A Novel Method for Software Effort Estimation Using by MLPNN, Meta-Heuristic Algorithms and Regression Based Methods

Mahdi Khazaiepoor[1], Amid Khatibi Bardsiri[2*]

[1] Computer engineering department, Kerman branch, Islamic Azad University, Kerman, IRAN
[2] Computer engineering department, Bardsir branch, Islamic Azad University, Bardsir, IRAN

Abstract
Today, effort estimation of software development has got a great importance in managing the projects properly. Proper and accurate cost estimation helps the customers and investing and also has a determinate role in logical decision making during doing and managing a project. Various models of cost estimation has been invented and used up to now. These models try to estimate the amount of necessary effort for doing a project. Most of the provided models act based on the data collected from the previous projects. Some of the efficient instruments for making these models are regression, neural network and meta-heuristic algorithms. In this paper we have used Multiple Layer Perceptron (MLP) neural network and Multi Linear Regression (MLR) and combined them with Genetic Algorithm (GA) and Imperialist Competition Algorithm (ICA). As a result we reached a model for effort estimation that can do the software projects effort estimation with less inaccuracy comparing with the previous methods. For this purpose, the data collections of COCOMO, Maxwell and Albrecht have been used which are standard and available for assessment and comparing the suggested model. Due to the given result, average performance improvement of suggested model for MMRE performance criterion on each data collections is 23%, 38% and 35%.

Keywords:
Effort Estimation of software development, Multi Linear Regression (MLR), neural network, Genetic Algorithm (GA), Imperialist Competition Algorithm (ICA), Maxwell, Albrecht, COCOMO.

1. Introduction

There is usually no chance of controlling, monitoring and accurate programing in software projects. Therefore effort estimation is one of the most important actions in managing a software project. Efficient management of software projects need an accurate and certain estimation [1]. On the other hand, no model is completely trustable in estimation process. Since, none of the estimation models is able to estimate the project's cost accurately. In recent years, researchers have provided lots of models for effort estimation of software development which can be divided in two groups of algorithmic and non-algorithmic. Algorithmic methods are mostly established based on mathematic methods which try to calculate the relation between project factors and effort development based on the mathematics contractions. Regression models such as multi linear regression (MLR), Step Wise Regression (SWR) [2], Regression tree [3] and Software Living Management (SLIM) [4], Constructive Cost Model (COCOMO) [5] and Software Evaluation and Estimation of Resources – Software Estimating Model (SEER-SEM) [6] are some of the well-known algorithm methods in effort estimation.

Non-algorithmic methods act based on active analysis of software project's factors. Some of the most practical methods are learning oriented models (LOM), analogy based estimation (ABE), and Expert Judgment (EJ). Researchers mostly like ABC model because it is more simple and practical [12, 13, 14]. This method works based on data comparison of previous projects which are similar to the present ones. One of the problems of ABE model is that it doesn't provide a proper estimation for complex and incompatible data collections [15]. Also it cannot cover a vast area of software projects [16, 17]. Some of the software project's features make effort estimation process harder than they seem to be [18]. Also some essays claim that the constructed models are not necessarily practical for all data collections [19, 20, 21, 22, and 23]. Models combination, features weighting and model's weighting are some of the methods which have been used recently [24, 25].

Lots of methods are provided for effort estimation of software development but unfortunately most of them are not very practical. Also, the other problem is depending performance to data collections. In this essay we decided to combine algorithm and non-algorithm methods to provide a model for effort estimation of software development which is highly practical dependent from data collections.

In the first phase of suggested method, the operation of choosing the most efficient features on software project's effort is being done combining genetic algorithm and MLP neural network, and then in next phase, features weighting is being done using MLR method due to the effect of each feature on project's effort and in final phase the weights are optimized.

The construction of paper is in 6 parts as following. In the second part, related works are provided and the requirements of proposed method are mentioned in part

Manuscript received January 5, 2026
Manuscript revised January 20, 2026
https://doi.org/10.22937/IJCSNS.2026.26.1.20

three. In section four, suggested method and in part five, evaluation of results are mentioned. In part seven, conclusion and future works are mentioned.

2. Related Works

Techniques of soft computing is somehow a new Idea that we are dealing with since 1994 [26] and its domain is vastly growing that somehow some technics like genetic algorithms and swarm intelligence are a part of this domain. Models of effort estimation in software projects by using the model of COCOMO and genetic algorithm are showed in [27]. These models are tested in NASA software projects. In [28] the writer had done many researches on using Genetic Programing (GP) and Neuro Programing (NP) and line regression on problem solving of estaminet of software projects. Adding to that he had done some studies on using soft computing for effort estimation shown on [29].
When you look closely on these works you can see that the general works were proposing for combining different strategies [30].

In many of the studies according to the simplicity and flexibility of the differencing method we have used this as the main method for proposal models. Combining the methods of analogy with genetic algorithms and analyzing [32, 33] are some of these examples. For instance the combination model of analogy and genetics were using for strategy of weighting.

Lately the Idea of localization in software projects is spreading and it had some good results. Menzies [34] had done some researches and results shown that global models are not working well with the localized areas. And it means some of the global models are not going well in localized areas even they are working pretty well in global models.
Battenberg [35] had done some studies on methods of regression based estimation in 2 parts. First in all of learning data and second clustered learning data. And the results show that models are working pretty fine in localized data.
In literature, estimation of software effort, partitioned models, are ones that make estimates locally [36.37] .

Researchers suggest models where projects are implemented using clustering algorithms divided into groups and for each cluster an equation based regression is calculated as estimated model ABE model is one of the . most popular models in the effort estimation which is widely used in hybrid models. The combination of ABE with genetic algorithms [17, 30, 31]. ABE with particle swarm optimization algorithm [38, 39] And Artificial (ANN) are some samples of Neural Network combinations. Though hybrid models have a fairly good accuracy based on ABE but the flexibility and versatility of these models are not good enough which can cover a wide range of software projects Figure (1) shows some of the . .models made to estimate the effort of software projects

