Journal of Scientific Research and Development 1 (1): 50-57, 2014
Available online at www.jsrad.org
ISSN 1115-7569

Approach of software cost estimation with hybrid of imperialist competitive and artificial neural network algorithms

Farhad Soleimanian Gharehchopogh [1,*], Awat Maroufi [2]

[1]Department of Computer Engineering, Hacettepe University, Beytepe, Ankara, Turkey
[2]Department of Computer Engineering, Urmia Branch, Islamic Azad University, Urmia, Iran

Abstract: Software is reckoned to be one of the most expensive the application tools in the computer system and accurate estimates the cost of software projects is always the most important concern of project managers. Accuracy in Software Cost Estimation (SCE) enables project manager to make various decisions while development cycle of software. Estimation accuracy of software cost helps project managers, analysts, designers, programmers and other member of development team to know how much effort and time is needed for software projects production. Different models have been used for SCE according to Line of Code (LOC) and Function Point (FP). COCOMO which is an algorithmic model for SCE, basically approximate LOC and the cost of developing software projects based on complexity of projects. In this paper, we have used a hybrid of Artificial Neural Networks (ANNs) and Imperialist Competitive Algorithm (ICA) for SCE in NASA's software project and we have used ICA to learn and to optimization. ICA with mathematically modeling the phenomenon of imperialism has been applied as an effective algorithm in optimization. Result show that the hybrid model has reduced Magnitude of Relative Error (MRE) and Mean MRE (MMRE) errors remarkably in comparison to COCOMO and ANN.

Key words: *Software Cost Estimation; COCOMO; Artificial Neural Networks; Imperialist Competitive Algorithm*

1. Introduction

The final outcome in design and implementation of software projects illustrated that majority of projects have been failed due to improper cost estimation and consequently inaccurate schematization and scheduling by manager (Gharehchopogh, 2011). Because software projects are complex and abstract, cost estimation and scheduling is too difficult. In software project management, cost estimation must not be undervalued; otherwise it may lead to rather bad get, no development, low quality and no completion (Ricardo de A.A et.al., 2012). Precise SCE helps to classify and to precede developing projects according to the general scheme of business. It also helps declaring how resources should be specified to the project and how much should be applied. A successful software project must be completed based on definite and predetermined cost and time. SCE includes determining one or more of the estimation follows (Boehm B.W., 2000, and 1991): Effort (Person/Month (P/M)), Project duration, and Cost. Most of the cost estimation models in the field calculate effort software projects have studied time and cost the number of persons per month (Cuadrado-Gallego et al., 2006).

Though effort and cost are so close and dependent, they are not essentially accounted according whit an easy correlated function. Effort is usually accounted according to P/M by programmers, analysts and project manager. Effort estimation can be changed to dollar cost through accounting average payment of per time until, then the result must be multiple into the estimated effort. Below are three major factors which experts must be concerned with during the estimated cost of software projects:

- Which software utilizes the cost estimation model?
- Which criterion should be applied for calculate size of the software, LOC or FP?
- What is a fine estimation?

Software development is one of the most sensitive factors in investment for software companies and software engineers are often worry about accurate estimations and quality of projects (AlYahya et al., 2010). Generally, there are many ways in order to assess cost of software projects. However, they are divided two major groups: algorithm and non-algorithmic. Application of both groups is necessary for accurate estimates. They all worked better if characteristic of software projects are well-defined. Algorithmic models operate a specific algorithm. At the beginning, these algorithms usually need information and dates about former projects and they result through mathematics. Some of the mathematical calculations deal with SCE through using simple formulas and specific data, other calculations is based on regression models and differential equations in order improve the accuracy

* Corresponding Author.

of algorithmic models, we need to adjust and calibrate the model to the local conditions. However, unlike the algorithm methods, cost estimation in software projects using non-algorithmic methods is based on comparison and inference. To use these methods, some information about previous objects that is similar to new projects is required. Usually the estimation process is according to an analysis of previous data. Models based on Artificial Intelligence (AI) and meta-heuristic are example of non-algorithmic estimation models (Khalifelu and Gharehchopogh, 2012).

