International Journal of Industrial Electronics, Control and Optimization © 2022 IECO
Vol. 5, No. 1, pp. 31-42 , Jan (2022)

3LEE: A 3-Layer Effort Estimator for Software Projects

Amin Moradbeiky [1], Vahid Khatibi Bardsiri [2,†], and Mehdi Jafari [3]

[1,2] Department of Computer Engineering, Faculty of Science, Kerman Branch, Islamic Azad University, Kerman, Iran
[3] Department of Electrical Engineering, Faculty of Engineering, Kerman Branch, Islamic Azad University, Kerman, Iran

ABSTRACT

Managing software projects due to its intangible nature is full of challenges when predicting the effort needed for development. Accordingly, there exist many studies with the attempt to devise models to estimate efforts necessary in developing software. According to the literature, the accuracy of estimator models or methods can be improved by correct application of data filtering or feature weighting techniques. Numerous models have also been proposed based on machine learning methods for data modeling. This study proposes a new model consisted of data filtering and feature weighting techniques to improve the estimation accuracy in the final step of data modeling. The model proposed in this study consists of three layers. Tools and techniques in the first and second layers of the proposed model select the most effective features and weight 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 data of lower layers. This arrangement significantly increased the accuracy of final estimation. Three datasets of real projects were used to evaluate the accuracy of proposed model, and the results were compared with those obtained from different methods. The results were compared based on performance criteria, indicating that the proposed model effectively improved the estimation accuracy.

Article Info

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

Article History:
Received 2021-09-14
Accepted 2021-12-30

I. INTRODUCTION

Software effort estimation is directly related to the development and success of software project, consequently, this estimation is considered as a major challenge for researchers and the practitioners in software industry. Estimation methods are divided of algorithmic and non-algorithmic categories [1], the first is based on mathematical and the second on heuristic and metaheuristic methods.

In both methods, usually a model is devised a number of tools and algorithms proposed to estimate of the software development effort. In each one of these models, 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:

- Neural networks are one of the most commonly adopted methods in AI; Elman's neural network is applied in [2] for software development effort estimation.
- Rankovic and et al. [3] proposed four new models based on artificial neural network and they utilized five datasets to test them.
- Kumar and et al. [4] used neural networks to deep learning in software effort estimation.
- The dilation-erosion-linear perceptron was introduced in 2012,and is applied in many articles for prediction, but if there exists complexity of input/output then, it will not be sufficient. Araujo et al.[5] optimized the structure of this

[†]**Corresponding Author**: ampayam@yahoo.com
Tel: +98-9153432720, Department of Computer Engineering, Faculty of Science, Kerman Branch, Islamic Azad University, Kerman, Iran

perceptron, using the descending gradient in the learning process, and used it in software effort estimation.
- A combination of satin bowerbird optimization algorithm (SBO) and the Neuro-fuzzy (ANFIS) is applied to increase the accuracy in predicting 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:
- ABE method has been commonly used for software effort estimation by researchers. The differential equation (DE) algorithm is applied in similarity function to weight the features, named Differential evolution in Analogy-Based Estimation (DABE) [7], to improve the efficiency of this method.
- There exists no exact definition on projects similarity. A similarity region is identified by [8] for feature selection in similar projects through Case-Based Reasoning (CBR) concept.
- One of the algorithms combined through ABE methods is genetic algorithm [9].
- Application of 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] is proposed.

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

Some researchers only use tools based on algorithmic methods:
- The COCOMO, proposed by [19], the COCOMOII, proposed by [20], SLIM, proposed by [21], Function Point Analysis, proposed by[22] and Dotti model proposed by [23].
- The regression-based methods like: linear regression methods [24, 25], non-linear regression methods [25], tree regression methods [26, 27].

