Comput Syst Sci & Eng (2017) 5: 369–382

International Journal of
Computer Systems
Science & Engineering

An optimized adaptive neuro-fuzzy inference system to estimate software development effort

Seyyed Hamid Samareh Moosavi and Vahid Khatibi Bardsiri*

Department of Computer Engineering, Kerman Branch, Islamic Azad University, Kerman, Iran E-mail: hamid.moosavi17@gmail.com

One of the most critical activities in software project management during the project inception phase is to estimate the effort and cost needed to complete the project tasks. Appropriate planning of software projects is strongly dependent to accurate software development effort estimation. Lack of information and unstable requirements make the process of effort estimation complicated. In spite of proposing numerous effort estimation models, the estimation accuracy needs to be improved. In this paper, a novel hybrid effort estimation model is proposed which is a combination of an adaptive neuro-fuzzy inference system (ANFIS) and firefly algorithm (FA). The proposed hybrid model is an optimized neuro-fuzzy based estimation model which is capable of producing accurate estimations. The evaluation of the proposed model, is performed using three real data sets (ISBSG, Kemerer and Albrecht). Results show that the proposed model can significantly improve the performance metrics.

Keywords: software development effort estimation, adaptive Neuro-fuzzy inference system, firefly algorithm, Performance metrics

1. INTRODUCTION

Accurate estimation of development effort is one of the challenging issues for software project management. Efficient resource allocation and scheduling is significantly dependent to reliable effort estimation. In the process of project estimation software development team usually estimates three aspects that are effort, schedule and cost among which effort estimation is the most important one. Underestimating software project effort causes schedule delays and cost over-runs, which, in the end, can lead to project failure. Conversely, overestimating software project effort can also be detrimental in effectively utilizing software development resources. Some of the reasons for the failure of software projects are as follows:

- Poor planning of the project
- Suddenly making decisions
- Insufficient engineering requirements
- Inaccurate estimations

*Corresponding Author: E-mail: kvahid2@live.utm.my

Therefore, different effort estimation models have been invented based on theoretical concepts and combination of existing models during the last decade [1-3].

In order to achieving more accurate results, researchers continue to develop new effort estimation models. A systematic literature review was conducted through Jorgenson and Shepperd in which 304 journal papers were investigated and 11 estimation techniques were identified [1].The investigated models can be classified into two main classes of parametric and machine learning models. Statistical and numerical investigation of historical projects is the main part of parametric models while artificial intelligence methods such as neural network, analogy based estimation, optimization algorithms and decision tree are located in machine learning models. The second group has attracted the attention of researchers due to their capability to overcome the complexity of non-linear relationship between project features and effort. The vital role of effort estimation field leads to appearance of various estimation models. From a comprehensive review [4], these models could be classified into the following categories:

However improvements achieved by existing estimation models are obvious, novel estimators are still developing. Due to

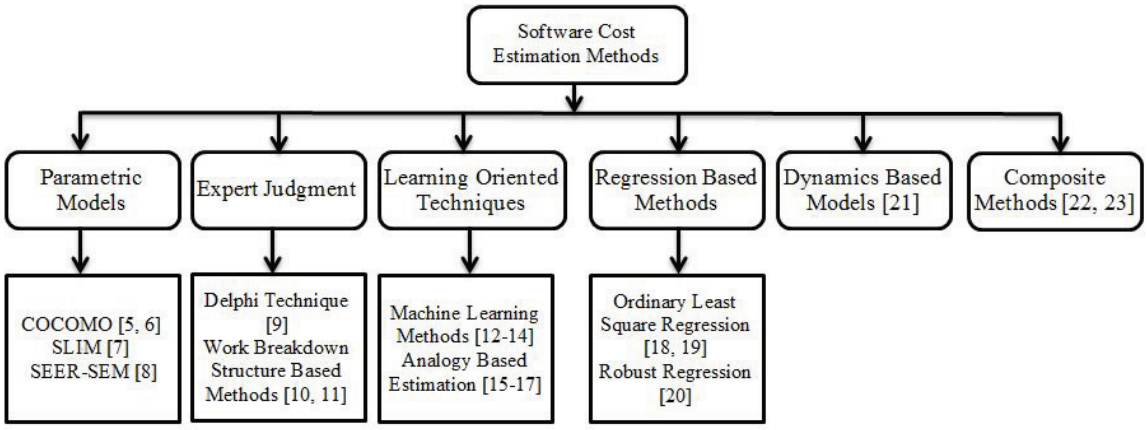

Figure 1 Software cost estimation methods.

wide applications of artificial intelligence methods [24-28] and uncertain and complicated features of software projects, an optimized neuro-fuzzy based model is proposed in this paper to estimate the effort. The main motivation behind this paper is the power of neuro-fuzzy systems in controlling and handling the complexity of effort estimation problem.

This paper is organized in 11 sections. Related works is presented in section 2. ANFIS structure is presented in section 3. In Section 4, firefly algorithm is described in details. Performance metrics and proposed method are presented in Sections 5 and 6, respectively. The experimental design is elaborated in Section 7. The results are presented in Section 8. In sections 9 and 10, the results are compared and the improvement is analyzed. Finally, the overall conclusion is presented in Section 11.

2. RELATED WORKS

Ziauddin A, et al, introduced a fuzzy logic based software cost estimation model. This study aims to utilize a fuzzy logic model to improve the accuracy of software effort estimation. The main idea in this study is fuzzifing input parameters of COCOMO II model and the result is defuzzified to get the resultant effort. Triangular fuzzy numbers are used to represent the linguistic terms in COCOMO II model. The results of this model are compared with COCOMO II and Alaa Sheta model. The proposed model yields better results in terms of MMRE, PRED (n) and variance account for (VAF). VAF is used in the context of statistical models whose main purpose is the prediction of future outcomes on the basis of other related information [29].

Vishal S., et al, introduced optimized fuzzy logic based framework for effort estimation in software development. The performance of the proposed framework is demonstrated in terms of empirical validation carried on real project data of the COCOMO public database. The results show that the proposed framework can be deployed on COCOMO II environment with information provided by experts for developing fuzzy sets and appropriate rule base [30].

Abeer H. introduced a model based on a fuzzy logic for enhancing the sensitivity of COCOMO cost model. This study enhances the accuracy and sensitivity of COCOMO 81 intermediate by fuzzifying the cost drivers. The dataset was collected from six NASA centers and covers a wide range of software domains, development process, languages and complexity, as well as fundamental differences in culture and business practices between centres. The results showed that the sensitivity of the proposed fuzzy model is superior to COCOMO81 intermediate [31].

Azzeh et al developed a new similarity measure based on integration of Fuzzy set theory and Grey Relational Analysis for analogy-based estimation. The proposed measure has the capability to deal with numerical and categorical attributes such that two levels of similarity measure have been defined: local and global measures. The results obtained suggested that the proposed model produces good accuracy when compared to other well Known estimation techniques such as case-based reasoning, stepwise regression and artificial neural network [32].

Idri et al. proposed a new Fuzzy analogy software cost estimation based on linguistic quantifiers. The model was designed for the datasets that are described by linguistic quantifiers (using an ordinal scale) such as the COCOMO dataset. They used Fuzzy aggregation operators to adjust estimates based on Fuzzy similarity between two software projects. This approach does not appear to perform well over other datasets that are not structurally similar to COCOMO dataset, and it is no suitable for early stage estimation [33].

Musflek et al developed a granular model for software cost estimation based on Fuzzy number called f-COCOMO. Both input (kilo line of code) and output (effort) are represented by their corresponding triangular Fuzzy numbers. The mapping between input and output was performed using the possibility distribution which assumes that the uncertainty in the input domain should be reflected on the uncertainty in the output domain. The model has a lack of validation in terms of prediction accuracy [34].

Kanmani et al proposed another technique based on with fuzzy logic using subtractive clustering technique for calculating the effort and have compared the result with that obtained using the concept of artificial neural network. They found that fuzzy system using subtractive clustering technique yields better result than that of artificial neural network [35].

Fei and Lui introduced the f-COCOMO model which applied fuzzy logic to the COCOMO model for software effort estimation. Since there was no comparison of the results between the f-COCOMO and other effort estimation models in their study, the estimation capability of the former is unknown [36].

Roger proposed a fuzzy COCOMO model which adopted the fuzzy logic method to model the uncertainty of software effort drivers, but the effectiveness of the proposed model is not mentioned [37].

Xue and Khoshgoftaar presented a fuzzy identification effort estimation modelling technique to deal with linguistic effort drivers, and automatically generated the fuzzy membership functions and rules by using the COCOMO81 database. The proposed fuzzy identification model provided significantly better effort estimates than the original three COCOMO models, i.e., basic, intermediate, and detailed [38].

