Applied Computer Systems

ISSN 2255-8691 (online)
ISSN 2255-8683 (print)
December 2019, vol. 24, no. 2, pp. 82–93
https://doi.org/10.2478/acss-2019-0011
https://content.sciendo.com

A Dataset-Independent Model for Estimating Software Development Effort Using Soft Computing Techniques

Mahdi Khazaiepoor[1], Amid Khatibi Bardsiri[2*], Farshid Keynia[3]
[1]*Computer Engineering Department, Kerman Branch, Islamic Azad University, Kerman, Iran*
[2]*Computer Engineering Department, Bardsir Branch, Islamic Azad University, Bardsir, Iran*
[3]*Department of Energy Management and Optimization, Graduate University of Advanced Technology, Kerman, Iran*

Abstract – **During the recent years, numerous endeavours have been made in the area of software development effort estimation for calculating the software costs in the preliminary development stages. These studies have resulted in the offering of a great many of the models. Despite the large deal of efforts, the substantial problems of the offered methods are their dependency on the used data collection and, sometimes, their lack of appropriate efficiency. The current article attempts to present a model for software development effort estimation through making use of evolutionary algorithms and neural networks. The distinctive characteristic of this model is its lack of dependency on the collection of data used as well as its high efficiency. To evaluate the proposed model, six different data collections have been used in the area of software effort estimation. The reason for the application of several data collections is related to the investigation of the model performance independence of the data collection used. The evaluation scales have been MMRE, MdMRE and PRED (0.25). The results have indicated that the proposed model, besides delivering high efficiency in contrast to its counterparts, produces the best responses for all of the used data collections.**

Keywords – **Clustering, estimation, feature selection, genetic algorithm, imperialist competitive algorithm, neural network, regression, software development effort.**

I. Introduction

Software development effort estimation is regarded as an important stage in software development projects. Therefore, many industry specialists and researchers have been devoting their attention to this field during recent years [1].

Software cost estimation incorporates the process of reaching a conclusion regarding the amount of effort required for the development of a software system [2]. The most important demanding requirement of the software development cost estimation is the accuracy of the estimation [3]. That is due to the fact that the overestimation of the software development cost might cause losing a project in a tender and, on the other hand, the underestimation might also cause the software company to be incurred by losses and/or it might cause the allocation of lower resources, as a result of which the project quality cannot be guaranteed [4], [5]. On the contrary, the accurate estimation can assist the managers and engineers in prioritisation of the project stages and proper allocation of the resources [6], [7]. This is why the accurate estimation of the software development effort is deemed both a serious challenge for project managers [8] and a major phase in project development [9].

There are many uncertain variables extant in software development effort estimation, for example, development method, programming organisation and language that make the software size take a fuzzy form [10]; also, a great many of the important and influential factors of effort estimation are determined during the final phases of software development. Therefore, these factors have to be taken into account before running an estimation effort [11]. Thus, it is difficult to estimate the costs and efforts required for software development during the early stages of software development cycle. Many of the project researchers and managers have become notoriously famous for their underestimation of the real software development costs [8]. According to the statistics presented by the International Society of Parametric Analysis (ISPA) [12] and Standish Group [13], two-thirds of software projects have failed before delivery due to being either time consuming or costly. There are two substantial reasons behind the software project failure: inappropriate project estimation in terms of size, costs and personnel required and uncertainty about the software and system requirements [8].

A majority of the studies in the area of effort estimation have been predominantly concentrated on the designing of a good estimator [14] and/or new metrics [15] and/or other aspects of the software project [16].

There are different methods for the estimation of software development effort, the most important of which can be categorised into two sets of model-based and expert-based [3]. The model-based group, as well, is classified into two sets of statistical and mathematical methods, such as regression, and smart methods, e.g., machine learning [17]. Nowadays, increasing attention is paid to machine learning for its high competency of modelling the relationships between the project effort and its components. Multilayer perceptron neural network (MLP) and neural networks drawn on radial basis functions (RBF) and metaheuristic algorithms, including

[*]Corresponding author's e-mail: a.khatibi@srbiau.ac.ir

genetic algorithm [18] and imperial competitive algorithm, are inter alia these methods.

Although there are various models offered for software development estimation, a majority of them, unfortunately, lack the required efficiency and a comprehensive model providing an acceptable and independent estimation of the data collection is yet to be presented [19]; the majority of the models are pertinent to a specific dataset, which is why the model performance is observed being highly dependent on the dataset [20]. Model blending, feature weight determination and model weighing for combining the models are amongst the methods that have been recently put into practice [19], [21].

The present article intends to combine machine learning methods during several phases in order to offer a software development effort estimation model that, besides having a high efficiency in estimating the software estimation, is maximally independent of the dataset used. To verify the model independence, six datasets, comprised of general and publicly available information featuring various sizes, will be used in effort estimation. The proposed method is composed of three general phases. During the first phase, the genetic algorithm and neural network are combined to perform the feature selection operation. During the next phase, the genetic algorithm and imperial competitive algorithm are employed to perform a clustering operation. During the final phase, the neural network is used to perform modelling and test the clusters. In the end, the obtained results are compared with the results of some other models to evaluate the efficiency of the proposed model.

The present article is organised in six sections. Prior research will be presented in Section II. Section III explains the proposed method requirements and Section IV gives details of the method. In Section V, the results are discussed and the analysis of research findings is performed in Section VI. Sections VII and VIII also discuss the validity and credibility of the method and its relevant results, and present the conclusions and suggestions for further research, respectively.

II. Related Works

Soft computation technique is a novel idea that has been around since 1994 [22]. Its domain has been speedily expanded in such a way that it is now covering such techniques as genetic algorithm and swarm intelligence. The software project effort estimation models using constructive cost model (COCOMO) and genetic algorithm are provided in [23]. The presented models were tested on NASA software projects.

In [24], the author runs a thorough study of using genetic style of programming (GP) and neural network (NN) and linear regression in problem-solving of software projects. Moreover, there are other studies offered about the use of soft computation techniques for software effort estimation in [25].

A glance at the prior studies makes it clear that a substantial fraction of the models focuses on combining various estimation strategies [26]. According to the simplicity and flexibility of analogy, comparison is the basis of the proposed models in the majority of studies. Combining analogy with genetic algorithm [27], analytics [28], particle swarm optimisation algorithm [29] and artificial neural network [30] has also been a method of choice.

Recently, the localisation idea has been introduced in software development projects and promising results have also been acquired. Khatibi in [19] offers a model using model combination and localization.

In [8], a complete analysis of various types of neural networks has been undertaken based on radial basis function, multilayer perceptron, general regression neural network and cascade correlation neural network.

III. The Proposed Method

In this section, we deal with the explication of the proposed method. First, the independent variables are normalised by mapping to [0, 1] interval using (1).

$$x_i' = \frac{x_{\max} - x_i}{x_{\max} - x_{\min}}, \quad (1)$$

where, x_i is the i-th data item and $x_{\max}$ and $x_{\min}$ are the highest and the lowest values for each independent variable.

After normalisation, the dataset is randomly divided into two parts of training data and test data (for a ratio of 70 % to 30 %). Since the results obtained from the model performance should be compared with the results of the other studies, the data assigned to two classes of the proposed model and compared reference models will be used. Therefore, the randomised division of the data will not exert a considerable effect on the comparison results.

The model consists of three general phases as stated below:

- GA and NN are used for feature selection;
- GA and ICA are used for clustering;
- MLP neural network used for modelling.

After the modelling phase, the test data are employed to perform model testing operation. At the end of this phase, such efficiency scales as MMRE, MdMRE, PRED (0.25) [31] are applied to compare the efficiency of the proposed model with that of the others. Figure 1 demonstrates the generalities of the proposed method.

A. Phase One: Feature Selection

The selection of a subset of features is one of the most important steps in enhancing the efficiency of the majority of software development effort estimation [20]. The proper selection of a subset of the most effective features, besides increasing the model speed, can considerably influence its efficiency.