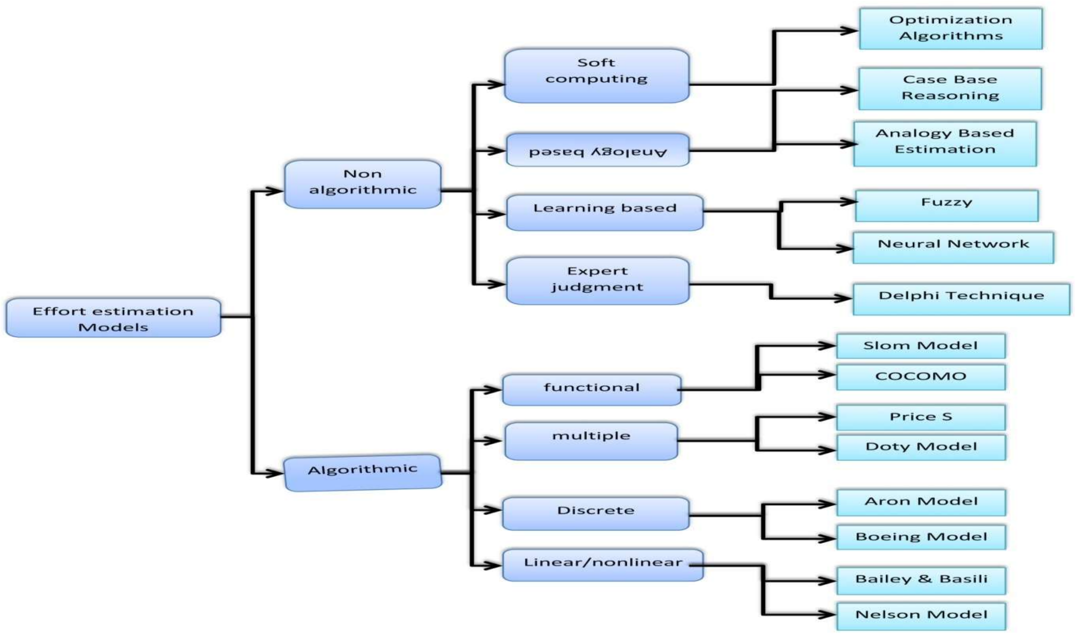


Figure (1): Classification of software effort estimation models

3. THE REQUIREMENTS OF THE PROPOSED METHOD

In this paper, an estimation model of software development effort is proposed by combining algorithmic and non-algorithmic methods. First, by using MLP neural network and genetic algorithm the feature selection operation is done, and in next phase using the MLR method and cross validation method (leave-one-out) weighing the selected features has been done. Finally, the coefficients are optimized using the ICA.

3.1 Neural network

Neural network is a computational device located in the computing intelligence branch. The neural network model is a computer system which is simulated according to the process of learning human brain and is widely used. Use of neural networks in effort estimation began in 1993 and has yielded good results. The benefits of neural networks in estimating costs are:

Eliminates the need to find a mathematical-costly good effort estimation relationship Similar to the function of the variables, in order to obtain the correct answer, There is no limitation in relation to the number of cost variables and the need to determine the estimation function in a concrete manner before training But its disadvantages can be highlighted by its many parameters. Neural network used in this paper is a two-perceptron network. The number of neurons in the hidden layer has been calculated with the aid of trial and error. In this network, the number of entries are equal to the number of properties of the data set.

3.2 Genetic Algorithms

Genetic Algorithm was introduced by John Holland in the early 1970s as a general tool for optimization. And then David Goldberg, a student of Mr. Holland, was instrumental in introducing a genetic algorithm. The genetic algorithm is a population-based algorithm that calls each one of the population members a chromosome. Each chromosome is one of the possible solutions to the problem solving atmosphere. At the start of the algorithm, a number of randomly generated responses are generated as initial population. These answers are evolved in each replication of the algorithm until they become the best possible solutions in the problem solving atmosphere. In each repetition of genetic algorithm, two operators of "intersection" and "mutation" are performed. The intersection operator is based on percentages with one population (usually 80%), and in the form of a mutation operator, a small percentage (usually 3%) of the population of each generation. The result of each of these operations is the production of a new member of the population. At the end of each repetition to select the number of members, the "Select" operator selects a certain number of answers for the next generation. The steps in this algorithm are in accordance with the flowchart of Fig. 2.

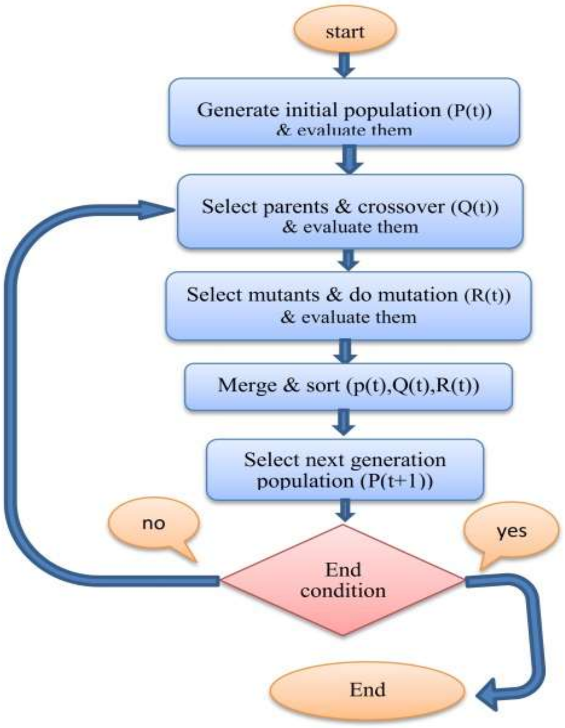

start
Generate initial population (P(t))
& evaluate them
Select parents & crossover (Q(t))
& evaluate them
Select mutants & do mutation (R(t))
& evaluate them
Merge & sort (p(t),Q(t),R(t))
Select next generation population (P(t+1))
no
yes
End condition
End

Figure (2): Genetic Algorithm Flowchart

3.3. Imperialist competition algorithm

The imperialist competition algorithm was introduced in 2007 by Dr. Atashpaz and Professor Lucas. This algorithm inspired by the same colonial phenomenon and colony in the real world which is based on the assumption that at first there are some entities in the name of the country And these countries are ranked according to a series of benchmarks.

This algorithm consists of two general phases:

- ✓ Competition within An Empire
- ✓ Rivalry between empires

In a rivalry within an empire, the colonies are trying to reach a degree of growth that could replace the colonial empire of that empire. This growth is based on factors such as attraction and revolution. In the rivalry between the

empires, each one is in the quest for the colonies of other empires. For this purpose, in each replication, a colony of the weakest empire is removed by the operator and it is given to one of the other empires. The steps of this algorithm are in accordance with the flowchart of Fig. 3.
In the next section, the proposed method is presented in detail.