In recent decades researchers have been able to archive great success in SCE by using AI and meta-heuristic techniques. A combination of meta-heuristic algorithm with other algorithms is used to achieve the optimum solution or a near optimal solution in solving optimization problems. One of the many applications in AI models to estimate the cost of software projects are ANNs (Nassif et al., 2013; Gharehchopogh, 2011; Gharehchopogh et al., 2012, 2013). One of the most common types of ANNs is Multi-Layer Perceptron (MLP) that is successfully used in a wide range of applications such as SCE (Alexander S. et.al, 2001, German S. et.al., 1992). The two fundamental problems we are face when working with MLP: selecting the appropriate architecture for MLP and selecting an appropriate training algorithm. Good architecture means choosing the optimal number of layers, number of neurons in each layer and the type of activation function for each neuron. Optimal architecture of ANNs is based on data sets and their characteristics. The most commonly used training algorithm for these networks is the back-propagation algorithm. In the back-propagation algorithm, new output value is calculated at each stage and it is compared with the actual value. Based on the calculated error, network weights are modified in such a way that at the end of each interaction, the value of the resulting error is less than the amount obtained in the previous interaction. Basis of this minimization is moving on gradient vector square error function of the network.

SCE using AI algorithms has not been determined with complete certainty until now. According to the studies that we've done, we have reached the conclusion that AI techniques perform better than algorithmic techniques.

ANNs have shown good accuracy (Gharehchopogh, 2011). In order to show the efficiency of ANNs, projects form 60 projects in NASA dataset software have been tested and trained through using ANNs. It is compared with COCOMO and result show that the error rate of COCOMO is more than the error rate of ANNs. In more than 90% of the time ANNs had better estimation than COCOMO it can be concluded that AI methods based on as a supplement are good alternative to algorithmic techniques.

According to (Maleki, 2014) a hybrid of Genetic Algorithm (GA) and Ant Colony Optimization (ACO) is used for SCE. In hybrid model the effective factors are tested by GA and they are trained by ACO during estimation. They had better comparison with COCOMO. Review and evaluation have been done on 60 software projects from NASA dataset. Test result show that the hybrid of GA and ACO compared to COCOMO has better performance in estimating the cost software projects and has less MMRE.

Data mining techniques are proper models for SCE. One of the most critical issues in software development is accurate estimate of costs. SCE has been simulating through using techniques such as Liner Regression (LR), ANNs, Support Vector Regression (SVR) and K-Nearest-Neighbors (KNN). Using the model LR the dependency of effective properties can be determined in SCE. The model LR finds the relationship between dependent and independent factors among data. ANNs try to have more accurate SCE by training and testing data. SVR is used to optimum effective factors in SCE. KNN is a technique in data mining that is used to closing data that have already been classified and their features have been specified. KNN is applicable to determine the weight of effective factors. Their results show that model SVR has less MRE in comparison with other models.

Hybrid of MLP and COCOMO II are for increase the accuracy of SCE used (Kumar K.V., et.al. 2008). Evaluation has been done on two dataset for 63 projects and other 100 projects. The major purpose of this hybrid is to have better training of this combination is to have better training of input data by MLP. In hybrid model there are two factors that are called Effort Multiplier (EM) and scale factor. They are trained by the middle layer and they are evaluated and tested by model COCOMO II. The input layer of MLP includes 5 neurons scale factor, 17 neurons EM and 2 neurons bias. The sigmoid function is used for the activation function of middle layer and back-propagation is used for training. Experimental result has shown the hybrid model has lesser error in comparison with COCOMO II. There for MMRE and PRED criterions have lesser error value in MLP-COCOMO II in comparison with COCOMO II.

C-means algorithm is used for SCE in order to train Radial Basis Function Network (RBFN) network (Gharehchopogh et al., 2014, 2012; Idri et al., 2006). In cases where the training data size is large, usually the clustering method is used to reduce the data. C-means algorithm is used for optimal selecting the RBFN function centers in hidden layer. RBFN networks have been evaluated on the data set of COCOMO81 software projects. Results show that PRED and MMRE have shown that the error is reduced.

Nowadays, despite numerous methods of estimation, accurate estimate of the software cost is still not easy and many researchers seek to use and combine different methods to archive more accurate results. The COCOMO II and MLP networks are combined for achieving more accurate results (Attarzadeh et al., 2012). In order to take advantage of the unique benefits of the models of the models COCOMO and MLP, the hybrid method is used.