3. ANFIS

Fuzzy inference system was developed in 1993 by Gang [39]. This model combines fuzzy logic with artificial neural networks to simplify the process of learning and adaptation. In fact, in neuro fuzzy models an adaptive network is used to solve the problem of identification of the fuzzy inference system parameters. An adaptive network is a feed forward and multilayer structure that its overall behavior output is determined by a set of modifiable parameters. By using an adaptive neural network, the main problem of fuzzy inference system is resolved (to obtain the fuzzy rules and optimization parameter model). The ANFIS architecture is shown in Fig 2.

The ANFIS network is composed of five layers. Each layer contains several nodes which are described by the node function. Let O_i^j denotes the output of the i_{th} node in layer j. The layers of ANFIS model have been described in following sections.

Layer1: Every node i is adaptive with node function. Equations (1) and (2) show the node functions.

$$o_i^1 = \mu A_i(x), \quad i = 1, 2 \tag{1}$$

$$o_i^1 = \mu B_{i-2}(y), \quad i = 3, 4 \tag{2}$$

Where x (or y) is the input to the i_{th} node and Ai (or Bi_2) is a linguistic label associated with this node; therefore, o_i^1 is the membership grade of a fuzzy set A (=A1, A2, B1, or B2). The main task of the first layer is fuzzification.

Layer2: At the second layer, all potential rules between the inputs are formulated by applying fuzzy intersection (AND). Hence, each output node represents the firing strength of a rule. Equation (3) shows this issue.

$$o_i^2 = w_i = \mu A_i(x)\mu B_i(y), \quad i = 1, 2 \tag{3}$$

Layer3: In this layer, weights obtained from the second layer are normalized by using Equation 4.

$$o_i^3 = \overline{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1, 2 \tag{4}$$

Layer4: Each node computes the contribution of the ith rule to the overall output. Equation 5 performs this computation.

$$o_i^4 = \overline{w}_i z_i = \overline{w}_i(a_i x + b_i y + c_i), \quad i = 1, 2 \tag{5}$$

Where $\overline{w}_i$ is the output of layer 3 and {ai, bi, ci} is the parameter set. Parameters of this layer are referred to consequent parameters.

Layer5: The final layer computes the overall output as the summation of all incoming signals from layer 4by using equation 6.

$$o_i^5 = \sum \overline{w}_i z_i = \frac{\sum_i w_i z_i}{\sum_i w_i} \tag{6}$$

Different methods are proposed for ANFIS training. The most common method is gradient descent to minimize the output error.

4. FIREFLY ALGORITHM

There are about two thousands firefly species, and most of fireflies produce short and rhythmic flashes. Two fundamental functions of such flashes are to attract mating partners (communication), and to attract potential prey. In 2007, firefly algorithm (FA) was developed by Xin-She Yang at Cambridge University [40], at the present time the following three idealized rules are taken into consideration:

1. All fireflies are unisex; as a result, one firefly will be attracted to other fireflies regardless of their sex.
2. Attractiveness of fireflies is proportional to their brightness, thus for any two flashing fireflies, the less bright one will move towards the brighter one. The attractiveness is proportional to the brightness and they both decrease as their distance increases. If there is no firefly brighter than a particular one, it will move randomly.
3. The brightness of a firefly is affected or determined by the landscape of the objective function.

Founded on these three rules, the basic steps of the firefly algorithm can be summarized as the pseudo code shown in following.

Objective function f(x), x = (x1, ...,xd)^T
Generate initial population of fireflies *Xi* (i = 1,2, ..., n)
Light intensity I_i at x_i is determined by f(x_i)
Define light absorption coefficient γ
while (t <MaxGeneration)
for i =1 : n all n fireflies
  for j = 1 : n all n fireflies (inner loop)
  if (I_i < I_j), Move firefly i towards j; end if
    Vary attractiveness with distance r via exp[-γr]
    Evaluate new solutions and update light intensity
  end for j
end for i
Rank the fireflies and find the current global best g*
end while
Post process results and visualization

4.1 Light Intensity and Attractiveness

In the firefly algorithm, there are two important issues:

1. Variation of light intensity.
2. Formulation of the attractiveness.

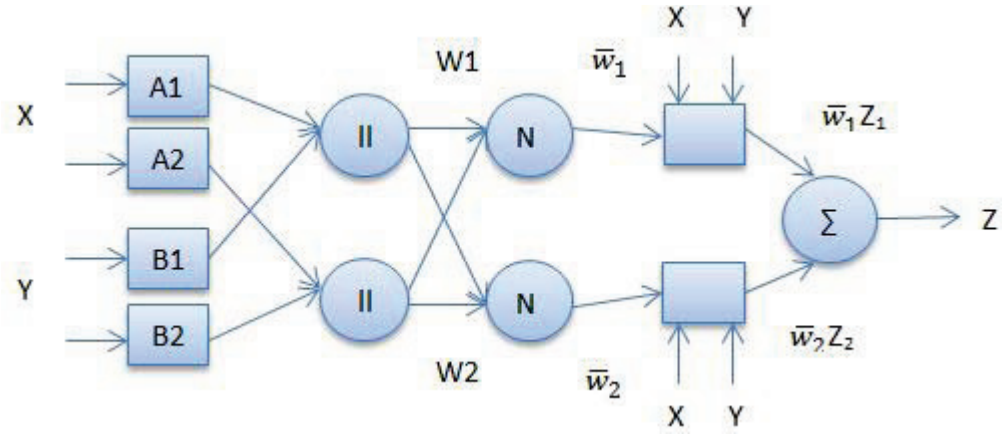

Figure 2 ANFIS architecture.

In the simplest form, the light intensity I_s is calculated in Equation 7. Where, I_s is the intensity at the source and r is the distance between two points. Due to singularity problem (r=0) in Equation 7, Equation 8 can be used, where I_0 is the original light intensity and γ is light absorption coefficient. By combining the two equations 7 and 8, equation 9 is obtained.

$$I(r) = \frac{I_s}{r^2} \tag{7}$$

$$I(r) = I_0 e^{-\gamma r} \tag{8}$$

$$I(r) = I_0 e^{-\gamma r^2} \tag{9}$$

Since the firefly's attractiveness is proportional to the light intensity seen by adjacent fireflies, the attractiveness of a firefly is calculated in by Equation 10, where ß$_o$ is the attractiveness at r = 0. Instead of Equation 10, Equation 11 can also be used to calculate the attractiveness.

$$\beta = \beta_0 e^{-\gamma r^2} \tag{10}$$

$$\beta = \frac{\beta_0}{1 + \gamma r^2} \tag{11}$$

By defining a characteristic distance $\Gamma = \frac{1}{\sqrt{\gamma}}$ and placing it in Equation 10 and Equation 11, the Equation 12 and 13 are obtained. In a more general form, the attractiveness function can be computed by Equation 14.

$$\beta(\Gamma) = \beta_0 e^{-1} \tag{12}$$

$$\beta(\Gamma) = \frac{\beta_0}{2} \tag{13}$$

$$\beta(r) = \beta_0 e^{-\gamma r^m}, \; m \geq 1 \tag{14}$$

For a fixed γ, the characteristic length is calculated in Equation 15. By calculating value Γ, γ can be calculated by Equation 16.

$$\Gamma = \frac{1}{m\sqrt{\gamma}} \tag{15}$$

$$\gamma = \frac{1}{\Gamma^m} \tag{16}$$

In 2-D space, the distance between any two fireflies i and j at xi and xj is obtained by Equation 17.

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{17}$$

The movement of a firefly i *that* is attracted to another more attractive (brighter) firefly j is calculated by Equation 18.

$$x_i = x_i + \beta_0 e^{-\gamma r_{ij}^2}(x_j - x_i) + \alpha \epsilon_i \tag{18}$$

In this equation, ϵ_i is a vector of random numbers drawn from a Gaussian distribution or uniform distribution and α is a randomization parameter. For calculation of α, Equation 19 can be used, where $\in 0$, $t_{\max}$ is the pseudo time for simulations and $t_{\max}$ is the maximum number of generations. α_0 is the initial randomization parameter while α_∞ is the final value. Instead of the Equation 19, Equation 20 can be utilized for calculation of α, where $\theta \in 0, 1$ is the randomness reduction constant.

$$\alpha = \alpha_\infty + (\alpha_0 - \alpha_\infty)e^{-t} \tag{19}$$

$$\alpha = \alpha_0 \theta^t \tag{20}$$

5. PERFORMANCE METRICS

Performance of estimation models is evaluated by several metrics including Relative Error (RE), Magnitude of Relative Error (MRE), and Mean Magnitude of Relative Error (MMRE) which are computed as the following equations:

$$RE = \frac{(estimate - actual)}{actual} \tag{21}$$

$$MRE = \frac{|estimate - actual|}{actual} \tag{22}$$

$$MMRE = \frac{\sum_{i=1}^{N} MRE}{N} \tag{23}$$

The other parameter used for evaluating the performance is Percentage of the Prediction (PRED) determined as:

$$PRED(X) = \frac{A}{N} \tag{24}$$

Where, A is the number of projects with MRE less than or equal to X while N is the number of considered projects. Usually, the acceptable level of X in software cost estimation methods is 0.25 and the various methods are compared based on this level [41–43]. Decrease of MMRE and increase of PRED are the main aims of all estimation techniques used in the field of software development effort.