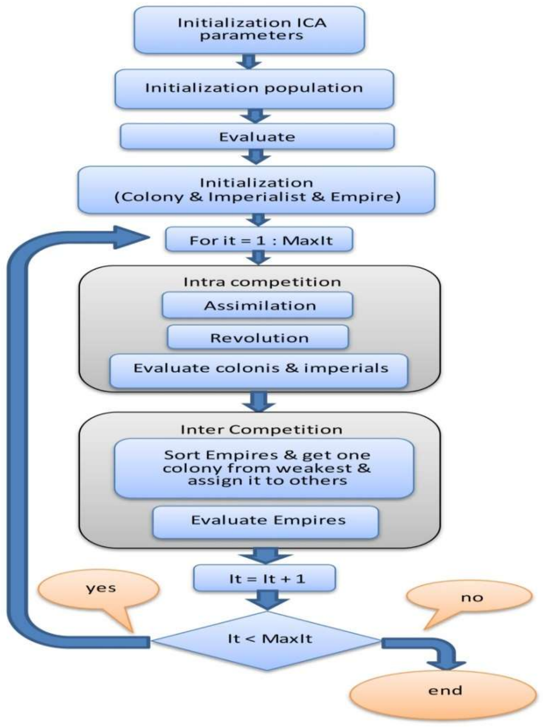

Initialization ICA parameters
Initialization population
Evaluate
Initialization (Colony & Imperialist & Empire)
For it = 1 : MaxIt
Intra competition
Assimilation
Revolution
Evaluate colonis & imperials
Inter Competition
Sort Empires & get one colony from weakest & assign it to others
Evaluate Empires
It = It + 1
yes
no
It < MaxIt
end

Figure (3): Imperialist Competitive Algorithm Flowchart

4. SUGGESTED MODEL

This section describes the details of the proposed model. According to the study of previous work, the main purpose of the models is to weight and combine models. The proposed model consists of three distinct sections:

I. *Feature Selection:*

In this section, for each data set, the most effective features in the project effort are first selected using the Genetic Algorithm and the MLP Neural Network.

II. *Calculate the coefficient of the effect of each attribute:*

In this step, using multiple linear regression (MLR), the coefficient of influence of each attribute is calculated.

III. *Optimization with ICA Algorithm:*

At this stage, using the colonial competition optimization algorithm, the coefficients obtained in the previous step are optimized.

IV. *Calculation of performance criteria:*

In the final step, using the MRE calculated on the test data in the intersection validation method (Section 3.1.2) Performance metrics MDMRE,MMRE and PRED (SECTION 5,1,2) [41,42,43] are Calculated and presented. Figure (4) shows the general approach of the proposed method.

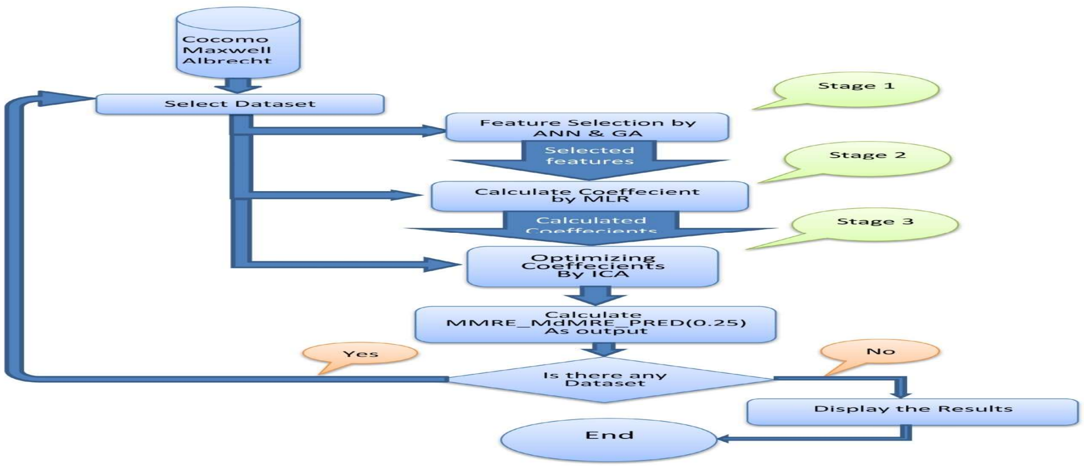


Figure (4): the general approach of the proposed method

4.1 First Phase: Feature Selection

In this phase, the selected dataset is given as an input to genetic algorithm. Cost function in genetic algorithm is named Feature Selection Cost (FSC). Chromosomes length is determined by amount of dataset's features. So, the first population is made randomly after determining chromosomes' length (equal to feature). Each chromosome contains a chain of '0' and '1', that means 'no choice' or 'choice' parallel features. Each member of population is evaluated by FSC function after making first population. After setting parameters of this neural network in the function for making swear about reliability, the function evaluate neural network with the function 'FITNN' in toolbox 5 times with the suitable software and mean score of 5 performances will return to genetic algorithm as cost response. The main loop of genetic algorithm starts after making and evaluating the first population. This loop is repeat for 100 iteration and contain following steps:

- ✓ *Crossover: for this reason doing crossover has been chosen by parents using roulette-wheel Selection function and one of the cross ways named single point, double point or uniform respectively with the chances of 0.2, 0.3, 0.5 are chosen and the children are made by the cross product.*
- ✓ *Response assessment: just after producing the answers made from cross product, this answers are assessed by FSC function.*
- ✓ *Doing mutation: in this level some responses are chosen and mutation will be done on them.*
- ✓ *Assessment of children produced by mutation: this action also is done by FSC function.*
- ✓ *Merging 3 groups of population (first population, population produced by crossover and population produced by mutation), collocating and choosing best ones as population of next generation.*

The output of this algorithm is the best answer as chosen features that will transfer to next phase. Fig (5) is showing the procedure of this phase.

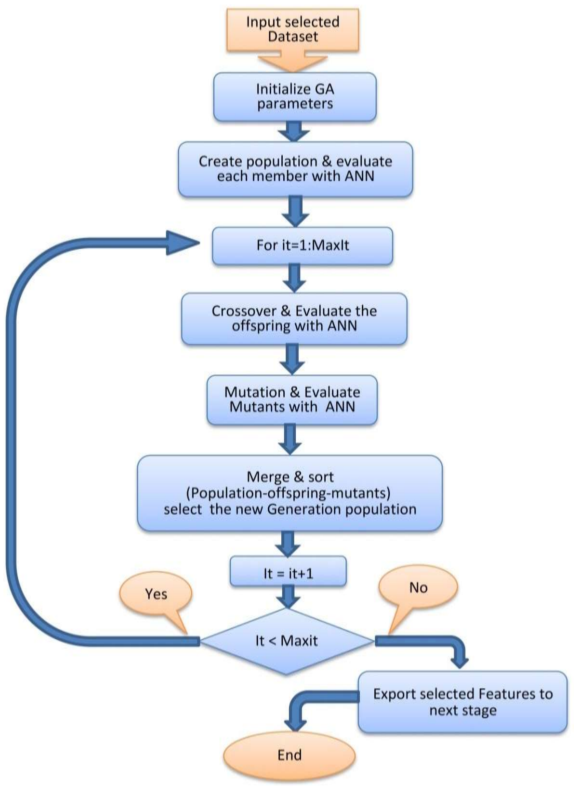

Input selected Dataset
Initialize GA parameters
Create population & evaluate each member with ANN
For it=1:MaxIt
Crossover & Evaluate the offspring with ANN
Mutation & Evaluate Mutants with ANN
Merge & sort (Population-offspring-mutants) select the new Generation population
It = it+1
Yes
No
It < Maxit
Export selected Features to next stage
End

Figure (5): Feature selection flowchart

4.2 Second phase: construct model with MLR

In this phase, firstly, in the given dataset, the selected features are extracted according to the output of the previous phase .Then normalization is performed according to (1) on each of the selected properties.