Artificial intelligence algorithms can improve the efficiency of formulated methods by searching for the appropriate configuration for these methods [28]. This approach has been followed in some articles:
- Updated K-modes clustering basic algorithms are applied in effort prediction. in the proposal model by [29] the Beysian belief network is constructed from of the COCOMO model, where the intervals are of fuzzy numbers, then, the PSO algorithm and Genetic algorithm (GA) are combined to improve the software effort estimation.
- Machine learning algorithms are commonly applied in problem estimation Two different types of Support Vector Machine (SVM) are applied by [30] to predict effort and compare with the other methods like neural networks, decision tree etc. Various feature selection methods have also been used to performance optimization of machine learning based methods [31].
- Meta-heuristic algorithms are commonly applied by researchers in many cases. A hybrid Meta-heuristic algorithm, consisting of Cuckoo Optimization Algorithm (COA), Harmonic Search [32], and DE algorithm is applied to optimizing COCOMO parameters [33], and to improve software effort estimation.

Some studies emphasize identifying key project features and their relationship with the software development effort. There has been an emphasis on identifying interrelated features influencing the software development effort [34]. Features influencing effort has been identified by a neural network [35]. The PSO algorithm [36] and the Bayesian technique [37] have been used to identify features influencing the software development effort.

A novel model is proposed in this study by analyzing models previously presented in the literature. Data preparation tools have been proposed in some studies to improve the estimation accuracy. Some studies have emphasized 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 project features and effectiveness of different features. The effectiveness has been defined as a coefficient in the literature. Data modeling by 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 input to the next layer, improving the final performance of the proposed model.

Table I
Various methods in software effort estimation

Study	Year	Dataset	Evaluation Method	Method	Ref No.
1	2019	21 Project(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 Project(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	[29]
12	2018	ISBSG	MAE	SVR	[30]
13	2017	COCOMO	MMRE	Cuckoo Search	[32]
14	2018	COCOMO	MMRE, Pred, MAE	DE	[33]
15	2020	ISBSG, Desharnais	MMRE, Pred	ACO, ABE	[38]

Section II discusses the ABE method used for estimating the software development effort. Section III introduces criteria for calculating the accuracy of the proposed model. Section IV introduces the proposed model. Section V discusses a cross validation method for evaluating the stability of the proposed model results. Section VI introduces three datasets of real projects used for testing the proposed model. Section VII introduces techniques compared with the proposed model. Section VIII presents the test results of the proposed model. The model results are analyzed in Section IX.

II. ANALOGY-BASED ESTIMATION METHOD (ABE)

Estimation methods are of the two algorithm and non-algorithm. Because the first methods are not appropriate to be adopted in dynamic environment of software projects, the second methods are applied in this context, making ABE one of the most applicable methods. 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.

A. Similarity Function

The similarity of projects through studying 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 P and P' projects, respectively. The result reveals the similarity level between the two projects.

$$\delta = 0.0001$$

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

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

$$\delta = 0.0001$$

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

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

B. Solution Function

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

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

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. Symbol C_{Pi} is a certain value to be estimated from the j^{th} more similar project.

C. The best value of K

The K value is applied in effort estimations with a high accuracy. An appropriate K value mostly depends on the study projects. If the difference in study projects is slightly high, the K value accuracy reduces, because the effective projects manifest more differences at the final stage of estimation. If the study 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 estimation process. Consequently, no constant value of K can be considered, 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.

III. EQUATIONS FOR ESTIMATION ERROR CALCULATION

In this section, utilized equations to evaluate the accuracy of the proposed model and compare it with other methods are introduced. These equations are commonly used for accuracy evaluation by researchers in the field. The results of equations are displayed as diagrams for better accuracy evaluation and comparison. utilized equations are presented in Equ (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):

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

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

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

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

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

IV. THE PROPOSED MODEL

This paper presents a new model called 3 Layer Effort Estimator (3LEE). 3LEE is composed of two sections: training and test. The training section of this model is consists of three layers, each responsible refine the data and enhance the precision of estimation. The model of layer 1 is shown in Fig (1), where, the best features are selected based on the feature selection and ABE method with several iteration. At every one of 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 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.