Dataset of NASA93 and COCOMO I have been evaluated. The result show the hybrid model which is called ANN-COCOMO II has improved the estimation accuracy to 8.36 percent in comparison with COCOMO II. The error value of PRED (25) on NASA93 dataset is 45% in COCOMO II and 52% in ANN-COCOMO II. Also, the error value of PRED (25) is equal to 37.5% on COCOMO I dataset for the model COCOMO II and it is equal to 45.5% for the mode ANN-COCOMO II. The error value of MMRE is equal to 0.5025 in COCOMO II and 0.4579 in ANN-COCOMO II.

The structure of the paper is as follows: in Section 2, we display related works and in Section 3, we will explain the proposed model; in Section 4, will be shown evaluation and result of the proposed model with others models and finally in Section 5, conclusion and future works is presented.

2. Related works

A hybrid of Firefly Algorithm (FA) and GA models is proposed for SCE (Maleki, I. et.al. 2014). The evaluation has been conducted on NASA93 Dataset. In the proposed model, using elitism operation GA attempts to find the best answer for effort factors, evaluate it in fitness function and present a solution with the lowest value of error as the final answer. The results show that the value of MMRE is 58.80 in the COCOMO model and respectively 38.31 and 30.34 in GA and FA models; it equals 22.53 in the hybrid model. The accuracy of PRED (25) in GA and FA models is respectively 77.41 and 80.64, and 88.17 in the hybrid model. Comparisons show that the hybrid model has increased the efficacy of estimation accuracy to about 2.88% compared with COCOMO model. The (LR) Liner Regression, Artificial Neural Network (ANN), SVR and (KNN) K Nearest Neighbors techniques have been utilized for SCE (Khalifelu Z.A., Gharehchopogh F.S, 2012).

By using LR model, we can determine the dependence of the effective characteristics on SCE. LR model finds the relationship between dependent and independent factors among the data. Training and testing data, ANN tries to reduce the value of MRE error. SVR model is proposed for optimizing the effective factors on SCE. KNN is a data-analysis technique which is used for classifying data in a set of data which have already been classified and whose properties have been determined. KNN is used to determine the weight of effective traits on SCE. Prediction accuracy on the training data is respectively 74%, 87%, 95% and 68% in LR, ANN, SVR and KNN models. Also prediction accuracy on the tested data is respectively 60%, 95%, 80% and 60% in LR, ANN, SVR and KNN models. The value of MAE is respectively 39.17%, 12.07%, 36.32% and 1.12% in LR, ANN, SVR and KNN models.

In addition, the value of RRSE error is respectively 11.6%, 2.37%, 8.20% and 0.77 in LR, ANN, SVR and KNN models. Their results show that KNN model has lower value of MAE in contrast to other models. NASA60 Dataset, which constitutes 15 effort indices, is tested and trained using ANN (Gharehchopogh, 2011). The results of 11 software projects on NASA60 Dataset show that the value of MRE in ANN is less than that of COCOMO model. The results show that in 90% of occasions, Ann has had better estimation than COCOMO model.

PSO-FCM and PSO-LA hybrid models have been proposed for the SCE (Gharehchopogh et al, 2014). The evaluation has been conducted on NASA60 Dataset. In PSO-FCM hybrid model, the minimum of inter-cluster distances and total distance within clusters, besides the number of clusters are used as fitness and algorithm-enhancement parameters of Particle Swarm Optimization (PSO). Using Fuzzy C-Means (FCM) makes the particles gather in the best cluster and causes the fitness function to have many local optimum points.

In order to improve the performance of PSO, Learning Automata (LA) is used to regulate the behaviour of particles. In PSO-LA hybrid model, all particles simultaneously search for a place in searching space. The results show that PSO-FCM hybrid model has lower MRE ratio than PSO-LA LA model. The value of MMRE in PSO-FCM model is 25.36, 24.56, 24.22 and 23.86 and 26.32 in the PSO-LA model. The accuracy of PRED (25) in the COCOMO Model is 40; it is 61.6, 58.3, 65 and 68.3 in PSO-FCM Model. It equals 63.3 in PSO-LA model.