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

In this case, x_i' is the normalized value of the sample i of the characteristic x and x_{max} and x_{min} are respectively the highest and lowest values for this property, and x_i is the initial value of the sample i for the attribute x. This relationship maps the values in the interval [0, 1]. After normalizing, a whole column is added as an intercept to the data set as an independent attribute. Experimental observations show that adding this column to the dataset can improve the performance of the model. After preparing the dataset, using the MLR method and validation LEAVE-ONE-OUT technique Construction and model testing is done. Note that the validation technique LEAVE-ONE-OUT is the best and most accurate statistical method for verifying the responses is considered [46]. In this technique,

each time a data record is passed as test data, and the remainder is considered as training data. This action repeats the number of datasets records So that all data are used once as test data. Finally, by averaging the results of the test data in the whole process, the final answer is calculated [47].
So in this phase for each data set, the number of records from the MLR method and the LEAVE-ONE-OUT technique the coefficients obtained are stored in an array and eventually the same array is sent as the output of this phase to the next stage. Figure 6 shows the process of performing this phase.

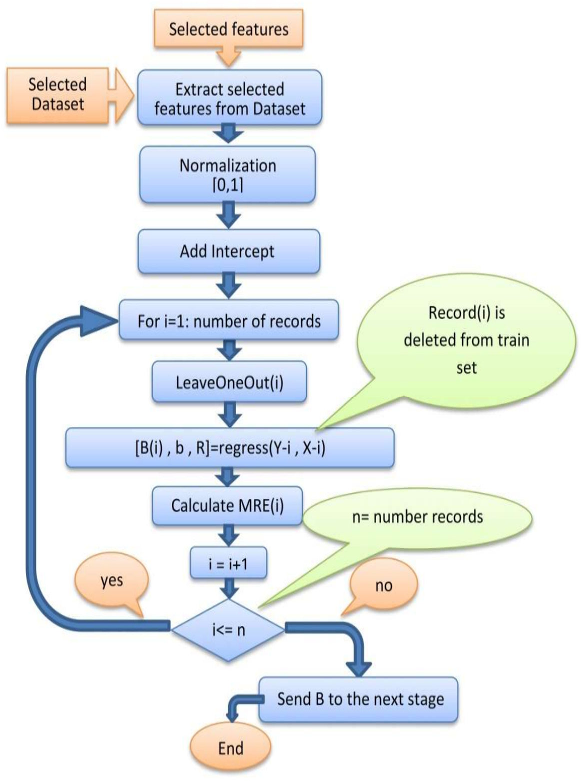


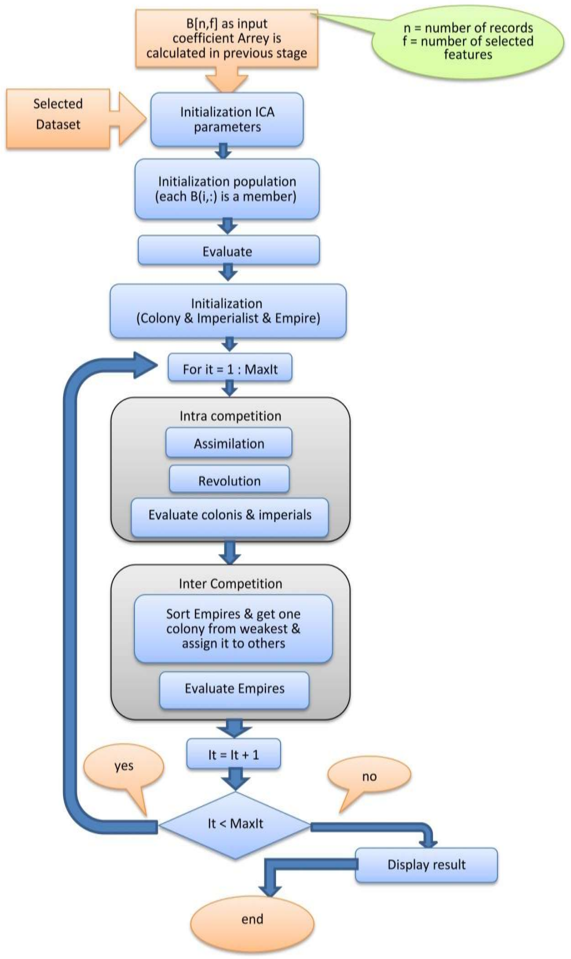
Figure (6): Construct Model with MLR

4.3 Third Phase: Optimization by ICA Algorithm

In the previous phase, the coefficients calculated by the MLR method in each step of the LEAVE-ONE-OUT technique it is stored in an array and sent as input to the ICA algorithm, in this phase, the ICA algorithm considers members of this array as the primary population and trying to optimize these answers. Because of the performance measures considered in measuring the proposed methods like MDMRE, MMRE and PRED (0.25), therefore, the objective function in the ICA algorithm is to optimize these criteria. Limit for MMRE and MDMRE is the smallest value and for PRED (0.25) is the highest value, so the objective function in this algorithm is to minimize the relationship (2).

$$Z = abs(PRED - (MMRE + MdMRE) \qquad (2)$$

The output of this phase is the weight of the best model. Figure 7 shows the steps involved in this phase.


B[n,f] as input coefficient Arrey is calculated in previous stage
n = number of records f = number of selected features
Selected Dataset
Initialization ICA parameters
Initialization population (each B(i,:) is a member)
Evaluate
Initialization (Colony & Imperialist & Empire)
For it = 1 : MaxIt
Intra competition
Assimilation
Revolution
Evaluate colonis & imperials
Inter Competition
Sort Empires & get one colony from weakest & assign it to others
Evaluate Empires
It = It + 1
yes
no
It < MaxIt
Display result
end

Figure(7): the steps involved in final phase

5. Experimental results

5.1 Experimental Design

In this section, we examine the performance of the proposed model on a set of different data. The reason for using multiple datasets is that each dataset has certain attributes so each of which can be a benchmark for our proposed model. In this chapter, three sets of data were first introduced, then, using this data set, we examine the results of the proposed model and draw up the corresponding diagrams. In the end, by comparing the proposed model with existing models, we will improve its performance.

5.1.1 Introducing the dataset

The use of past experiences in software projects is inevitable To this end, a number of data sets related to various software projects are being collected and for use by researchers. From the most important collection of data, reserves of software application estimation data as a framework for analyzing methods and estimation models. In recent decades, researchers have used various types of software data collections. We also use three datasets to evaluate our proposed model the name and general characteristics of this data set are presented in Table (1) .