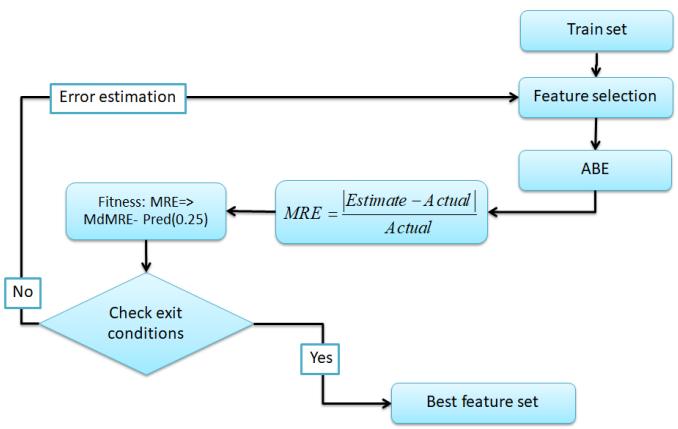


Fig. 1. The flowchart of training section model, layer 1

The layer 2 model is shown in Fig. (2). which undergoes training through the selected features as its input. This model is iterated for many times through LSA algorithm, and at each iteration the LSA algorithm suggests an appropriate setting for ABE. The ABE method processes projects and estimates them based on 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 method. The ABE method searches for the most similar projects with the target project to estimate software development effort based on features adaptation. The ABE method uses the LSA algorithm to increase estimation accuracy. The LSA algorithm tries to propose the most appropriate configuration for ABE method and helps it to provide a more accurate estimation. The configuration proposed by the LSA algorithm differs based on project conditions and its features. On the other hand, the first layer helps increase the second layer's accuracy by processing input data to the first layer. Simply speaking, 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 of the estimator of first and second layers and based on their input and output. This layer leads to improved final estimated accuracy.

The layer 3 model is shown in Fig. (3) which undergoes predicting proper estimation error based on a project's data. To estimate the prediction error, Artificial Neural Network (ANN) is applied. The proper configuration of ABE obtained in layer 2 is received as its input and best features 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 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 and Error is the error proposed by the 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 final estimation. In the proposed model, the third layer has very important role. This layer tries to provide a more accurate estimate of the project through building a model. The built model receives the project characteristics and the estimated amount of effort to make the estimation by using equation 9. In other words, this layer tries to get a more accurate perspective of the project status. On the other hand, the sub layers of this layer have also strengthened its accuracy by refining and providing data.

After 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 settings suggested by 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 prediction error Through ANN. These proposed settings reduce final estimation error.

