Engineering Applications of Artificial Intelligence 60 (2017) 1–15

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

# Satin bowerbird optimizer: A new optimization algorithm to optimize ANFIS for software development effort estimation

Seyyed Hamid Samareh Moosavi, Vahid Khatibi Bardsiri

*Department of Computer Engineering, Kerman Branch, Islamic Azad University, Kerman, Iran*

ARTICLE INFO

*Keywords:*
Development effort estimation
Adaptive Neuro-fuzzy inference system
Satin bowerbird optimization algorithm
Software project

ABSTRACT

Accurate software development effort estimation is crucial to efficient planning of software projects. Due to complex nature of software projects, development effort estimation has become a challenging issue which must be seriously considered at the early stages of project. Insufficient information and uncertain requirements are the main reasons behind unreliable estimations in this area. Although numerous effort estimation models have been proposed during the last decade, accuracy level is not satisfying enough. This paper presents a new model based on a combination of adaptive neuro-fuzzy inference system (ANFIS) and satin bower bird optimization algorithm (SBO) to reach more accurate software development effort estimations. SBO is a novel optimization algorithm proposed to adjust the components of ANFIS through applying small and reasonable changes in variables. The proposed hybrid model is an optimized neuro-fuzzy based estimation model which is capable of producing accurate estimations in a wide range of software projects. The proposed optimization algorithm is compared against other bio inspired optimization algorithms using 13 standard test functions including unimodal and multimodal functions. Moreover, the proposed hybrid model is evaluated using three real data sets. Results show that the proposed model can significantly improve the performance metrics.

## 1. Introduction

Software development effort estimation has always been a challenge for software engineering communities. Accurate and reliable software effort estimation is important for allocating resources and creating a reasonable schedule during software project planning. Project planning is a crucial activity in software projects which significantly affects the success or failure of project. One of the most important aspects of project planning is accurate estimation of effort required to complete the project. Underestimating of software project effort causes delay and cost over-run, which can lead to project failure. Conversely, over-estimating can also be detrimental for effective utilization of project resources. Therefore, during the last decade, a number of software effort estimation methods have been developed using different theoretical concepts (Jorgensen and Shepperd, 2007) and combining existing estimation methods (MacDonell and Shepperd, 2003a; Mittas and Angelis, 2008).

Jorgensen and Shepperd conducted a systematic review in which 11 estimation approaches were identified from 304 selected journal papers (Jorgensen and Shepperd, 2007). These approaches fall into two major categories: parametric models, which are derived from the statistical and/or numerical analysis of historical project data, and machine learning models, which are based on a set of artificial intelligence techniques such as artificial neural networks, genetic algorithms, analogy-based or case-based reasoning, decision trees, and genetic programming. Machine learning techniques are attracted the attention of software effort estimation researchers, as they can model the complex relationship between effort and software attributes, especially when this relationship is not linear and does not seem to have any predetermined form. The importance of accurate effort estimation has led to extensive research efforts in this field. The current methods can be classified into the following categories (Boehm et al., 2000):

1. Parametric models: COCOMO (Huang et al., 2007), SLIM (Putnam and Myers, 1992) and SEER-SEM (Jensen, 1983).
2. Expert judgment: Delphi technique and work breakdown structure based methods (Jorgensen, 2004).
3. Learning oriented techniques: machine learning methods (Oliveira, 2006) and analogy based estimation (Shepperd and Schofield, 1997).
4. Regression based methods: ordinary least square regression and robust regression (Costagliola et al., 2005).
5. Dynamics based models (Madachy, 1994).
6. Composite methods (Chulani et al., 1999; MacDonell and Shepperd, 2003b).

*E-mail addresses:* hamid.moosavi17@gmail.com (S.H. Samareh Moosavi), kvahid2@live.utm.my (V. Khatibi Bardsiri).

http://dx.doi.org/10.1016/j.engappai.2017.01.006
Received 6 May 2016; Received in revised form 14 December 2016; Accepted 10 January 2017

However prior models achieved improvements in estimation of effort, the attempts are ongoing to produce more accurate and reliable models. Due to complexity of effort estimation problem and difficulty of attributes relation analysis, the optimization process plays a vital role in this area. The optimization can directly be applied to effort estimation process like attribute weighting in analogy based estimation or indirectly applied to machine learning methods such as ANN and ANFIS.

One of the most common applications of neuro-fuzzy systems is producing rules for complex and uncertain problems (Pan et al., 2014, 2013; Li et al., 2015; Pratama et al., 2013). On the other hands, software projects are inherently uncertain and complex so that the available information is not enough at the early stage of project and the problem of effort estimation is completely vague. In this situation, fuzzy and neuro-fuzzy models can handle the uncertainty and increase the estimation accuracy (Trendowicz and Jeffery, 2014). Moreover, promising results have recently been reported from fuzzy-based models applied to the field of software effort estimation (Idri et al., 2016; Satapathy et al., 2016a). Therefore, the model proposed in this paper is constructed using a neuro-fuzzy system optimized by a new optimization algorithm.

In recent years, different optimization algorithms have been presented such as gray wolf optimization (GWO) (Mirjalili et al., 2014), artificial bees colony (ABC) (Karaboga and Basturk, 2007), cuckoo search (CS) (Yang and Deb, 2009, 2010), dolphin echolocation (DE) (Kaveh and Farhoudi, 2013; Kaveh, 2014), ray optimization (RO) (Kaveh et al., 2013; Kaveh and Khayatazad, 2013), krill herds (KH) (Hossein and Hossein-Alavi, 2012), biogeography-based optimization (BBO) (Simon, 2008), gravitational search algorithm (GSA) (Rashedi et al., 2009), artificial chemical reaction optimization algorithm (ACROA) (Alatas, 2012), coral reef optimization (CRO) algorithm (Salcedo-Sanz et al., 2013), charged system search (CSS) algorithm (Kaveh and Talatahari, 2010), central force optimization (CFO) (Formato, 2008), symbiotic organisms search (SOS) (Cheng and Prayogo, 2014), fire fly optimization algorithm (FA) (Yang, 2010a, 2010b). In spite of proposing so many optimization algorithms, the attempts are ongoing to propose new ones because the adaptability and optimization capability of algorithms are completely different. In this paper, a novel optimization algorithm is presented to solve the problem of effort estimation in software projects. The proposed algorithm is organized so that it is an appropriate choice for problems such as attribute weighting, ANN adjustment (weight and bias), ANFIS adjustment and problems which need small changes in position of variables. Moreover, increase in the number of variables does not affect the performance of the proposed algorithm.

In fact, this paper aims at making a link between software development effort estimation problem, ANFIS and SBO. Due to uncertainty, complexity and lack of information in software development effort estimation, ANFIS can be an appropriate choice to handle the problem. On the other hands, the structure of ANFIS can be optimized using an optimization algorithm. The proposed SBO algorithm is a suitable choice for this purpose. The reasons behind combining SBO and ANFIS is the ability of SBO to avoid from the trap of local optimum and its moderate changes applied to solutions, which makes SBO capable of adjusting ANFIS.

This paper is organized in 8 sections. Related work is presented in Section 2. In Section 3, satin bowerbird optimization algorithm is described in details. In Section 4, evaluation of the proposed algorithm is presenting using mathematical functions. ANFIS structure is presented in Section 5. SBO for software development effort estimation is presented in Section 6. The experimental results are presented in Section 7. Finally, the overall conclusion is presented in Section 8.

## 2. Related work

During the last decade, complexity of software development effort estimation has attracted the attention of researchers so that a wide domain of machine learning methods has been employed in this field. Most of existing estimation models in this area are benefited from supervised learning methods because various real data sets exist that include information related to software projects completed in the past. Optimization algorithms, neural networks and fuzzy methods have widely been utilized in the field of software development effort estimation. The application of optimization algorithms is mostly focused on adjusting the parameters of effort estimators such as ABE (a, 2013; Azzeh et al., 2015; Khatibi and Khatibi Bardsiri, 2015), COCOMO (Agrawal et al., 2016) and UCP (Kumari and Ishdeep, 2015; Satapathy et al., 2016b). The results reported from combining effort estimators and optimization algorithms indicate the significant improvement in estimation accuracy.

On the other hands, different types of neural network have extensively been employed to estimate the effort in software projects (Nassif et al., 2016). Due to complexity and uncertainty level of effort estimation problem, performance enhancement of neural network has been considered in prior studies (Rijwani and Jain, 2016; Hota et al., 2015; Azzeh and Nassif, 2016). Bias and weight tuning and structure configuration are the main improvements considered in the proposed estimation models.

Investigation of literature in the field of software development effort estimation shows that fuzzy and neuro-fuzzy systems play a vital role in increase of accuracy. The previous research studies based on these systems can be categorized into three main groups. The first group tries to increase the accuracy of COCOMO model through fuzzification of its cost drivers (Ziauddin et al., 2013; Vishal and Kumar Verma, 2010; Abeer, 2012; Sheenu et al., 2014; Kumar and Chopra, 2013; Xua and Khoshgoftaar, 2004). The second group of research studies are concentrated on classification of software projects to remedy the problem of inconsistency and facilitate the estimation process through assigning fuzzy rules to each class of projects (Agrawal and Shrivastava, 2015; Cuauhtemoc Lopez-Martin, 2010). The similarity function of ABE is improved by the fuzzy models proposed in the third group of studies (Idri et al., 2016; Azzeh et al., 2009, 2011). It must be noted that the similarity measurement is one of the most important steps in analogy based effort estimation and its improvement through fuzzy models significantly affects the final results.

Regarding the COCOMO improvement, a fuzzy logic based estimation model was introduced to increase the accuracy of software effort estimation. The main idea in this study is fuzzification of 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 (Ziauddin et al., 2013). A similar fuzzy framework was proposed to estimate the effort (Vishal and Kumar Verma, 2010). The main difference is that the proposed framework is optimized through meta-heuristic algorithms. The performance of the proposed framework is demonstrated by 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 (Vishal and Kumar Verma, 2010). Enhancing the sensitivity of COCOMO is the other idea followed by researchers in this field (Abeer, 2012). In this regard, the accuracy and sensitivity of intermediate COCOMO 81 are enhanced by fuzzification of the cost drivers. The dataset is collected from six NASA organizations and covers a wide range of software domains, development process, languages and complexity, as well as fundamental differences in culture and business practices among centres. The results showed that the sensitivity of the proposed fuzzy model is superior to intermediate COCOMO81 (Abeer, 2012).

Neuro-fuzzy based models have also been utilized in this field. For an instance, ANFIS was experienced to estimate the effort based on COCOMO model (Sheenu et al., 2014). In this study, the efficiency of the proposed ANFIS based model is evaluated by COCOMO81 datasets. The obtained results were compared with artificial neural network and intermediate COCOCMO model developed by Boehm. The comparison was analyzed using Magnitude of Relative Error (MRE) and Root Mean Square Error (RMSE). It is observed that the ANFIS provided better results than ANN and COCOMO model (Sheenu et al., 2014). In prior studies, a comparison has been conducted between neural network and fuzzy logic in term of effort estimation accuracy (Kumar and Chopra, 2013). NASA dataset was employed as the input for both frameworks. These frameworks are validated using the parameters of MMRE and PRED. The results show that the fuzzy logic based framework works better as compared to the neural network framework (Kumar and Chopra, 2013). Dealing with linguistic cost drivers of COCOMO is a challenging issue considered in research studies (Xua and Khoshgoftaar, 2004). Automatically generation of fuzzy membership functions is the main goal and proposed fuzzy based model outperforms the original organic, basic and detailed versions of COCOMO 81 (Xua and Khoshgoftaar, 2004).