6. PROPOSED MODEL

Due to the complexity and inconsistency of software projects, ANFIS is confronted by serious challenges to reach accurate effort estimations. In order to overcome this problem, the ANFIS parameters can be optimized by a meta-huristic optimization algorithm, which is the main goal of this paper. Firefly algorithm is utilized to optimize ANFIS in this paper, which makes a high performance effort estimation model. Indeed, FA tries to find the best possible parameters to be employed by ANFIS so that the most accurate estimations are obtained. Two main steps are designed in the proposed model as follows:

6.1 ANFIS optimization using FA

This step includes the process of adjusting ANFIS parameters to find the best possible structure, which is carried out through exhaustive investigation of structures. Indeed, an optimization problem is defined to reach the best configuration of ANFIS. In this step, the traditional and classical training of ANFIS is replaced with intelligent meta-heuristic optimization algorithms. Tagaky-Sugeno fuzzy system is employed in the proposed model. Figure 3 depicts the process of ANFIS optimization in the proposed model. This step can be described as follows:

6.1.1 Training and testing groups

Albrecht, Kemerer and ISBSG real data sets are utilized in this paper to estimate the software development effort by the proposed model. At the first step, the ANFIS input data must be normalized in the range of zero and one. After that, based on three fold cross validation, three equal groups (almost having the same size) are randomly created. One of these groups is selected as the testing group while the other two groups are considered as the training group.

6.1.2 Designing the basic fuzzy system

Genfis2 system is utilized as a base fuzzy system in this paper. This system creates a Sugeno FIS by means of subtractive clustering. Two groups of input and output are separately employed by this system. At first, the subclust function is used based on the extraction rule to identify the number of rules and antecedents membership functions. Afterward, the rules consequent equations are determined using least linear squares estimation.

6.1.3 ANFIS parameters adjusting

In order to optimize the performance of ANFIS, the main stage is adjusting the configuration parameters. This stage is explained in the following section.

1. Obtain the base fuzzy system parameters: This step includes obtaining the input and output parameters of membership functions. The obtained parameters are then sequentially sorted in a vector.

2. Modify ANFIS parameters using FA: it is assumed that the optimal values must be the coefficients of initial parameters. Equation 25 shows the process of obtaining the optimal value of parameter i (p_i^*) where p_i^0 is the initial value for parameter i and x_i is the related coefficient.

$$p_i^* = x_i p_i^0 \tag{25}$$

The role of FA is determining the value of xi, which can be performed in two ways:

In this equation, the is determined by FA. There are two viewpoints about the x_i:

1. The size is modified and sign is stable so that $x_i \in 10^{-\alpha}, 10^{\alpha}$ and α must be less or equal to one.

2. The size and sign are both modified and $x_i \in -M, +M$ where M is a number less than 10.

3. Configure ANFIS using the optimal parameters found by FA and calculating the performance parameters of MMRE and PRED(0.25).

6.1.4 Fitness function definition

The best parameters determined by FA are employed to configure ANFIS in the next stage. Both performance metrics of MMRE and PRED(0.25) are considered in evaluation of parameters carried out by FA. Indeed, MMRE-PRED(0.25) is the fitness function which must be minimized by FA. This is because MMRE needs to be minimized while PRED(0.25) must be maximized so that the best possible performance is achieved.

6.2 Performance evaluation

The evaluation of the proposed hybrid model is presented at this stage. MMRE and PRED(0.25) are the parameters used to evaluate the accuracy level. At this stage, the normalized testing data are applied to basic genfis2 system to evaluate the performance using unseen data. The default parameters of ANFIS are replaced by those determined in the prior stage. Then, the values of MMRE and PRED(0.25) are calculated to show the real performance of the proposed model. The optimal parameters increase the accuracy of ANFIS because the training efficiency is substantially improved. Different parameters are proposed by FA and the best of which is employed for evaluation purpose. Figure 4 depicts the evaluation process of the proposed model as explained.

7. EXPERIMENTAL RESULTS

Three real data sets (Albrecht, Kemerer and ISBSG) are employed to evaluate the performance of the proposed model in this paper. Matlab software was utilized to implement the proposed model. The input data are normalized in the range of zero to one and are then randomly divided into three groups with the same size based on three fold cross validation. After conducting the training stage, the optimal parameters are used by ANFIS and the effort is estimated and finally MMRE and PRED(0.25) are computed for three groups of testing data.

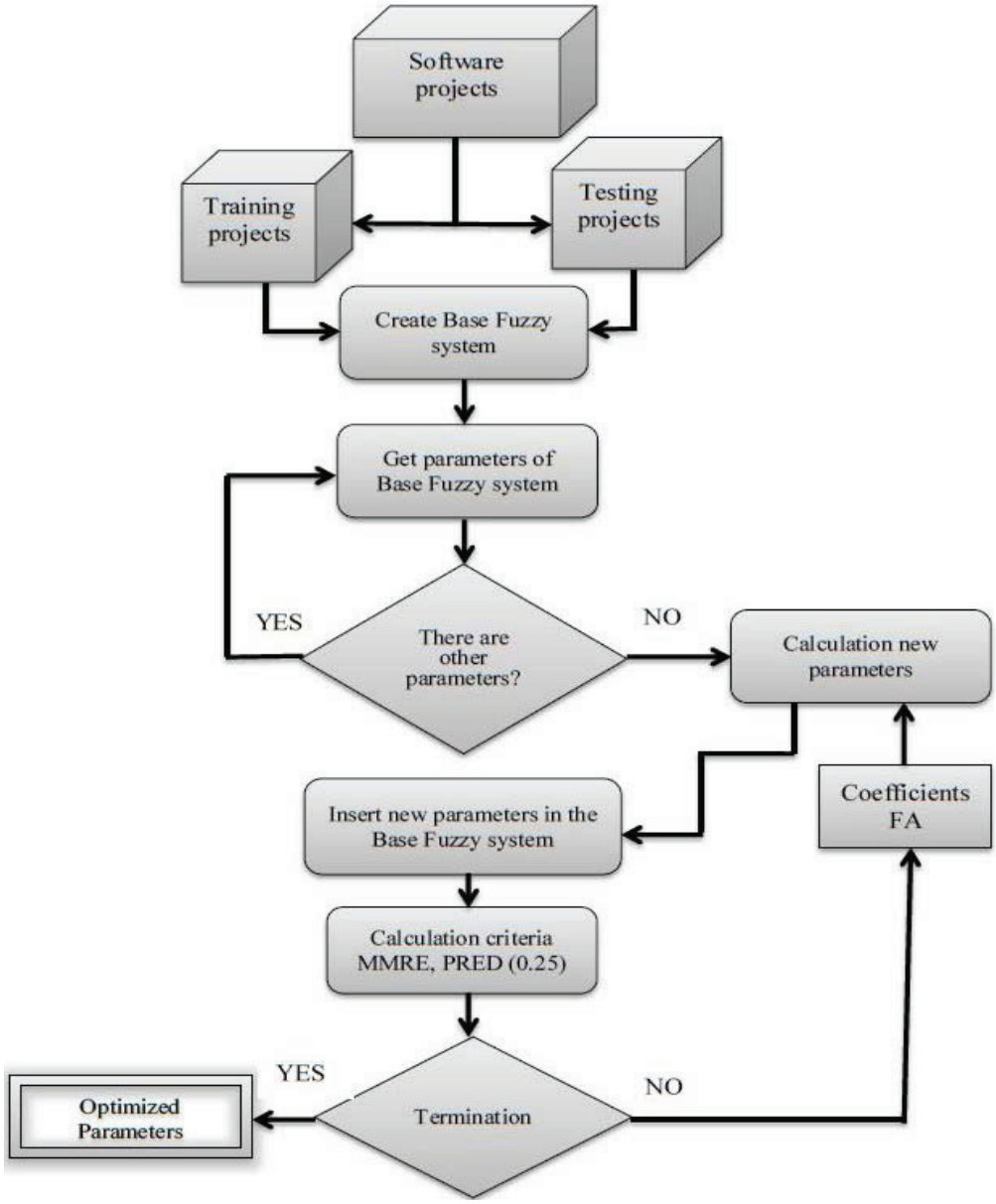

Figure 3 Optimized Neuro-fuzzy system design process in the first stage.