3. Proposed method

Problems of using form algorithm models in SCE have led to much effort in optimizing these algorithms and a different method is presented for estimation. Among these, we can say that ANNs are highly efficient according to the characteristics such as high speed and noise immunity, ability to training and generality and resistance to change parameters more proficiency. ANNs are one of the computational methods that with help of the learning process and the use of process called neurons and recognize the relationships between data try to provide a mapping between the input space (the input layer) and favorable space (output layer). Layer (s) or hidden layers process the data processed from the input layer and gives it to the output layer. Training is a process that will ultimately lead to learning. Learning network occurs when network connection weights between the layers change in a way that the difference between measured and predicted values is acceptable. By achieving these requirements, learning process take place. These weights show memory and knowledge of the network of the network. Some of the significant features of ANNs are high speed processing, ability to learn the patters by following those patterns, generalizability after learning, resilience in against unexpected errors and not creating significant disruption in case of troubles in case of troubles due to the weight distribution network connection. In this paper, we use MLP that that is briefly explained below:

This network includes one input layer, one or more hidden layers and one output layer. In training stage to training information perceptron is given and the network weights are adjusted in such a way that error between the current output and the target is minimal or number of training reaches to the present value. Then to assess the accuracy of the learning process, some inexperienced input is applied to the network.

These inputs are different from those inputs used in the network training process. Generally training ANNs is very complex and it is an optimization problem with the number of variables. For MLP training that multi-layer feed forward network using back propagation algorithm. During MLP training with the help of back propagation learning algorithm, first, the calculation of the network input to the output of the network is done and calculated error values spread to the prior layer. At first, calculation of the output can be done layer by layer and the output of each layer will be the input of the next layer. In propagation mode, output layer are adjusted at the beginning, because there is a good amount for each of neurons and whit regard to this fact and the update rule, the weights can be adjusted.

In this paper, the multi-layer feed forward perceptron network along with the back propagation learning algorithm is composed of three simple process layers that are connected together. In this network the relationship between the outputs (O_t) and inputs $(O_{t-1}, O_{t-2}, \ldots, O_{t-p})$ is the Eq. 1.

$$O_t = b_0 + \sum_{j=1}^{Q} w_j . g(b_{0j} + \sum_{i=1}^{P} w_{i,j} . O_{t-1}) + \varepsilon_t \quad (1)$$

In the Eq. 1, w_j and $w_{i,j}$ are the parameters of the model and they often called the connection weights and ε_t is an appropriate for Q_t that on reaching the less it can be accepted that the network is well trained. P is the number of input neurons and Q represent the number of hidden neurons. Neurons in the hidden layer can use different active functions in order to produce the output. The most common ones are sigmoid logarithm functions, tangent sigmoid and linear active functions. That in this paper, we have used sigmoid active functions that are calculated according to the Eq. 2.

$$\varphi(v) = \frac{1}{1 + \exp(-av)} \quad (2)$$

In Eq. 2, $\varphi(v)$ is the value of sigmoid active function that is applied to the neurons in hidden layer in order to achieve the proper output. MLP introduced in Eq. 1 is actually a non-linear mapping from post observations to future value that is clearly shown in Eq. 3.

$$O_t = f(O_{t-1}, O_{t-2}, \ldots, O_{t-p}, W) + \varepsilon_t \quad (3)$$

In the Eq. 3 w is the vector of all parameters and f is a function that is determined by the neural network structure and connection weights. Selecting the number of hidden layer and number of neurons is so important for each of them. If the number is low, networks to solve complex and non-linear problems with learning error will be faced problems. And if it is increase will cause two problems: The first, training time increases and the second, network may learn the trivial network system of training data and solve poorly. One another form problem of MLP networks is that, despite the back-propagation algorithm has given very good result in solving problems; it performs poorly in solving some problems. It could be due to long or unclear time of learning, inappropriate selection of the learning coefficient or random distribution of initial weight. In some cases of the existence of local minima, learning process is disrupted. This is due to the answer which is on the verge of local parts of the threshold functions. To resolve this problem and to enhance perceptron network performance, we have hybrid it with ICA. In Fig. 1 shows the hybrid model of ANN and ICA.