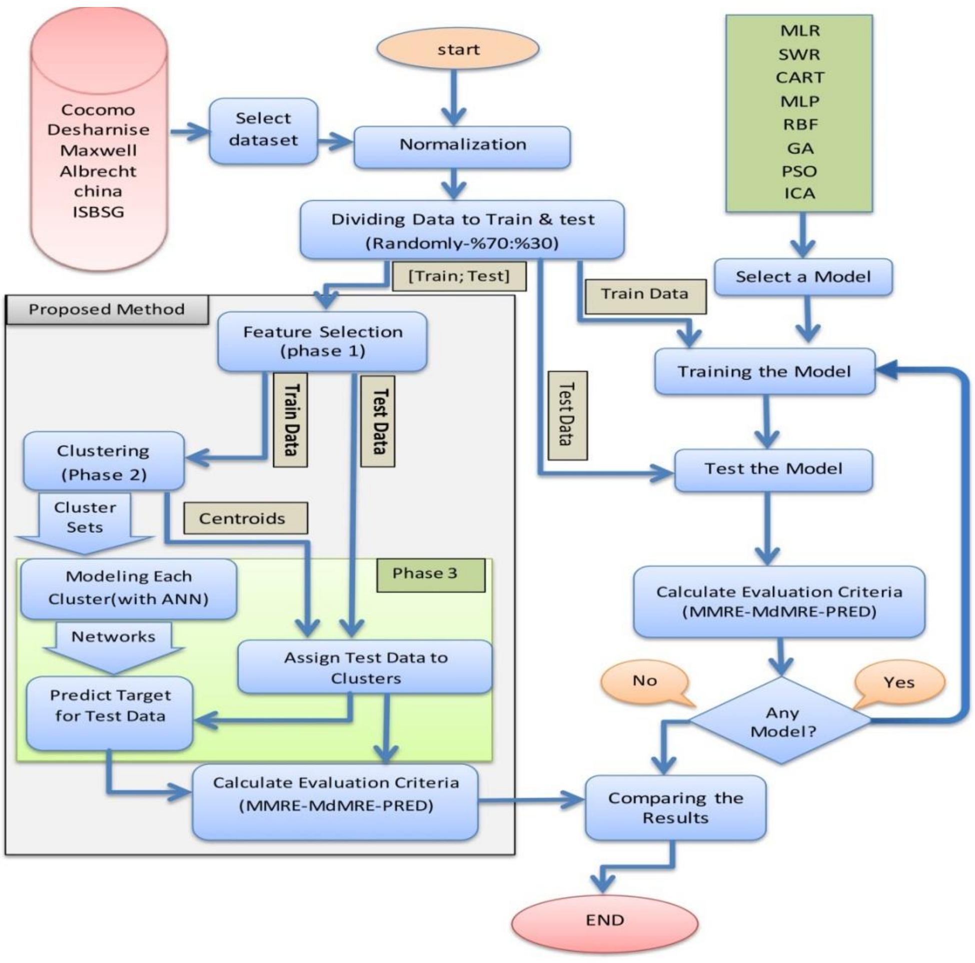

Fig. 1. Generalities of the proposed method.

During this phase, the collection of the selected data is fed as input to the genetic algorithm. The cost function of the genetic algorithm is named FeatureSelectionCost (FSC). The chromosome length is determined according to the number of the selected dataset features. Each chromosome is a bit-thread composed of “0s” and “1s” that, respectively, denote the selection and/or rejection of the corresponding feature. After the preliminary population is created, each member of the population is evaluated by FSC function. The cost function is calculated using neural networks created by the fitnn() function in the MATLAB neural network toolbox. To render results more reliable, five neural networks are evaluated and the mean value of the results is returned.

After creating and evaluating the preliminary responses, the main loop of the genetic algorithm is initiated. The loop, repeated for a hundred times, includes the following steps:

a) **Performing crossover:** the parents are selected by the use of RoulettWheelSelection function to randomly undergo one of the single-point, two-point and/or even crossover operations with probability values of 0.2, 0.3 and 0.5, respectively, selected by RoulettWheelSelection function resulting in the generation of the offspring;
b) **Response Evaluation:** immediately after the generation of responses resulting from the cross-over, the responses are evaluated by FSC functions;
c) **Mutation:** at this stage, a number of responses are selected based on a given rate and subjected to mutation;
d) **Evaluation of Offspring Resulting from Mutation:** the action is also carried out by FSC function;
e) **Blending of the three population groups:** (initial population, population resulting from crossover and the population resulting from mutation) it is followed by ordering and selecting the best as the next generation population.

The output of the algorithm forms the training and test dataset with the selected features that are sent to the forthcoming phase.

B. Phase Two: Clustering

Generally, a clustering operation is an unsupervised classification. Since our objective in the current research paper is to create a model with no dependency on any specific set of data, a total number of six datasets are selected from the effort estimation study field. Considering the difference these datasets have in terms of the nature and size, a different number of clusters is produced for each dataset. Therefore, clustering is accomplished in two stages. At the first stage, the genetic algorithm [32] is used for each dataset to calculate the number of clusters. During this stage, the genetic algorithm carries out the cluster number determination assisted by DB [33] and CS [34] indices.

Conceptually, DB index can be considered as intending to minimise the intra-cluster distance and maximise the inter-cluster distance, which correspond to the two principles of cohesion inside the cluster and separation between the clusters, respectively. Relations (2), (3), (4) and (5) show the index calculation method.

$$S_{i.q} = \sqrt[q]{\frac{1}{N_i}\sum_{x\in c_i} d\,(x, m_i)^q}; \tag{2}$$

$$d_{i.j.t} = \sqrt[t]{\sum_{p=1}^{d}\left|m_{i.p} - m_{j.p}\right|^t}; \tag{3}$$

$$R_{i.q.t} = \max_{j\in k.j\neq i}\left(\frac{s_{i.q}+s_{j.q}}{d_{i.j.t}}\right); \tag{4}$$

$$DB = \frac{1}{k}\sum_{i=1}^{k} R_{i.q.t}. \tag{5}$$

Let us suppose that x_p is a member to cluster c_i. In this case, CS index can be defined according to relations (6), (7) and (8)

$$d_p^{\max} = \max_{x_q\in c_i} d(x_p, x_q); \tag{6}$$

$$\text{mean}(d_i) = \frac{1}{N_i}\sum_{x_p\in c_i} d_p^{\max}; \tag{7}$$

$$CS = \frac{\frac{1}{k}\sum_{i=1}^{k}\text{mean}(d_i)}{\frac{1}{k}\sum_{i=1}^{k}\min_{j\in k.j\neq i} d(m_i, m_j)} \tag{8}$$

in such a way that k is the number of clusters and m_i and m_j are the centres of clusters i and j, respectively.

The lesser values of CS indices are indicative of the idea that the distance between the cluster centres is larger and the distance between the members of a cluster is smaller, i.e., the cluster cohesion and separation principles have been met.

The reason why a genetic algorithm has been applied during this stage is its speed and binary nature. Moreover, the calculation of the DB and CS indices takes a lot of operation to accomplish. At the second stage, having the number of clusters at hand, the imperialist competitive algorithm is used to perform clustering for offering good cohesion and good capability of setting the parameters. The phase outputs are the clusters and their centres.

Imperial competitive algorithm was introduced in 2007 by Atashpaz and Lucas [35]. The algorithm that is inspired by the imperialist-colony phenomenon in the real world is laid on the foundation of the assumption that there are at first a number of entities called countries, which are ranked based on certain scales. The algorithm consists of two general phases:

- competition in an empire;
- competition between the empires.