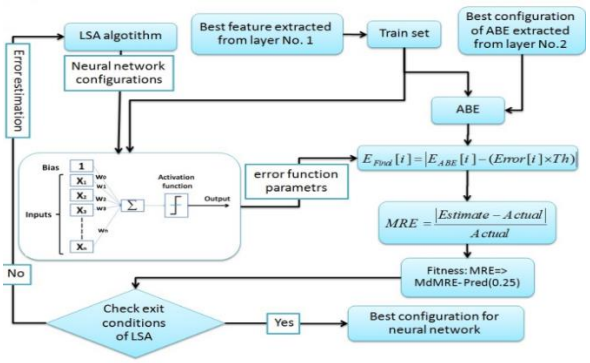


Fig. 2. The flowchart of training section model, layer 2

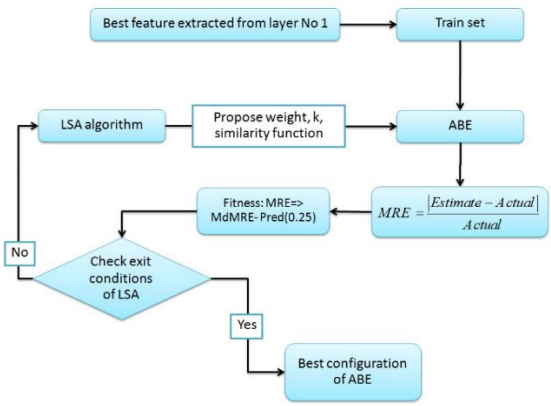


Fig. 3. The flowchart of training section model, layer 3

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

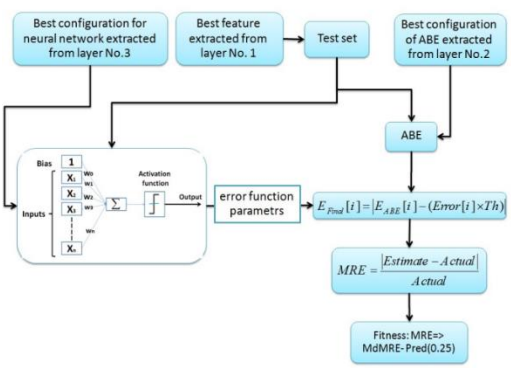


Fig. 4. The test section model flowchart

V. CROSS VALIDATION METHOD

Based on the proposed model, projects must be divided into two groups of training and test. The arrangement of projects in dividing process effects on the accuracy of the

proposed model *[39]*. For sustainability provement of the models, different cross validation methods including 3 fold, 10 fold, etc. can be used. Each one of these methods provides a specific arrangement for projects. Based on the performed study *[39]*, leave-one-out (LOO) is the best method for evaluation and its achieved accuracy is independent from arrangement of projects. In this paper, LOO method is adopted.

VI. INTRODUCING DATASETS

In testing stage of proposed model, dataset of real projects are utilized. These datasets are applied by many researchers. The details of the data analyzes of this datasets are tabulated in Table II. The desharnais dataset consists of 81 real software projects. This dataset is collected in canadian software houses. The projects in 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' are independent. In this paper, only 77 projects of this dataset are used for tests because 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.

Table II

Datasets

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

VII. TECHNIQUES

This proposed model is compared with the following methods for evaluating 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 estimation accuracy in unusual data [40].
- Multivariate Adaptive Regression Splines (MARS): is a non-linear and non-parametric regression method indicative of some interesting features like ease in interpretation, the ability to model complex non-linear correlation, with a rapid output [41].
- 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 [42].
- M5: The M5 method utilizes 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. [43].
- Multi Layered Perceptron (MLP): Neural network is a non-linear modeling technique. Multi Layered Perceptron-based neural network is 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 [44].
- Case based learning reasoning (CBR): The CBR operator search for the most similar sample to the sample we intend to estimate. The similarity of samples is calculated through this method. In this method, the K determines the number of most similar samples that must be used for data estimation [45].

VIII. 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 dataset. Precision of this model in these tests is calculated and displayed through the criteria and equations introduced in section III. In this paper, MATLAB software has been used for modeling. The tests of the proposed model and all compared models in this paper are performed on a computer with 4th generation i5 CPU and 4GB RAM. In the configuration of LSA algorithm, the size of initial population factor and the maximum numbers of iterations are considered 50 and 150 times respectively. The adopted neural network in this study is Feedforward network. The network settings are suggested by the LSA algorithm. Determining the vector of bias values, weight, and the best number of hidden layers for neural network are the suggested configuration by the LSA algorithm. These settings have been selected based on the results of multiple experiments.

A. Desharnais dataset test

Desharnais dataset test is selected as the first, the specifications of which are presented in section VI. The MRE value obtained from implementing this proposed model is expressed in Fig. (5). The MdMRE value for this test is 0.22 and the PRED value is 0.51.

