp. 628-640, 2007.

[10] V. K. Bardsiri, D. N. A. Jawawi, S. Z. M. Hashim, and E. Khatibi, "A PSO-based model to increase the accuracy of software development effort estimation," *Software Quality Journal*, vol. 21, pp. 501-526, 2013.

[11] 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*, 2021.

[12] V. Resmi, S. Vijayalakshmi, and R. S. Chandrabose, "An effective software project effort estimation system using optimal firefly algorithm," *Cluster Computing*, vol. 22, pp. 11329-11338, 2019.

[13] J. F. Vijay, "Enrichment of accurate software effort estimation using fuzzy-based function point analysis in business data analytics," *Neural Computing and Applications*, vol. 31, pp. 1633-1639, 2019.

[14] I. Kaur, G. S. Narula, R. Wason, V. Jain, and A. Baliyan, "Neuro fuzzy—COCOMO II model for software cost estimation," *International Journal of Information Technology*, vol. 10, pp. 181-187, 2018.

[15] A. Idri, I. Abnane, and A. Abran, "Missing data techniques in analogy-based software development effort estimation," *Journal of Systems and Software*, vol. 117, pp. 595-611, 2016.

[16] P. R. Sree and R. SNSVSC, "Improving efficiency of fuzzy models for effort estimation by cascading and clustering techniques," *Procedia Computer Science*, vol. 85, pp. 278-285, 2016.

[17] A. Karimi and T. J. Gandomani, "Software development effort estimation modeling using a combination of fuzzy-neural network and differential evolution algorithm," *International Journal of Electrical and Computer Engineering (2088-8708)*, vol. 11, 2021.

[18] 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, p. 2050005, 2020.

[19] B. Barry, "Software engineering economics," *New York*, vol. 197, 1981.

[20] B. Boehm, C. Abts, A. Brown, S. Chulani, B. Clark, E. Horowitz *et al.*, "Software cost estimation with COCOMO II. Prentice Hall PTR," *Upper Saddle River, NJ*, 2000.

[21] L. H. Putnam, "A general empirical solution to the macro software sizing and estimating problem," *IEEE transactions on Software Engineering*, pp. 345-361, 1978.

[22] A. J. Albrecht and J. E. Gaffney, "Software function, source lines of code, and development effort prediction: a software science validation," *IEEE transactions on software engineering*, pp. 639-648, 1983.

[23] S. A. Abbas, X. Liao, A. U. Rehman, A. Azam, and M. Abdullah, "Cost estimation: A survey of well-known historic cost estimation techniques," *Journal of Emerging Trends in Computing and Information Sciences*, vol. 3, pp. 612-636, 2012.

[24] G. R. Finnie, G. E. Wittig, and J.-M. Desharnais, "A comparison of software effort estimation techniques: Using function points with neural networks, case-based reasoning and regression models," *Journal of systems and software*, vol. 39, pp. 281-289, 1997.

[25] P. Sentas, L. Angelis, I. Stamelos, and G. Bleris, "Software productivity and effort prediction with ordinal regression," *Information and software technology*, vol. 47, pp. 17-29, 2005.

[26] L. C. Briand, K. El Emam, D. Surmann, I. Wieczorek, and K. D. Maxwell, "An assessment and comparison of common software cost estimation modeling techniques," in *Proceedings of the 1999 International Conference on Software Engineering (IEEE Cat. No. 99CB37002)*, 1999, pp. 313-323.

[27] L. C. Briand, T. Langley, and I. Wieczorek, "A replicated assessment and comparison of common software cost modeling techniques," in *Proceedings of the 22nd international conference on Software engineering*, 2000, pp. 377-386.

[28] F. Zare, H. K. Zare, and M. S. Fallahnezhad, "Software effort estimation based on the optimal Bayesian belief network," *Applied Soft Computing*, vol. 49, pp. 968-980, 2016.

[29] A. García-Floriano, C. López-Martín, C. Yáñez-Márquez, and A. Abran, "Support vector regression for predicting software enhancement effort," *Information and Software Technology*, vol. 97, pp. 99-109, 2018.

[30] S. Beiranvand and Z. Chahooki, "Bridging the semantic gap for software effort estimation by hierarchical feature selection techniques," *Journal of AI and Data Mining*, vol. 4, pp. 157-168, 2016.

[31] A. Puspaningrum and R. Sarno, "A hybrid cuckoo optimization and harmony search algorithm for software cost estimation," *Procedia Computer Science*, vol. 124, pp. 461-469, 2017.

[32] S. P. Singh, V. P. Singh, and A. K. Mehta, "Differential evolution using homeostasis adaption based mutation operator and its application for software cost estimation," *Journal of King Saud University-Computer and Information Sciences*, 2018.