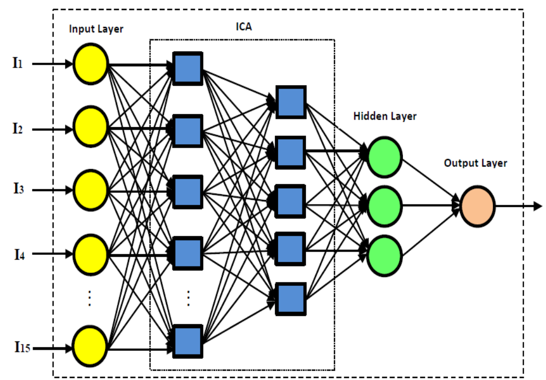

Fig. 1: Proposed method: hybrid model of ANN and ICA

As shown in Fig. 1 architecture of a proposed method consists of five layers. The first layer is input layer and the two next layer train network by ICA and the fourth layer which is related to the hidden layer network, is used for greater convergence of network and the lost layer is the output. The first layer that includes 15 neurons is receives EMs of each project from the input. These EMs include RELY, CPLX, DATA, ACAP and etc. (Menzies et al., 2005). Each EM shows the importance of projects with respect to factors that have been considered. Then data are given to the second layer which is related to ICA for better optimization and finding the necessary resources. Second layer includes 15 neurons that those neurons from the initial population (countries). Each of them considers just one of the factors as power function. After formation of countries, primary empires are formed and according to their EMs each of from empires number of countries belongs to them. In this stage, we have considered the number of empires equal to 5. Now competition between countries of each empire and between all the empires has started and as it is said in the introduction of ICA, empires are trying to converge subsidiary countries by changing the value of EM and are trying to move them toward themselves. In every time imperialistic competitions the power of each country and the weakest country of a weak colony granted to strong empires. This is repeated until the total elimination of a weak empire (Atashpaz and Lucas, 2007). Deleting of empires will continue until an empire remains. The input of the fourth layer that is the output of ICA, receives optimizes EMs from ICA. And from Eq. 1 is used for calculating the weights between third and fourth layer. Each layer has its own weight. In fourth layer, their values and their weights are updated by applying Eq. 2 each node. The percent of estimation accuracy is calculated by applying Eq. 4.

$$MRE_i = \frac{|Actual_i - Estimate_i|}{Actual_i} \times 100 \tag{4}$$

If it is acceptable, it will be sent to the output and otherwise by multiplying particular values of ANN and weights, data are entered into the network again. Steps are repeated until reaching the optimal solution. Fig. 2 shows the pseudo code hybrid model of ICA and ANN.

01: Enter 15 Cost Driver (EM_I)
02: Initialize the weights
Set learning rate γ ($0 < \gamma \leq 1$)
03: **Do**
04: **For** each training ANN
05: Begin
06: Initial Population (No. Countries),
07: Decided to form the initial empires.
08: **Loop**
09: Move the colonies toward the imperialist
10: **If** a colony in an empire which has lower cast than that of the

Exchange the positions of that imperialist and colony
11: Compute the total cost of all empires
12: Peak the weakest colony from the weakest empire and give it

13: **If** an empire with no colonies **then** eliminate this empire
14: **While** (condition satisfied)
15: Compute the predicted of each node

$$W_{EM} = b_0 + \sum_{i=1}^{Q} w_j . g(b_{0j} + \sum_{i=1}^{P} w_{i,j} . W_{EM-1}) + \varepsilon_t$$

$H_{PM} = \Pi EM_I$ for i= 1 to 15
PM= a *(size) ^{b} + W_{PM}
16: Update the weights between hidden and output layer.
W_{EM} (new) = W_{EM} (old) + γ * H_{EM}
17: Evaluate Fitness
MRE=|Estimate – Actual| / Actual;
MMRE=∑ (MRE) / N;
18: **While** (not terminate condition)
19: End

Fig. 2: Pseudo code of proposed method

In proposed method, MMRE is considered as the objective function. The target of fitness function in the proposed method is to minimize the value of MMRE compared with ANN and COCOMO models. Hybrid model is repeated until the value of MMRE reduced to acceptable levels. MMRE is defined according to Eq. 5 (MacDonell S.G. and Gray A.R., 1997).

$$MMRE = \frac{1}{N} \sum_{i=1}^{N} MRE_i \tag{5}$$

Using Eq. 5, the total error between the proposed method and ANN and COCOMO models is comparable. Also percentage relative error deviation is considered as an important measure of fore cost accuracy. The most common methods of checking accuracy in prediction are MMRE and PRED. PRED(x) is defined according to Eq. 6 (MacDonell S.G. and Gray A.R., 1997).

$$PRED(x) = \frac{1}{n} \times \sum_{i=1}^{n} \begin{cases} 1, & if MRE \leq x \\ 0, & otherwise \end{cases} \tag{6}$$

4. Evaluation and results

Several indices should be considered for estimating SCE and one of the most important parameters that need to be considered is the size of the projects. SCE depend on the accurate estimation about the LOC. Software projects are often not unique and there is no back ground or previous experience about them. So estimation seems complex. On the other hand, production of such projects is not tangible. Therefore, the measurement of cost and the rate of progress are too difficult in software projects.

