Journal of Artificial Intelligence and Data Mining (JAIDM), Vol. 9, No. 3, 2021, 329-340.

Shahrood University of Technology

Journal of Artificial Intelligence and Data Mining (JAIDM)
Journal homepage: http://jad.shahroodut.ac.ir

Research paper

Hybrid PSO-SA Approach for Feature Weighting in Analogy-Based Software Project Effort Estimation

Zahra shahpar[1], Vahid Khatibi Bardsiri[2*] and Amid Khatibi Bardsiri[2]

1. Department of Computer engineering, Kerman Branch, Islamic Azad University, Kerman, Iran.
2. Department of Computer engineering, Bardsir Branch, Islamic Azad University, Bardsir, Iran.

Article Info

Article History:
Received 01 October 2020
Revised 26 April 2021
Accepted 07 June 2021

DOI:10.22044/jadm.2021.10119.2152

Keywords:
Software Effort Estimation, Analogy-based Estimation, Feature Weight Optimization, Particle Swarm Optimization, Simulated Annealing.

**Corresponding author: khatibi78@yahoo.com (V. Khatibi Bardsiri).*

Abstract

The software effort estimation plays an important role in software project management, and analogy-based estimation (ABE) is the most common method used for this purpose. ABE estimates the effort required for a new software project based on its similarity to the previous projects. A similarity between the projects is evaluated based on a set of project features, each of which has a particular effect on the degree of similarity between the projects and the effort feature. The present study examines the hybrid PSO-SA approach for feature weighting in the analogy-based software project effort estimation. The proposed approach is implemented and tested on two well-known datasets of software projects. The performance of the proposed model is compared with the other optimization algorithms based on the MMRE, MDMRE, and PRED (0.25) measures. The results obtained showed that the weighted ABE models provide more accurate and better effort estimates relative to the unweighted ABE models and that the hybrid PSO-SA approach leads to better and more accurate results compared to the other weighting approaches in both datasets.

1. Introduction

Analogy-based estimation (ABE) is one of the most commonly used methods for the software project effort estimation. Introduced by Shepperd in 1979, this method has achieved a widespread popularity due to its ease of use and simplicity [1]. ABE starts with collecting data from the previous software projects in the form of an n*p matrix, with each row reporting one software project's information and describing the project with a set of features and their values. These features are often defined and used at random and based solely on their availability, without a formulated relationship with the target feature, i.e. the amount of effort required by the software project. Obviously, these features are not equally important in all the software projects [2, 3]. The researchers, therefore, use the weighting techniques in order to optimize the feature sets. In these techniques, an appropriate 'weight' or coefficient is determined for each feature by analyzing the effect of the independent feature on the target feature [4, 5]. Feature weighting expands or compresses the feature space, which, in turn, affects the proximity of the previous projects to the target projects, changing the set of adjacent and neighboring projects that might be similar to the current project and provide a solution for it. Such problems are NP-hard and pose significant computational challenges if the number of projects and feature sets does not matter. Selecting a feature subset, which might appear to be an easier solution, is also NP-hard but less challenging since each feature is weighted by zero or one, and for this reason, feature selection has so far been the dominant approach in software projects effort estimation [6]. After reviewing the techniques reported for this purpose, we present our hybrid feature weighting approach. In the next section, we analyze the relevant studies reported in the literature. The ABE model is then introduced and reviewed. The research methodology is elaborated in the fourth section. The fifth section describes how the proposed model is implemented on the dataset. Finally we analyze and discuss the results of implementing the proposed approach in the sixth section.

2. Related Works

Feature weighting is a common approach in machine learning in order to deal with the challenge of very large datasets. Even in small datasets, a large and diverse feature space can lead to overfitting [7]. In such cases, feature weighting and selection is an appropriate approach for determining which features affect the dependent feature by weighting the features with different coefficients based on their relationship with the target feature. For example, the irrelevant or redundant features are given lower weights. Due to its focus on the important features, therefore, this approach leads to a smaller dataset, and as a result, the research work will be faster, more accurate, and more extensive. Various objectives have been mentioned for feature weighting in the literature, the most important of which include [8, 9]:

- improved model performance and prevention of overfitting (for example, selecting better clusters in clustering and improved prediction performance in supervised classification).
- faster and more cost-effective models.
- acquiring a deeper understanding of the data and the processes that have generated the data.
- finding a feature subset through feature weighting in order to improve the prediction accuracy or reduce the size of the dataset without a significant negative impact on the accuracy of classification prediction, using only the selected features.

Various feature selection and weighting techniques have been used in order to improve ABE. These techniques can be classified into three general approaches including correlation analysis, exhaustive search, and optimization.

Correlation analysis is a statistical technique used for estimating the software development effort. In this method, the dependency level between the project features is determined by statistical analysis. The traits that have a weak correlation with the development effort are assigned lower weights; the traits with higher correlation coefficients will have higher weights, and the traits with no correlation are excluded. The previous studies have shown that this technique can improve the ABE's efficiency [10-13].