In intra-empire competition, the colonies try to achieve a stage of growth so that they can take the position of the imperialist of the empire; such a growth is carried out based on such operators as attraction and revolution. In the intra-empire competition, the empires do their best to occupy the colonies of the other empires. To do so, a colony from the weakest empire is taken away by the operator “omission” and given to one of the other empires in each repetition.

C. Phase Three: Modelling and Testing

During this phase, the MLP neural network [8] is first employed to run modelling on the training data in such a manner that a network will be constructed for every cluster. The test data are assigned to the nearest cluster based on the cluster centres computed from the previous phase. Now, the network calculates the predicted effort for each cluster of the test data assisted by the network corresponding to that cluster. In the end, the predicted effort is compared for the data sets of all clusters with the real values followed by the calculation of the efficiency scales.

IV. Experimental Design

We deal, in this section, with the introduction of the used datasets, preliminary settings of the algorithms and efficiency scales. The present article has endeavoured to compare the obtained results with the results acquired from the other common effort estimation models so as to better validate the proposed model. The compared models can be divided into one of the following classes:

- **Regression methods:** they encompass the multiple linear regression (MLR), stepwise regression (SWR) and classification and regression tree (CART);
- **Metaheuristic algorithm methods:** they embrace genetic algorithm, particle swarm algorithm and imperialist competitive algorithm;
- **Neural network methods:** they incorporate multilayer perceptron (MLP) neural network and the neural network based on radial basis functions (RBF).

A. Introducing the Datasets

To evaluate the proposed model, we use the following commonly used public datasets:

- **COCOMO81 dataset:** this dataset contains information of NASA software projects [36];
- **Albrecht dataset:** it includes the information of IBM software projects made in the 1970s [37];
- **Desharnais dataset:** in this dataset, there is information of some software projects accomplished in Canada [38];
- **Maxwell dataset:** it contains information on several projects pertinent to the trade banks in Finland [39];
- **ISBSG dataset:** this collection of data incorporates information of a large number of industrial software projects from all around the globe that have been collected and offered by ISBSG group [40];
- **China dataset:** This data collection that is the newest dataset in the area of effort estimation contains information on 499 software projects with 18 features belonging to various software companies and firms [2].

TABLE I
THE SIX SELECTED DATASETS CONSISTING OF 792 SOFTWARE PROJECT CASES

Data set	***F***	***F-cat***	***SS***	***Num***	**Unit**	**Min**	**Mean**	**Median**	**Max**
Cocomo81	17	1	1	63	months	5.9	683.5	98.0	11400.0
Albrecht	7	0	2	24	months	0.5	21.9	11.5	105.2
Desharnais	11	1	2	81	hours	546.0	4833.9	3542.0	23940.0
Maxwell	27	6	1	62	hours	583.0	822.0	5190.0	63694.0
ISBSG	7	0	2	63	hours	64.0	5588.65	3216.0	60826.0
China	18	0	1	499	hours	26.0	3921.1	1829.0	54620.0

F – number of features, *F-cat* – number of categorical variables,
SS – number of software size variables, *Num* – number of projects.

According to the existence of some missing data, various kinds of features and different measurement metrics, the present study, after implementing a preparatory operation, has made use of the information of 63 projects with seven features. A summary of the information from these data sets is provided in Table I.

B. Efficiency Scales

Numerous and diverse efficiency scales have been employed in various articles. The objective of the majority of these scales is the assessment of the model estimation accuracy.

A majority of the articles have made use of relative error size scale in lieu of the relative error that can be defined as shown in relation (9).

$$MRE_i = \frac{|\,\mathrm{Estimated}_i - \mathrm{Actual}_i\,|}{\mathrm{Actual}_i}. \tag{9}$$

A majority of the extant efficiency scales measure the estimation accuracy so they have been formed based on the same scale. The present article makes use of MMRE, MdMRE and PRED(0.25) efficiency scales defined corresponding to relations (10), (11) and (12):

$$\mathrm{MMRE} = \mathrm{mean}(MRE); \tag{10}$$

$$\mathrm{MdMRE} = \mathrm{median}(MRE); \tag{11}$$

$$\mathrm{PRED}(0.25) = \frac{A}{N}, \tag{12}$$

where *A* is the number of observations in which their MRE values are below 0.25 and *N* is the total number of observations.

C. Preliminary Settings

The present article makes use of genetic algorithm and multilayer perceptron neural network for feature selection; also, the genetic algorithm and imperialist competitive algorithm are applied to perform clustering and, finally, the neural network is utilised to carry out the modelling operation. The preliminary settings and the parameter values of genetic algorithm and imperialist competitive algorithm have been obtained based on trial and error as well as according to an observation of the prior and similar works as summarised in Tables II and III.

The neural network used herein is a two-layer perceptron in the first layer of which the number of neurons is determined according to the intended dataset based on trial and error. The first input layer is the values of the features selected from every dataset. Thus, the number of the neural network inputs is different in every dataset. Trainscg function (conjugate gradient) has been used to train the neural network and the test data; training data and validation data have been divided in ratios of 70 %, 15 % and 15 %, respectively.

TABLE II
THE INITIAL SETTINGS AND THE VALUES OF PARAMETERS OF GENETIC ALGORITHMS

Name	**Description**	**Value**
MaxIt	Maximum of iteration	100
Npop	Num of initial population	30
Pc	Crossover percentage	0.8
Pm	Mutation percentage	0.3
Mu	Mutation rate	0.1

TABLE III
THE INITIAL SETTINGS AND THE VALUES OF PARAMETERS OF ICA

Name	**Description**	**Value**
MaxIt	Maximum of iteration	1000
Npop	Num. of initial population	Num. of dataset records
Nemp	Num. of empires	10
Alpha	Selection pressure	3
Beta	Assimilation coefficient	1
Prevolution	Revolution probability	0.2
Mu	Revolution rate	0.3
Zeta	Effect coefficient of the colony value on the empire value	0.5

V. EXPERIMENTAL RESULTS

The COCOMO dataset is usually applied in the evaluation process of software project effort estimation models. Table IV, Figs. 2 and 3 summarise the effort estimation values for each of the aforesaid models.

The Albrecht dataset includes information pertaining to 24 projects with eight features, all of which have been applied in the effort estimation process. The results obtained from the various models based on this dataset for three parameters of

MMRE, MdMRE and PRED(0.25) have been given in Table V, Figs. 4 and 5.

The Desharnais dataset is one of the most widely used dataset in studies on effort estimation area. The dataset was presented in 1997 by Desharnais et al. The results of various models using this dataset for three parameters of MMRE, MdMRE and PRED (0.25) have been summarised in Table VI, Figs. 6 and 7.

The Maxwell dataset is one of the most well-known and frequently applied datasets in the area of effort estimation. The results of various models using this dataset for three parameters of MMRE, MdMRE and PRED(0.25) are summarised in Table VII, Figs. 8 and 9.

The ISBSG dataset has been offered by the International Software Benchmarking Standards Group (ISBSG) and it is continuously updated. The results of various models using this dataset for three parameters of MMRE, MdMRE and PRED(0.25) are summarised in Table VIII, Figs. 10 and 11.

The results of various models using the China dataset for three parameters of MMRE, MdMRE and PRED(0.25) are summarised in Table IX, Figs. 12 and 13.