7.1 Data normalization

Lots of methods exist for data normalization which mostly focused on data conversion so that the desired conditions are achieved. Equation 26 shows the method used in this paper to normalize the data in which all the values are converted to the range of zero to one.

$$z_i = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}} \tag{26}$$

The normalized value is shown by z_i, x_i is a feature value, $x_{\min}$ is the minimum value and $x_{\max}$ is the maximum value of a feature column, respectively.

7.2 Cross validation

Three fold cross validation is employed to evaluate the performance of the proposed model in this paper. At first, all the data are randomly divided into three groups having the same size. A group is treated as the testing data while the other two groups are joined to make the training data. The model is trained using the training data and then the trained model is evaluated using the testing data. The performance metrics are calculated and saved for testing projects. This process is repeated three times so that all the data are evaluated. Finally, the average of performance metrics computed for testing data is considered as the final result.

7.3 Data sets

7.3.1 ISBSG

ISBSG (International Software Benchmarking Standards Group) has developed and refined its data collection standard over a ten-year period based on the metrics that have proven to be very useful to improve software development processes. The recent data release of this organization is the ISBSG R11 data repos-

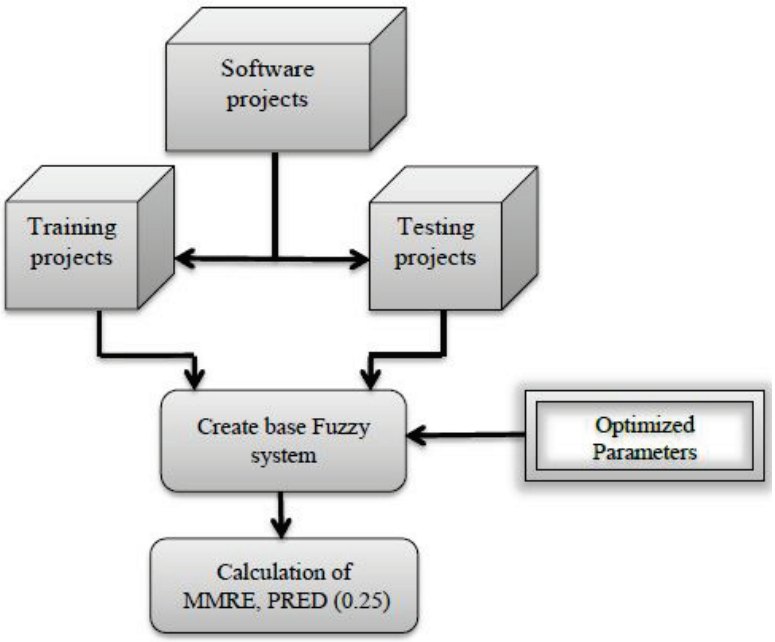

Figure 4 Optimized Neuro-fuzzy system in the second stage.

itory [44] which contains totally 4,106 projects (with 105 features) coming from 22 countries and various organizations such as banking, communications, insurance, business services, government and manufacturing. Due to the heterogeneous nature and the huge size of the entire repository, ISBSG recommends extracting out a suitable subset for any cost estimation practice. At the first step, only the relevant features characterizing projects should be considered to create the subset. Thus, we select out 14 important features (include project effort) suggested by ISBSG: 'DevType', 'OrgType', 'BusType', 'App- Type', 'DevPlat', 'PriProLan', 'DevTech', 'ProjectSize' (consisting of six sub features: 'InpCont', 'OutCont', 'EnqCont', 'FileCont', 'IntCont', and 'AFP'), and 'NorEffort'. Then, the projects with missing values in any of the selected feature are excluded from the subset. Afterward, a further step is taken to refine the subset. In ISBSG dataset, project data quality is rated and only projects with A or B rating are used in published research works. Therefore the projects with the ratings other than A and B are excluded from the subset. Moreover, since the normalized effort ('NorEffort') is used as the target for estimation, the risk of using normalized effort should be noted. For project covering less than a full development life cycle, normalized effort is an estimate of the full development effort and this may introduce biasness. Hence the normalized ratio (normalized effort / summary effort) is used to refine the project subset. As suggested by ISBSG that the ratio up to 1.2 is acceptable, we filter out the projects with normalized ration larger than 1.2. Finally, the subset is further reduced to the projects with 'Banking' as 'OrgType'. After all, the above procedures results to a subset with 118 projects. The descriptive statistics of all features are summarized in Table 1.

7.3.2 Albrecht

The Albrecht dataset is a popular dataset [45]. This dataset includes 24 projects developed by third generation languages. Features of this dataset include:

Table 1 Descriptive statistics of all features of ISBSG data set.

SD	Skewness	mean	maximum	minimum	variable
0.5	−0.07	1.52	2	1	DevType
6.36	0.29	7.55	15	2	BusType
2.14	0.18	5.76	9	1	AppType
4.5	0.03	6.25	12	1	DevPlat
0.77	1.87	1.45	4	1	PriProLan
3.96	0.10	10.19	16	4	DevTech
172.2	3.37	107.7	1240	6	Inpcount
303.8	3.42	165.4	2455	4	Outcount
137.3	2.70	98.8	1306	3	Enqcount
188.7	2.24	114.7	1732	7	Filecount
215.88	1.83	99.42	1572	5	Intcount
864	2.81	599.1	7633	57	Afp
6257.1	2.86	5898.9	36255	426	Noreffort

- Effort
- Function point (fp)
- File count (Filcount)
- Input count (Inpcount)
- Enquiry count (EnqCont)
- Output count (Outcount)
- Source line of cod (SLOC)

The descriptive statistics is presented in Table 2.

7.3.3 Kemerer

This data set contains data from 15 large completed business data-processing projects of the same company. Each project has six input features: (1) programming language, (2) hardware, (3) duration, (4) KSLOC, (5) AdjFP (adjusted function points), and (6) RAWFP (raw function points). [46].

Table 2 Descriptive statistics of all features of albrecht data set.

skewness	SD	median	mean	max	min	count	feature
2.3	28.4	11.5	21.9	105	0.5	24	effort
1.5	493	506	643.3	1902	100	24	fp
1.5	15.5	11.5	17.4	60	3	24	file
3.3	36.9	33.5	40.3	193	7	24	input
2.1	19.3	13.5	16.9	75	0	24	inquiry
1.4	35.2	39	47.3	150	12	24	output
3.1	63.7	51.7	61.1	318	3	24	SLOC

7.4 Initial settings of parameters

The base fuzzy system is genfis2 in this paper. In order to achieve the best possible result, the system parameters must accurately be determined. Table 3 shows the parameter values determined for the system. Instead of using default parameters, ANFIS uses optimal parameters determined by FA to reach more reliable estimation results. The parameters used in FA are shown in Table 4. These parameters are determined by conducting an exhaustive trial and error process.

Table 3 ANFIS parameter values.

parameter	value
Input MF function type	Gaussian
Output MF type	Linear
Learning algorithm	FA
Based fuzzy system	Genfis2
And method	prod
Or method	probor
Defuzz method	wtaver
Im method	prod
Agg method	sum

8. RESULTS

Tables 5 represent the results of applying the proposed approach to Albrecht, ISBSG and Kemerer data. The obtained results show that applying FA to ANFIS makes the training process more accurate. Due to accurate training of Network, acceptable results are obtained (MMRE and PRED (0.25)) for three data sets.

Table 4 FA parameter values.

100	Max it
30	npop
1	gama
2	beta
0.2	alpha
0.99	Apha-damp
2	m

Table 5 Results of applying the proposed model to Albrecht, Kemerer and ISBSG.

Data set	PRED (0.25)	MMRE
Albrecht	0.50	0.347
ISBSG	0.610	0.557
Kemerer	0.60	0.368

9. COMPARING THE PROPOSED MODEL WITH OTHER MODELS

Table 6 shows the results related to comparison of the proposed model with other models on Albrecht data set. Multiple linear regression (MLR), stepwise regression (SWR) and classification and regression tree (CART) are the estimation methods which have been compared to the proposed model. The selection of these methods is due to their wide application in prior studies [41, 47-51]. Artificial neural network (ANN) is the other model involved in comparison process.

Moreover, the estimation accuracy of the proposed model is compared with that of reported in recent two studies [43,47]. The results of MMRE and PRED(0.25) on Albrecht data set are shown in Figure 5. It is seen that the proposed model achieves more accurate results compared to the other models. Indeed, these parameters are significantly improved by the proposed model.

Table 6 Results of the Albrecht data set in comparison with other models.

PRED(0.25)	MMRE	method
0.44	0.40	OABE
0.375	1.03	CART
0.25	0.84	SWR
0.25	0.975	ANN
0.50	**0.347**	ANFIS-FA
0.36	0.41	NABE
0.125	1.17	MLR

Table 7 shows the comparison between the proposed model and other models based on performance metrics on ISBSG data set. In addition to MLR, CART and SWR, the results of the proposed model is compared with that of reported in [41]. Figure 6 depicts the MMRE and PRED(0.25) results related to the proposed model and other models on ISBSG data set. It is obvious that the proposed model presents more accurate results compared to the other models.