In the exhaustive search technique, appropriate weights are selected for the independent features based on the pre-determined criteria. Analysis of the dependencies between the features leads to the formation of several feature subsets. The features that form the intersection of all subsets are regarded as the most important features, and weighting is based on the number of reappearances of the same feature in various subsets [14, 15].

The optimization techniques are used for adjusting the feature weights in the ABE similarity function. Genetic algorithm is one of the most common optimization methods employed for determining the feature weights and the number of neighbors (KNN) [16-20]. In the previous studies, genetic algorithm has been used for weighting the features and for minimizing the error parameter [16-21]. Particle swarm optimization is another method used for this purpose [22]. Particle swarm optimization and multi-objective particle swarm optimization have been used for determining the number of nearest neighbors, selecting the optimal feature set, and weighting the features in software effort estimation [23]. Araujo et al. [24] have proposed a hybrid linear perceptron method and used a genetic algorithm for optimizing the perceptron parameters and selecting the optimal feature subset with the goal of enhancing the accuracy of effort estimation. Khatibi et al. [25] have presented a flexible method for estimating software development effort, the core idea of which is to localize adaptation and the weighting process by clustering the software projects. They classified the projects into several clusters based on the key features and used a combination of ABE and the PSO algorithm for feature weighting, with different clusters assigning different weights to each project feature. Instead of comparing the new project with all the previous projects, it is only compared with the projects that fall within the related clusters (based on the common features). Khatibi et al. [26] have proposed a PSO-based model for improving the estimation accuracy. This model is a combination of PSO algorithms and the ABE approach. In the present work, a framework is presented in which the appropriate weights are assigned to the project features, and thus increase the accuracy of estimates. The framework includes a training and a testing phase through which the proposed estimation model is developed and evaluated. In this method, the PSO algorithm extracts the possible weights and selects those weights that lead to more accurate estimates. Wu et al. [27] have used the PSO optimization algorithm for weighting features in the Manhattan and Euclidean similarity functions, and gray relationship degree. Benala et al. [28] have employed the differential evolution algorithm in order to optimize the feature weights for ABE similarity functions, applying five mutation strategies. Ranichandra [29] has made an attempt to optimize the non-orthogonal space distance, which is a measure of the similarity between the software projects based on the feature weights and feature redundancy using the ant colony optimization algorithm. Khatibi [30] has utilized the differential evolution algorithm as a tool for feature weighting. Shah et al. [31] have proposed a model for software effort estimation that ensemble Artificial Bee Colony (ABC) with Analogy-based estimation. Zakrani et al. [32] have designed a model for effort estimation using the Support Vector Regression (SVR) and feature selection methods. Their SVR model was optimized by a grid search procedure.

The previous studies indicate that the main challenges in applying the optimization techniques are the entrapment of optimization algorithms in local optimization and the high computational burden. In the present work, therefore, we intended to use an approach that could address these two challenges.

3. Analogy-based Estimation

Analogy-based estimation was first proposed by Sheppard and Schofield as an approach for estimating the software project effort [1]. The main idea of this method is to compare the new project with the previous projects. It measures the similarity between the current project and the previous projects, and then estimates the effort required for it based on that required for the most similar projects [1, 17]. The ABE approach consists of four steps [16, 30]:

- Collecting information from the previous projects in the form of datasets.
- Selecting and weighting the features.
- Using the similarity function in order to determine the degree of similarity between the new and past projects and select the most similar ones.
- Determining the effort required for the new project using the solution function based on the efforts reported for the most similar projects.

3.1. Similarity Function

Similarity function plays a pivotal role in the analogy-based effort estimation. Using different similarity measures can lead to different results [33]. In this work, three similarity functions were employed and examined. Euclidean similarity function, calculated by (1), has been the most widely used similarity function in the previous studies. In this equation, p and p' represent the projects; w_i is the weight assigned to each feature, and ranges between 0 and 1; f_i and f_i' are the i*th* feature of each project; and n is the number of features. δ (δ=0.0001) is added to obtain the non-zero results [13, 16-19, 21, 25-27, 30].

Another widely used function is the Manhattan similarity function (2). Its equation closely resembles that of Euclidean similarity, and the only difference is that the former calculates the absolute value of the difference between the features [13, 16, 21, 25-27, 30].