TABLE IV
THE RESULTS ON COCOMO DATASET

	MetaHeuristic Methods			**Neural Networks**		**Regression Methods**			
	GA	**ICA**	**PSO**	**MLP**	**RBF**	**CART**	**MLR**	**SWR**	**Proposed**
MMRE	4.7227	5.8391	4.6650	2.7064	2.2826	1.0592	4.5950	3.2778	**0.86011**
MdMRE	0.9475	0.9297	0.9478	1.0367	0.9245	0.8846	0.8104	0.8106	**0.645129**
PRED(%)	5.263	15.789	5.263	10.526	15.789	15.789	15.789	21.052	**31.578**

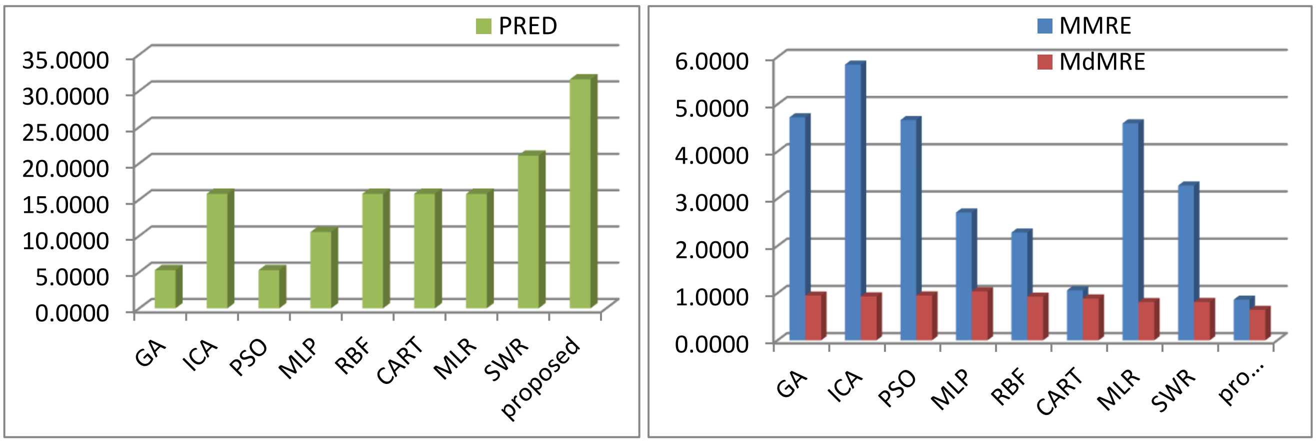

Fig. 2. Comparing methods (MMRE, MdMRE, PRED) on COCOMO dataset.

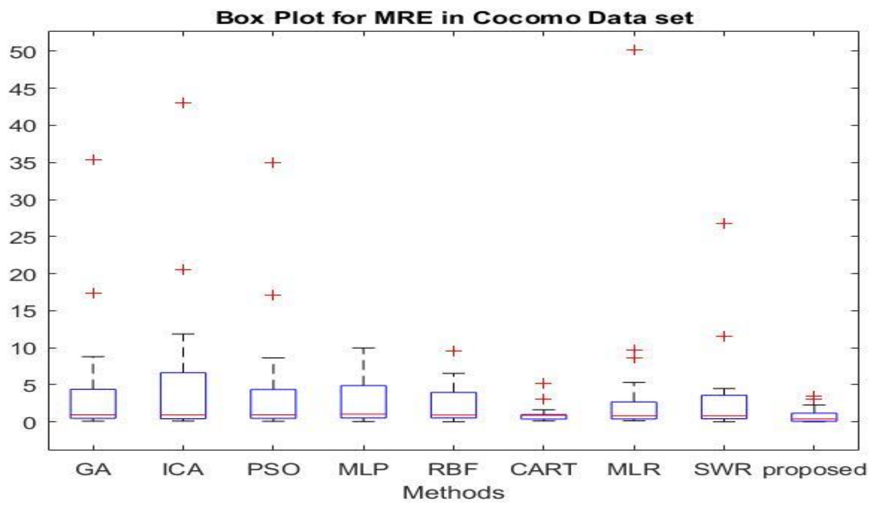

Fig. 3. Box plot for MRE in COCOMO dataset.

TABLE V

THE RESULTS ON ALBRECHT DATASET

	MetaHeuristic Methods			**Neural Networks**		**Regression Methods**			
	GA	**ICA**	**PSO**	**MLP**	**RBF**	**CART**	**MLR**	**SWR**	**Proposed**
MMRE	0.8515	0.4757	0.7395	0.5029	0.4845	0.7306	0.5635	0.5444	**0.4038**
MdMRE	0.3634	0.4419	0.4759	0.4744	0.3162	0.6742	0.3900	**0.2633**	0.3042
PRED(%)	14.2857	28.5714	28.5714	14.2857	28.5714	0.0000	14.2857	**42.8571**	**42.8571**

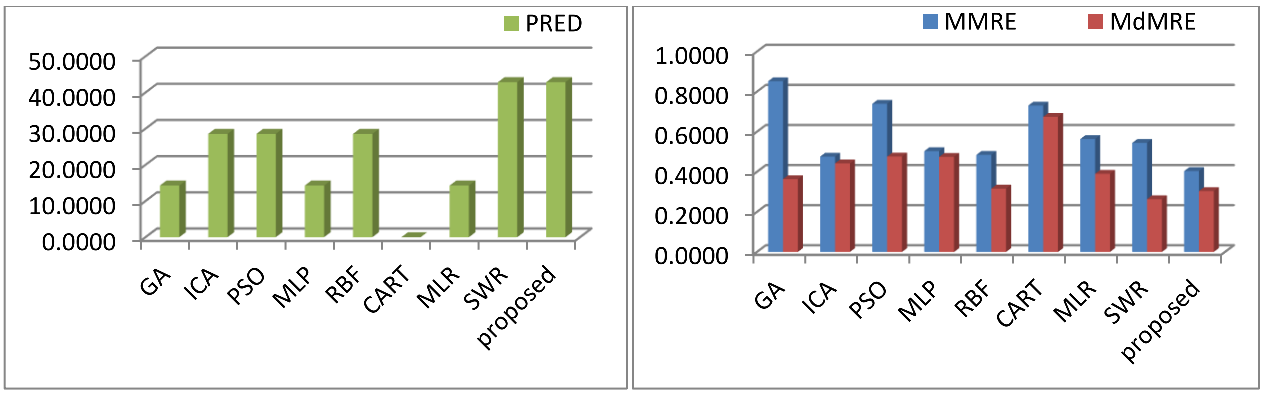

Fig. 4. Comparing methods (MMRE, MdMRE, PRED) on Albrecht dataset.

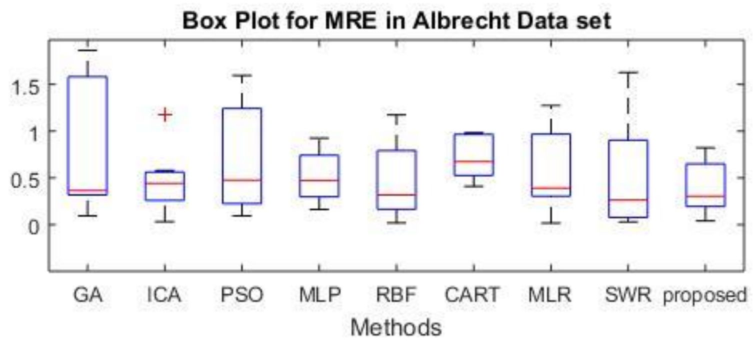

Fig. 5. Box plot for MRE in Albrecht dataset.