[33] A. Ali and C. Gravino, "Improving software effort estimation using bio-inspired algorithms to select relevant features: An empirical study," *Science of Computer Programming*, vol. 205, p. 102621, 2021.

[34] S. Goyal and P. K. Bhatia, "Feature selection technique for effective software effort estimation using multi-layer perceptrons," in *Proceedings of ICETIT 2019*, ed: Springer, 2020, pp. 183-194.

[35] A. Setiadi, W. F. Hidayat, A. Sinnun, A. Setiawan, M. Faisal, and D. P. Alamsyah, "Analyze the Datasets of Software Effort Estimation With Particle Swarm Optimization," in *2021 International Seminar on Intelligent Technology and its Applications (ISITIA)*, 2021, pp. 197-201.

[36] P. Phannachitta, "On an optimal analogy-based software effort estimation," *Information and Software Technology*, vol. 125, p. 106330, 2020.

[37] E. Kocaguneli and T. Menzies, "Software effort models should be assessed via leave-one-out validation," *Journal of Systems and Software*, vol. 86, pp. 1879-1890, 2013.

[38] P. W. Holland and R. E. Welsch, "Robust regression using iteratively reweighted least-squares," *Communications in Statistics-theory and Methods*, vol. 6, pp. 813-827, 1977.

[39] J. H. Friedman, "Multivariate adaptive regression splines," *The annals of statistics*, pp. 1-67, 1991.

[40] J. R. Quinlan, *C4. 5: programs for machine learning*: Elsevier, 2014.

[41] Y. Wang and I. H. Witten, "Induction of model trees for predicting continuous classes," 1996.

[42] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, pp. 359-366, 1989.

[43] J. Li, G. Ruhe, A. Al-Emran, and M. M. Richter, "A flexible method for software effort estimation by analogy," *Empirical Software Engineering*, vol. 12, pp. 65-106, 2007.

[44] S. Ranichandra, "Optimizing non - orthogonal space distance using ACO in software cost estimation," *Mukt Shabd J*, vol. 9, pp. 1592-1604, 2020.

[45] H. Mustapha and N. Abdelwahed, "Investigating the use of random forest in software effort estimation," *Procedia computer science*, vol. 148, pp. 343-352, 2019.

FEEM : یک مدل انعطاف پذیر مبتنی بر هوش مصنوعی برای تخمین تلاش نرم افزار

امین مرادبیکی*

گروه مهندسی کامپیوتر، دانشگاه آزاد اسلامی واحد زابل، زابل، ایران.

ارسال ۲۰۲۱/۰۹/۱۳؛ بازنگری ۲۰۲۱/۱۰/۲۴؛ پذیرش ۲۰۲۲/۰۱/۰۴

چکیده:

بدلیل ماهیت ناملموس نرم افزار، مدیریت پروژه‌های تولید نرم افزار همواره با چالش‌های فراوانی در حوزه پیشبینی میزان تلاش مورد نیاز برای توسعه آن مواجه است. از این رو پژوهش‌های فراوانی در جهت توسعه ابزارهای دقیق برای تخمین تلاش مورد نیاز برای توسعه نرم افزار انجام شده است. بر اساس ادبیات تحقیق، دقت روش‌ها یا مدل‌های تخمینگر با بکارگیری صحیح شیوه‌های پالایش داده یا وزندهی ویژگی، قابل افزایش است. همچنین مدل‌های زیادی برمبنای روش‌های یادگیری ماشین ارائه شده است که از مدل سازی داده استفاده کرده‌اند. مدل ارائه شده در این مقاله متشکل از سه لایه است. در لایه‌های اول و دوم این مدل، ابزارها و تکنیکهایی پیشنهاد داده شده است که کار انتخاب موثرترین ویژگی‌ها و وزن دهی ویژگی‌ها را با کمک الگوریتم جستجوی نور (LSA) انجام می‌دهد. همچنین در لایه سوم این مدل با استفاده از ترکیبLSA و شبکه عصبی، یک مدل از تخمینگر لایه یک و دو ساخته شده است و با کمک آن دقت تخمین نهایی به شکل چشم‌گیری افزایش یافته است. برای ارزیابی دقت مدل‌های پیشنهادی، از سه مجموعه داده از پروژه‌های واقعی استفاده شده است و نتایج بدست آمده با نتایج روشهای مختلف مقایسه شده است.

کلمات کلیدی: الگوریتم جستجوی نور, پروژه نرم افزار, تخمین تلاش توسعه, شبکه‌های عصبی.