In addition to these two commonly used similarity functions, there are other functions that, despite their importance, have not received due attention in the feature selection and weighting studies. Thus Minkowski similarity function was also used and examined in the present work. Minkowski similarity function is a generalization of Euclidean and Manhattan functions, calculated by (3) [25, 26, 27, 34].

$$sim(p,p') = \frac{1}{\sqrt{\sum_{i=1}^{n} w_i Dis(f_i, f_i') + \delta}} \tag{1}$$

$$w_i Dis(f_i, f_i') =, \begin{cases} w_i(f_i - f_i')^2, & if\ f_i and f_i' are numerical or ordinal \\ w_i, & if\ f_i and\ f_i' are\ nominal and\ f_i \neq f_i' \\ 0, & if\ f_i\ and\ f_i' are nominal and\ f_i = f_i' \end{cases}$$

$$sim(p,p') = \frac{1}{\sum_{i=1}^{n} w_i Dis(f_i, f_i') + \delta} \tag{2}$$

$$w_i Dis(f_i, f_i') =, \begin{cases} w_i(f_i - f_i'), & if\ f_i and f_i' are numerical or ordinal \\ w_i, & if\ f_i and\ f_i' are\ nominal and\ f_i \neq f_i' \\ 0, & if\ f_i\ and\ f_i' are nominal and\ f_i = f_i' \end{cases}$$

$$sim(p,p') = \frac{1}{\sqrt[q]{\sum_{i=1}^{n} w_i Dis(f_i, f_i') + \delta}} \tag{3}$$

$$w_i Dis(f_i, f_i') =, \begin{cases} w_i(f_i - f_i')^q, & if\ f_i and f_i' are numerical or ordinal \\ w_i, & if\ f_i and\ f_i' are\ nominal and\ f_i \neq f_i' \\ 0, & if\ f_i\ and\ f_i' are nominal and\ f_i = f_i' \end{cases}$$

3.2. K-Nearest Neighbor (KNN)

KNN specifies a number (k) of projects closest to the current project. It is an important parameter of ABE, and affects the accuracy of effort estimation. Some authors opt for K=1 in the KNN algorithm, and the others recommend k equals two or three. [16, 19, 35]. The previous studies have shown that finding the optimal KNN is an important step, and will affect the ABE efficiency.

3.3. Adaptation Function

After identifying the most similar projects, the effort required for the target project can be estimated using the statistical methods termed the adaptation functions. The ABE approach uses the adaptation function as a mechanism for aggregation of the effort of most similar previous projects as the estimation effort of the current project [36]. The adaptation function is called "closed project" (CP) if only one project similar to the target project is selected. If more than two of the most similar projects are selected, various adaptation techniques such as the mean matching function, median matching function, and reverse distance weighted mean matching function can be used. Most studies use the weighted mean matching function, which is calculated by (4), where k is the number of similar projects, $\hat{E_0}$ is the estimated effort, E_i is the effort for project x_i and $s(x_0, x_i)$ is a measure of similarity between the target project x_0 and a similar project x_i [27, 28, 30].

$$E_0^{\wedge} = \sum_{i=1}^{k} \left(\frac{s(x_0, x_1)}{\sum_{i=1}^{k} s(x_0, x_i) E_i} \right) \qquad k = 1, 2 or 3 \tag{4}$$

4. Model Description

This work presents a weighting approach based on the hybrid PSO-SA optimization algorithm for ABE. As noted earlier, the feature weights are used in the ABE similarity function. Accordingly, the PSO-SA optimization algorithm is employed in order to determine the feature weights in the similarity determination step of ABE. The proposed model is implemented in two phases: training and testing. The feature weights are determined by PSO-SA in the training phase, and the proposed model is evaluated in the testing phase. The evaluation parameters and training and testing phases are described in the following sections.

4.1. Performance Measures

The performance measures are used for assessing the estimation methods. Several parameters have been utilized for this purpose in the previous works, among which three measures including the mean magnitude of relative error (MMRE), median magnitude of relative error (MDMRE), and PRED(0.25) are the most widely used. MMRE represents the average estimation error for all samples (training or testing), and MDMRE is the median error in the value determined by the algorithm relative to the actual effort of the samples. MMRE and MDMRE are calculated by (5) and (6), respectively [27, 30].

$$MMRE = \frac{1}{n} \sum_{i=1}^{n} \frac{|ActualEffort - EstimatedEffort|}{ActualEffort} \tag{5}$$

$$MDMRE = median(MRE) \tag{6}$$

Where n is the number of projects being evaluated. The lower the MMRE and MDMRE, the lower the algorithm's estimation error and the higher its accuracy [27, 30].

PRED(0.25), calculated by (7), represents the percentage of samples whose estimation error is less than or equal to 0.25 [27, 30].

$$PRED(0.25) = \frac{k}{n} \tag{7}$$

where k is the number of samples for which the difference between the effort estimated by the algorithm and the actual effort is less than or equal to 0.25, and n is the total number of samples evaluated [27, 30].

4.2. Training Phase

The first step in implementing and evaluating the proposed model on the dataset is data normalization through which the values of the features describing the project are brought into the range [0, 1]. The dataset is then randomly divided into the training and testing sets using the 3-fold method. The training and testing sets are used in order to "train" the model and to assess its performance, respectively. At the training step, each similarity function is tuned separately.