TABLE VI

THE RESULTS ON DESHARNAIS DATASET

	MetaHeuristic Methods			**Neural Networks**		**Regression Methods**			
	GA	**ICA**	**PSO**	**MLP**	**RBF**	**CART**	**MLR**	**SWR**	**Proposed**
MMRE	0.8318	0.5627	0.7050	0.7953	0.5474	0.6261	0.5048	**0.4460**	0.4776
MdMRE	0.8581	0.4839	0.7952	0.4775	0.3720	0.7069	0.3533	0.3880	**0.2104**
PRED(%)	0.0000	13.0435	8.6957	30.4348	39.1304	17.3913	39.1304	30.4348	**52.1739**

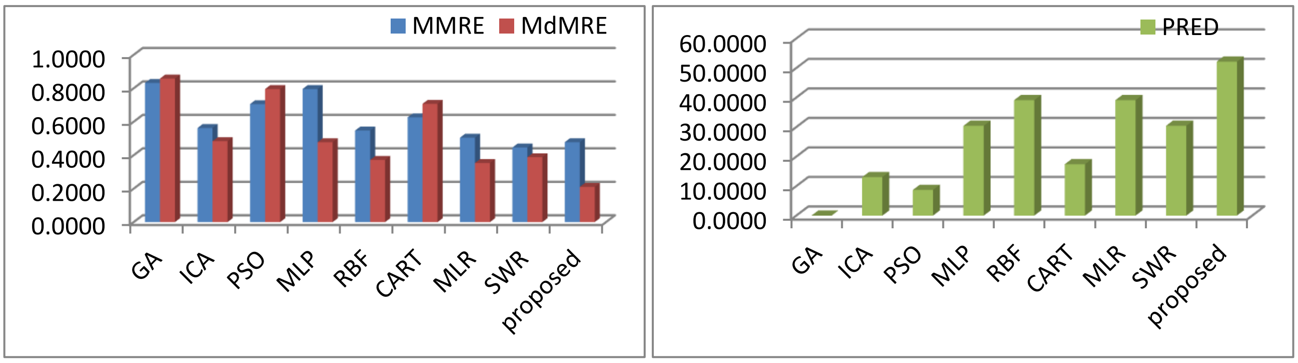

Fig. 6. Comparing methods (MMRE, MdMRE, PRED) on Desharnais dataset.

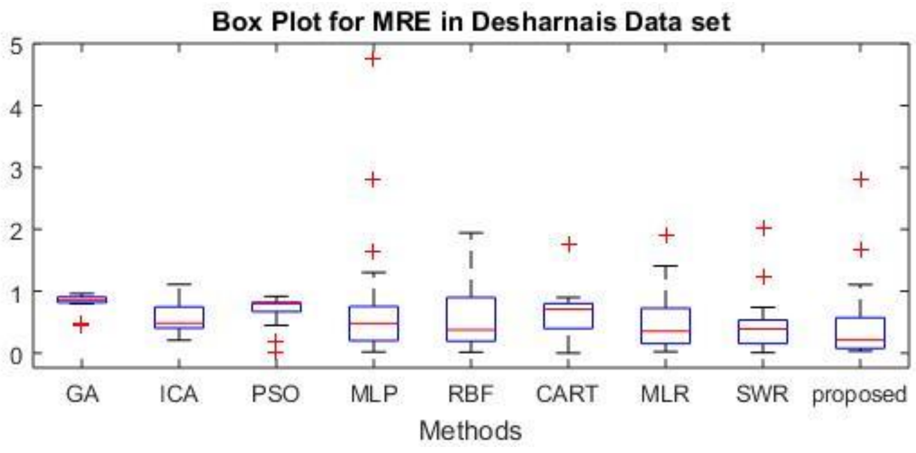

Fig. 7. Box plot for MRE in Desharnais dataset.

TABLE VII
THE RESULTS ON MAXWELL DATASET

	MetaHeuristic Methods			Neural Networks		Regression Methods			
	GA	**ICA**	**PSO**	**MLP**	**RBF**	**CART**	**MLR**	**SWR**	**Proposed**
MMRE	0.7482	0.6275	0.8217	0.7421	0.5538	0.6692	0.9286	0.5234	**0.5083**
MdMRE	0.4407	0.4558	0.5233	0.4661	0.5552	0.8054	0.6174	0.3381	**0.2483**
PRED(%)	36.8421	36.8421	26.3158	15.7895	15.7895	10.5263	26.3158	26.3158	**52.6316**

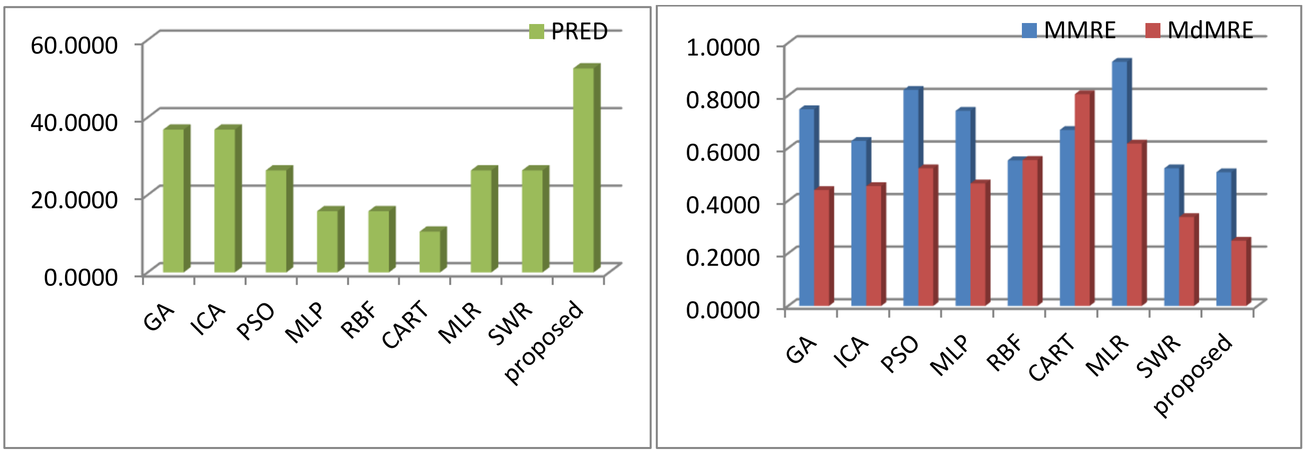

Fig. 8. Comparing methods (MMRE, MdMRE, PRED) on Maxwell dataset.

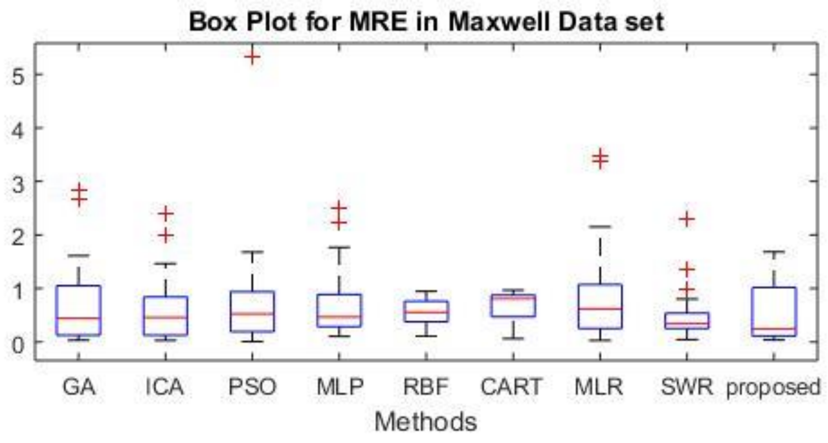

Fig. 9. Box plot for MRE in Maxwell dataset.

TABLE VIII
THE RESULTS ON ISBSG DATASET

	MetaHeuristic Methods			Neural Networks		Regression Methods			
	GA	ICA	PSO	MLP	RBF	CART	MLR	SWR	Proposed
MMRE	0.8773	0.6900	0.8773	1.1218	0.7543	0.7863	0.5478	0.5360	**0.3523**
MdMRE	0.8859	0.7099	0.8859	0.6258	0.5114	0.7278	0.5208	0.4224	**0.2927**
PRED(%)	0.0000	5.2632	0.0000	15.7895	21.0526	10.5263	21.0526	36.8421	**42.1053**

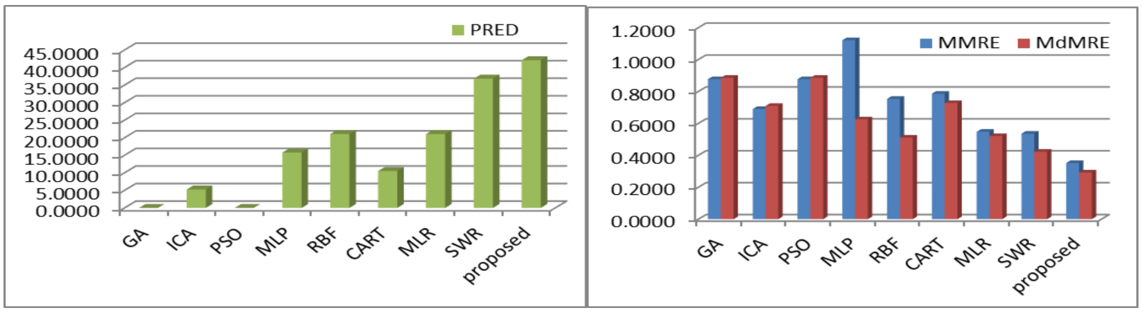

Fig. 10. Comparing methods (MMRE, MdMRE, PRED) on ISBSG dataset.