In Table 8, MLR, SWR and CART are compared with the proposed model on Kemerer data set in terms of MMRE and PRED(0.25). In addition, the results reported in [43] are compared to those of obtained by the proposed model. The comparison of MMRE and PRED(0.25) results are seen in Figure 7. It is observed that the proposed model can significantly improve the performance metrics on Kemere data set.

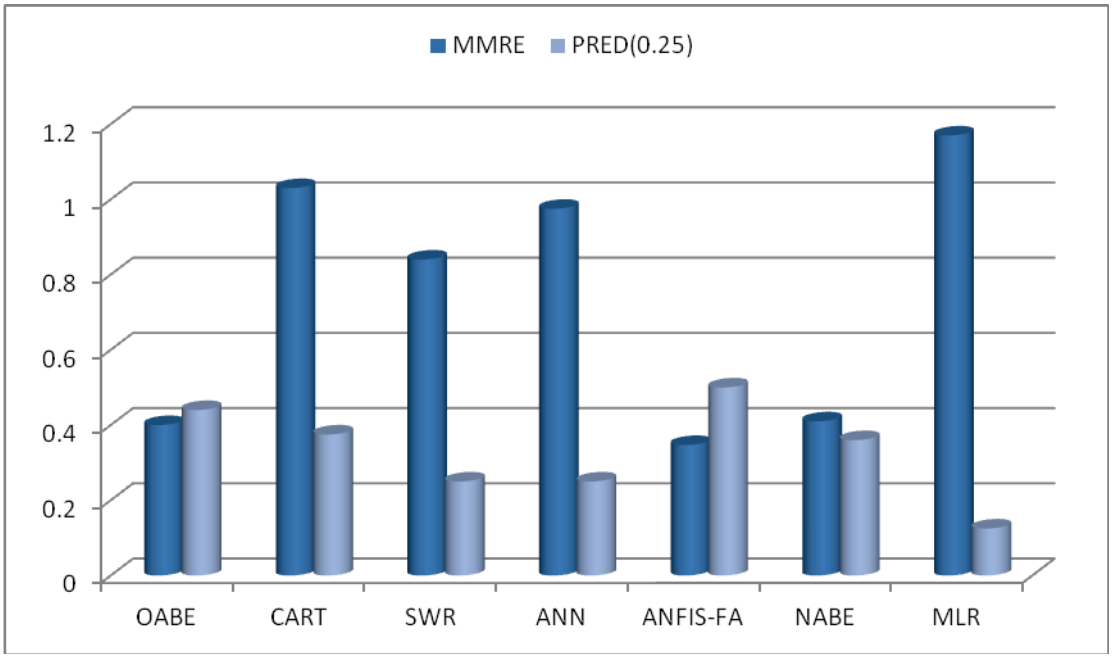

Figure 5 Values obtained from PRED (0.25) and MMRE of the Albrecht data set in comparison with other models

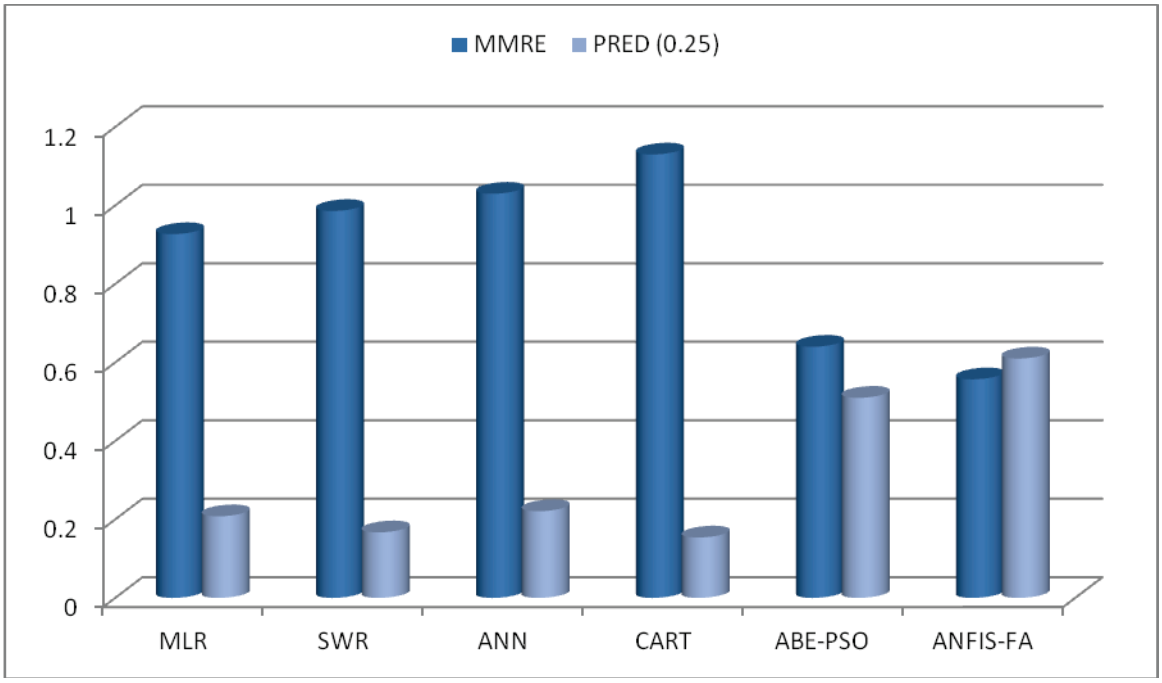

Figure 6 Values obtained from MMRE and PRED (0.25) of the ISBSG data set in comparison with other models.

10. IMPROVEMENT ANALYSIS

An accurate and reliable estimate of software development effort is a very important issue for project managers. Due to uncertainties in the software projects, a method should be selected not only to handle this uncertainty, also to provide an accurate estimation of the development effort. ANFIS can handle uncertainties of software projects with the correct choice of membership functions and training algorithms. Although ANFIS is a well-known estimation model and is widely used in software development effort estimation, this model is still not able to produce an accurate estimate in many situations. In this section the improvements achieved by the proposed model are investigated. Figure 8 shows a comparison between the proposed model and other models in terms of improvement percentage on Albrecht data set. The comparison is based on MMRE and PRED(0.25). It is observed that the proposed model has significantly improved MLR by 237% while this percentage is 15% for OABE method. PRED(0.25) has also been substantially improved so that improvement of 300% and 13% is seen for MLR and OABE, respectively.

The box plot related to MRE results of estimation models is shown in Figure 9. According to the figure, the proposed model

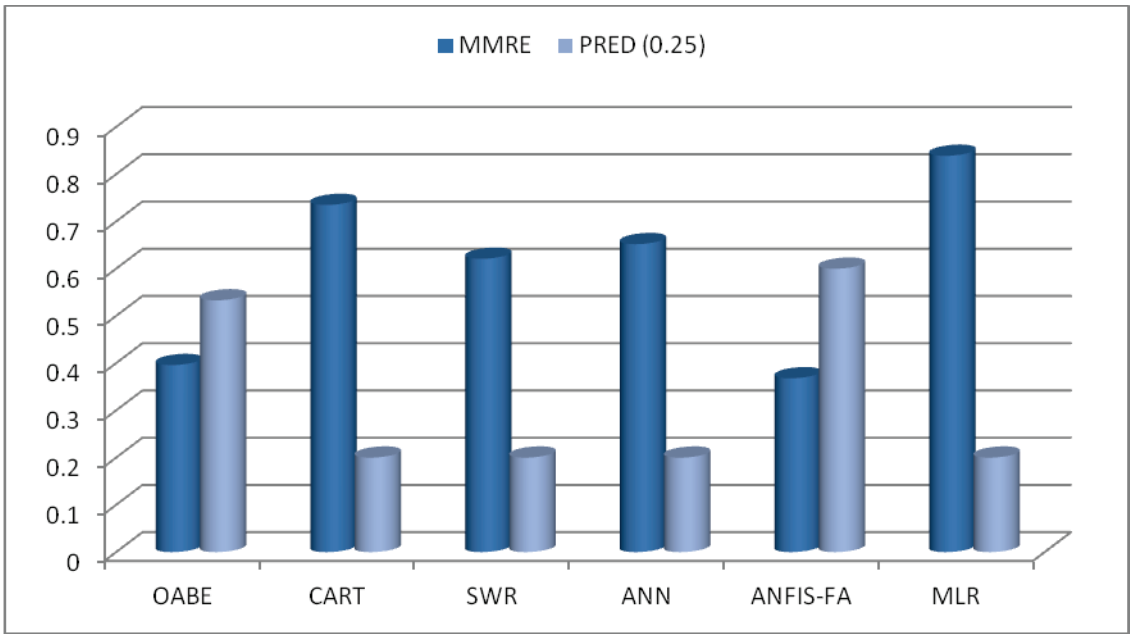

Figure 7 Values obtained from MMRE and PRED (0.25) of the Kemerer data set in comparison with other models.