During the training process, the weights are dynamically presented to the similarity function by the PSO-SA optimization algorithm (Figure 1). Using the adaptation function and based on the most similar projects that were selected, the effort required for the target project is estimated, and the MRE performance measure is calculated based on the estimated effort and actual effort of the current project. This process is repeated and the efforts of all training projects are estimated. After processing the entire training dataset, MMRE, MDMRE, and PRED (0.25) are calculated and the fitness and cost of the proposed model (or more accurately, the fitness of the weights provided by PSO-SA at each iteration) are determined based on the values of performance parameters by (8).

$$Fitness = MMRE + MDMRE - PRED(0.25) \tag{8}$$

The training procedure using the PSO-SA algorithm is repeated three times to tune each similarity function. At the end of the training phase, the optimized weights of features in each similarity function are determined.

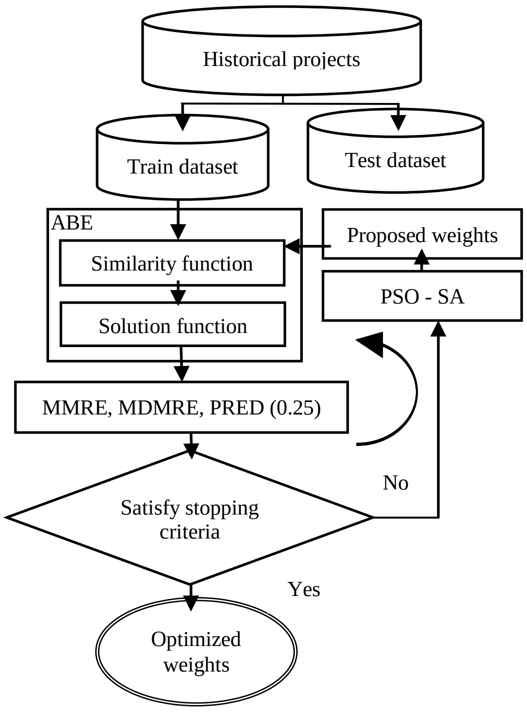

Figure 1. Flowchart of training phase for the proposed model.

4.2.1. PSO-SA

The PSO-SA algorithm is a combination of two optimization algorithms, namely PSO and SA. PSO is a population-based algorithm inspired by the movement of birds in a flock and performs a global exploration process in the search space. As mentioned in the Research Background, the PSO algorithm faces two challenges: slow convergence and entrapment in local optimization. On the other hand, the SA algorithm - which simulates the annealing process in metallurgy – performs a very good local search for each solution. By a combination of these two algorithms, therefore, the problem space can be well-covered by both the global and local search. First, a random initial answer is generated and the PSO algorithm starts searching the problem space (or, in other words, finding the optimal weight for features), and then these values are fed as the initial answers into the SA algorithm. SA prevents the answers from getting entrapped in the local optimization by performing a local search and making minor modifications in the weights generated by PSO, and the best and most optimal weights are thus provided for the features. The final output of SA is feed as the final feature weight into the ABE similarity function. Using a combination of the two optimization algorithms (PSO and SA), the advantages and strengths of both algorithms can be leveraged. Figure 2 depicts the PSO-SA algorithm.

```
(1) iter ←0, $cpt$ ← 0,
nitialize $swarm\_size$ particles
(2) stop_criterion←maximum number of function valuations or $Optimal\_solution$ is not attained
(3) while  Not stop_criterion  do
(4)   for each particle $i$ ← 1 to $swarm\_size$ do
(5)      Evaluate ( $rticle$( ) ) if the fitness value is better than the best fitness value (cbest) in history  then
(6)      Update current value as the new $cbest$.
(7) end
(8) end
(9) Choose the particle with the best fitness value in the neighborhood ($gbest$)
(10) for each particle $i$ ← 1 to $swarm\_size$ do
(11)              Update particle velocity
(12)              Enforce velocity bounds
(13)              Update particle position
(14)              Enforce particle bounds
(15)  end for
(16) if there is no improvement of global best solution  then
(17)     $cpt \leftarrow cpt + 1$
(18) end if
(19)  Update global best solution
(20)  $cpt$ ← 0
(21)   if  $pt = K$  then
(22)     $cpt$ ← 0
(23)     / /Apply SA to global best solution
(24)     iterSA ← 0, Initialize $T$
(25)     $current\_solution$ ← global_best_solution
(26) cu$rrent\_cost$ ←Evaluate($current\_solution$)
(27) while  Not SA_stop_criterion  do
(28)    while  inner-loop stop criterion  do
(29)
$Neighbor$ ←Generate($current\_solution$)
(30)     $Neighbor\_cost$ ← Evaluate($Neighbor$)
(31)     if Accept(current_cost, Neighbor_cost, $T$)  then
(32)     $current\_solution \leftarrow Neighbor$
(33) $current\_cost \leftarrow Neighbor\_cost$
(34) end
(35) iterSA ← iterSA + 1
(36) Update (global_best_solution)
(37)     end
(38) Update($T$)
(39) Update (SA_stop_criterion)
(40) end
(41) end
(42)  iter ← iter + 1, Update (stop_criterion)
(43) end
```

Figure 2. A summary of the PSO-SA algorithm.