Table (1): Data Set Attributes

Data set	Num projects	Num. features	Min. Effort	Max. Effort	Ave. Effort
Albrecht	24	8	9.2	2.105	8.22
Maxwell	62	26	583	63694	8223
COCOMO	63	17	9.5	11400	683

5.1.2 Performance metrics

Different quantities of various performance measures have been used in different environments. The goal of most of these criteria is to measure the accuracy of model estimation for example, the RE parameter shows the relative error of the difference between the predicted value of the model and the actual value. This parameter is calculated according to (3).

$$RE = |\, Estimated - Actual \,| \qquad (3)$$

Estimated is the predicted value by the model and Actual is the actual value in most cases, instead of relative error, the criterion of relative error size is used which is defined according to relation (4).

$$MRE_i = \frac{|\, Estimated_i - Actual_i \,|}{Actual_i} \qquad (4)$$

Most of the performance criteria are based on measuring the accuracy of estimates based on this criterion. In this article, the three performance criteria MdMRE, MMRE and PRED (0.25) are defined in relations (5, 6, and 7).

$$MMRE = mean(MRE) \qquad (5)$$

$$\mathrm{MdMRE} = \mathrm{median(MRE)} \qquad (6)$$

$$PRED(0.25) = \frac{A}{N} \qquad (7)$$

Where A is the number of observations whose MREs are less than %25 and N is the total number of observations.

5.1.3 Evaluation method

One of the evaluation methods is the cross validation method. By this way, the sets of data are divided by k into equal parts, and one of these k sections is used as test data and (K-1), the other part being used as training data. The smaller the size, the (K) more rendering will be, and the larger the (k) the larger the image, the less impact. Normally, the value of (K) can be equal to the number of datasets records. In this case, the method is the intersection validation method (Leave-one-out). Leave-One-Out method has been used to evaluate the performance of the proposed model. Each time, a record of data set is considered as the test data and the other data as training data. This action repeats the number of records So that each data is used once as test data. At each stage, the MRE value for the test data is computed and at the end of MMRE values and (0.25) PRED for the test data is calculated.

5.1.4 Initial settings

In this paper, Genetic Algorithm and MLP Neural Network for selecting features and multiple linear regression of MLR for weighting selected features. Finally, colonial competition algorithm is used to optimize weights. The initial settings and the values of parameters of genetic algorithms and colonial competition, which are obtained with error and try method, are in accordance with Table (2) and (3).

Table (2): The initial settings and the values of parameters of genetic algorithms

Name	Value	Discription
MaxIt	100	Maximum of iteration
Npop	30	Num of initial population
Pc	8.0	Crossover Percentage
Pm	3.0	Mutation Percentage
Mu	1.0	Mutation Rate

Table (3): The initial settings and the values of parameters of ICA

Name	value	Discription
MaxIt	1000	Maximum of iteration
Npop	Num of dataset records	Num of initial population
Nemp	10	Num. of Empire
Alpha	3	Selection preture
Beta	1	Assimilation coefficient
Prevolution	0.2	Revolution Probability
Mu	0.3	Revolution Rate
Zeta	0.5	Effect coefficient of Colony's value on Empire's value

The neural network used is a two-layer MLP, in which the number of neurons in the first layer is determined by trial and error according to the data set. The first layer entries are the values of the properties selected in each data set. Therefore, the number of neural network inputs varies in each set of data to train the neural network, the train LM . . levemberg-marquardt propagation function has been used(Train, test and validation data are divided by 70, 13 and 13 percent, respectively. Figure 8 is a schematic of this kind .of neural network

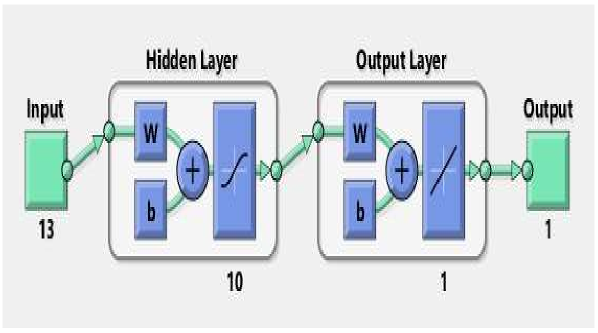


Figure (8): a schematic of two layer neural network

5.2 Experimental results

In this paper, it has been tried to compare the results with the results of other common models of estimation of effort in order to validate the proposed model.
These models are include the RTM model [50], MLFE [49], LSE [48], OABE [51], (LOG + OLS) [54], CBR [53], LMS [52], RIR [55], and (OLS + BC) [56], and GA [57].

5.2.1 Results on the COCOMO Dataset

The COCOMO dataset is commonly used in the process of evaluating software appraisal model estimation models. This dataset contains data on software projects, including 62 projects with 17 features Table (4) shows the estimate of the estimated effort by each of the models reported.

Table (4): the estimated effort by each of the models on COCOMO

method	Mmre	MdMre	Pred(0.25)
OABE	0.5	0.48	0.2
LSE	0.66	0.38	0.32
MLFE	1.48	0.72	0.14
RTM	0.54	0.47	0.25
GA	1.59	0.81	0.14
Log+OLS	***0.32***	***0.28***	***0.49***
LMS	0.58	0.48	0.27
BC+OLS	0.67	0.63	0.19
RIR	0.66	0.65	0.21
CBR k=1	0.67	0.65	0.19
Proposed	0.58	0.62	00.3

As can be seen, the lowest value for the MMRE parameter is related to OABE model with a value of 0.30 and the highest value for the GA model with a 1.39 value. The proposed model for this parameter is 0.58, which is in the second place. For MdMRE parameter, the lowest value is the LOG + OLS model with the value of 0.28 and the highest value for the GA model with a value of 0.81.The proposed model for this parameter is 0.62, which is ranked sixth. For the parameter (0.25), the PRED is also better than the LOG + OLS model with the value of 0.49 and the worst case for the GA and MLFE models with a value of 0.14.The proposed model for this parameter measures 0.20, which is ranked fourth. The results show that the proposed model on the COCOMO dataset does not work well and yields relatively weak results The diagram (9) illustrates this fact.