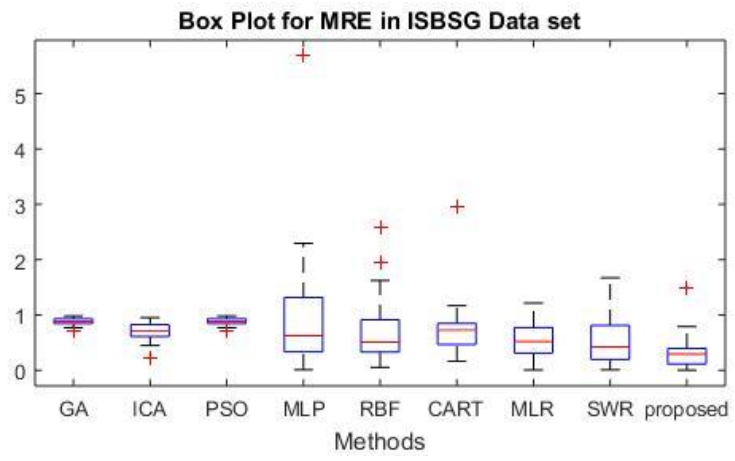

Fig. 11. Box plot for MRE in ISBSG dataset.

TABLE IX
THE RESULT ON THE CHINA DATASET

	MetaHeuristic Methods			Neural Networks		Regression Methods			
	GA	ICA	PSO	MLP	RBF	CART	MLR	SWR	Proposed
MMRE	0.5039	0.4571	0.2361	0.7053	2.9239	0.3696	0.1877	0.2566	**0.1076**
MdMRE	0.2942	0.1962	0.1127	0.2765	0.8876	0.2658	0.0675	0.0941	**0.0512**
PRED(%)	44.6667	60.6667	75.3333	47.3333	16.0000	47.3333	80.0000	72.0000	**84.0000**

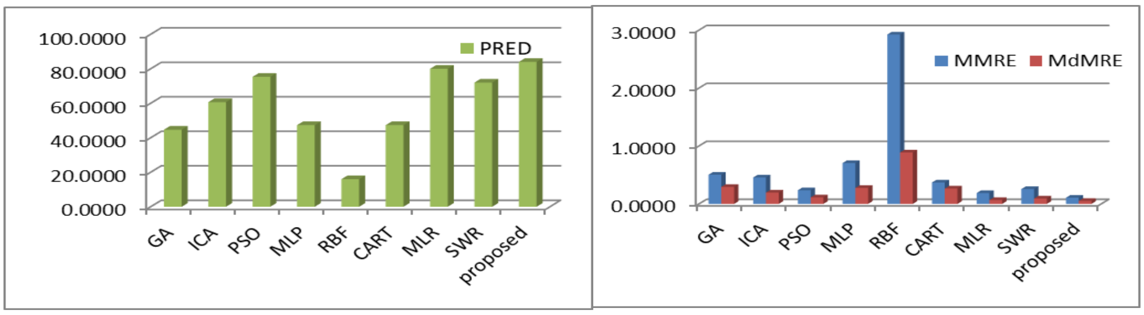

Fig. 12. Comparing methods (MMRE, MdMRE, PRED) on the China dataset.

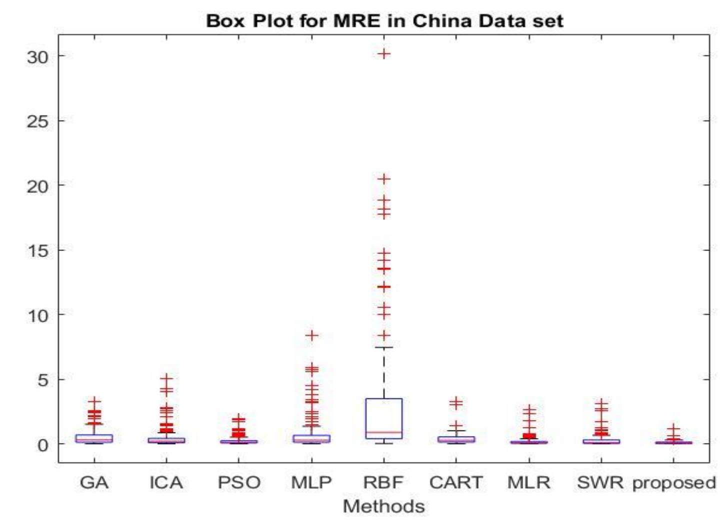

Fig. 13. Box plot for MRE in the China dataset.

VI. Result Analysis

The results indicate that the efficiency of the majority of the models is undeniably dependent on the dataset used. Table X summarises the model efficiency rates according to three levels: good, medium and weak. The table indicators have been computed corresponding to the following algorithm:

```
DataSet = {Cocomo, Albrecht, Desharnais, Maxwell, ISBSG, China}
Method = {GA, ICA, PSO, MLP, RBF, CART, MLR, SWR,
        Proposed}
For each DatsSet_i
    For each Method_j
        $A_{ij} = 1 - \frac{\text{MMRE}[\text{DataSet}_i \ \text{Method}_j] - \min(\text{MMRE}[\text{DataSet}_i])}{\max(\text{MMRE}[\text{DataSet}_i]) - \min(\text{MMRE}[\text{DataSet}_i])}$
        $B_{ij} = 1 - \frac{\text{MdMRE}[\text{DataSet}_i \ \text{Method}_j] - \min(\text{MdMRE}[\text{DataSet}_i])}{\max(\text{MdMRE}[\text{DataSet}_i]) - \min(\text{MdMRE}[\text{DataSet}_i])}$
        $C_{ij} = \frac{\text{PRED}[\text{DataSet}_i \ \text{Method}_j] - \min(\text{PRED}[\text{DataSet}_i])}{\max(\text{PRED}[\text{DataSet}_i]) - \min(\text{PRED}[\text{DataSet}_i])}$
        $X_{ij} = (A_{ij} + B_{ij} + C_{ij})/3$
        If ($X_{ij}$ <= 30%) then EFFICIENCY[i,j] = 'weak'
        Else if ($X_{ij}$ > 30% & $X_{ij}$ <= 70%) then EFFICIENCY[i,j] =
        'Medium'
        Else EFFICIENCY[i,j] = 'Strong'
    END (for j)
END (for i)
```

It can be stated based on Table X that the regression-based models have had better performance in comparison to that of the other models and the evolutionary algorithm models have shown the weakest performance. Moreover, the results obtained for each model largely depend on the used dataset. As an example, PSO model has been proved strong for some datasets, weak for some others and yet intermediately performed for the third group of datasets. Although parameter setting plays an effective role in the performance of these algorithms, it can be observed for the proposed method that it has exhibited the best performance for all of the datasets. The issue is reflective of the reality that the proposed model is least dependent on the used set of data. One factor largely influential on the model performance can be the number of the training samples because the models, particularly the proposed model, have displayed the best performance for the China dataset whose records are a lot higher than those of the other data collections. According to the fact that the proposed model makes use of two phases, i.e., feature selection and clustering, two factors, namely the selection of the most effective features and localisation, have been able to corroborate the model performance. The enhancement rates are more distinct in regard of the China dataset; in addition, the localisation operation has been found having no significant effect on the improvement of the proposed model performance due to a low number of samples. Table XI presents the mean percentage of performance improvement of the proposed model for various scales on the datasets used herein.