4.3. Testing Phase

The accuracy and efficiency of the proposed model were examined in the testing phase. The weights proposed in the training phase for the ABE similarity function were used and evaluated on the testing dataset. At each iteration, a project was selected from the testing dataset, and its similarity with other test projects is determined by the similarity function and taking into account the weights obtained in the training phase; and then its effort is estimated by the adaptation function. The process is repeated, each time with one of the projects of the testing dataset, until there are no projects left untested. Finally, MMRE, MDMRE and PRED (0.25) are calculated for the evaluation purposes. These steps are depicted in Figure 3.

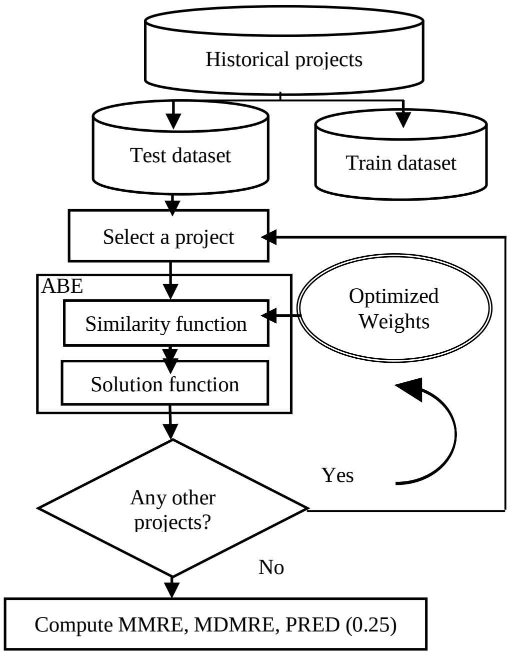

Figure 3. Flowchart of testing phase for the proposed method.

5. Experimental Results

5.1. Description of Datasets

The Albrecht and Desharnais datasets, used in almost all the previous studies, were selected for evaluating the proposed model.

The Albrecht dataset contains 24 projects developed with the third-generation languages. Completed in the 1970s by IBM, these include 8 projects written in COBOL, 4 written in PL1, and 2 in DMS. The dataset has 5 independent features including the number of inputs, number of outputs, number of queries, number of files, and number of lines of source code, and its dependent feature is the effort, expressed in terms of 1000 man-hours. Desharnais is one of the most commonly used datasets for effort estimation. The dataset is based on the Canadian projects (1989). It contains 81 software project samples, four of which have incomplete data, and thus only 77 projects are used. Each sample in this dataset is described by 11 features, of which 10 are independent and one is dependent. In the present work, effort (measured in person-hours) was used as the dependent variable [28, 30, 37, 38].

5.2. Implementation

As noted in the model description, the 3-fold cross-validation approach was used for evaluating the model, and the projects were randomly divided into the training and testing sets. This process was repeated thrice and the accuracy and efficiency of the model were calculated based on the average of the results of the three repeats. The ABE model and the hybrid PSO-SA optimization algorithm were implemented in MATLAB. The PSO and SA parameters were set as follows: particle-size$_{PSO}$ = 100, max-iter$_{PSO}$ = 100, c1$_{PSO}$ = 2, c2$_{PSO}$ = 2, max-iter$_{SA}$ = 1000, T-initial$_{SA}$ = 0.001, T-final$_{SA}$ = 0, in which particle-size$_{PSO}$ is the population size. The max-iter$_{PSO}$ and max-iter$_{SA}$ define the maximum number of iteration for PSO and SA respectively. The c1$_{PSO}$ and c2$_{PSO}$ are the “cognitive parameter” and “social parameter” respectively. The T-initial and T-final are the initial temperature and final temperature respectively (temperature T is considered to be decreased linearly from T-initial to T-final, during of the execution the algorithm).

5.3. Results of Albrecht Dataset

Table 1 shows the results of implementing various ABE approaches on the Albrecht dataset. These results were obtained by implementation of six ABE models (unweighted Euclidean, weighted Euclidean, unweighted Manhattan, weighted Manhattan, unweighted Minkowski, and weighted Minkowski similarity functions) together with four adaptation approaches (CP, Mean, Median, IRWM) and based on three evaluation measures (MMRE, MDMRE, and PRED (0.25) on the testing and training datasets. The results obtained show that the three weighted approaches are clearly more efficient, and provid more accurate estimates compared to the three unweighted approaches. The three similarity functions (Euclidean, Manhattan, and Minkowski) with optimal weights led to lower estimation errors (MMRE and MDMRE values) and higher PRED (0.25)s in comparison with the same similarity functions without weighting. The Manhattan and Minkowski similarity functions exhibited no significant difference but both had a higher efficiency than the Euclidean similarity function. The results also indicate that the K value affects the accuracy and efficiency of the models, with the lowest errors and the highest efficiencies of all the three weighted similarity functions obtained with K = 2 and mean adaptation function.