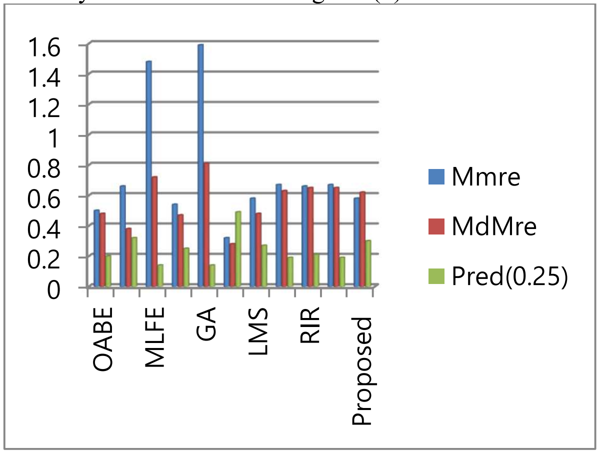


Figure (9): MMRE, MdMRE, and Pred(0.25) results on COCOMO by each models

5.2.2 Maxwell Dataset

One of the most famous and most widely used datasets for estimating effort is Maxwell dataset. This data set includes 63 projects with 36 features. Many researchers have used this dataset to evaluate the performance of their proposed model Table 5 shows the amount of effort estimated by different models on this dataset.

Table (5): the estimated effort by each of the models on Maxwell

Method	Mmre	MdMre	Pred(0.25)
OABE	0.42	0.44	0.34
LSE	0.71	0.48	0.27
MLFE	0.71	0.48	0.27
RTM	0.46	0.41	0.32
GA	1.17	0.6	0.18
RIR	0.41	0.32	0.35
Log+OLS	0.43	0.37	0.32
CBR K=2	0.51	0.39	0.39
CBR K=3	0.47	0.38	0.36
CBR K=5	0.44	0.36	0.32
Proposed	**0.35**	***0.22***	***0.58***

According to Table (5), the best value for the MMRE parameter is 0.35 and is related to the proposed model, and the worst value for this parameter is equal 0.71 which is related to the MLFE and LSE models.
For the MdMER parameter, the best value is 0.22, which is commonly related to the CBR-FA model and the proposed model. The worst value is 0.60, which is related to the GA model. Finally, the best value for the parameter is (0.25) (PRED is equal to 0.58 Which is related to the proposed model and the worst value for this parameter is related to the GA model and to 0.18. The results show that the proposed model has worked very well on this data set, and the accuracy achieved is desirable. Diagram (10) depicts these results.

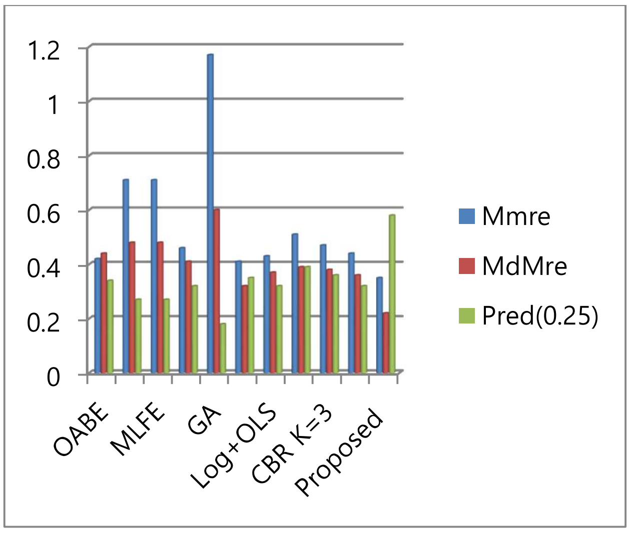


Figure (10): MMRE, MdMRE, and Pred(0.25) results on Maxwell by each models

5.2.3 Albrecht Dataset

This dataset contains information on 24 projects and has 8 features that are used in estimation process. Although the number of projects in this data set is low but has been used by many researchers to estimate the effort to evaluate the models. We also use this dataset to evaluate our proposed model. The results of various models on this dataset are presented for the three parameters MdMRE, MMRE and (0.25) PRED shown in Table (6).

Table (6): the estimated effort by each of the models on Albrecht

method	Mmre	MdMre	Pred(0.25)
OABE	0/4	0/37	0/45
LSE	0/63	0/3	0/37
MLFE	0/65	0/3	0/37
RTM	0/61	0/4	0/33
GA	0/45	0/38	0/33
CBR+FA	0/38	0/29	0/48
CBR+PSO	0/52	0/24	0/57
Proposed	***0/34***	***0/18***	***0/58***

As can be seen, the best value for the MMRE parameter for the proposed model is 0.34 and the worst case is 0.65 and is related to the MLFE model the best value for the MdMRE parameter is related to the proposed model and equals 0.18 and the worst value is 0.40, which is related to RTM model. Finally, the best value for the parameter (0.25) is PRED equals 0.58, which is related to the proposed model. The worst value is 0.33, which is shared by RTM and GA models. The results show that the proposed model also has a good performance on this data set. Figure 11 shows these results.

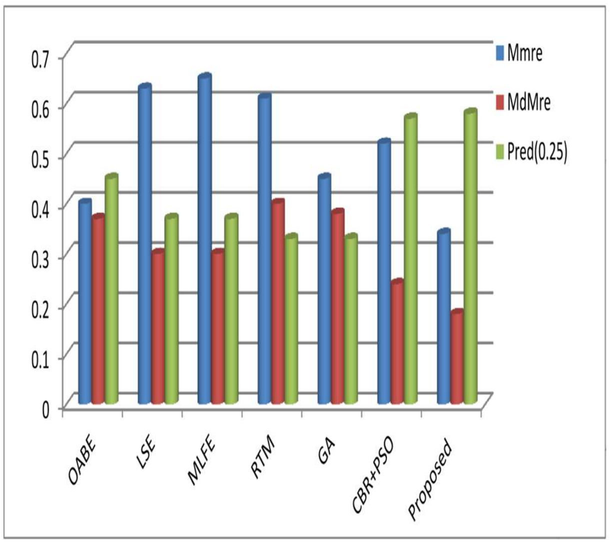


Figure (11): MMRE, MdMRE, and Pred(0.25) results on Albrecht by each models

6. RESULTS ANALYZES

The results show that the proposed model failed to perform well on the COCOMO dataset for the criterion (0.25) (PRED) The reason for this is the limited variety of projects in this data set In the Maxwell dataset, according to the proposed results, the proposed model in each of the parameters of the MdMRE, MMRE and (0.25) PRED ranked first. Therefore, it provides an acceptable performance on this data set. Albrecht dataset is a relatively standard dataset, which can lead to an improved model accuracy. The results of the proposed model have improved dramatically on this data set. The most important one can be the proper weighting and the exact choice of the solution function despite the low number of projects in this dataset, the proposed model has been able to deliver satisfactory results. And this certainly depends on the quality of the projects in this dataset. Table 7 shows the average improvement performance of the suggested model in each of the data sets for each of the MdMRE, MMRE and (0.25) PRED models.