TABLE X
Comparison of Performance Rate of the Proposed Model

DataSet	MetaHeuristic Methods			Neural Networks		Regression Methods			
	GA	**ICA**	**PSO**	**MLP**	**RBF**	**CART**	**MLR**	**SWR**	**Proposed**
Cocomo	Weak	Weak	Weak	Medium	Medium	**Strong**	Weak	Medium	**Strong**
Albrecht	Weak	Medium	Medium	Weak	Medium	Weak	Weak	**Strong**	**Strong**
Desharnais	Weak	Weak	Weak	Weak	Medium	Weak	Medium	**Strong**	**Strong**
Maxwell	Medium	Medium	Medium	Weak	Weak	Weak	Medium	Medium	**Strong**
ISBSG	Weak	Weak	Weak	Weak	Medium	Weak	Medium	**Strong**	**Strong**
China	Weak	Medium	**Strong**	Weak	Weak	Weak	**Strong**	**Strong**	**Strong**

TABLE XI
PERFORMANCE IMPROVEMENT OF THE PROPOSED MODEL

Dataset	China	ISBSG	Maxwell	Desharnais	Albrecht	Cocomo
MMRE	86 %	54 %	27 %	24 %	34 %	77 %
MdMRE	86 %	56 %	52 %	62 %	28 %	57 %
PRED(0.25)	38 %	67 %	53 %	57 %	50 %	64 %

According to Table XI, it can be stated that the highest improvement percentage has been documented for MMRE and MdMRE scales on the China dataset. One possible reason behind such an improvement can be the large number of the records in this set of data that, per se, causes the localisation phase of the proposed method exhibit more appropriate performance. The improvement results are also considerable for the other datasets.

VII. THREATS TO VALIDITY

Metaheuristic and neural networks have been used during various phases of the present research. Considering the random nature of these tools, the results obtained from every implementation might appear a little different from one another. MDMRE, MMRE and PRED(0.25) used in the present article are biased. They have only been chosen herein because they were found most frequently employed in the prior research.

To perform training and test operations, all of the datasets have been assigned randomly to training and test groups in a 70 % to 30 % ratio, respectively. The random assignment of the data can have a considerable influence on the model results. However, considering that all the models are run based on a single classification, there will be made not much of an effect on the overall work because the objective has been to compare the performance of various models and to evaluate the model dependency on the dataset applied.

The datasets used herein contain a low number of records, except for the China dataset. The low number of the records in a dataset reduces the effect of the localisation operation of the clustering phase. According to the results, it can be seen that the model proposed herein has had better performance in regard of the China dataset.

VIII. CONCLUSION

The accuracy of the software development project effort estimation plays a substantial role in the project management, cost overestimation and/or underestimation. Having a highly accurate model independent of the used dataset has always been demanded by the researchers in this study field. After performing normalisation operation in the model, the dataset has been assigned to two groups of training and test data (for a ratio of 70 % to 30 %, respectively). The data assignment has been conducted in a randomised manner. The model consists of three general phases as described below:

- **Feature Selection:** during this phase, genetic algorithm and MLP neural network are first applied for every dataset to select the most effective features influencing the project development effort.
- **Clustering:** during this phase, genetic algorithm and imperialist competitive algorithm are used to run clustering over the training set of the data collection.
- **Modelling and Test:** this phase performs the modelling with the help of the MLP neural network; then, the test data are employed to perform the model test operation. In the end, MMRE, MdMRE and PRED(0.25) are applied as the efficiency scales to compare the efficiency rate of the model with that of the other models.

The results have indicated that the proposed model outperforms all the other methods for all the datasets and regression-based methods come next to it. Moreover, the largest superiority of the proposed model is its independence of the datasets used. The problem of the model efficiency dependence on the used dataset can be seen in the majority of the models designed previously. Since the proposed model uses localisation and clustering, its performance on larger datasets might be better due to better clustering. Based on the obtained results, the following topics can be suggested for further research in line with model performance improvement:

- using regression methods in modelling;
- using fuzzy methods in clustering;
- using new sets of data with a larger number of records.

REFERENCES

[1] X.-Y. Jing, F. Qi, F. Wu, and B. Xu, “Missing Data Imputation Based on Low-Rank Recovery and Semi-Supervised Regression for Software Effort Estimation” in *Proceedings of the 38th International Conference on Software Engineering (ICSE 2016)*, 2016, pp. 607–618. https://doi.org/10.1145/2884781.2884827

[2] F. Qi, X.-Y. Jing, X. Zhu, X. Xie, B. Xu, and S. Ying, “Software Effort Estimation Based on Open Source Projects: Case Study of Github,” *Information and Software Technology*, vol. 92, pp. 145–157, Dec. 2017. https://doi.org/10.1016/j.infsof.2017.07.015

[3] 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, Dec. 2016. https://doi.org/10.1016/j.asoc.2016.08.004

[4] M. Jørgensen, “The Influence of Selection Bias on Effort Overruns in Software Development Projects,” *Information and Software Technology*, vol. 55, no. 9, pp. 1640–1650, Sep. 2013. https://doi.org/10.1016/j.infsof.2013.03.001

[5] S. Grimstad, M. Jørgensen, and K. Moløkken-Østvold, “Software Effort Estimation Terminology: The Tower of Babel,” *Information and Software Technology*, vol. 48, no. 4, pp. 302–310, Apr. 2006. https://doi.org/10.1016/j.infsof.2005.04.004

[6] B. Kitchenham, S. Lawrence Pfleeger, B. McColl, and S. Eagan, “An Empirical Study of Maintenance and Development Estimation Accuracy,” *Journal of Systems and Software*, vol. 64, no. 1, pp. 57–77, Oct. 2002. https://doi.org/10.1016/S0164-1212(02)00021-3

[7] M. Jorgensen and M. Shepperd, “A Systematic Review of Software Development Cost Estimation Studies,” *IEEE Transactions on Software Engineering*, vol. 33, no. 1, pp. 33–53, Jan. 2007. https://doi.org/10.1109/TSE.2007.256943

[8] A. B. Nassif, M. Azzeh, L. F. Capretz, and D. Ho, “Neural Network Models for Software Development Effort Estimation: A Comparative Study,” *Neural Computing and Applications*, vol. 27, no. 8, pp. 2369–2381, Nov. 2015. https://doi.org/10.1007/s00521-015-2127-1

[9] M. Jørgensen and D. I. Sjøberg, “Impact of Effort Estimates on Software Project Work,” *Information and Software Technology*, vol. 43, no. 15, pp. 939–948, Dec. 2001. https://doi.org/10.1016/S0950-5849(01)00203-8

[10] J. Khan, Z. A. Shaikh, and A. B. Nauman, “Development of Intelligent Effort Estimation Model Based on Fuzzy Logic Using Bayesian Networks” in *International Conference on Advanced Software Engineering and Its Applications*, Springer, 2011, pp. 74–84. https://doi.org/10.1007/978-3-642-27207-3_9

[11] R. Fuentetaja, D. Borrajo, C. L. López, and J. Ocón, "Multi-Step Generation of Bayesian Networks Models for Software Projects Estimations," *International Journal of Computational Intelligence Systems*, vol. 6, no. 5, pp. 796–821, 2013. https://doi.org/10.1080/18756891.2013.805583

[12] D. Eck, et al., *Parametric Estimating Handbook*, The International Society of Parametric Analysts, 2009.

[13] J. Lynch, "Chaos Manifesto," The Standish Group, 2009.