In this section, we have evaluated and compared the result of the proposed method. Dataset software projects simulation on MATLAB 2012b has made in 60 projects that has 15 EM factors. In proposed method, appropriate value for the parameters of ICA affects ANN training a lot. Table 1 shows the parameters of the hybrid model.

Table 1: Parameter values

Parameters	Values
No. Population	15
No. Imperials	5
Iteration	20
β	2
γ	0.5
Revolution rate	0.2
Training	80%
Testing	20%

As shown in Table 1, the number of initial countries is equal to 15 and the number of empires is equal to 5 and the number of iterations of the algorithm is equal to 20. Parameter β makes the colony approaching the imperialist from different directions. Increase of γ causes increase in search around the imperialist by decrease of γ colonies will move as closely possible to the vector which connects the colony to the colonizer. Training rate and network testing are respectively equal to 80% and 20%. Table 2 shows that 60 projects of software projects from NASA have been evaluated and compared. Also, Thousands of Source Lines of Codes (KSLOC) projects are shown. That results of shown that the proposed method has reduced the value of MRE by comparison with ANN and COCOMO models. So proposed method is for effective estimation and it has less estimation error in comparison with COCOMO.

Table 2: Comparison of MRE models on 60 projects from software projects NASA dataset

No.	KSLOC	Actual Effort	MRE COCOMO	MRE ANN	MRE Propose Method
1	2.2	8.4	24.15	2.11	1.48
2	3.5	10.8	3.95	3.09	1.65
3	5.5	18	7.36	5.23	1.12
4	6	24	58.88	22.75	2.45
5	9.7	25.2	20.05	18.05	5.14
6	7.7	31.2	23.91	15.29	2.11
7	11.3	36	30.83	4.70	0.98
8	8.2	36	29.55	9.55	1.68
9	6.5	42	28.22	17.98	4.89
10	8	42	22.22	28.56	10.57
11	20	48	27.21	23.41	7.12
12	10	48	41.66	10.82	4.60
13	15	48	46.19	20.65	2.85
14	10.4	50	34.90	22.89	8.23
15	13	60	9.36	14.25	4.25
16	14	60	25.88	9.14	6.11
17	19.7	60	6.10	6.20	8.11
18	32.5	60	93.91	14.12	2.14
19	31.5	60	3.81	8.51	1.15
20	12.8	62	27.96	12.55	0.85
21	15.4	70	22.51	10.87	4.17
22	20	72	60.76	18.12	5.98
23	7.5	72	41.75	8.01	2.45
24	16.3	82	29.79	11.20	17.59
25	15	90	39.54	7.10	5.14
26	11.4	98.8	42.04	20.4	5.87
27	21	107	36.75	6.40	6.15
28	16	114	34.48	7.26	1.74
29	25.9	117.6	27.85	7.12	1.25
30	24.6	117.6	31.65	18.50	12.18
31	29.5	120	18.94	11.26	14.90
32	19.3	155	35.78	5.45	2.98
33	32.6	170	29.88	6.12	1.45
34	35.5	192	32.10	7.58	1.68
35	38	210	28.46	19.24	2.45
36	48.5	239	24.31	2.58	4.65
37	47.5	252	37.81	3.69	0.59
38	70	278	21.28	5.12	0.98
39	66.6	300	23.76	7.15	2.18
40	66.6	352.8	35.17	10.16	3.17
41	50	370	36.90	14.28	5.19
42	79	400	45.74	12.87	10.58
43	90	450	38.29	16.27	12.98
44	78	571.4	24.50	12.67	11.48
45	100	215	120.66	28.94	9.16
46	150	324	49.50	17.19	10.58
47	100	360	44.97	14.12	5.14
48	100	360	15.85	18.59	9.25
49	190	420	1.89	8.45	2.14
50	115.8	480	11.37	2.01	1.04
51	101	750	19.87	5.45	2.48
52	161.1	815	4.76	1.45	0.58
53	284.7	973	38.36	12.67	5.75
54	227	1181	3.93	2.48	3.15
55	177.9	1228	3.64	1.12	0.25
56	282.1	1368	17.21	15.6	7.14
57	219	2120	29.00	8.12	2.18
58	423	2300	25.78	7.90	2.85
59	302	2400	0.46	1.45	1.11
60	370	3240	25.21	1.78	0.33

In Fig. 3 is shown the comparison of MRE between the proposed method, ANN and COCOMO models on 60 software projects from NASA dataset. As it can be seen, the proposed method has considerably reduced the MRE error. As shown in Fig. 3, in most of the projects ANN-ICA models have less percentage of MRE error in comparison with other models. In Table 3, in order to illustrate the excellence of proposed method in MMRE and PRED in comparison with COCOMO and ANN are compared on 60 projects on NASA's software projects dataset. In comparison of models, PRED is equal to 15.