Table 7 Results of the ISBSG data set in comparison with other models

PRED(0.25)	MMRE	method
0.208	0.927	MLR
0.167	0.986	SWR
0.221	1.03	ANN
0.154	1.13	CART
0.51	0.64	ABE-PSO
0.610	**0.557**	ANFIS-FA

Table 8 Results of the Kemerer data set in comparison with other models.

PRED(0.25)	MMRE	method
0.533	0.396	OABE
0.20	0.735	CART
0.20	0.621	SWR
0.20	0.652	ANN
0.60	**0.368**	ANFIS-FA
0.20	0.839	MLR

has lowest inter quartile range and median as compared to the other models. The MRE results are also statistically analyzed through Wilcoxon test as seen in Table 9. It is observed that majority of p-values are less than 0.05, which statistically approved the significant difference between the proposed model and other models.

The percentage of improvement achieved by the proposed model against the other models on ISBSG data set is shown in Figure 10 based on MMRE and PRED(0.25). It is seen that the proposed model generates more accurate results according to the improvement percentage of PRED(0.25). The highest and lowest improvement percentage is related to CART and ABE-PSO by 296% and 19%, respectively. It must be noted that the percentage improvement for PRED(0.25) is greater than that of MMRE but the superiority of the proposed model is totally confirmed on this data set. The percentage of MMRE improvement is 102% and 14% for CART and ABE-PSO, respectively, which are the highest and lowest values in this regard. The promising result is improvement of ABE-PSO for MMRE and PRED(0.25) by 14% and 19%, respectively.

The box plot of MRE results on ISBSG data set is shown in Figure 11. As seen in the figure, the proposed model has the lowest median and inter quartile range while the widest inter quartile range is related to SWR. The results of statistical analysis of MREs are presented in Table 10. It is observed that all the p-values are less than 0.05, which strongly confirms the difference of the proposed model and the other models in accuracy level.

The improvement percentage achieved by the proposed model against the other models on Kemerer data set is displayed in Figure 12 based on MMRE and PRED(0.25). According to the figure, the lowest and greatest improvement percentage of MMRE is related to OABE and MLR by 7% and 127%, respectively. In terms of PRED(0.25), SWR, ANN and CART are improved by 200% while the percentage of improvement for OABE is 12%.

The MRE result analysis of Kemerer data set in form of box plot is drawn in Figure 13. However the proposed model has the lowest median, CART achieves the lowest inter quartile range. The statistical analysis of MRE results is presented in Table 11. According to the table, only two p-values are less than 0.05. Therefore, the superiority of the proposed model is statistically confirmed.

In total, analysis of improvements achieved by the proposed model shows that it has the capability of providing reliable estimation results in all the three data sets.

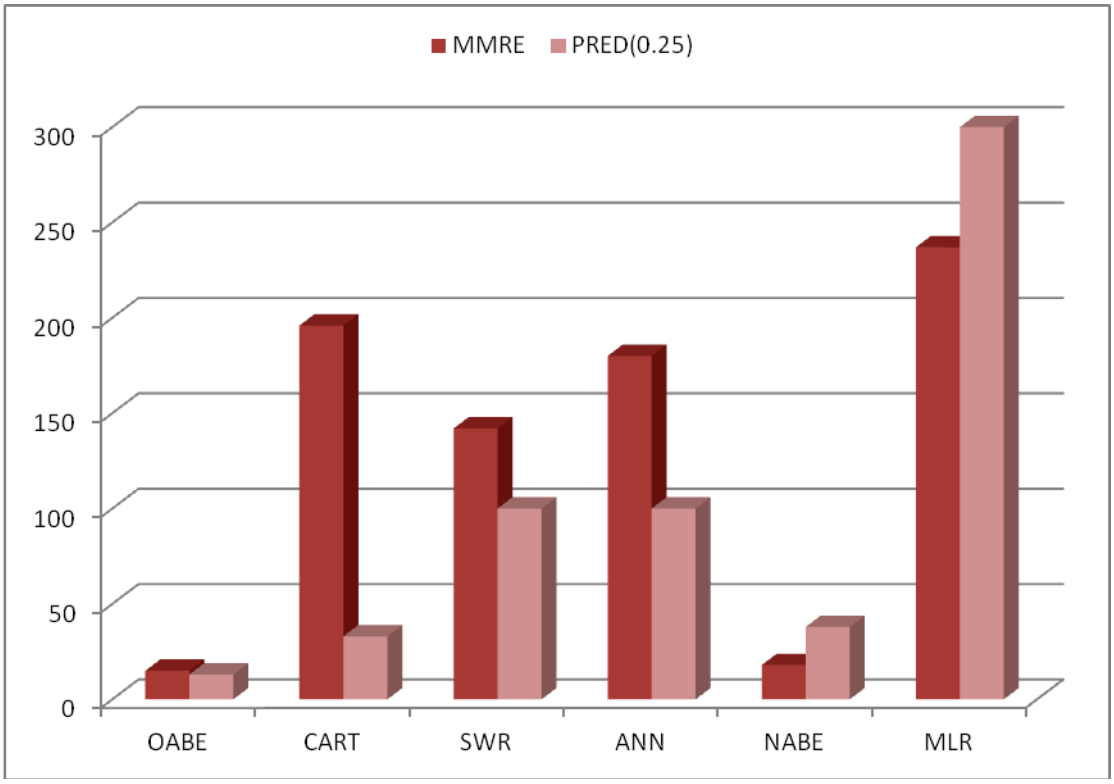

Figure 8 Improvement percentage obtained by applying the proposed method to Albrecht data set (MMRE and PRED (0.25)).

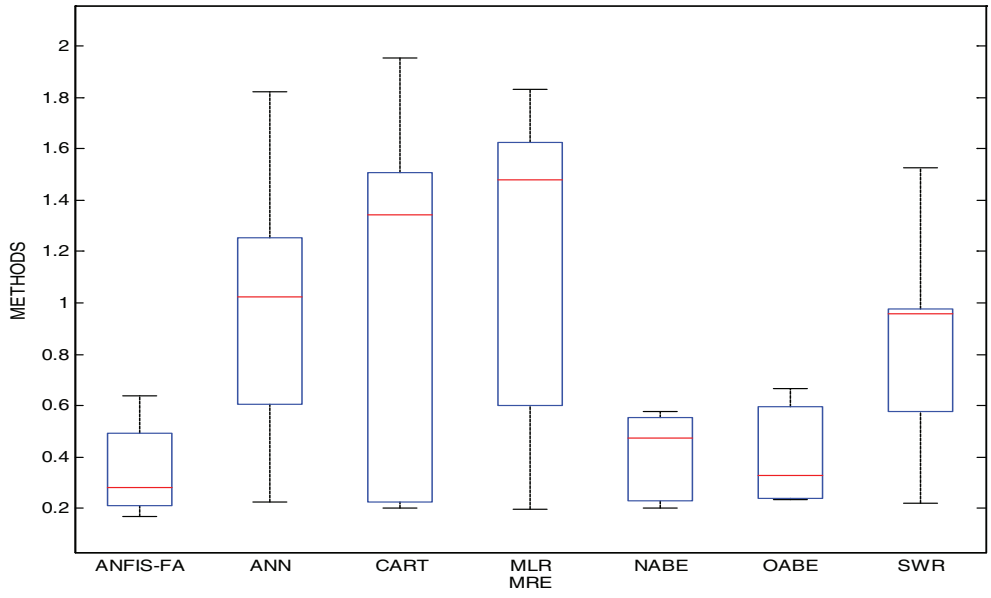

Figure 9 Box plot on MRE results obtained from Albrecht.

Table 9 P-values of Wilcoxon test (Albrecht. other models).