[14] J. Moeyersoms, E. Junqué de Fortuny, K. Dejaeger, B. Baesens, and D. Martens, "Comprehensible Software Fault and Effort Prediction: A Data Mining Approach," *Journal of Systems and Software*, vol. 100, pp. 80–90, Feb. 2015. https://doi.org/10.1016/j.jss.2014.10.032

[15] S. R. Chidamber and C. F. Kemerer, "A Metrics Suite for Object Oriented Design," *IEEE Transactions on Software Engineering*, vol. 20, no. 6, pp. 476–493, Jun. 1994. https://doi.org/10.1109/32.295895

[16] T. Menzies, Z. Chen, J. Hihn, and K. Lum, "Selecting Best Practices for Effort Estimation," *IEEE Transactions on Software Engineering*, vol. 32, no. 11, pp. 883–895, Nov. 2006. https://doi.org/10.1109/TSE.2006.114

[17] C. Lopez-Martin, C. Isaza, and A. Chavoya, "Software Development Effort Prediction of Industrial Projects Applying a General Regression Neural Network," *Empirical Software Engineering*, vol. 17, no. 6, pp. 738–756, Dec. 2012. https://doi.org/10.1007/s10664-011-9192-6

[18] A. Idri, F. azzahra Amazal, and A. Abran, "Analogy-Based Software Development Effort Estimation: A Systematic Mapping and Review," *Information and Software Technology*, vol. 58, pp. 206–230, Feb. 2015. https://doi.org/10.1016/j.infsof.2014.07.013

[19] A. Khatibi Bardsiri, S. M. Hashemi, and M. Razzazi, "GVSEE: A New Global Model to Estimate Software Services Development Effort," *Journal of the Chinese Institute of Engineers*, vol. 39, no. 6, pp. 765–776, 2016. https://doi.org/10.1080/02533839.2016.1176873

[20] J. Keung, E. Kocaguneli, and T. Menzies, "Finding Conclusion Stability for Selecting the Best Effort Predictor in Software Effort Estimation," *Automated Software Engineering*, vol. 20, no. 4, pp. 543–567, May 2012. https://doi.org/10.1007/s10515-012-0108-5

[21] D. Wu, J. Li, and Y. Liang, "Linear Combination of Multiple Case-Based Reasoning With Optimized Weight for Software Effort Estimation," *The Journal of Supercomputing*, vol. 64, no. 3, pp. 898–918, Dec. 2010. https://doi.org/10.1007/s11227-010-0525-9

[22] L. A. Zadeh, "Soft Computing and Fuzzy Logic," in *Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems: Selected Papers by Lotfi A. Zadeh*, World Scientific, 1996, pp. 796–804. https://doi.org/10.1142/9789814261302_0042

[23] A. F. Sheta, "Estimation of the COCOMO Model Parameters Using Genetic Algorithms for NASA Software Projects," *Journal of Computer Science*, vol. 2, no. 2, pp. 118–123, Feb. 2006. https://doi.org/10.3844/jcssp.2006.118.123

[24] J. J. Dolado and L. Fernandez, "Genetic Programming, Neural Networks and Linear Regression in Software Project Estimation" in *Proceedings of International Conference on Software Process Improvement, Research, Education and Training*, 1998.

[25] A. Sheta, D. Rine, and A. Ayesh, "Development of Software Effort and Schedule Estimation Models Using Soft Computing Techniques" in *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pp. 1283–1289, Jun. 2008. https://doi.org/10.1109/CEC.2008.4630961

[26] 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, no. 4, pp. 628–640, Apr. 2007. https://doi.org/10.1016/j.jss.2006.06.006

[27] S.-J. Huang and N.-H. Chiu, "Optimization of Analogy Weights by Genetic Algorithm for Software Effort Estimation," *Information and Software Technology*, vol. 48, no. 11, pp. 1034–1045, Nov. 2006. https://doi.org/10.1016/j.infsof.2005.12.020

[28] Q. Song and M. Shepperd, "Predicting Software Project Effort: A Grey Relational Analysis Based Method," *Expert Systems with Applications*, vol. 38, no. 6, pp. 7302–7316, Jun. 2011. https://doi.org/10.1016/j.eswa.2010.12.005

[29] 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, no. 3, pp. 501–526, Sep. 2012. https://doi.org/10.1007/s11219-012-9183-x

[30] V. K. Bardsiri, D. N. A. Jawawi, S. Z. M. Hashim, and E. Khatibi, "Increasing the Accuracy of Software Development Effort Estimation Using Projects Clustering," *IET software*, vol. 6, no. 6, pp. 461–473, Dec. 2012. https://doi.org/10.1049/iet-sen.2011.0210

[31] A. K. Bardsiri, S. M. Hashemi, and M. Razzazi, "Statistical analysis of the most popular software service effort estimation datasets," *Journal of Telecommunication, Electronic and Computer Engineering*, vol. 7, no. 1, pp. 87–96, 2015.

[32] D. E. Goldberg and J. Richardson, "Genetic Algorithms With Sharing for Multimodal Function Optimization" in *Proceedings of the Second International Conference on Genetic Algorithms*, Lawrence Erlbaum, 1987.

[33] D. L. Davies and D. W. Bouldin, "A Cluster Separation Measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, no. 2, pp. 224–227, Apr. 1979. https://doi.org/10.1109/TPAMI.1979.4766909

[34] C.-H. Chou, M.-C. Su, and E. Lai, "A New Cluster Validity Measure and Its Application to Image Compression," *Pattern Analysis and Applications*, vol. 7, no. 2, pp. 205–220, Jun. 2004. https://doi.org/10.1007/s10044-004-0218-1

[35] E. Atashpaz-Gargari and C. Lucas, "Imperialist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition" in *2007 IEEE Congress on Evolutionary Computation*, IEEE, 2007, pp. 4661–4667. https://doi.org/10.1109/CEC.2007.4425083

[36] B. W. Boehm, "Software Engineering Economics," *IEEE Transactions on Software Engineering*, vol. 10, no. 1, pp. 4–21, Jan. 1984. https://doi.org/10.1109/TSE.1984.5010193

[37] 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*, vol. 9, no. 6, pp. 639–648, Nov. 1983. https://doi.org/10.1109/TSE.1983.235271

[38] J. M. Desharnais, "Analyse statistique de la productivitie des projets informatique a partie de la technique des point des fonction," Master's Thesis, University of Montreal, 1989.

[39] K. D. Maxwell, *Applied Statistics for Software Managers*, Prentice Hall, 2002.

[40] International Software Benchmarking Standards Group. [Online]. Available: https://www.isbsg.org/

Mahdi Khazaiepoor received a M.S. degree in computer-software engineering from science and research branch, IAU university in 2008, and now he is a PhD candidate in Kerman branch, IAU university. He is currently dean of software engineering department, IAU university, Birjand branch, Iran.
E-mail: mkhazaiepoor@gmail.com

Amid Khatibi Bardsiri received his B.S. degree in computer software engineering from Shahid Bahonar university, Kerman, Iran in 2008, and his M.S. and PhD degree in software engineering from Islamic Azad University, Tehran, Iran, in 2014. He published about 45 research papers in international journals and conference proceedings. His areas of research include information systems engineering, software development, software metrics, grid computing, and cloud computing.
E-mail: a.khatibi@srbiau.ac.ir
ORCID iD: https://orcid.org/0000-0001-9640-498X

Farshid Keynia, Associated Professor Department of Energy Management and Optimization Graduate University of Advanced Technology, Kerman, IRAN Ph.D., Electrical Engineering, Semnan University. M.S., Electrical Engineering, Shahid Bahonar University, Kerman, IRAN. B.S., Electrical Engineering, Shahid Bahonar university, Kerman, Iran.
E-mail: fkeynia@gmail.com
ORCID iD: https://orcid.org/0000-0001-6286-3600