Analogy based effort estimation has widely been studied and improved though machine learning techniques. Fuzzy method has also been employed to develop a new similarity measure based on integration of fuzzy set theory and gray relational analysis for analogy-based estimation (Azzeh et al., 2009). 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 obtained results suggested that the proposed model produces good accuracy as compared to other well Known estimation techniques such as ABE, stepwise regression and artificial neural network (Azzeh et al., 2009). In the field of ABE improvement, the ensemble of effort estimation based on classical and fuzzy analogy has recently been carried out (Idri et al., 2016). The evaluation shows that fuzzy analogy ensembles obtain better performance than classical analogy ensembles. In prior studies, a combination of analogy based estimation and fuzzy numbers has also been evaluated (Azzeh et al., 2011). A novel similarity function and an adaptation technique are achieved by the proposed combination and the results show the superiority of this combination compared with ABE and stepwise regression.

In terms of software projects classification and rules definition, a neuro-fuzzy based model was developed (Agrawal and Shrivastava, 2015). In this study, the estimated time for neuro-fuzzy model created for three membership functions is compared with the existing neural network models. The neuro-fuzzy model for Gaussian, Triangular and Trapezoidal membership functions is compared with the neural network models. For this experiment, Lopez Martin dataset is used with 41 modules. Finally it is observed from the comparison that the neuro-fuzzy model using Trapezoidal membership function gives better results than all the other models. It is also observed that Trapezoidal MF gives better results for all the five parameters (Agrawal and Shrivastava, 2015). Moreover, this technique has been employed to estimate the effort particularly in small programs. New, changed and reused codes from small programs developed by 74 programmers are employed as the input for the fuzzy model. The accuracy of output was compared with that of statistical regression model using the comparison criterion Mean Magnitude Error Relative to the estimate (Cuauhtemoc Lopez-Martin, 2010).

In total, prior studies in the field of software development effort estimation have widely utilized intelligent methods to improve the estimations. This is because of the complicated relationship among software project features which makes the estimation quite difficult. Due to analysis capability and flexibility of fuzzy and neuro-fuzzy systems, they can be suitable solutions in this area. However, the application of these systems has mostly been restricted to COCOMO model in previous research studies. COCOMO is a parametric estimation model and there are several standard data sets including software projects having COCOMO format. Therefore, fuzzy rules definition is simply performed in this model. In order to widen the application of fuzzy systems in the field of software development effort estimation, a neuro-fuzzy system is optimized in this paper. The optimization process is carried out using a novel optimization algorithm so that the proposed model can be employed in a wide range of software project data sets.

## 3. Satin bower bird optimization algorithm

Complex and multifaceted displays in the context of mating are common, especially in polygynous species and birds (Møller and Pomiankowski, 1993). These displays can involve auditory, chemical, visual or tactile components and may provide searching females with various types of information including species, sex, individual identity, relatedness or quality (Boogert et al., 2008).

Bowerbirds are close relatives of the birds of paradise. Satin bowerbirds are fruit and insect-eating passerines native to mesic forests and rainforest in eastern Australia. Mainly occurring in forests, they live in a particular local area throughout their lives. During autumn and winter, satin bowerbirds leave their forest habitat and move into open woodlands to forage for fruit and insects. However, with the arrival of the spring breeding season they collect together in small groups, inhabiting territories which they apparently occupy year after year.

Male satin bowerbirds build specialized stick structures, called bowers, where courtship and copulation take place. Bowers are decorated with flowers, feathers, berries etc. These decorations are important in female choice and male mating success (Coleman et al., 2004). Males compete by stealing decorations from other males and will destroy the bowers of neighbors (Borgia, 1985). Male courtship behavior includes presentation of decorations and dancing displays accompanied by loud vocalizations, and females prefer males that display at high intensity. Another indication of strong sexual competition is that not all adult males are successful at constructing, maintaining and defending bowers, and as a result, there is considerable variability in male mating success. In simple terms, male bowerbirds attract mates by constructing a bower, a structure built from sticks and twigs, and decorating the surrounding area. Females visit several bowers before choosing a mating partner and returning to his bower.

In SBO algorithm, adult males begin to build bower with different materials on their land during mating season. They utilize a variety of materials such as flowers, fruits, sparkling objects, branches as well as dramatic gestures that are all part of variables to attract females. Females due to the bower beauty and dramatic gestures of males are attracting to bower. It is important to note that male birds use their natural instinct and imitation of other males for building the bower. According to the principles of stain bower bird life, SBO algorithm is organized in the several stages as follows.

### 3.1. Generating a set of random bower

Similar to other meta-heuristic algorithms, SBO algorithm begins with creating an initial population randomly. In fact, initial population includes a set of positions for bowers. Each position is defined as an n-dimensional vector of parameters that must be optimized. These values are randomly initialized so that a uniform distribution is considered between the lower and upper limit parameters. Indeed, the parameters of each bower are the same as variables in the optimization problem. The combination of parameters determines the attractiveness of bower.

### 3.2. Calculating the probability of each population member

The probability is the attractiveness of a bower. A female satin bower bird selects a bower (built) based on its probability. Similarly, a

male mimics bower building through selecting a bower based on the probability assigned to it. This probability is calculated by Eq. (1). In this equation, $fit_i$ is fitness of ith solution and NB is the number of bower. In this equation, the value of $fit_i$ is achieved by Eq. (2).

$$Prob_i=\frac{fit_i}{\sum_{n=1}^{NB} fit_n} \tag{1}$$

$$fit_i=\begin{cases} \frac{1}{1+f(x_i)}, f(x_i)\geq 0 \\ 1+|f(x_i)| \quad , f(x_i)<0 \end{cases} \tag{2}$$

In this equation, $f(x_i)$ is the value of cost function in ith position or ith bower. The cost function is a function that should be optimized.

Eq. (2) has two parts. The first part calculates the final fitness where values are greater than or equal to zero, while the second part calculates the fitness for values less than zero. This equation has two main characteristics:

1. For $f(x_i)=0$ both parts of this equation have fitness value of one.
2. Fitness is always a positive value.

### 3.3. Elitism

Elitism is one of the important features of evolutionary algorithms. Elitism allows the best solution (solutions) to be preserved at every stage of the optimization process. All the birds normally build their nests using their natural instincts. The male satin bower bird likes all other birds in the mating season and uses his natural instinct to build his bower and decorate it. This means that all males use materials to decorate their bower. However, an important factor to attract more attention to the bower of a particular male is experience. This experience helps a lot in both the dramatic gestures and building bower. This means that older males can attract more attention of others to their bower. In other words, experienced males build a better bower so these bowers have more fitness than the other bowers. In this work, the position of the best bower built by birds (best position) is intended as elite of iteration. Since the position of elite has the highest fitness, it should be able to influence the other positions.

### 3.4. Determining new changes in any position

In each cycle of the algorithm, new changes at any bower are calculated according to Eq. (3).

$$x_{ik}^{new}=x_{ik}^{old}+\lambda_k\left(\left(\frac{x_{jk}+x_{elite,k}}{2}\right)-x_{ik}^{old}\right) \tag{3}$$

In this equation, $x_i$ is ith bower or solution vector and $x_{ik}$ is kth member of this vector. $x_j$ is determined as the target solution among all solutions in the current iteration. In Eq. (3), value j is calculated based on probabilities derived from positions. In fact, the value j is calculated by the roulette wheel procedure, which means that the solution having larger probability will have more chance to be selected as $x_j$. $x_{elite}$ indicates the position of elite, which is saved in each cycle of the algorithm. In fact, position of elite is the position of a bower whose fitness is the highest in current iteration.

Parameter $\lambda_k$ determines the attraction power in the goal bower. $\lambda_k$ determines the amount of step which is calculated for each variable. This parameter is determined by Eq. (4).

$$\lambda_k=\frac{\alpha}{1+p_j} \tag{4}$$

In Eq. (4), α is the greatest step size and $p_j$ is the probability obtained by Eq. (1) using the goal bower. Since the obtained probability values are between 0 and 1, the denominator of this equation is collected by 1 to avoid 0 in the denominator of the Eq. (4). As is obvious from Eq. (4), the step size is inversely proportional to the probability of target position. In other words, when probability of the target position is greater (due to the constant α), movement to that position is more carefully done. The highest step size occurs when the probability of the target position is 0 while the step size will be α. On the other hands, the lowest step size occurs when the probability of target position is 1 and the step size will be α/2.

### 3.5. Mutation

When males are busy by building a bower on the ground, they may be attacked by other animals or be completely ignored. In many cases, stronger males steal materials from weaker males or even destroy their bower. Therefore, at the end of each cycle of the algorithm, random changes are applied with a certain probability. In this step, random changes are applied to $x_{ik}$ with a certain probability. Here, for mutation process, a normal distribution (N) is employed with average of $x_{ik}^{old}$ and variance of $\sigma^2$, as seen in Eq. (5).

$$x_{ik}^{new}\sim N(x_{ik}^{old}, \sigma^2) \tag{5}$$

$$N(x_{ik}^{old}, \sigma^2)=x_{ik}^{old}+(\sigma * N(0, 1)) \tag{6}$$

In Eq. (6), the value of σ is a proportion of space width, as calculated in Eq. (7):

$$\sigma=z*(var_{max}-var_{min}) \tag{7}$$

In Eq. (7), $var_{max}$ and $var_{min}$ are upper bound and lower bound assigned to variables, respectively. Z parameter is percent of the difference between the upper and lower limit which is variable.