Table (7): results analyzes

	Average improvement		
Dataset	PRED(0.25)	MdMRE	MMRE
COCOMO	↑ % 25	-% 12 ↓	↑ % 23
MaxWell	↑ % 87	↑% 47	↑ % 38
Albrecht	↑ % 40	↑%45	↑ % 35

7. CONCLUSION

The proper estimation of the software development effort is crucial for software projects. The tendency of researchers in recent decades and many researches in this area confirms this importance Therefore, it is important to provide a model for estimating project effort. Although there are many models for this purpose But according to the results presented by these models, the type of project has a significant effect on the accuracy of the estimates The proposed method is set in three functional phases.in the first phase, combining the genetic algorithm and the perceptron neural network, a feature selection operation was performed. Then in the second phase using multiple linear regression methods the weighting operation of the selected features is performed in the previous phase. Finally, in the last phase, with the help of the colonial competition algorithm, the weights obtained by multiple linear regressions are optimized. As the results of the proposed model show, this model offers good results on the Maxwell and Albrecht data sets In the case of the COCOMO Collection, the results are relatively weaker The fault of the proposed model is the same sensitivity to the set. To evaluate the proposed model, the MdMRE, MMRE and (0.25) PRED have been used. According to Table 7, the results of the proposed method are desirable.

It seems that the construction of a hybrid model that can withstand this sensitivity is more stable and its results are not dependent on the data set. In the future, it's a good solution In addition, a "multi-model" solution can be used to overcome this problem in such a way that the original model is a combination of models, so that for each dataset its appropriate model is chosen.

References

[1] Jorgensen, M., Shepperd, M.: ‘A systematic review of software development cost estimation studies‘, IEEE Trans. Softw. Eng., 2007, 33, (1), pp. 33–53.

[2] Costagliola, G., Ferrucci, F., Tortora, G., Vitiello,G.: ‘Class point: an approach for the size estimation of object-oriented systems‘, IEEE Trans. Softw. Eng., 2005, 31,(1), pp. 52–74.

[3] Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: ‘Classification and regression trees’ (Wadsworth, Pacific Grove, CA, 1984.

[4] Putnam, L.H.: ‘A general empirical solution to the macro software sizing and estimating problem‘, IEEE Trans. Softw. Eng., 1978, 4, (4), pp. 345–361.

[5] Boehm, B.W.: ‘Software engineering economics’ (Prentice-Hall, Englewood Cliffs, NJ, 1981.

[6] Jensen, R.: ‘An improved macrolevel software development resource estimation model’. Proc. of Fifth Conf. of Int. S Parametric Analysts, 1983, pp. 88–92.

[7] Azzeh, M., Neagu, D., Cowling, P.I.: ‘Analogy-based software effort estimation using fuzzy numbers‘, J. Syst. Softw., 2011, 84, (2), pp. 270–284

[8] El-Sebakhy, E.A.: ‘Functional networks as a novel data mining paradigm in forecasting software development efforts‘, Expert Syst. Appl., 2011, 38, (3), pp. 2187–2194.

[9] Bonetti, A., Bortot, S., Fedrizzi, M., Marques Pereira, R.A., Molinari, A.:‘Modelling group processes and effort estimation in project management using the Choquet integral: an MCDM approach’, Expert Syst. Appl., 2012, 39, (18), pp. 13 366–13p 375

[10] Shepperd, M., Schofield, C.: ‘Estimating software project effort using analogies‘, IEEE Trans. Softw. Eng., 1997, 23, (11), pp. 736–743

[11] Grimstad, S., Jørgensen, M.: ‘Preliminary study of sequence effects in judgment-based software development work-effort estimation’, IET Softw., 2009. 3, (5), pp.435–441

[12] Gupta, S., Sikka, G., Verma, H.: ‘Recent methods for software effort estimation by analogy’, SIGSOFT Softw. Eng. Notes, 2011, 36, (4), pp. 1–5.

[13] Kocaguneli, E., Menzies, T., Bener, A., Keung, J.W.: ‘Exploiting the essential assumptions of analogy-based effort estimation’, IEEE Trans. Softw. Eng., 2012, 38, (2), pp. 425–438

[14] Khatibi Bardsiri, V., Norhayati Abang Jawawi, D., Zaiton Mohd Hashim, S., Khatibi, E.: ‘Increasing the accuracy of software development effort estimation using projects clustering’, IET Softw., 2012, 6, (6), pp. 461–473.

[15] Huang, Y.S., Chiang, C.C., Shieh, J.W., Grimson, E.: ‘Prototype optimization for nearest-neighbor classification’, Pattern Recognit., 2002, 35, pp. 1237–1245

[16] Khatibi Bardsiri, V.K., Jawawi, D.N.A., Bardsiri, A.K., Khatibi, E.: ‘LMES: a localized multi-estimator model to estimate software development effort’, Eng. Appl. Artif. Intell., 2013, 26, (10), pp. 2624–2640

[17] Li, Y.F., Xie, M., Goh, T.N.: ‘A study of project selection and feature weighting for analogy based

software cost estimation', J. Syst. Softw., 2009, 82, (2), pp. 241–252

[18] Bardsiri, A. K., and S. M. Hashemi. 2013. "Electronic Services, the Only Way to Realize the Global Village." International Journal of Mechatronics, Electrical and Computer Technology 3 (6): 1039–1041.

[19] Azzeh, M. 2012. "A Replicated Assessment and Comparison of Adaptation Techniques for Analogy-based Effort Estimation." Empirical Software Engineering 17 (1–2): 90–127.

[20] Čeke, D., and B. Milašinović. 2015. "Early Effort Estimation in Web Application Development." Journal of Systems and Software 103: 219–237. doi:10.1016/j.jss.2015.02.006.

[21] Idri, A., F. A. Amazal, and A. Abran. 2015. "Analogy-based Software Development Effort Estimation: A Systematic Mapping and Review." Information and Software Technology 58: 206–230. doi:10.1016/j. infsof.2014.07.013.

[22] Khatibi, E., and V. K. Bardsiri. 2015. "Model to Estimate the Software Development Effort Based on In-depth Analysis of Project Attributes." IET Software 9 (4): 109–118. doi:10.1049/iet-sen.2014.0169.

[23] Menzies, T., A. Butcher, A. Marcus, T. Zimmermann, and D. Cok. 2011. "Local vs. Global Models for Effort Estimation and Defect Prediction." In Proceedings of 26th IEEE/ACM International Conference on Automated Software Engineering, Lawrence, KS, USA, 6–10 November 2011: 343–351. Washington, DC: IEEE.

[24] Hsu, C.-J., and C.-Y. Huang. 2011. "Comparison of Weighted Grey Relational Analysis for Software Effort Estimation." Software Quality Journal 19 (1): 165–200.

[25] Wu, D., J. Li, and Y. Liang. 2013. "Linear Combination of Multiple Case-based Reasoning with Optimized Weight for Software Effort Estimation." The Journal of Supercomputing 64 (3): 898–918. doi:10.1007/s11227-010-0525-9.