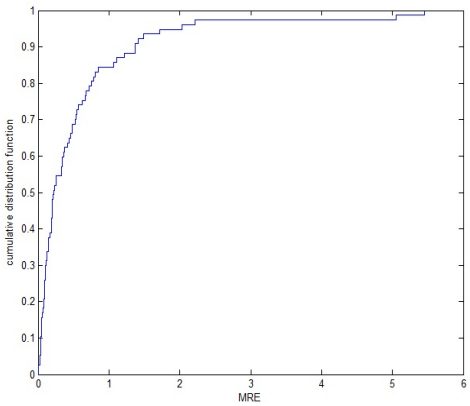


Fig. 5. MRE error frequency distribution with 3LEE model for desharnais dataset

The frequency distribution diagram of MRE error is graphed in Fig. (5), where the percentage 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 projects with a specific MRE quantity. As observed in Fig. (5), a high percentage of errors fall within a range less than 0.5. The higher slope of this diagram in one area signifies 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.

B. COCOMO dataset test

Specifications of this dataset are presented in section VI. The MRE value obtained from implementing this proposed model is expressed in Fig. (6). The MdMRE value for this test is 0.53 and the PRED value is 0.19.

TABLE III

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 [38]	0.36	0.48	0.75
RF [46]	0.39	0.32	1.86
3LEE	0.22	0.24	0.53

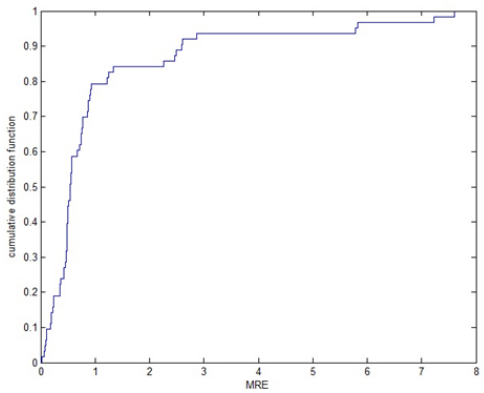


Fig. 6. MRE error frequency distribution with the 3LEE model for the COCOMO dataset

The frequency distribution diagram of MRE error for the COCOMO dataset is shown in Fig. (6), where the horizontal axis, represent the MRE quantity and the vertical axis represent the percentage of projects with a specific MRE quantity. As observed here, a high percentage of errors fall within a range of 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 low estimation accuracy.

C. Maxwell dataset test

The next test is run on the maxwell dataset, specifications of which are given in section VI. The MRE value obtained from implementing this proposed model is expressed in Fig. (7). The MdMRE value for this test is 0.24 and the PRED value is 0.5.

Table IV

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 [38]	0.36	0.32	0.09
RF [46]	0.36	0.40	0.12
3LEE	0.51	0.5	0.19

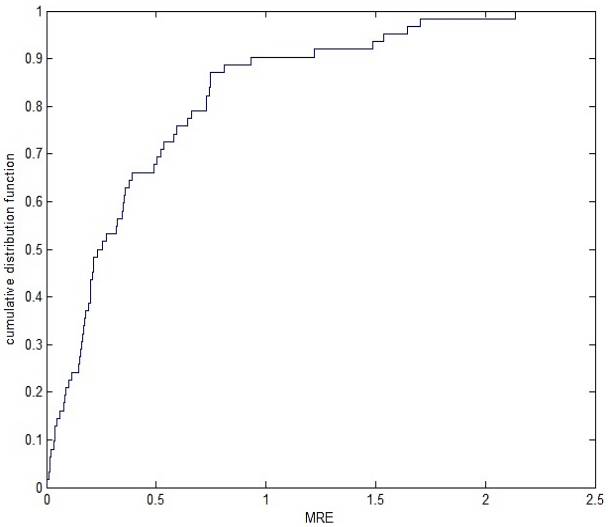


Fig. 7. MRE error frequency distribution with the 3LEE model for the maxwell dataset

The frequency distribution diagram of MRE error for the Maxwell dataset is drawn in Fig. (7), where, the horizontal axis, represent the MRE quantity and the vertical axis represent the percentage of projects with a specific MRE quantity. As observed here, about 70% of the projects are estimated with less than 0.5 error and big errors are of a small distribution.