### 3.6. Combining old population and the population obtained from changes

In this stage, at the end of each cycle, the old population and the population obtained from changes are evaluated. After the evaluation, these two populations are combined and sorted (based on the values obtained from the cost function) and the new population is created according to the previously defined number while the others are deleted. In this algorithm, steps 3.2 through 3.5 continue in the event of not meeting the termination conditions. Pseudo-code of SBO algorithm is designed as follows:

```
Initialize the first population of bowers randomly
Calculate the cost of bowers
Find the best bower and assume it as elite
While the end criterion is not satisfied
Calculate the probability of bowers using Eqs. (1) and (2)
For every bower
   For every element of bower
   Select a bower using roulette wheel
   Calculate λk using Eq. (4)
   Update the position of bower using Eqs. (3) and (6)
   End for
End for
Calculate the cost of all bowers
Update elite if a bower becomes fitter than the elite
End while
Return best bower
```

## 4. Evaluating the performance of SBO with mathematical functions

In this section, the performance of the proposed algorithm is evaluated using 13 well-known mathematical functions. These functions are widely used to evaluate the optimization algorithms (Yao et al., 1999; Digalakis and Margaritis, 2001). The functions are divided

**Table 1**
Unimodal functions.

| Function | Dim | Range | $f_{min}$ |
|---|---|---|---|
| $F_1(X)=\sum_{i=1}^{n} X_i^2$ | 30 | [−100,100] | 0 |
| $F_2(X)=\sum_{i=1}^{n} \lvert X_i\rvert+\prod_{i=1}^{n} \lvert X_i\rvert$ | 30 | [−10,10] | 0 |
| $F_3(X)=\sum_{i=1}^{n}\left(\sum_{j-1}^{i} X_j\right)^2$ | 30 | [−100,100] | 0 |
| $F_4(X)=\max\{\lvert X_i\rvert,\ \ 1 \le i \le n\}$ | 30 | [−100,100] | 0 |
| $F_5(X)=\sum_{i=1}^{n-1} [100(X_{i+1}-X_i^2)^2+(X_i-1)^2]$ | 30 | [−30,30] | 0 |
| $F_6(X)=\sum_{i=1}^{n} ([X_i+0.\ 5])^2$ | 30 | [−100,100] | 0 |
| $F_7(X)=\sum_{i=1}^{n} iX_i^4+\text{random}[0,\ 1)$ | 30 | [−1.28,1.28] | 0 |

into two categories: unimodal and multimodal. Unimodal functions have a global optimum and have no local optimum. In fact, this type of functions is suitable for exploitation capability of algorithm. These functions are shown in Table 1. Multimodal functions have a wide number of local optimum. This type of functions is useful to assess the exploration and local optimum avoidance capability of algorithm. These functions are shown in Table 2.

To evaluate the performance of the SBO algorithm, the results obtained by this algorithm are compared with particle swarm optimization algorithm (PSO) (one of the best among swarm based algorithms) (Kennedy and Eberhart, 1995), and genetic algorithm (GA) (one of the best among evolutionary algorithms) (Holland, 1992; Holland and Reitman, 1977). In addition, the SBO algorithm is compared with several recent algorithms like ant lion optimizer (ALO) (Mirjalili, 2015), FA and ABC. In fact, the SBO algorithm is compared with the latest version of the well-known algorithms. Each algorithm is executed on test functions for 30 times. Every test function is solved with 30 solutions on the 1000 iteration. The parameter setting for each algorithm is described as follows:

1. SBO: Parameter values of α and z are considered, 0.94 and 0.02, respectively. The mutation probability is 0.05. Maximum iteration =1000 and initial population =50.
2. PSO (Kennedy and Eberhart, 1995): $C_1$=2, $C_2$=2, W=1. Maximum iteration =1000 and initial population =50.
3. ABC (Karaboga and Basturk, 2007): limit=100. Maximum iteration =1000 and initial population =50.
4. ALO (Mirjalili, 2015): Maximum iteration =1000 and initial population =50.
5. FA (Yang, 2010a; Yang, 2010b): $\beta_0$=1, α ϵ [0,1], γ=1. Maximum iteration =1000 and initial population =50.
6. GA (Holland, 1992; Holland and Reitman, 1977): roulette wheel selection, single point crossover with a crossover probability of 0.9, mutation probability of 0.01. Maximum iteration =1000 and initial population =50.

Mean and standard deviation are calculated as the final result of SBO at the end of 30 times of running. A statistical test is required to determine the mean and standard deviation of the overall performance of SBO. The statistical test helps to confirm the significance of differences between the results achieved by different algorithms. In this paper, a non-parametric statistical test called Wilcoxon test rank sum (Derrac et al., 2011) is employed. The statistical test has an output parameter called p-value by which the significance level of two algorithms is determined. In this paper, an algorithm is statistically different if and only if its result in p-value is less than 0.05.

### 4.1. The results on unimodal test functions

In Table 3, the results of optimization algorithms on 7 unimodal test function are shown. As the results show, the SBO algorithm outperforms the other algorithms in most cases. In other words, the SBO algorithm has the most accurate results in four test functions of F3, F4, F5 and F7. The PSO algorithm obtained the best results from three other test functions. SBO achieved a very competitive result to PSO in function F3. Values obtained from the parameter of P are shown in Table 4. It is observed that p-values are less than 0.05 in most cases, which shows statistically superiority of SBO in comparison with the other algorithms. In Fig. 1, the convergence process of algorithms is shown on several unimodal test functions. As seen in the figure, exploitation of SBO (according to the results of test functions) makes the convergence process faster than the other algorithms.

### 4.2. The results on multi-modal test functions

The results obtained from optimization algorithms on 6 multi-modal test functions are shown in Table 5. As it is clear from the results, SBO algorithm is able to achieve the best results for functions F8, F9 and F11 in comparison to the other algorithms. In Table 6, the results obtained from parameter of P are shown. According to the results, it can be concluded that the SBO algorithm is statistically superior in comparison to the other algorithms. In Fig. 2, the convergence process of algorithms is shown on several multi-modal test functions. It is observed that SBO is converged in a fewer number of iterations in comparison to the other algorithms. Since SBO has high exploration level, this algorithm can avoid from the local optimum and move to the overall optimum in multimodal test functions.

According to the overall results obtained from the algorithms on 13 test functions, it can be concluded that the SBO algorithm is able to

**Table 2**
Multimodal functions.

| Function | Dim | Range | $f_{min}$ |
|---|---|---|---|
| $F_8(X)=\sum_{i=1}^{n} -X_i \sin(\sqrt{\lvert X_i\rvert})$ | 30 | [−500,500] | −418*Dim |
| $F_9(X)=\sum_{i=1}^{n} [X_i^2-10\cos(2\pi X_i)+10]$ | 30 | [−5.12,5.12] | 0 |
| $F_{10}(X)=-20\exp\left(-0.\ 2\sqrt{\frac{1}{n}\sum_{i=1}^{n} X_i^2}\right)\ -\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi X_i)\right)+20+e$ | 30 | [−32,32] | 0 |
| $F_{11}(x)=\frac{1}{4000}\sum_{i=1}^{n} X_i^2-\prod_{i=1}^{n}\cos\frac{X_i}{\sqrt{i}}+1$ | 30 | [−600,600] | 0 |
| $F_{12}(X)=\frac{\pi}{n}\{10\sin(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_n-1)^2\}+\sum_{i=1}^{n} u(X_i,10,\ 100,\ 4)$ $y_i=1+\frac{X_i+1}{4}$ $u(X_i,a,\ k,\ m)=\begin{cases} k(X_i-a)^m x_i > a \\ 0 \quad -a < x_i < a \\ k(-X_i-a)^m x_i < a \end{cases}$ | 30 | [−50,50] | 0 |
| $F_{13}(x)=0.\ 1\{\sin^2(3\pi x_1)+\sum_{i=1}^{n}(x_i-1)^2[1+\sin^2(3\pi x_i+1)]+(x_n-1)^2[1+\sin^2(2\pi x_n)]\}+\sum_{i=1}^{n} u(X_i,5,\ 100,\ 4)$ | 30 | [−50,50] | 0 |

**Table 3**
Results of optimization algorithms on unimodal test function.

| Function | | SBO | ALO | FA | ABC | PSO | GA |
|---|---|---|---|---|---|---|---|
| **F1** | **ave** | 0.00337 | 4.60E−7 | 1.47E+4 | 4.44 | **6.26E−14** | 0.146 |
| | **std** | 0.0016 | 4.61E−7 | 1.94E+3 | 3.2672 | 1.62E−13 | 0.089 |
| **F2** | **ave** | 0.0146 | 34.56 | 99.72 | 9.68E+1 | **2.07E−4** | 0.077 |
| | **std** | 0.0037 | 47.42 | 24.49 | 22.600 | 3.06E−4 | 0.014 |
| **F3** | **ave** | **2.09E+2** | 2.67E+2 | 1.99E+4 | 5.93E+4 | 2.16E+2 | 1.85E+3 |
| | **std** | 55.33 | 116.84 | 4.37E+3 | 9.51E+3 | 54.74 | 740.60 |
| **F4** | **ave** | **0.262** | 7.84 | 48.26 | 55.84 | 2.81 | 2.06 |
| | **std** | 0.0395 | 3.94 | 5.177 | 4.68 | 0.67 | 0.25 |
| **F5** | **ave** | **85.44** | 273.75 | 6.05E+6 | 1.25E+6 | 92.07 | 99.89 |
| | **std** | 67.39 | 443.74 | 3.16E+6 | 7.33E+5 | 93.07 | 39.65 |
| **F6** | **ave** | 0.0028 | 6.69E−7 | 1.55E+4 | 3.505 | **5.42E−18** | 0.132 |
| | **std** | 0.0012 | 4.30E−7 | 3.43E+4 | 2.168 | 5.78E−18 | 0.071 |
| **F7** | **ave** | **0.0037** | 0.0465 | 2.271 | 0.377 | 0.013 | 0.036 |
| | **std** | 8.116E−4 | 0.0183 | 1.01 | 0.1147 | 0.003 | 0.010 |

**Table 4**
P-Values obtained from unimodal test function.

| Function | SBO versus ALO | SBO versus ABC | SBO versus FA | SBO versus PSO | SBO versus GA |
|---|---|---|---|---|---|
| **F1** | 2.96E−11 | 3.019E−11 | 2.74E−11 | 2.89E−11 | 2.99E−11 |
| **F2** | 2.09E−8 | 3.018E−11 | 2.99E−11 | 2.85E−11 | 2.94E−11 |
| **F3** | **0.14** | 2.978E−11 | 2.98E−11 | **0.63** | 2.89E−11 |
| **F4** | 2.97E−11 | 2.971E−11 | 2.98E−11 | 2.98E−11 | 2.94E−11 |
| **F5** | **0.19** | 2.893E−11 | 2.97E−11 | **0.88** | **0.061** |
| **F6** | 2.89E−11 | 2.991E−11 | 2.93E−11 | 2.20E−11 | 2.96E−11 |
| **F7** | 2.76E−11 | 2.737E−11 | 2.51E−11 | 2.73E−11 | 2.67E−11 |

have the best results in 7 functions. By assessing the P values obtained from comparison of SBO and 5 other algorithms, it is understood that only 6 out of 65 P values are higher than 0.05. These results show the statistical superiority of SBO over the other algorithms. On the other hands, from convergence diagram of algorithms, it is observed that SBO is faster than the other algorithms.

## 5. ANFIS

Fuzzy inference system was developed in 1993 by Gang (Jang, 1993). 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.