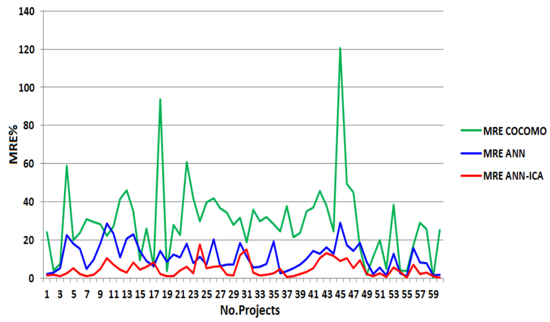

Fig. 3: Comparison chart of MRE in COCOMO, ANN and proposed method

Table 3: Comparison models considering error of MMRE and PRED

Models	MMRE	PRED (15%)
COCOMO	29.64	18.33
ANN	11.10	73.33
Proposed Method	4.78	98.33

According to Table 3, 98.33% percent of software projects that are using ANN-ICA have less than 15 percent error. However, in COCOMO model have just 18.33% from estimation less than 15% error and in ANN it is 73.33% models. In Figure (4), MMRE comparison chart of proposed method, COCOMO and ANN is shown for 60 projects dataset of software projects from NASA.

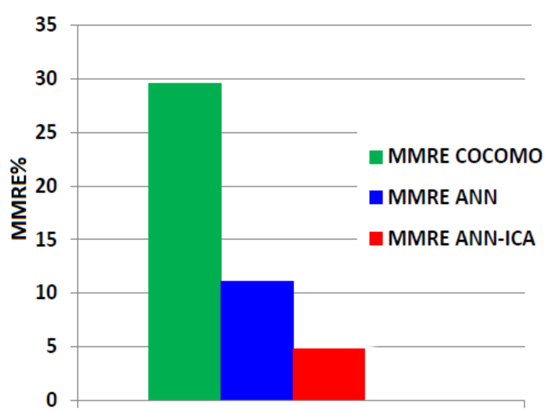

Fig. 4: Comparison Chart for MMRE of Models

According to Fig. 4, estimation accuracy of hybrid model is better than the COCOMO and ANN models. It shows that the hybrid model has reduced the error value of MMRE and it is a suitable method for SCE.

5. Conclusion and future works

Inaccuracy in SCE and efforts through estimations of optimistic or pessimistic may cause several problems for software projects. So, in this paper, hybrid ANNs and ICA is used for SCE. In the proposed method for training ANNs we have used ICA and preliminary data for testing and training dataset are NASA software projects. Design and implementation of the proposed method is on MATLAB 2012 b. ANNs have dumb and not analyzable structure to way that you cannot figure out what is the network going to do in the first sight to the neurons and their weights. But, by hybrid ICA with ANNs the structure of ANNAs can be completely interpretable for experts and evaluate and analyze it more easily. Results show that the error value of PRED (15%) for 60 projects of software projects in COCOMO, ANN and ANN-ICA is respectively equals to 18.33%, 73.33% and 98.33%. The accuracy of PRED in the hybrid model shows the fact that applying ICA and hybrid it with ANNs helps us to have more accurate estimation in software projects. We hope this paper will help us use other meta-heuristic in ANNs training in the future.

References

Alexander S. Poznyak, Edgar N. Sanchez, Wen Yu, (2001), Differential Neural Networks for Robust Nonlinear Control, World Scientific Publishing Co. Pre. Ltd, pp. 10-21.

AlYahya M., Ahmad R., Lee S., (2010), Impact of CMMI Based Software Process Maturity on COCOMO II's Effort Estimation, The International Arab Journal of Information Technology, Vol. 7, No. 2, pp. 129-137.

Atashpaz-G.E., Lucas C., (2007), Imperialist Competitive Algorithm: an Algorithm for Optimization Inspired by Imperialistic Competition, IEEE Congress on Evolutionary Computation, pp. 4661-4667.