In order to demonstrate the improvement caused by the application of PSO-SA optimization for feature weighting, the "improved value" was calculated by (9) and (10) for the weighted ABE methods.

$$improved value = \frac{OV - EV}{OV} * 100\% \quad for\ MMRE and\ MdMRE \tag{9}$$

$$improved value = \frac{EV - OV}{OV} * 100\% \quad for\ \mathrm{Pred}(0.25) \tag{10}$$

where OV is the magnitude of each one of the performance measures for the ABE method without weighting, and EV represents the magnitude of each of the performance measures for ABE with optimal weights [27]. Table 2 shows the improvement of each one of the performance measures in the testing and training datasets (based on (9) and (10). The data in Table 2 are indicative of the improvement of both MMRE and Pred in both the testing and training datasets. The highest MMRE improvements are 44% in the testing phase and 36% in the training phase. Pred has also been significantly improved, up to a maximum of 201% in the training phase and 55% in the testing phase. MDMRE has improved in all cases except for the testing phase of the ABE model with weighted Euclidean function, with the maximum improvements being 36% in the testing phase and 48% in the training phase.

Table 1. Estimation results for different ABE models in the Albrecht dataset.

Methods	Number of projects	Adaptation technique	MMRE		MDMRE		PRED (0.25)	
			Training	Testing	Training	Testing	Training	Testing
UEue	1	Cp	0.83	1.22	0.65	0.31	0.19	0.40
	2	Mean	0.60	1.38	0.39	0.42	0.31	0.28
		WM	0.66	1.48	0.46	0.44	0.25	0.24
	3	Mean	0.53	1.44	0.37	0.34	0.25	0.25
		WM	0.59	1.50	0.41	0.42	0.25	0.38
		Median	0.59	1.50	0.44	0.36	0.19	0.38
WEue	1	Cp	0.46	1.11	0.40	0.45	0.37	0.40
	2	Mean	0.37*	0.94	0.22*	0.35*	0.61*	0.25
		WM	0.36	0.92	0.21	0.32	0.53	0.37
	3	Mean	0.40	1.20	0.29	0.48	0.46	0.25
		WM	0.39	0.90	0.23	0.28	0.52	0.38
		Median	0.48	0.92	0.40	0.34	0.28	0.38
UMan	1	Cp	0.71	1.26	0.54	0.56	0.25	0.38
	2	Mean	0.60	1.16	0.37	0.52	0.44	0.38
		WM	0.63	1.09	0.36	0.50	0.38	0.25
	3	Mean	0.62	1.36	0.43	0.51	0.13	0.13
		WM	0.61	1.40	0.39	052	0.25	0.38
		Median	0.67	1.38	0.48	0.40	0.06	0.40
WMan	1	Cp	0.43	1.03	0.40	0.39	0.33	0.25
	2	Mean	0.40*	0.84*	0.21*	0.25*	0.63*	0.50
		WM	0.36	0.92	0.20	0.49	0.58	0.38
	3	Mean	0.42	0.86	0.56	0.40	0.43	0.25
		WM	0.37	0.90	0.24	0.36	0.50	0.25
		Median	0.48	0.98	0.40	0.34	0.32	0.36
UMINK	1	Cp	0.67	1.47	0.42	0.53	0.19	0.18
	2	Mean	0.68	1.27	0.52	0.62	0.31	0.36
		WM	0.67	1.27	0.46	0.44	0.19	0.15
	3	Mean	0.64	1.20	0.38	0.35	0.13	0.15
		WM	0.62	1.17	0.41	0.30	0.25	0.25
		Median	0.69	1.17	0.46	0.36	0.13	0.19
WMINK	1	Cp	0.45	1.04	0.24	.34	0.56	0.25
	2	Mean	0.40*	0.52*	0.20*	0.39*	0.63*	0.50
		WM	0.44	0.59	0.17	0.29	0.60	0.38
	3	Mean	0.50	0.63	0.20	0.48	0.63	0.25
		WM	0.40	0.73	0.19	0.35	0.57	0.25
		Median	0.46	0.73	0.38	0.35	0.36	0.25

Table 2. Improvement of different ABE models for the Albrecht dataset.

Methods	MMRE		MDMRE		PRED (0.25)	
	Training (%)	Testing (%)	Training (%)	Testing (%)	Training (%)	Testing (%)
WEue - ABE	34.24	29.01	35.17	-0.58	90.48	7.24
WMan - ABE	35.93	27.15	23.08	36.69	148.65	16.24
WMink - ABE	33.17	44.09	47.64	9.33	200.55	54.89

As mentioned in Section 2, many researchers have used the PSO and GA optimization algorithms for feature weighting. Therefore, In order to comprehensively examine and validate the proposed approach, we compared it with two feature weighting approaches based on the PSO and GA optimization algorithms. Moreover, the proposed approach was compared with the reported results of some existing techniques in the literature: a support vector regression based on feature selection (SVR-FS) [32] and a differential evolution in analog-based estimation (DABE) [28].