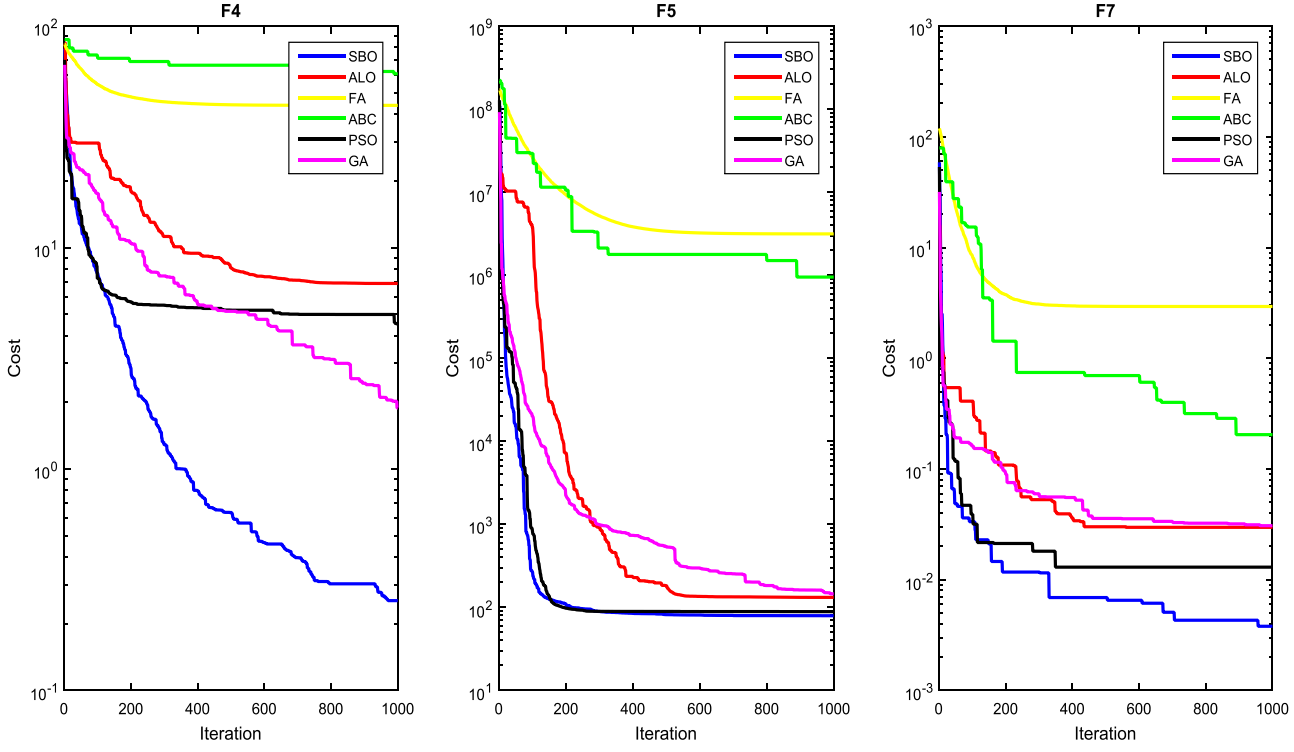


**Fig. 1.** Convergence of algorithms on F4, F5 and F7.

**Table 5**
Results of optimization algorithms on multimodal test function.

| function | | SBO | ALO | FA | ABC | PSO | GA |
|---|---|---|---|---|---|---|---|
| **F8** | ave | **−8.83E+3** | −5.48E+3 | −7.11E+3 | −6.30E+3 | −6.70E+3 | −6.19E+3 |
| | std | 290.09 | 57.89 | 742.99 | 978.86 | 757.79 | 1.178E+3 |
| **F9** | ave | **16.96** | 69.58 | 1.77E+2 | 2.24E+2 | 49.41 | 1.30E+2 |
| | std | 3.315 | 25.53 | 19.90 | 13.93 | 14.42 | 67.64 |
| **F10** | ave | 0.0134 | 1.79 | 18.89 | 3.014 | **5.22E−9** | 0.078 |
| | std | 0.0025 | 0.918 | 0.206 | 0.684 | 8.007E−9 | 0.021 |
| **F11** | ave | **0.0082** | 0.010 | 1.55E+2 | 0.958 | 0.024 | 0.16 |
| | std | 0.0059 | 0.0095 | 29.20 | 0.138 | 0.016 | 0.061 |
| **F12** | ave | 1.89E−5 | 8.65 | 2.28E+6 | 3.15E+6 | **3.76E−19** | 7.89E−4 |
| | std | 3.39E−5 | 3.495 | 1.27E+6 | 1.46E+6 | 6.82E−19 | 9.49E−4 |
| **F13** | ave | 6.57E−6 | 1.16E−2 | 2.23E+7 | 4.41E+6 | **4.71E−14** | 0.004 |
| | std | 5.74E−6 | 0.013 | 1.09E+7 | 2.27E+6 | 1.04E−13 | 0.004 |

**Table 6**
P-Values obtained from multimodal test function.

| function | SBO Versus ALO | SBO versus FA | SBO versus ABC | SBO versus PSO | SBO versus GA |
|---|---|---|---|---|---|
| **F8** | 1.84E−11 | 1.99E−9 | 2.415E−10 | 1.17E−11 | 9.88E−10 |
| **F9** | 3.01E−11 | 9.91E−11 | 2.976E−11 | 2.99E−11 | 2.94E−11 |
| **F10** | 1.01E−6 | 2.99E−11 | 2.991E−11 | 2.80E−11 | 2.83E−11 |
| **F11** | **0.80** | 2.97E−11 | 2.991E−11 | 3.22E−6 | 2.90E−11 |
| **F12** | 3.01E−11 | 2.98E−11 | 2.987E−11 | 2.79E−11 | 4.30E−11 |
| **F13** | 0.01 | 2.98E−11 | 3.001E−11 | 2.79E−11 | 2.88E−11 |

An adaptive network is a feed forward and multilayer structure which 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 parameters). The ANFIS architecture is shown in Fig. 3.

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 ith node in layer j. The layers of ANFIS model have been described in the following sections.

### 5.1. Layer1

Every node i is adaptive with node function. Eqs. (8) and (9) show the node functions.

$$o_i^1=\mu A_i(x), \quad i=1, 2 \tag{8}$$

$$o_i^1=\mu B_{i-2}(y), \quad i=3, 4 \tag{9}$$

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

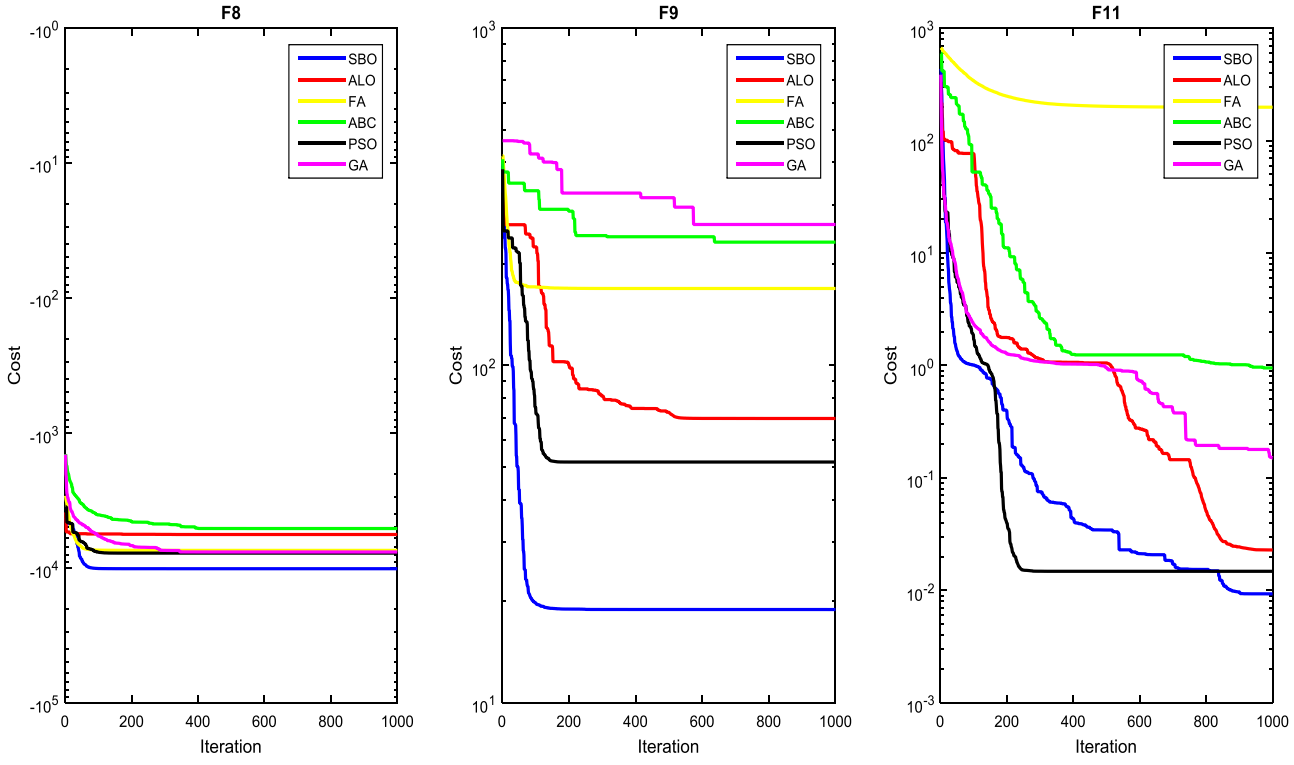


**Fig. 2.** Convergence of algorithms on F8, F9 and F11.

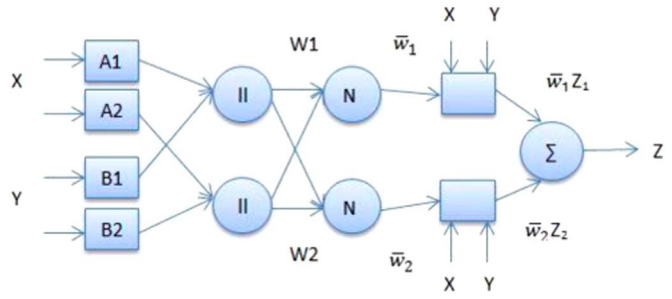


**Fig. 3.** ANFIS architecture.

of a fuzzy set A (=A1, A2, B1, or B2). The main task of the first layer is fuzzification.

### 5.2. 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. Eq. (10) shows this issue.

$$o_i^2 = w_i = \mu A_i(x)\mu B_i(y), \quad i=1, 2 \tag{10}$$

### 5.3. Layer3

In this layer, weights obtained from the second layer are normalized using Eq. (11).

$$o_i^3 = \overline{w_i} = \frac{w_i}{w_1 + w_2}, \quad i=1, 2 \tag{11}$$

### 5.4. Layer4

Each node computes the contribution of the ith rule to the overall output. Eq. (12) performs this computation.

$$o_i^4 = \overline{w_i} z_i = \overline{w_i}(a_i x + b_i y + c_i), \quad i=1, 2 \tag{12}$$

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.

### 5.5. Layer5

The final layer computes the overall output as the summation of all incoming signals from layer 4 using Eq. (13).

$$o_i^5 = \sum \overline{w_i} z_i = \frac{\sum_i w_i z_i}{\sum_i w_i} \tag{13}$$

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

## 6. SBO for software development effort estimation

Although ANFIS is a strong and fast method for estimation, the non-normality of software project data sets makes estimation process challenging and complicated. To deal with this problem one way is adjusting accurate parameters for ANFIS. The main objective of this paper is to adjust the accurate parameters for ANFIS using meta-heuristic algorithm. In this paper, SBO is employed to adjust the parameters. Indeed, the model focuses on improving the performance of ANFIS. The main role of SBO is to find the best parameters for ANFIS to reach the most accurate estimate. The optimization parameters are those employed by input and output membership functions. In this paper, Gaussian and Linear functions are utilized for input and output, respectively. Average and standard deviation are parameters of input function while attributes coefficients along with a constant number are the parameters of output function.