ANN	CART	MLR	NABE	SWR	OABE
0.020	0.007	0.014	0.38	0.020	0.23

11. CONCLUSION

In order to improve the performance of ANFIS in the field of software development effort estimation, modern meta-heuristic algorithms can be used instead of classic training algorithms. This paper focused on the use of ANFIS combined by FA as a novel training algorithm. The proposed model consists of two

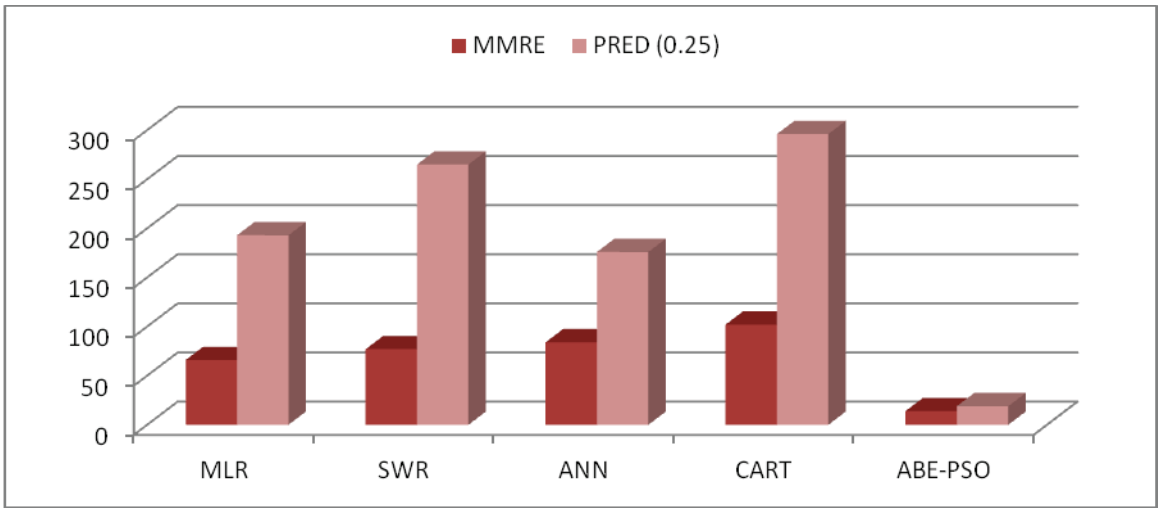

Figure 10 Improvement percentage obtained by applying the proposed method to ISBSG data set (MMRE and PRED (0.25)).

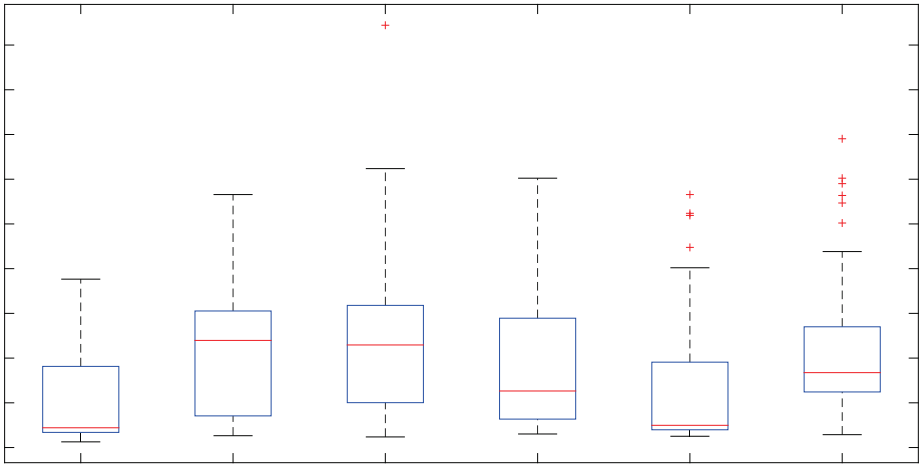

Figure 11 Box plot on MRE results obtained from ISBSG.

Table 10 P-values of Wilcoxon test (ISBSG. other models).

ANN	CART	ABE-PSO	MLR	SWR
6.98E-15	3.07E-15	0.004	2.39E-11	1.96E-12

Table 11 P-values of Wilcoxon test (kemerer. other models).

ANN	CART	MLR	SWR	OABE
0.095	0.0317	0.055	0.015	0.069

stages in which the model is constructed and evaluated. The role of FA in the proposed model is generation of the optimized parameters for ANFIS. In fact, by choosing the best parameters generated by FA, the ANFIS performance favorably increased. To evaluate the performance of the proposed method, Albrecht, ISBSG and Kemerer data sets were employed along with performance metrics of MMRE and PRED (0.25). The obtained results were compared with those reported by popular models. Based on the obtained results, the lowest MMRE and the highest PRED (0.25) in three data sets have been related to the proposed model. In fact, the combination of FA and ANFIS is able to provide the best performance among the existing models. Finally, statistical analysis of results supported the superiority of the proposed model against the other models. For the future study, new optimization algorithms and feature selection will be used to improve the accuracy of software development effort estimation.

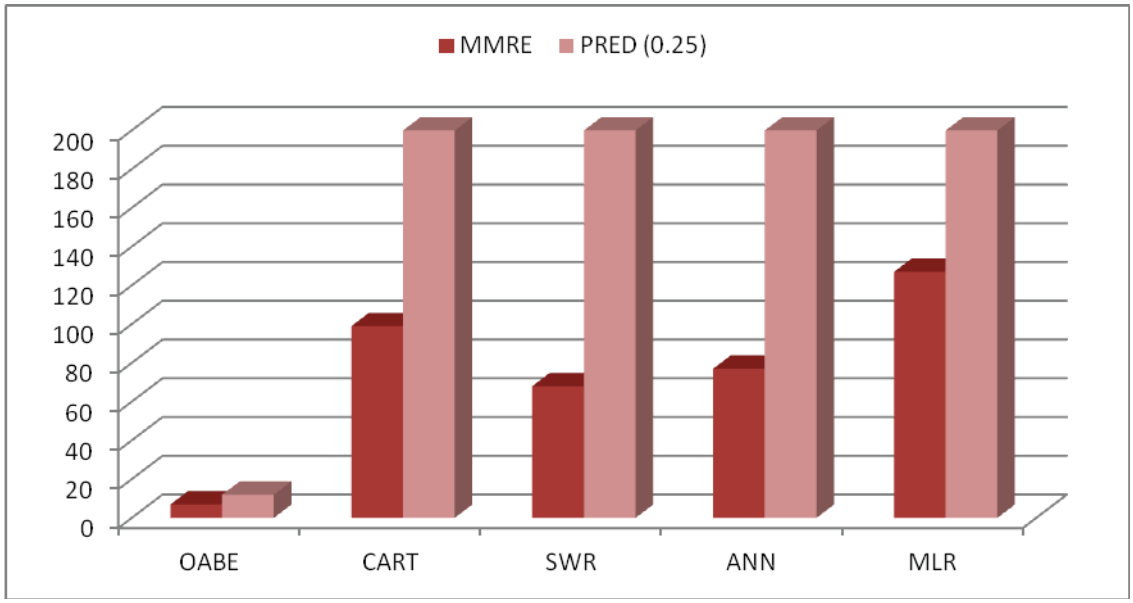

Figure 12 Improvement percentage obtained by applying the proposed method to Kemerer data set (MMRE and PRED (0.25)).

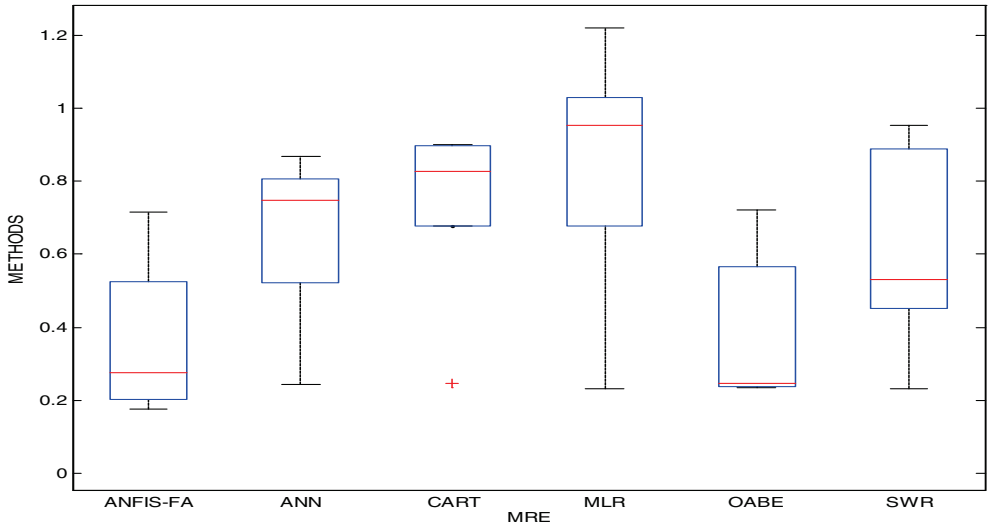

Figure 13 Box plot on MRE results obtained from Kemerer.