With the Albrecht dataset, three of the various combinations including the weighted Euclidean function and mean adaptation function and KNN = 2, Manhattan similarity function and mean adaptation function and KNN = 2, and Minkowski function and mean adaptation function and KNN = 2 gave the best results. These three combinations were thus selected for comparison with the other weighting methods. Table 3 shows the estimates made by various weighting approaches on the Albrecht dataset. These results obtained indicate the superiority of the PO-SA weighting approach over the other approaches (except DABE) according to all the three performance measures in both the testing and training datasets. The PSO and GA weighting approaches exhibited a high performance in the training dataset but their performance in the testing dataset was not satisfactory, which is indicative of overlearning in these two approaches. AS seen in Table 3, the proposed model achieves outperforms compared to the SVR-FS model. DABE achieved MMRE = 0.02 and MDMRE = 0.02, less than our results. In the case of PRED (0.25) the PSO-SA approach with PRED (0.25) = 0.50 achieved the best results.

Table 3. Results of comparisons for the Albrecht dataset.

Methods		MMRE		MDMRE		PRED (0.25)	
		Training	Testing	Training	Testing	Training	Testing
PSO-SA	WEue	0.37*	0.94	0.22	0.35	0.61	0.25
	WMan	0.40	0.84	0.21	0.25*	0.63*	0.50*
	WMink	0.40	0.52*	0.20*	0.39	0.63*	0.50*
PSO	WEue	0.39	1.13	0.24	0.39	0.56	0.25
	WMan	0.39	1.14	0.22	0.35	0.57	0.38
	WMink	0.50	1.57	0.24	0.30	0.61	0.36
GA	WEue	0.39	1.09	0.23	0.33	0.58	0.38
	WMan	0.47	1.10	0.27	0.34	0.55	0.38
	WMink	0.46	1.55	0.23	0.30	0.60	0.38
DABE, 2017 [28]		0.10	0.02	0.001	0.02	0.17	0.37
SVR-FS, 2019 [32]		N/A	0.58	N/A	0.32	N/A	0.29

Figures 4 and 5 were drawn based on the three performance measures for the testing and training datasets, respectively, in order to enable a more accurate comparison of the results.

As it can be seen in Figure 4, which depicts the training phase for the Albrecht dataset (except the SVR-FS model, because its training data is not available), the DABE model has the lowest MMRE AND MDMRE. After that, the ABE methods with a weighted Euclidean similarity function have the lowest MMRE, and among the three weighting approaches, PSO-SA has the lowest MMRE followed by PSO. On the other hand application of the PSO-SA optimization algorithm for weighting prominently improved the PRED (0.25) measure. The ABE methods with Manhattan and Minkowski similarity functions, which are weighted with PSO-SA and GA, respectively, have lower MDMRE values relative to those weighted with PSO. Overall, the PSO-SA weighting approach led to better results compared to the PSO and GA approaches in the training phase on the Albrecht dataset.

The accuracies of different approaches in the testing phase on the Albrecht dataset are shown in Figure 5. The DABE and SVR-FS models have the lowest MMRE and MDMRE values. For the MMRE measure, weighting with PSO-SA is evidently the best approach compared to the GA and PSO approaches (for all the three similarity functions). MDMRE is the lowest for the ABE methods with Euclidean and Manhattan similarity functions weighted compared to the PSO-SA and GA approaches. Judging by the PRED (0.25) measure, the ABE methods with Manhattan and Minkowski similarity functions weighted with PSO-SA are the best approaches compared to the other options. It can be concluded from the testing and training results that the PSO-SA optimization algorithm has prominently improved the PRED (0.25) measures**.**

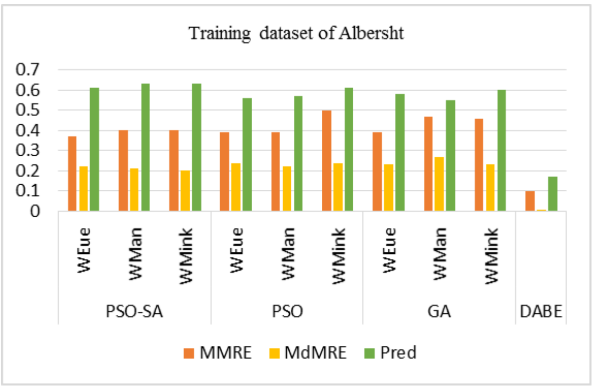

Figure 4. Accuracy comparisons for the Albrecht training dataset.