Since the performance metrics play a vital role in the proposed model, they are explained as follows prior to explanation of the proposed model. The 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:

$$RE = \frac{(estimate - actual)}{actual} \tag{14}$$

$$MRE = \frac{|estimate - actual|}{actual} \tag{15}$$

$$MMRE = \frac{\sum_{i=1}^{N} MRE}{N} \tag{16}$$

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

$$PRED(X) = \frac{A}{N} \tag{17}$$

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 (Khatibi Bardsiri et al., 2013a, 2013b; Azzeh et al., 2014). Decrease of MMRE and increase of PRED are the main aims of all estimation techniques used in the field of software development effort estimation. The proposed model includes two stages described in the following sections.

### 6.1. Obtain the optimal parameters for ANFIS by SBO

In this paper, the proposed model uses SBO to adjust the parameters of ANFIS. In other words, the optimal structure of ANFIS is found through investigation of numerous structures. In fact, the training of neuro-fuzzy systems is expressed as an optimization problem. In order to reach accurate results, the classical training algorithms and networks are replaced with an optimization process. In the proposed model, Tagaky-Sugeno fuzzy system is used. The first stage of optimal neuro-fuzzy system is described as follows:

1. Getting training and testing data
2. Creating a base fuzzy system
3. Adjusting the parameters of fuzzy system by SBO based on error function
4. Returning best values as the end result

The optimized Neuro-fuzzy system designed in the first stage of the proposed model is shown in Fig. 4.

In this paper, Albrecht, Kemerer and ISBSG data sets are employed for software development effort estimation. At first, ANFIS input data are normalized between zero and one. Then, this data is randomly divided into three equal groups (almost the same size) using three fold cross validation. At each step, one of the categories is used for testing while training is done using the other two categories. In the proposed model, genfis2 is used as a base fuzzy system. Genfis2 produces a Sugeno type FIS structure using subtractive clustering. This system requires two sets of input and output, separately. Firstly, the rule of extraction method uses the subclust function to determine the number of rules and antecedent membership functions and then uses the least linear squares estimation to determine each rule's consequent equations. Adjusting the parameters of the fuzzy system is the most important stage of ANFIS optimal design. This process is described as follows:

1. Read the parameters of base fuzzy system: This includes reading parameters that are input and output membership functions. After

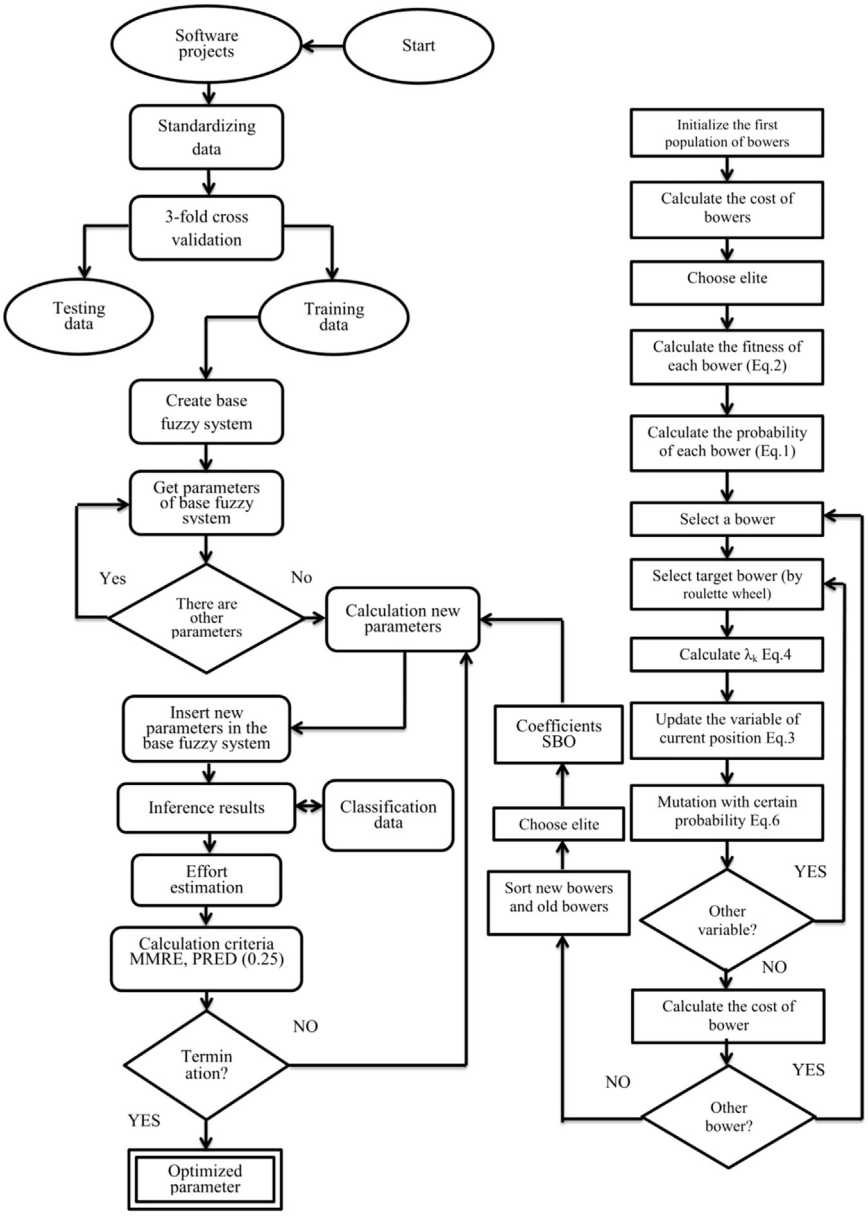


**Fig. 4.** Optimized neuro-fuzzy system design process in the first stage.

reading these parameters, they should sequentially be stored in a vector.

2. Alter parameters of base fuzzy system by SBO: To do this, we assume that the optimal parameter values are coefficient of primary parameters. In other words, if we assume that $p_i^*$ is optimal value of parameter i and $p_i^0$ is initial value of the parameter i and $x_i$ is a coefficient in this case, the value of $p_i^*$is obtained from Eq. (18):

$$p_i^* = x_i p_i^0 \tag{18}$$

In this equation, $x_i$is determined by SBO. There are two viewpoints about $x_i$:

I. Without changing signs and only changing the size. $x_i \epsilon [10^{-\alpha}, 10^{\alpha}]$ and optimal value of α could be less or equal to one.
II. By changing signs and size. $x_i \epsilon$[-M,+M] where value of M could be 10 or less.

3. Insert the new parameters obtained by the optimization algorithm in the fuzzy system and calculate the system error metrics (MMRE and PRED (0.25)).

At this stage, the best parameters obtained by SBO will be saved for the next step. These parameters were obtained according to the metrics of MMRE and PRED (0.25). In fact, SBO calculates these parameters to make MMRE-PRED (0.25) minimal. In general, the main goal of all software effort estimation techniques is to decrease MMRE and increase PRED (0.25).

### 6.2. Using the optimized parameters in ANFIS

The main objective of this stage is to evaluate the accuracy of the proposed model. Indeed, outputs of this stage are values of MMRE and PRED (0.25). Similar to the previous stage, the normalized training and testing data are applied to the base fuzzy systems genfis2, but the difference is that optimization parameters obtained from the previous stage are used instead of the default system parameters. After inserting the optimal parameters, final values of MMRE and PRED (0.25) are calculated. The evaluation of the optimized Neuro-fuzzy system in the second stage of the proposed model is shown in Fig. 5. In the proposed model, the optimal parameters help ANFIS to have a more accurate estimate. In each

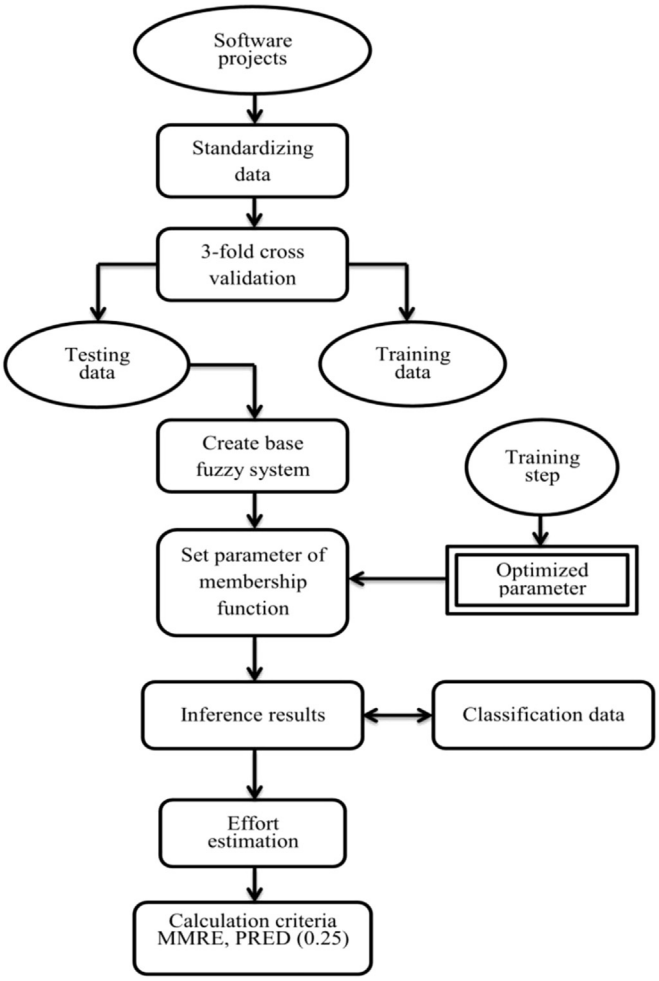


**Fig. 5.** Optimized Neuro-fuzzy system in the second stage.

iteration, SBO can produce different parameters so that the best of which is utilized in the final model.

## 7. Experimental results

In this paper, Albrecht, Kemerer and ISBSG data sets are used for software development effort estimation. At first, ANFIS input data is normalized between zero and one. Then, this data is randomly divided into three equal groups (almost the same size) using three fold cross validation. After the testing stage, the effort estimation is performed and then in the final stage MMRE and PRED (0.25) are calculated for each category of data. There are several methods to standardize data. In these methods, the data is converted so that desired conditions are provided. One of the most important methods is converting data to a new set in which all values fall between zero and one. Eq. (19) shows this normalization.

$$z_i = \frac{x_i - x_{min}}{x_{max} - x_{min}} \tag{19}$$

In this equation, $z_i$ is normalized data, $x_i$ is the value of primary data, $x_{min}$ and $x_{max}$ are minimum and maximum amount of data, respectively.