REFERENCES

1. M. Jorgensen, M.J. Shepperd, 2007, A systematic review of software development cost estimation studies, IEEE Transactions on Software Engineering 33 (1), 33–53.
2. S.G. MacDonell, M.J. Shepperd, 2003, Combining techniques to optimize effort predictions in software project management, Journal of Systems and Software 66 (2), 91–98.
3. N. Mittas, L. Angelis, 2008, Combining regression and estimation by analogy in a semi-parametric model for software cost estimation, in: Proc. Second ACMIEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '08), pp. 70–79.
4. Boehm, B., Abts, C., Chulani, S., 2000. Software development cost estimation approaches – a survey. Annals of Software Engineering 10, 177–205.
5. Boehm, B., 1981. Software Engineering Economics. Prentice-Hall, Englewood Cliffs, NJ.
6. Huang, X.S., Ho, D., Ren, J., Capretz, L.F., 2007. Improving the COCOMO model using a neuro-fuzzy approach. Applied Soft Computing Journal 7 (1), 29–40.
7. Putnam, L., Myers, W., 1992. Measures for Excellence. Yourdon Press Computing Series.
8. Jensen, R., 1983. An improved macrolevel software development resource estimation model. In: Proceedings of 5th Conference of International S Parametric Analysts, pp. 88-92.
9. Helmer, O., 1966. Social Technology. Basic Books, NY.
10. Tausworthe, R.C., 1980. The work breakdown structure in software project management. Journal of Systems and Software 1 (3), 181–186.
11. Jorgensen, M., 2004. Top-down and bottom-up expert estimation of software development effort. Information and Software Technology 46, 3–16.
12. Heiat, A., 2002. Comparison of artificial neural network and regression models for estimating software development effort. Information and Software Technology 44, 911–922.
13. Shin, M., Goel, A.L., 2000. Empirical data modeling in software engineering using radial basis functions. IEEE Transactions on

Software Engineering 26 (6), 567–576.
14. Oliveira, A.L.I., 2006. Estimation of software project effort with support vector regression. Neurocomputing 69, 1749–1753.
15. Shepperd, M., Schofield, C., 1997. Estimating software project effort using analogies. IEEE Transactions on Software Engineering 23 (12), 733–743.
16. Auer, M., Trendowicz, A., Graser, B., Haunschmid, E., Biffl, S., 2006. Optimal project feature weights in analogy-based cost estimation: improvement and limitations. IEEE Transactions on Software Engineering 32 (2), 83–92.
17. Huang, S.J., Chiu, N.H., 2006. Optimization of analogy weights by genetic algorithm for software effort estimation. Information and Software Technology 48, 1034– 1045.
18. Mendes, E., Mosley, N., Counsell, S., 2005. Investigating Web size metrics for early Web cost estimation. Journal of Systems and Software 77 (2), 157–172.
19. Costagliola, G., Ferrucci, F., Tortora, G., Vitiello, G., 2005. Class point: an approach for the size estimation of object-oriented systems. IEEE Transactions on Software Engineering 31 (1), 52–74.
20. Miyazaki, Y., Terakado, K., Ozaki, K., Nozaki, H., 1994. Robust regression for developing software estimation models. Journal of Systems and Software 27, 3–16.
21. Madachy, R., 1994. A Software Project Dynamics Model for Process Cost, Schedule and Risk Assessment, Ph.D. Dissertation, University of Southern California.
22. Chulani, S., Boehm, B., Steece, B., 1999. Bayesian analysis of empirical software engineering cost models. IEEE Transactions on Software Engineering 25 (4), 573–583.
23. MacDonell, S.G., Shepperd, M.J., 2003. Combining techniques to optimize effort predictions in software project management. Journal of Systems and Software 66, 91–98.
24. *Qingyuan Zhou and Jianjian Luo*, 2015, Artificial neural network based grid computing of e-governance scheduling for emergency management. International Journal of Computer Systems Science and Engineering, 30 (5).
25. *Xiaowei Zheng, Jiaxuan Li, Yong Zhang and Qingkun Liu*. 2016, An optimization model of Hadoop cluster performance prediction based on Markov process. International Journal of Computer Systems Science and Engineering. 31 (2).
26. cKhanna H Nehemiah, Angelin Gladston and A Kannan, 2016, An optimal tabu prioritization algorithm for regression testing. International Journal of Computer Systems Science and Engineering. 31 (5).
27. *Rawaa Dawoud Al-Dabbagh, Saad Mekhilef and Mohd Sapiyan Baba*. 2015, Parameters' fine tuning of differential evolution algorithm. International Journal of Computer Systems Science and Engineering. 30 (2).
28. *A. M. Kalpana, K. Tamizarasu and A. Ebenezer Jeyakumar*. 2014, A fuzzy logic based framework for assessing the maturity level of Indian small scale software organizations. International Journal of Computer Systems Science and Engineering. 29 (2).
29. Ziauddin, Shahid Kamal, Shafiullah khan and Jamal Abdul Nasir. 2013, A Fuzzy Logic Based Software Cost Estimation Model. International Journal of Software Engineering and Its Applications (IJSEIA). 7. Issue 2.
30. Vishal Sharma and Harsh Kumar Verma. 2010, Optimized Fuzzy Logic Based Framework for Effort Estimation in Software Development. International Journal of Computer Science Issues (IJCSI). 7. Issue 2. No 2.
31. Abeer Hamdy. 2012, Fuzzy Logic for Enhancing the Sensitivity of COCOMO Cost Model. Journal of Emerging Trends in Computing and Information Sciences. 3(9).
32. Azzeh, M., Neagu, D., Cowling, P., 2009. Fuzzy grey relational analysis for software effort estimation. Journal of Empirical Software Engineering, doi:10.1007/s10664-009-9113-0.
33. Idri, A., Abran, A., Khoshgoftaar, T., 2001. Fuzzy analogy: anewapproach for software effort estimation. In: 11th International Workshop in Software Measurements, pp. 93–101.
34. Musflek, P., Pedrycz, W., Succi, G., Reforment, M., 2000. Software cost estimation with Fuzzy models. Applied Computing Review 8, 24–29.
35. Kanmani S, Kathiravan J, Kumar SS, Shanmugam M, 2008, Class point based effort estimation of OO systems using fuzzy subtractive clustering and ANNs. In Proceedings of the 1st India software engineering conference, ISEC '08, ACM, pp 141–142, New York, NY, USA.
36. Fei Z, Liu X, 1992, f-COCOMO: fuzzy constructive cost model in software engineering. In: IEEE international conference on fuzzy systems, pp 331–337.
37. Roger JS,1993, ANFIS: adaptive network based fuzzy inference systems. IEEE Trans Syst Man Cybern 23(3):665–685.
38. Xua Z, Khoshgoftaar TM, 2004, Identification of fuzzy models of software cost estimation. Fuzzy Sets Syst 145(1):141–163.
39. Jang, J.-S.R. 1993, ANFIS: Adaptive network based fuzzy inference system. IEEE Transactions on Systems Man and Cybernetics. 23 (3). pp 665-68.
40. Yang X.-S. 2009, Firefly Algorithms for Multimodal Optimization. In *Stochastic Algorithms: Foundations and Applications*. pp 169–178.
41. Khatibi Bardsiri Vahid . Dayang Norhayati Abang Jawawi, Siti Zaiton Mohd Hashim, Elham Khatibi. 2013, A PSO-based model to increase the accuracy of software development effort estimation. Software Qual. 21(3). Pp 501–526.
42. Khatibi Bardsiri Vahid. Amid Khatibi and Elham Khatibi. 2013, An Optimization-Based Method to Increase the Accuracy of Software Development Effort Estimation. *J. Basic. Appl. Sci. Res.* 3(2). pp 159-166.
43. Azzeh Mohammad. Yousef Elsheikh. Marwan Alseid. 2014, *An Optimized Analogy-Based Project Effort Estimation*. International Journal of Advanced Computer Science and Applications. 5. Pp 6–11.
44. ISBSG International software benchmark and standard group, Data CDRelease 10, www.isbsg.org, 2007.
45. A. Albrecht and J. Gaffney, 1983, Software function, source lines of code and development effort prediction: A software science validation. IEEE Transactions on Software Engineering. 9. Pp 639–648.
46. Kemerer, F. C. 1987. An empirical validation of software cost estimation models. Communications of the ACM, 30 (5), 416–429.
47. Li YF, Xie M. Goh TN . 2009, A study of the non-linear adjustment for analogy based software cost estimation. Empir Softw Eng 14(6). Pp 603–643.
48. Li YF. Xie M, Goh TN. 2009, A study of project selection and feature weighting for analogy based software cost estimation. J Syst Softw. 82(2). pp 241–252.
49. Mittas, N. & Angelis, L. 2010, LSEbA: Least squares regression and estimation by analogy in a semi parametric model for software cost estimation. Empirical Software Engineering. 15(5). pp 523–555.
50. Hsu C-J, Huang C-Y. 2011, Comparison of weighted grey relational analysis for software effort estimation. Softw Qual J. 19(1). pp 165–200..
51. Khatibi Bardsiri, V., et al. 2013, A flexible method to estimate the software development effort based on the classification of projects and localization of comparisons. Empir Software Eng. 19(4). pp 857–884.