IX. ANALYSIS AND COMPARISON OF THE RESULTS

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

This precision is due to fact that refining filters are separated which in turn increase data precision. In estimation research, in cases where data refining methods are applied, they join estimation process which leads to many problems. Combining the refinement and estimation processes lead to problems for the model leading to less precision in the results. In the early tests run, here 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 face drastic changes.

Another reason for 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 projects with a high estimation error percentage. Applying, Eq. (9) and layer 3 lead to a considerable decline in upper limit of MRE With a drastic decline in an acceptable range.

In running tests on the 3LEE 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 other methods as shown in Fig. (8, 9 and 10), respectively. 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 Fig. (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, Fig. (10), where, 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 accuracy here model with other methods is based on the PRED and MdMRE criteria, Fig. (10).

Another comparison is made based on MAE benchmark to better assess the accuracy of this proposed model. This criterion represents the mean error of estimation in the projects, Fig. (11), whereas observed the 3LEE model is more accurate than all its counterparts. The accuracy of this model, according to the MAE criteria, 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 data sets. This point reflects the ability of this model to be adaptive in project conditions.

To determine 3lee overall performance, Wilcoxon, a statistical test, is executed which would confirm the superiority of this model. Wilcoxon test specifies the difference between two data samples that the difference is determined by P-value parameter. Based on this method two samples of data statistically different when p-value quantity is less than 0.05. In this article, the P-value quantity of different methods is compared with the 3LEE method. The P-value of each one of the assessed methods in comparison to the 3LEE model is tabulated in Table V. The P-value quantity of Wilcoxon test in all methods and all dataset is less than 0.05. The results of this test confirm the statistical significance of this model.

TABLE V

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 [38]	0.041	0.0389	0.037
RF [46]	0.048	0.05	2.3853e-005