The normalized data is randomly divided into three groups (almost the same size). Firstly, two groups are considered (for example $s_2$ , $s_3$) for training. Then, the other group is used (for example, $s_1$) for testing. In other words, when data sets are divided into three sets of $s_1$, $s_2$ and $s_3$, one of them (for example, $s_1$) is selected for testing while the other two sets (s2 and s3) are individually used for training. This process continues until all of the sets are used for testing. Finally, the average of three obtained accuracy is considered as the final result.

### 7.1. Data sets

ISBSG (International Software Benchmarking Standards Group) has developed and refined its data collection over a ten-year period based on the metrics that have proven to be very useful to improve software development processes (ISBSG International software benchmark and standard group, 2007). The current release of this data set is the ISBSG R11 data repository (ISBSG), which includes a total of 5052 projects from 24 countries. Selected features are: {Input count (Inpcont), Output count (Outcont), Enquiry count (EnqCont), File count (FileCont), Interface Count (Intcont), Adjusted function point(AFP), and Normalized effort in hours (NorEffort)}.

Albrecht is a popular dataset which includes 24 projects developed by third generation languages. There are five independent features: "Inpcout", "Outcount", "Quecount", "Filcount "and "SLOC" and two dependent features: "Fp" and "Effort" in this data set. The effort, which is recorded in 1000 person hours, is the targeting feature.

The Kemerer data set contains data collected from 15 large completed business data-processing projects. Each project has seven input features: (1) programming language, (2) hardware, (3) duration, (4) KSLOC, (5) AdjFP (adjusted function points), (6) Effort and (7) RAWFP (raw function points) (Kemerer, 1987).

### 7.2. Initial settings of parameters

In this paper, genfis2 is used as a base fuzzy system. In this system, variables must accurately be adjusted to reach the best possible result. The values used for these variables are shown in Table 7. Since ANFIS does not use the default training function in this paper, accurate adjusting of SBO parameters has a significant impact on the final results. SBO parameter values are shown in Table 8. The values are obtained through an exhaustive trial-error process.

### 7.3. Comparing the proposed model with other models

The results of the Albrecht data set are shown in Table 9 in comparison with other models. The results of the proposed model are compared with models of regression-based including CART, stepwise regression (SWR), and multiple regression (MLR). These models are involved in the comparison process because they have been widely used in other research studies (Khatibi Bardsiri et al., 2013; Mittas and Angelis, 2010; Hsu and Huang, 2011). Artificial neural network (ANN) is also compared with the proposed method. Also, the results obtained from the proposed model are compared with the results reported in two recent studies (Azzeh et al., 2014; Li et al., 2009). In Fig. 6, the values obtained from PRED (0.25) and MMRE on Albrecht data set are compared with the other models. The results show that the proposed model generates more accurate estimations compared with the other

**Table 7**
ANFIS parameter values.

| Value | Parameter |
|---|---|
| Gaussian | Input MF function type |
| Linear | Output MF type |
| SBO | Learning algorithm |
| Genfis2 | Based fuzzy system |
| prod | And method |
| probor | Or method |
| wtaver | Defuzz method |
| prod | Im method |
| sum | Agg method |

**Table 8**
SBO parameter values.

| | |
|---|---|
| Max it | 100 |
| npop | 50 |
| α | 0.94 |
| z | 0.02 |
| Mutation probability | 0.05 |

**Table 9**
Results of the Albrecht data set in comparison with other models.

| Method | MMRE | PRED (0.25) |
|---|---|---|
| OABE (Azzeh et al., 2014) | 0.40 | 0.44 |
| CART | 0.98 | 0.25 |
| SWR | 0.76 | 0.125 |
| ANN | 0.925 | 0.25 |
| ANFIS-SBO | **0.235** | 0.625 |
| NABE (Li et al., 2009) | 0.41 | 0.36 |
| MLR | 1.01 | 0.25 |
| ANFIS-PSO | 0.867 | 0.375 |

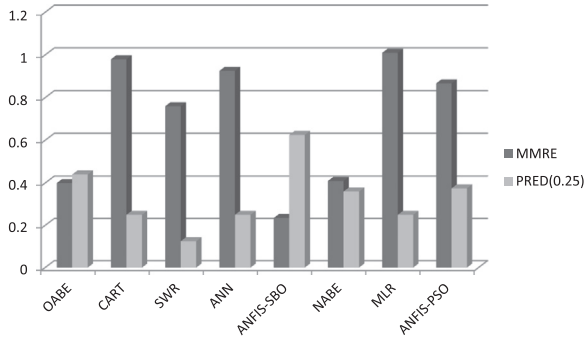


**Fig. 6.** PRED (0.25) and MMRE of the Albrecht data set in comparison with other models.

models. The proposed model can significantly improve both MMRE and PRED (0.25).

The results of applying the proposed model to ISBSG dataset compared with the other popular models are shown in Table 10. The results of the proposed model are compared with models of regression-based estimation methods including CART, stepwise regression (SWR), multiple regression (MLR) and ANN. Moreover, the results obtained from the proposed model are compared with those of reported in a recent paper (Khatibi Bardsiri et al., 2013a). In Fig. 7, values obtained from MMRE and PRED(0.25) on ISBSG data set are compared with the other models. Among all the models, the proposed model can make a significant improvement.

The results of applying the proposed model to Kemerer dataset compared with other popular models are shown in Table 11. The results of the proposed model are compared with models of regression-based estimation methods including CART, stepwise regression (SWR), multiple regression (MLR) and ANN. Moreover, the results obtained

**Table 10**
Results of the ISBSG data set in comparison with other models.

| Method | MMRE | PRED (0.25) |
|---|---|---|
| MLR | 0.891 | 0.234 |
| SWR | 0.969 | 0.174 |
| ANN | 1.00 | 0.241 |
| CART | 1.11 | 0.167 |
| ABE-PSO (Khatibi Bardsiri et al., 2013a) | 0.64 | 0.51 |
| ANFIS-SBO | **0.490** | **0.637** |
| ANFIS-PSO | 0.767 | 0.275 |

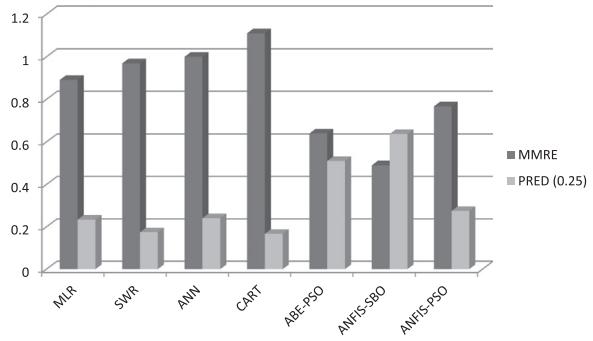


**Fig. 7.** MMRE and PRED (0.25) of the ISBSG data set in comparison with other models.

**Table 11**
Results of the Kemerer data set in comparison with other models.

| Method | MMRE | PRED (0.25) |
|---|---|---|
| OABE (Azzeh et al., 2014) | 0.396 | 0.533 |
| CART | 0.65 | 0.20 |
| SWR | 0.64 | 0.20 |
| ANN | 0.57 | 0.40 |
| ANFIS-SBO | **0.268** | **0.60** |
| MLR | 0.71 | 0.20 |
| ANFIS-PSO | 0.551 | 0.40 |

from the proposed model are compared with those of reported in a recent paper (Azzeh et al., 2014). In Fig. 8, the values obtained from MMRE and PRED (0.25) on Kemerer data set are compared with other models. Among all the models, the proposed model reaches the best results.

### 7.4. Improvement analysis

In Table 12, the improvement percentage obtained by applying the proposed method to Albrecht data set is demonstrated by MMRE and PRED (0.25). As shown in Table 12, the largest improvement of MMRE, 329%, is occurred for MLR while the lowest percentage of improvement, 70%, is related to OABE. Similar to MMRE, PRED (0.25) is also improved. Lowest percentage of improvement, 42%, is related to OABE and the largest improvement of PRED (0.25), 400%, is occurred for SWR.

Fig. 9 displays the box plots drawn for MRE values achieved on the Albrecht data set. The lowest median and the inter quartile range is related to the proposed model. Table 13 shows the p-values achieved by Wilcoxon test on MRE results of different models. All p-values are less than 0.05, which statistically confirms the superiority of the proposed model.

In Table 14, the improvement percentage obtained by applying the proposed method to ISBSG data set is presented for metrics of MMRE and PRED (0.25). As shown in Table 14, the proposed model can

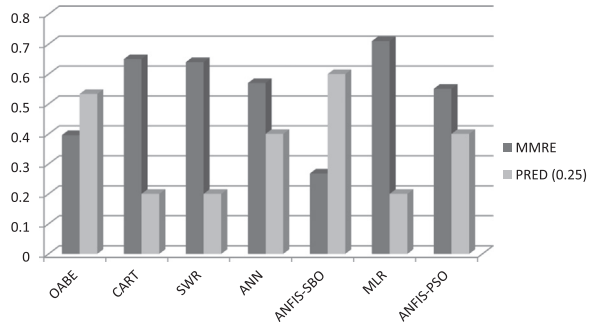


**Fig. 8.** MMRE and PRED (0.25) of the Kemerer data set in comparison with other models.

**Table 12**
Improvement percentage obtained by applying the proposed method to Albrecht data set (MMRE and PRED (0.25)).

| Method | MMRE | PRED (0.25) |
|---|---|---|
| OABE | **70%** | **42%** |
| CART | 317% | 150% |
| SWR | 223% | **400%** |
| ANN | 293% | 150% |
| NABE | 74% | 73% |
| MLR | **329%** | 150% |
| ANFIS-PSO | 268% | 66% |

significantly provide more improvement for metric of PRED (0.25) than the other models. Lowest improvement for PRED (0.25) is related to ABE-PSO model with 24% while the highest is related to the CART model with 281%. It is obvious that the improvement percentage obtained for PRED (0.25) is higher than that for MMRE, but the proposed model is still stronger than the other models.

The highest improvement for MMRE is related to CART model with 126% and the lowest is related to the ABE-PSO model with 30%. It is observed that the proposed model compared to the best model (ABE-PSO) can improve the value of MMRE and PRED (0.25) up to 30% and 24%, respectively. Fig. 10 displays the box plots drawn for MRE values achieved on ISBSG. The lowest median is related to the proposed model but the inter quartile range is related to the SWR model. Table 15 shows the p-values achieved by Wilcoxon test on MRE results of different models. All six p-values are less than 0.05.

In Table 16, the improvement percentage obtained by applying the proposed method to Kemerer data set is demonstrated by MMRE and PRED (0.25). As shown in Table 16, the largest improvement of MMRE, 164%, is occurred for MLR while the lowest percentage of improvement, 47%, is related to OABE. Similar to MMRE, PRED (0.25) is also improved. The lowest percentage of improvement, 12%, is related to OABE and the largest improvement of PRED (0.25), 200%, is occurred for SWR, CART and MLR.

Fig. 11 displays the box plots drawn for MRE values achieved on the Kemerer data set. The lowest median is related to the proposed model but the inter quartile range is related to the CART model. Table 17 shows the p-values achieved by Wilcoxon test on MRE results of different models. Three p-values are less than 0.05, which statistically confirms the superiority of the proposed model.