Attarzadeh I., Mehranzadeh A., Barati A., (2012), Proposing an Enhanced Artificial Neural Network Prediction Model to Improve the Accuracy in Software Effort Estimation, Fourth International Conference on Computational Intelligence, Communication Systems and Networks, pp. 167-172.

Boehm B.W. (1981), Software Engineering Economics", Prentice-Hall, Englewood Cliffs, New Jersy, 1981.

Boehm B.W. (2000), Software Cost Estimation with COCOMO II", Prentice Hall PTR, Englewood Cliffs, New Jersy, 2000.

Cuadrado-Gallego J.J., Fernandez-Sanz L., Sicilia Miguel-Angel (2006), "Enhancing Input Value Selection in Parametric Software Cost Estimation Models through Second Level Cost Drivers", Software Quality Journal, Vol. 14, Issue 4, pp. 339-357.

German S., Bienenstock E., Doursat R., (1992), Neural Networks and the Bias/Variance Dilemma, Neural Computation, Vol. 4, No.1, pp. 1–58.

Gharehchopogh F.S. (2011), Neural Networks Application in Software Cost Estimation: A Case

Study, 2011 International Symposium on Innovations in Intelligent Systems and Applications (INISTA 2011), pp. 69-73, IEEE, Istanbul, Turkey, 15-18 June 2011.

Gharehchopogh F.S., Maleki I., Farahmandian M., (2012), New Approach for Solving Dynamic Traveling Salesman Problem with Hybrid Genetic Algorithms and Ant Colony Optimization, International Journal of Computer Applications (IJCA), Vol .53, No.1, pp. 39-44.

Gharehchopogh F.S., Maleki I., Khaze S.R., "A New Optimization Method for Dynamic Travelling Salesman Problem with Hybrid Ant Colony Optimization Algorithm and Particle Swarm Optimization", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), Vol. 2, Issue 2, pp. 352-358, February 2013.

Gharehchopogh, F.S,, Jabbari N., Ghaffari_Azar Z., (2012), Evaluation of Fuzzy K-Means and K-Means Clustering Algorithms in Intrusion Detection systems", International Journal of Scientific & Technology Research (IJSTR), ISSN 2277-8616, Vol: 1, Issue 11, pp.66-72.

Gharehchopogh. F.S, Ebrahimi. L, Maleki. I., Gourabi. S.J, (2014), A Novel PSO based Approach with Hybrid of Fuzzy C-Means and Learning Automata in Software Cost Estimation, Indian Journal of Science and Technology, 7(6): 795-803.

Idri A., Abran A., Mbarki S., (2006), An Experiment on the Design of Radial Basis Function Neural Networks for Software Cost Estimation, Information and Communication Technologies (ICTTA '06), Damascus, Vol. 1, pp. 1612-1617.

Khalifelu Z.A., Gharehchopogh F.S., (2012), Comparison and Evaluation Data Mining Techniques with Algorithmic Models in Software Cost Estimation", Elsevier, Procedia-Technology Journal, Vol. 1, pp. 65-71.

Kumar K.V., Ravi V., Carr M., Kiran N.R., (2008), Software Development Cost Estimation using Wavelet Neural Networks, The Journal of Systems and Software, Vol. 81, pp. 1853-1867.

MacDonell S.G., Gray A.R., (1997), A Comparison of Modeling Techniques for Software Development Effort Prediction", in Proceedings of International Conference on Neural Information Processing and Intelligent Information Systems, pp. 869-872.

Maleki, I. Ebrahimi, L. Gharehchopogh, F.S., (2014), A Hybrid Approach of Firefly and Genetic Algorithms in Software Cost Estimation, MAGNT Research Report, 2(6): 372-388.

Menzies T., Port D., Chen Zh., Hihn J.,(2005), Validation Methods for Calibrating Software Effort Models, ICSE ACM, USA.

Nassif A.B., Ho D., Capretz L.F., (2013), Towards an Early Software Estimation using log-linear Regression and a Multilayer Perceptron Model, Journal of Systems and Software, Vol. 86, Issue 1, pp.144-160.

Ricardo de A.A., Adriano L.I. Oliveira, Soares S., Meira S., (2012), "An Evolutionary Morphological Approach for Software Development Cost Estimation", Neural Networks, Vol. 32, pp. 285-291.