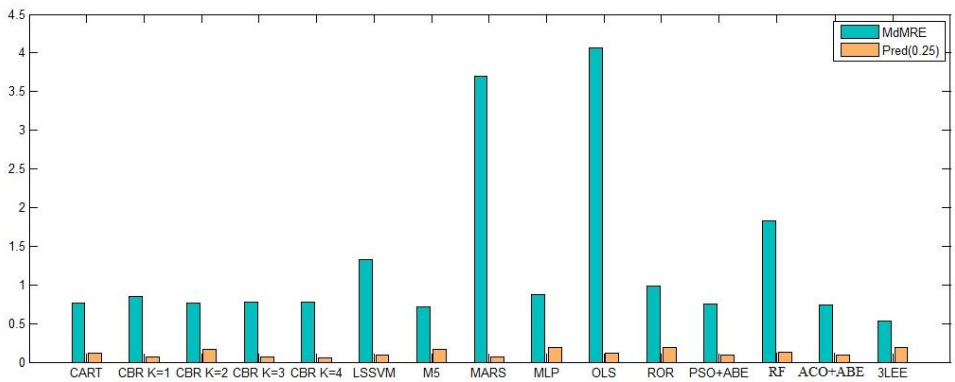


Fig. 8. Comparing methods on cocomo dataset

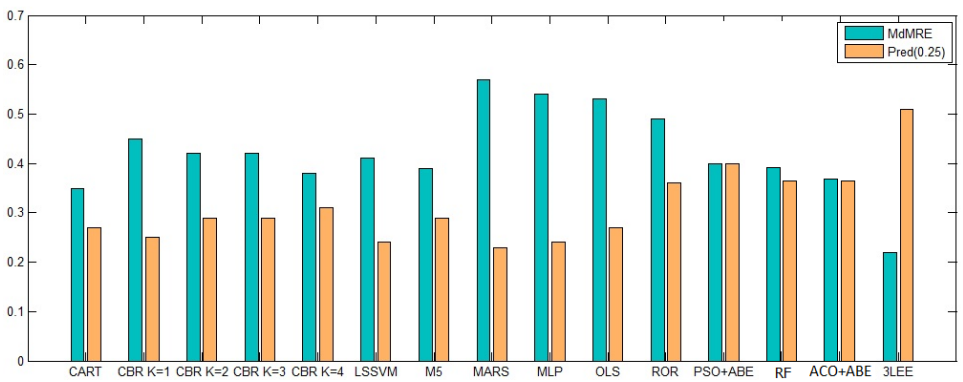


Fig. 9. Comparing methods on desharnise dataset

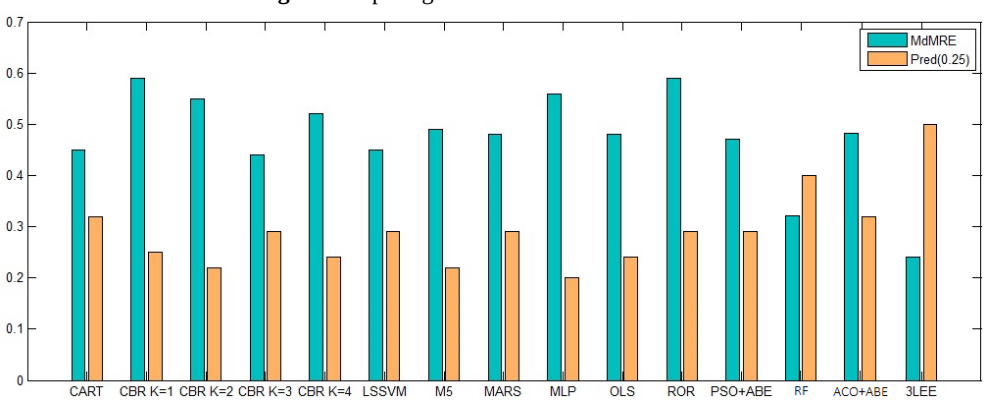


Fig. 10. Comparing methods on maxwell dataset

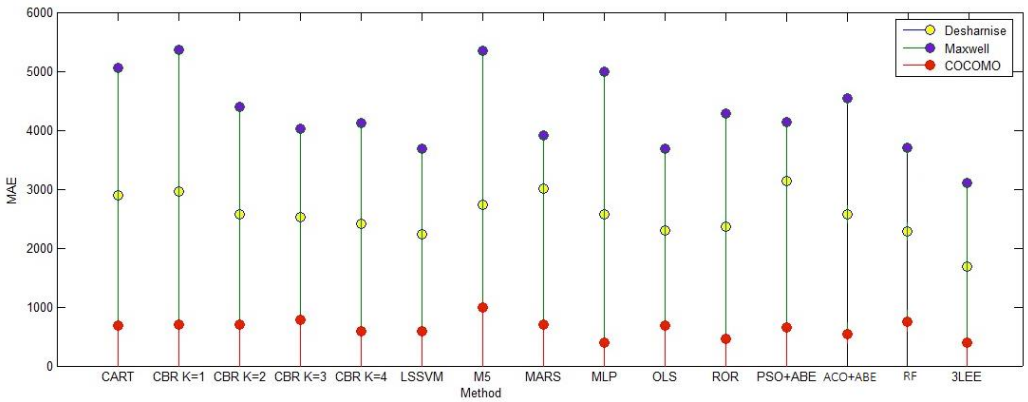


Fig. 11. Comparison of MAE criterion in datasets

X. CONCLUSION

According to the literature, the use of data processing methods, identification methods of effective features, and identification of types of relationships between project features on software project effort, or data modeling increase the estimation accuracy. Moreover, the correct application of heuristic algorithms for configuring methods and tools plays a key role in increased efficiency. A novelty of this study is to present sub-models with the above objectives for identifying features and their effective relationships, exactly configuring data modeling techniques, and estimating by the LSA algorithm based on feature similarity. The other novelty is to propose a model consisting of three layers in which sub-models are organized in their layers to improve their accuracy. The first layer of the proposed model acts on project features. In the second layer, the ABE method, an estimation method based on feature similarity, is configured using the LSA algorithm. The accuracy of the second layer is increased by using the analysis results of the first layer on 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 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 layers and their consistency with project conditions. This model is tested and its precise results were displayed. Precision of the results here suggests that many models presented by researchers so far can become more precise if redesigned based on the procedures presented here. Here, a new method is applied to increase the accuracy of the estimation model. In addition to data modeling to estimate the effort, a separate modeling is performed to estimate the model error. The error modeling is made in Layer 3. The result of this model indicates the final value of the estimation. Separate error modeling is contributive in reducing error estimation.

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 multilayer 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.

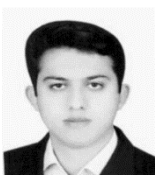
[9] N.-H. Chiu and S.-J. Huang, "The adjusted analogy-based software effort estimation based on similarity distances," *Journal of Systems and Software*, vol. 80, pp. 628-640, 2007.

[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 & 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 & 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] M. Dehghani, Z. Montazeri, O. P. Malik, A. Ehsanifar, and A. Dehghani, "OSA: Orientation Search Algorithm," *International Journal of Industrial Electronics Control and Optimization*, vol. 2, pp. 99-112, 2019.
[29] 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.
[30] 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.