**Table 13**
P-values of Wilcoxon test (Albrecht. other models).

| ANN | CART | MLR | NABE | SWR | OABE | ANFIS-PSO |
|---|---|---|---|---|---|---|
| 0.007 | 0.007 | 0.01 | 0.028 | 0.001 | 0.0499 | 0.0379 |

**Table 14**
Improvement percentage obtained by applying the proposed method to ISBSG data set (MMRE and PRED (0.25)).

| Method | MMRE | PRED (0.25) |
|---|---|---|
| MLR | 81% | 172% |
| SWR | 97% | 266% |
| ANN | 104% | 164% |
| CART | **126%** | **281%** |
| ABE-PSO | **30%** | **24%** |
| ANFIS-PSO | 56% | 131% |

Finally, by examining the obtained results, it is observed that the proposed model has achieved more accurate results in three data sets compared to the other models.

## 8. Conclusion

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 correct selection 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. As a solution to this problem, modern meta-heuristic algorithms can be used instead of classic training algorithms in ANFIS.

This paper focused on the use of ANFIS combined by SBO as a novel training algorithm. SBO is focused on the best solution while the diversity of population is also guaranteed. This feature of SBO makes it suitable to be used in the field of attribute weighting, ANN weight and bias adjusting and ANFIS parameters adjusting. Moreover, based on

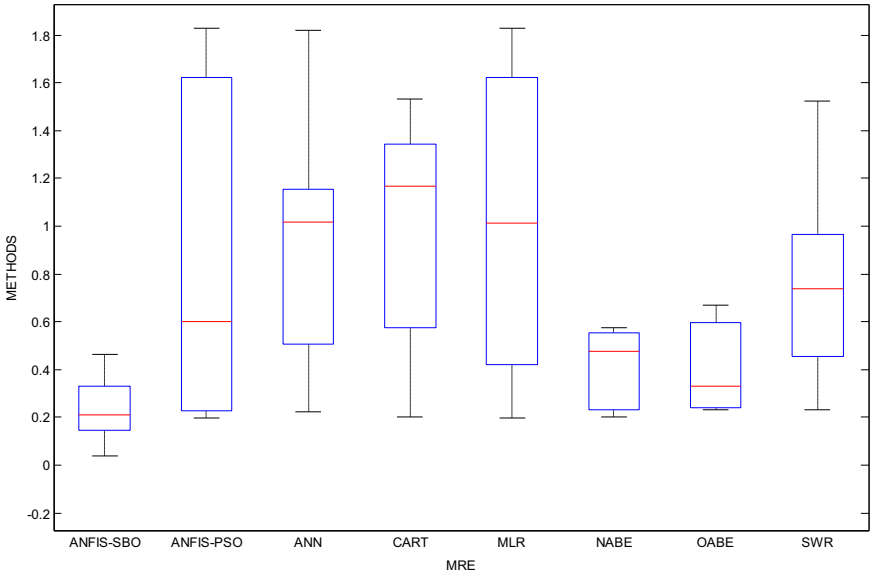


**Fig. 9.** Box plot on MRE results obtained from Albrecht.

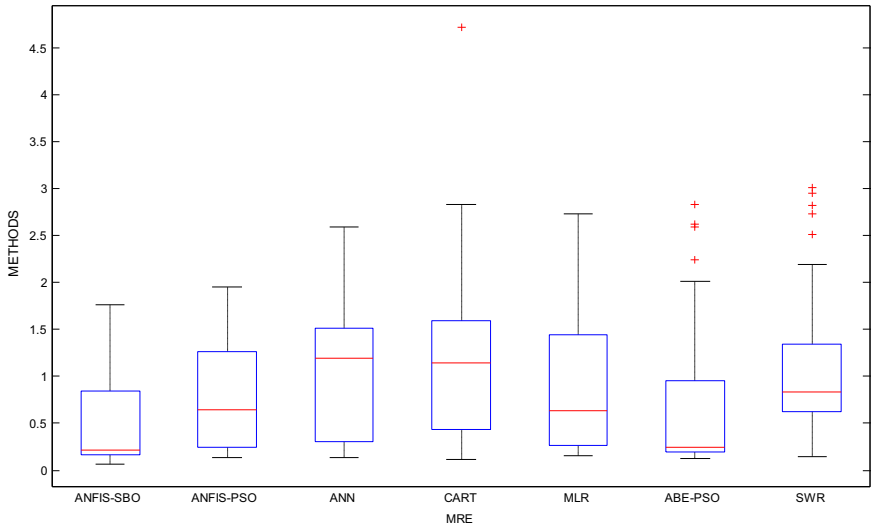


**Fig. 10.** Box plot on MRE results obtained from ISBSG.

**Table 15**
P-values of Wilcoxon test (ISBSG. other models).

| ANN | CART | ABE-PSO | MLR | SWR | ANFIS-PSO |
|---|---|---|---|---|---|
| 3.01E−16 | 2.61E−17 | 8.99E−4 | 6.39E−13 | 3.66E−14 | 3.66E−10 |

**Table 16**
Improvement percentage obtained by applying the proposed method to Kemerer data set (MMRE and PRED (0.25)).

| Method | MMRE | PRED (0.25) |
|---|---|---|
| OABE | **47%** | **12%** |
| CART | 142% | **200%** |
| SWR | 138% | **200%** |
| ANN | 112% | 50% |
| MLR | **164%** | **200%** |
| ANFIS-PSO | 105% | 50% |

**Table 17**
P-values of Wilcoxon test (kemerer. other models).

| ANN | CART | MLR | SWR | OABE | ANFIS-PSO |
|---|---|---|---|---|---|
| 0.095 | 0.0317 | 0.0310 | 0.031 | 0.22 | 0.095 |

the benchmark conducted in Section 4, increase in the number of variables does not have significant effect on the performance of SBO. According to the results obtained from SBO (with uni-modal and multi-modals functions), this algorithm can be useful to achieve the exploitation (Section 4.1), exploration (Section 4.2) and high convergence rate. SBO algorithm was compared with five well-known recent algorithms (ALO, PSO, FA, GA and ABC). The results show that the SBO algorithm achieves better answer than the other algorithms in many test functions. The results obtained from the statistical test showed that the SBO algorithm is statistically superior compared with the other algorithms (Tables 4, 6).

The proposed hybrid estimation model consists of two stages in which the model is constructed and evaluated. The role of SBO in the

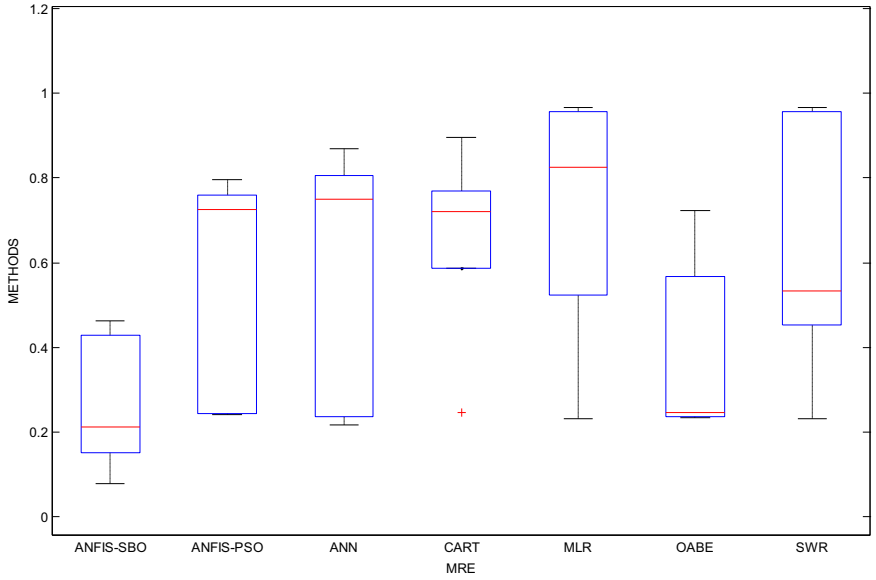


**Fig. 11.** Box plot on MRE results obtained from Kemerer.

proposed model is generation of the optimized parameters for ANFIS. In fact, by choosing the best parameters generated by SBO, the ANFIS performance is significantly improved. To evaluate the performance of the proposed hybrid 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 presented in Section 7.3, the lowest MMRE and the highest PRED (0.25) are related to the proposed model in the three data sets. In fact, the combination of SBO and ANFIS is able to have the best performance among the existing models. Finally, statistical analysis of results supported the superiority of the proposed model against the other models (Tables 13,15,17).

According to the overall results, it can be concluded that the SBO algorithm can be an appropriate complementary algorithm for solving the problem of software development effort estimation. Particularly, the hybrid model of ANFIS-SBO is capable of producing accurate estimations in this field. New optimization algorithms and feature selection techniques can be employed to improve the accuracy of software development effort estimation, as the future work.

## References


Abeer, Hamdy, 2012. Fuzzy logic for enhancing the sensitivity of COCOMO cost model. J. Emerg. Trends Comput. Inf. Sci. 3 (9), 1292–1297.

Agrawal, A., Jain, V., Sheikh, M., 2016. Quantitative Estimation of Cost Drivers for Intermediate COCOMO towards Traditional and Cloud Based Software Development, In: Proceedings of the 9th Annual ACM India Conference. ACM: Gandhinagar, India. pp. 85–95.

Agrawal, V., Shrivastava, V., 2015. Performance evaluation of software development effort estimation using neuro-fuzzy model. Int. J. Emerg. Res. Manag. Technol. 4 (7), 193–199.

Alatas, B., 2012. A novel chemistry based meta heuristic optimization method for mining of classification rules. Expert Syst. Appl. 39, 11080–11088.

Azzeh, M., Nassif, A.B., 2016. A hybrid model for estimating software project effort from use case points. Appl. Soft Comput. 49, 981–989.

Azzeh, M., Neagu, D., Cowling, P., 2009. Fuzzy grey relational analysis for software effort estimation. J. Empir. Softw. Eng. 15 (1), 60–90.

Azzeh, M., Neagu, D., Cowling, P.I., 2011. Analogy-based software effort estimation using Fuzzy numbers. J. Syst. Softw. 84 (2), 270–284.

Azzeh, M., Yousef, E., Marwan, A., 2014. An optimized analogy-based project effort estimation. Int. J. Adv. Comput. Sci. Appl. 5, 6–11.

Azzeh, M., Nassif, A.B., Minku, L.L., 2015. An empirical evaluation of ensemble adjustment methods for analogy-based effort estimation. J. Syst. Softw. 103, 36–52.

Boehm, B., Abts, C., Chulani, S., 2000. Software development cost estimation approaches – a survey. Ann. Softw. Eng. 10 (1), 177–205.

Boogert, N.J., Giraldeau, L.-A., Lefebvre, L., 2008. Song complexity correlates with learning ability in zebra finch males. Anim. Behav. 76 (5), 1735–1741.

Borgia, G., 1985. Bower destruction and sexual competition in the satin bowerbird (Ptilonorhynchus violaceus). Behav. Ecol. Sociobiol. 18, 91–100.