[26] L. A. Zadeh,"Soft computing and fuzzy logic," IEEE Softw., vol. 11, pp. 48-56,Nov. 1994.

[27] A. Sheta,"Estimation of the COCOMO model parameters using genetic algorithms for nasa software projects," Journal of Computer Science, vol. 2,pp. 118-123, 2006.

[28] J. J. Dolado and L. F. andez, "Genetic programming,neural network and linear regression in software project estimation," in Proceedings of the fNSPIRE Ifl, Process Improvement through training and education, pp. 157-171,British Company Society,1998.

[29] A. Sheta, D. Rine, and A. Ayesh, "Development of software effort and schedule estimation models using soft computing techniques," in Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE CEC 2008) within the 2008 IEEE World Congress on Computational Intelligence (WCCI 2008), Hong Kong, pp. 1283-1289,June 2008.

[30] Chiu, N.-H., Huang, S.-J.: 'The adjusted analogy-based software effort estimation based on similarity distances', J. Syst. Softw., 2007, 80, (4), pp. 628–640

[31] Huang, S.J., Chiu, N.H.: 'Optimization of analogy weights by genetic algorithm for software effort estimation', Inf. Softw. Technol., 2006, 48, pp. 1034–1045

[32] Hsu, C.-J., Huang, C.-Y.: 'Comparison of weighted grey relational analysis for software effort estimation', Softw. Qual. J., 2011, 19, (1), pp. 165–200

[33] Song, Q., Shepperd, M.: 'Predicting software project effort: a grey relational analysis based method', Expert Syst. Appl., 2011, 38, (6), pp. 7302–7316

[34] Menzies, T., Butcher, A., Marcus, A., Zimmermann, T., Cok, D.: 'Local vs. globalmodels for effort estimation and defect prediction'. 26th IEEE/ACM Int. Conf. on Automated Software Engineering, Lawrence, KS, 2011.

[35] Bettenburg, N., Nagappan, M., Hassan, A.E.: 'Think locally, act globally: improving defect and effort prediction models'. Ninth Working Conf. on Mining Software Repositories, Zurich, Switzerland, 2012.

[36] Cuadrado-Gallego, J.J., Sicilia, M.A., Garre, M., Rodrguez, D.: 'An empirical study of process-related attributes in segmented software cost-estimation relationships', J. Syst. Softw., 2006, 79, (3), pp. 353–361.

[37] Rodrguez, D., Cuadrado, J.J., Sicilia, M.A., Ruiz, R.: 'Segmentation of software engineering datasets using the M5 algorithm'. Sixth Int. Conf. on Computational Science – Volume Part IV, Reading, UK, 2006.

[38] Khatibi Bardsiri, V., Jawawi, D., Hashim, S., Khatibi, E.: 'A PSO-based model to increase the accuracy of software development effort estimation', Softw. Qual. J., 2012, 21, (3), pp. 501–526

[39] Wu, D., Li, J., Liang, Y.: 'Linear combination of multiple case-based reasoning with optimized weight for software effort estimation', J. Supercomputing., 2010, 64, (3), pp. 898–918

[40] Li, Y.F., Xie, M., Goh, T.N.: 'A study of the non-linear adjustment for analogy based software cost estimation', Empir. Softw. Eng., 2009, 14, pp. 603–643.

[41].Atashpaz-Gargari, E., Lucas, C., 2007. Imperialist competitive algorithm: an algorithm for optimization

inspired by imperialistic competition. In: IEEE Congress on Evolutionary Computation, Singapore, pp. 4661e4667.

[42].Atashpaz-Gargari, E., Hashemzadeh, F., Rajabioun, R., Lucas, C., 2008. Colonial competitive algorithm: a novel approach for PID controller design in MIMO distillation column process. International Journal of Intelligent Computing and Cybernetics 1 (3), 337e355

[43]. Bardsiri, A. K., Hashemi, S. M. & Razzazi, M. (2016). GVSEE: A Global Village Service Effort Estimator to Estimate Software Services Development Effort. *Applied Artificial Intelligence, An International Journal, 30*(5), 396–428.

[44]. Bardsiri, V. K., Jawawi, D. N. A., Bardsiri, A. K., & Khatibi, E. (2013). LMES: A localized multi-estimator model to estimate software development effort. *Engineering Applications of Artificial Intelligence, 26*(10), 2624-2640

[45].Bardsiri, A. K., Hashemi, S. M. & Razzazi, M. (2015). Statistical Analysisof the Most Popular Software Service Effort Estimation Datasets. Journal of Telecommunication, Electronic andComputer Engineering, 7(1),87-96.

[46] Keung J, Kocaguneli E, Menzies T (2013) Finding conclusion stability for selecting the best effort predictorin software effort estimation. Automated Software Eng 20(4):543–567.

[47] Zhang, W., Y. Yang, and Q. Wang. 2015. Using Bayesian regression and em algorithm with missing handling for software effort prediction. Information and Software Technology 58:58–70. doi:10.1016/j.infsof.2014.10.005.

[48] Elsheikh, y., Alseid, M., Azzeh, M., 2014, An optimized analogy-based project effort estimation, *International Journal of Advanced Computer Science and Applications*, 5(4): 6-11.

[49] Walkerden ,F., Jeffery, D.R., 1999, An empirical study of analogy-based software effort Estimation, *Journal of Empircal Software Engineering*, 4(2): 135–158.

[50] Mendes, E., Mosley, N., Counsell, S., 2003, A replicated assessment of the use of adaptation rules to improve Web cost estimation, *International Symposium on Empirical Software Engineering*, 100-109.

[51] Jorgensen, M., Indahl, U., Sjoberg, D., 2003, Software effort estimation by analogy and regression toward the mean, *Journal of System and Software*, 68(3): 253-262.

[52]Hoerl, A.E., 1962, Application of ridge analysis to regression problems, *Chemical Eng.Progress*, 58(5): 54-59.

[53] Rousseeuw, P., 1984, Least median of squares regression, *Journal of the American Chemical Society(ACS)* , 79(388): 871-880.

[54] Finnie, G., Wittig, G., Desharnais, J.M., 1997, A comparison of software effort estimation techniques: using function points with neural networks, case-based reasoning and regression models, *Journal of Systems and Software*, 39(3): 281-289.

[56] Hastie, T., Tibshirani, R., Friedman, J., 2005, The elements of statistical learning, data mining, Inference, and Prediction*, Springer*, 27(2): 83-85.

[57] Box, G., Cox, D., 1964, An Analysis of Transformations, J*ournal of Royal Statistical Soc*, 26(2): 211-252.

[58] Chiu, N.H., Huang, S.J., 2007, The adjusted analogy-based software effort estimation based on similarity distances, *Journal of Systems and Software*, 80(4): 628-640.