[31] 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.
[32] 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.
[33] 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.
[34] 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.
[35] 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.
[36] 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.
[37] P. Phannachitta, "On an optimal analogy-based software effort estimation," *Information and Software Technology*, vol. 125, p. 106330, 2020.
[38] S. Ranichandra, "Optimizing non-orthogonal space distance using ACO in software cost estimation," *Mukt Shabd J*, vol. 9, pp. 1592-1604, 2020.
[39] 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.
[40] 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.
[41] J. H. Friedman, "Multivariate adaptive regression splines," *The annals of statistics*, pp. 1-67, 1991.

[42] J. R. Quinlan, *C4. 5: programs for machine learning*: Elsevier, 2014.
[43] Y. Wang and I. H. Witten, "Induction of model trees for predicting continuous classes," 1996.
[44] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, pp. 359-366, 1989.
[45] 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.
[46] H. Mustapha and N. Abdelwahed, "Investigating the use of random forest in software effort estimation," *Procedia computer science*, vol. 148, pp. 343-352, 2019.



Amin Moradbeiky received his B.Sc. from University of Sistan and Baluchestan, Iran in 2009 and M.Sc. from Islamic Azad University, Kerman Branch, Iran in 2014, both in engineering of information technology. He is currently Ph.D. student of software engineering in Islamic Azad University, Kerman Branch, Iran. His research interests are soft computing techniques, software test and software measurement.



Vahid Khatibi Bardsiri, he is a lecturer at the Department of Computer Science, Islamic Azad University, Bardsir Branch, Iran. He holds B.Sc. and M.Sc. Degrees in software engineering from Ferdowsi University of Mashhad, Iran (2002) and Science and Research Branch of Islamic Azad University, Iran (2004), respectively. He received his Ph.D. in the area of software development effort estimation at Universiti Teknologi Malaysia (UTM), 2013. He is a senior member of International Association of Computer Science and Information Technology (IACSIT). His research interests are agile software development methods, soft computing techniques and software measurement.



Mehdi Jafari, received the B.S. in electronic engineering and M.S. degrees in communication systems from Shiraz University of technology in 1992 and 1996, respectively. During 1992-1998 he stayed in signal processing research laboratory, ministry of telecommunications of Iran. He received Ph. D degree in communication systems in science and research branch of Islamic Azad University in January 2008. He is currently head of electrical engineering department of Islamic Azad University, Kerman Branch. His main research interests are artificial intelligence, image and video processing, pattern recognition and prediction.