Cheng, M.-Y., Prayogo, D., 2014. Symbiotic organisms search: a new metaheuristic optimization algorithm. Comput. Struct. 139, 98–112.

Chulani, S., Boehm, B., Steece, B., 1999. Bayesian analysis of empirical software engineering cost models. IEEE Trans. Softw. Eng. 25 (4), 573–583.

Coleman, S.W., Patricelli, G.L., Borgia, G., 2004. Variable female preferences drive complex male displays. Nature 428, 742–745.

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

Cuauhtemoc Lopez-Martin, 2010. A fuzzy logic model for predicting the development effort of short scale programs based upon two independent variables. Appl. Soft Comput. 11 (1), 724–732.

Derrac, J., Garca, S., Molina, D., Herrera, F., 2011. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol. Comput. 1, 3–18.

Digalakis, J., Margaritis, K., 2001. On benchmarking functions for genetic algorithms. Int. J. Comput. Math. 77, 481–506.

Formato, R.A., 2008. (ed)Central force optimization: a new nature inspired computational framework for multidimensional search and optimization. In Nature Inspired Cooperative Strategies for Optimization (NICSO 2007) 129. Springer, 221–238.

Holland, J.H., 1992. Genetic algorithms. Sci. Am. 267, 66–72.

Holland, J.H., Reitman, J.S., 1977. Cognitive systems based on adaptive algorithms. ACM SIGART Bull., (49–49).

Hossein, A., Hossein-Alavi, 2012. A: krill herd: a 000000000new bio-inspired optimization algorithm. Commun. Nonlinear Sci. Numer. Simul. 17, 4831–4845.

Hota, H.S., Shukla, R., Singhai, S., 2015. Predicting Software Development Effort Using Tuned Artificial Neural Network, In: Jain, L.C. et al. (Eds.), Computational Intelligence in Data Mining – Proceedings of the International Conference on CIDM, 20–21 December 2014. Springer India: New Delhi. 3, pp. 195–203.

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

Huang, X.S., Ho, D., Ren, J., Capretz, L.F., 2007. Improving the COCOMO model using a neuro-fuzzy approach. Appl. Soft Comput. J. 7 (1), 29–40.

Idri, A., Hosni, M., Abran, A., 2016. Improved estimation of software development effort using Classical and Fuzzy Analogy ensembles. Appl. Soft Comput. 49, 990–1019.

ISBSG International software benchmark and standard group, 2007. Data CDRelease 10, ⟨www.isbsg.org⟩.

Jang, J.-S.R., 1993. ANFIS: adaptive network based fuzzy inference system. IEEE Trans. Syst. Man Cybern. 23 (3), 665–668.

Jensen, R., 1983. An improved macrolevel software development resource estimation model. In: Proceedings of the 5th Conference of International S Parametric Analysts, pp. 88–92.

Jorgensen, M., 2004. Top-down and bottom-up expert estimation of software development effort. Inf. Softw. Technol. 46 (1), 3–16.

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

Karaboga, D., Basturk, B., 2007. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J. Glob. Optim. 39, 459–471.

Kaveh, A., 2014. Dolphin echolocation optimization. In: Advances in Metaheuristic Algorithms for Optimal Design of Structures. Springer, 157–193.

Kaveh, A., Talatahari, S., 2010. A novel heuristic optimization method: charged system search. Acta Mech. 213, 267–289.

Kaveh, A., Khayatazad, M., 2013. Ray optimization for size and shape optimization of truss structures. Comput Struct. 117, 82–94.

Kaveh, A., Farhoudi, N., 2013. A new optimization method: dolphin echolocation. Adv. Eng. Softw. 59, 53–70.

Kaveh, A., Ghazaan, M.I., Bakhshpoori, T., 2013. An improved ray optimization algorithm for design of truss structures. Civ. Eng. 57, 97–112.

Kemerer, F.C., 1987. An empirical validation of software cost estimation models. Commun. ACM 30 (5), 416–429.

Kennedy J., Eberhart R., 1995. Particle swarm optimization. In: Proceedings IEEE International Conference on Neural Networks, vol. 6, pp. 1942–1948.

Khatibi, E., Khatibi Bardsiri, V., 2015. Model to estimate the software development effort based on in-depth analysis of project attributes. Softw., IET 9 (4), 109–118.

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. Softw. Eng. 19 (4), 857–884.

Khatibi Bardsiri, V., Dayang Norhayati, Abang Jawawi, Siti Zaiton, Mohd Hashim, Khatibi, E., 2013a. A PSO-based model to increase the accuracy of software development effort estimation. Softw. Qual. 21 (3), 501–526.

Khatibi Bardsiri, V., Khatibi, A., Khatibi, E., 2013b. An optimization-based method to increase the accuracy of software development effort estimation. J. Basic. Appl. Sci. Res. 3 (2), 159–166.

Kumar, Shiyna, Chopra, Vinay, 2013. Neural network and fuzzy logic based framework for software development effort estimation. Int. J. Adv. Res. Comput. Sci. Softw. Eng. 3 (5), 19–24.

Kumari, Poonam, Ishdeep, Singla, 2015. Tuning of use case point (UCP) analysis Parameter using PSO. Commun. Appl. Electron. 1 (5), 25–28.

Li, X., Lu, W.F., Zhai, L., Er, M.J., Pan, Y., 2015. Remaining life prediction of cores based on data-driven and physical modeling methods. In: Nee, A.Y.C. (Ed.), Handbook of Manufacturing Engineering andTechnology. Springer London, London, 3239–3264.

Li, Y.F., Xie, M., Goh, T.N., 2009. A study of the non-linear adjustment for analogy based software cost estimation. Empir. Softw. Eng. 14 (6), 603–643.

MacDonell, S.G., Shepperd, M.J., 2003a. Combining techniques to optimize effort predictions in software project management. J. Syst. Softw. 66 (2), 91–98.

MacDonell, S.G., Shepperd, M.J., 2003b. Combining techniques to optimize effort predictions in software project management. J. Syst. Softw. 66, 91–98.

Madachy, R., 1994. A Software Project Dynamics Model for Process Cost, Schedule and Risk Assessment (Ph.D. dissertation). University of Southern California.

Mirjalili, S., 2015. The Ant Lion Optimizer. Adv. Eng. Softw. 83, 80–98.

Mirjalili, S., Mirjalili, S.M., Lewis, A., 2014. Grey wolf optimizer. Adv. Eng. Softw. 69, 46–61.

Mittas, N. Angelis, L., 2008. Combining regression and estimation by analogy in a semi-parametric model for software cost estimation, In: Proceedings of the Second ACMIEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '08), pp. 70–79.

Mittas, N., Angelis, L., 2010. LSEbA: least squares regression and estimation by analogy in a semi parametric model for software cost estimation. Empir. Softw. Eng. 15 (5), 523–555.

Møller, A.P., Pomiankowski, 1993. A. Why have birds got multiple sexual ornaments? Behav. Ecol. Sociobiol. 32 (3), 167–176.

Nassif, A.B., Azzeh, M., Capretz, L.F., Ho, D., 2016. Neural network models for software development effort estimation: a comparative study. Neural Comput. Appl. 27 (8), 2369–2381.

Oliveira, A.L.I., 2006. Estimation of software project effort with support vector regression. Neurocomputing 69 (13–15), 1749–1753.

Pan, Y., Hu, X., Er, M.J., Li, X., Gouriveau, R., 2013. Bearing condition prediction using enhanced online learning fuzzy neural networks, In: Nee, A.Y.C., Song, B., Ong, S.-K. (Eds.), Re-engineering Manufacturing for Sustainability, Proceedings of the 20th CIRP International Conference on Life Cycle Engineering, Singapore 17–19 April 2013, Springer Singapore, Singapore. pp. 175–182.

Pan, Y., Er, M.J., Li, X., Yu, H., Gouriveau, R., 2014. Machine health condition prediction via online dynamic fuzzy neural networks. Eng. Appl. Artif. Intell. 35, 105–113.

Pratama, M., et al., 2013. Data driven modeling based on dynamic parsimonious fuzzy

neural network. Neurocomputing 110, 18–28.
Putnam, L., Myers, W., 1992. Measures for Excellence. Yourdon Press Computing Series.
Rashedi, E., Pour, H.N., Saryazdi, S., 2009. GSA: a gravitational search algorithm. Inf. Sci. 179, 2232–2248.
Rijwani, P., Jain, S., 2016. Enhanced software effort estimation using multi layered feed forward artificial neural network technique. Procedia Comput. Sci. 89, 307–312.
Salcedo-Sanz, S., Pastor-Snachez, A., Gallo-Marazuela, D., Portilla-Figueras, 2013. A. novel coral reefs optimization algorithm for multi-objective problems. Intell. Data Eng. Autom. Learn. – IDEAL 8206, 326–333.
Satapathy, S.M., Kumar, M., Rath, S.K., 2016a. Optimized class point approach for software effort estimation using adaptive neuro-fuzzy inference system model. Int. J. Comput. Appl. Technol. 54 (4), 323–333.
Satapathy, S.M., Acharya, B.P., Rath, S.K., 2016b. Early stage software effort estimation using random forest technique based on use case points. IET Softw. 10 (1), 10–17.
Sheenu, Rizvi, Abbas, S.Q., Rizwan, Beg, 2014. A hybrid fuzzy-Ann approach For software effort estimation. Int. J. Found. Comput. Sci. Technol. (IJFCST) 4 (5), 45–56.
Shepperd, M., Schofield, C., 1997. Estimating software project effort using analogies. IEEE Trans. Softw. Eng. 23 (12), 733–743.
Simon, D., 2008. Biogeography-based optimization. IEEE Trans. Evolut. Comput. 12, 702–713.
Trendowicz, A., Jeffery, R., 2014. Software Project Effort Estimation. Foundations and Best Practice Guidelines for Success. Springer.
Vishal, Sharma, Kumar Verma, Harsh, 2010. Optimized fuzzy logic based framework for effort estimation in software development. Int. J. Comput. Sci. Issues (IJCSI) 7 (2), 30–38.
Xua, Z., Khoshgoftaar, T.M., 2004. Identification of fuzzy models of software cost estimation. Fuzzy Sets Syst. 145 (1), 141–163.
Yang, X.-S., 2010a. Firefly algorithm, Levy flights and global optimization. In: Research and Development in Intelligent Systems XXVI. Springer, 209–218.
Yang, X.-S., 2010b. Firefly algorithm, stochastic test functions and design optimizationJ Bio-Inspired Comput, 78–84.
Yang, X.-S., Deb, S., 2009. Cuckoo search via Lévy flights. In: World Congress On Nature & Biologically Inspired Computing. NaBIC, 210–214.
Yang, X.-S., Deb, S., 2010. Engineering optimisation by cuckoo search. Int J. Math. Model Numer Optim. 1, 330–343.
Yao, X., Liu, Y., Lin, G., 1999. Evolutionary programming made faster. IEEE Trans. Evol. Comput. 3, 82–102.
Ziauddin, Shahid Kamal, khan, Shafiullah, Abdul Nasir, Jamal, 2013. A fuzzy logic based software cost estimation model. Int. J. Softw. Eng. Appl. (IJSEIA) 7 (2), 7–18.