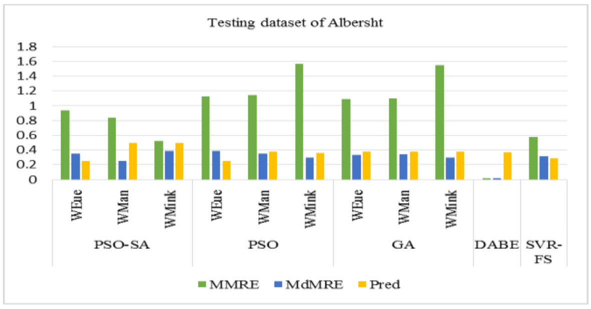

Figure 5. Accuracy comparisons for the Albrecht testing dataset.

5.4. Results of Desharnais Dataset

Table 4 presents the results of applying various ABE combinations to the Desharnais dataset, demonstrating the superiority of the ABE methods weighted with the Euclidean and Manhattan similarity functions over the other combinations in both the testing and training datasets. The ABE methods with all the three similarity functions (Euclidean, Manhattan, and Minkowski) performed better in the weighted mode than in the corresponding unweighted mode (as indicated by all the performance measures). Euclidean and Manhattan weighted ABE outperformed Minkowski weighted ABE, and there is no significant difference between the former two combinations according to the performance measures.

Table 5 represents the degree of improvement of the three weighted ABE approaches in the Desharnais dataset at both the training and testing phases. Improvement of all measures except MDMRE at the testing phase of weighted ABE with Manhattan similarity function is quite evident in all cases. The Minkowski similarity function led to the highest MMRE improvements (42.39% and 22.90% at the training and testing phases, respectively). For MDMRE, the maximum improvement is 35.51% at the training phase and 24.12% at the testing phase. The corresponding values for the Pred performance measure are 82.39% at the testing phase and 67.82% at the training phase.

Table 4. Estimation results for different ABE models in the Desharnais dataset.

Methods	Number of projects	Adaptation technique	MMRE		MDMRE		PRED (0.25)	
			Training	Testing	Training	Testing	Training	Testing
UEue	1	Cp	0.53	0.51	0.34	0.45	0.35	0.35
	2	Mean	0.59	0.47	0.30	0.33	0.45	0.46
		WM	0.55	0.46	0.30	0.29	0.43	0.46
	3	Mean	0.68	0.47	0.38	0.34	0.37	0.35
		WM	0.59	0.45	0.37	0.31	0.39	0.38
		Median	0.67	0.46	0.35	0.30	0.27	0.42
WEue	1	Cp	0.40	0.46	0.28	0.35	0.47	0.38
	2	Mean	0.38*	0.38*	0.22*	0.20*	0.55*	0.38*
		WM	0.39	0.45	0.25	0.31	0.50	0.46
	3	Mean	0.40	0.53	0.24	0.27	0.53	0.50
		WM	0.39	0.47	0.23	0.27	0.55	0.46
		Median	0.38	0.42	0.30	0.25	0.50	0.50
UMan	1	Cp	0.50	0.48	0.32	0.46	0.35	0.35
	2	Mean	0.63	0.48	0.32	0.34	0.41	0.38
		WM	0.56	0.45	0.26	0.34	0.45	0.42
	3	Mean	0.66	0.46	0.38	0.31	0.37	0.38
		WM	0.60	0.44	0.37	0.31	0.43	0.42
		Median	0.65	0.47	0.32	0.26	0.33	0.46
WMan	1	Cp	0.40	0.50	0.27	0.39	0.47	0.35
	2	Mean	0.39	0.40	0.22	0.21	0.52	0.54
		WM	0.36	0.44	0.24	0.32	0.52	0.46
	3	Mean	0.36*	0.38*	0.21*	0.24*	0.58*	0.50*
		WM	0.37	0.46	0.22	0.31	0.56	0.35
		Median	0.38	0.42	0.24	0.24	0.54	0.50
UMINK	1	Cp	0.43	1.02	0.27	0.57	0.45	0.12
	2	Mean	1.17	0.9	0.55	0.45	0.24	0.27
		WM	0.86	0.93	0.45	0.51	0.27	0.19
	3	Mean	0.98	0.82	0.38	0.38	0.35	0.31
		WM	0.93	0.87	0.43	0.47	0.35	0.23
		Median	1	0.95	0.46	0.55	0.25	0.19
WMINK	1	Cp	0.44	1.01	0.27	0.46	0.46	0.27
	2	Mean	0.39*	0.49*	0.21*	0.34*	0.57*	0.38*
		WM	0.47	0.84	0.31	0.38	0.51	0.35
	3	Mean	0.50	0.61	0.24	0.31	0.52	0.42
		WM	0.57	0.79	0.24	0.33	0.51	0.42
		Median	0.43	0.51	0.28	0.40	0.46	0.43

The proposed PSO-SA weighting approach was compared with the PSO weighting approach, GA weighting approach, and the reported results of some existing techniques in the literature: a case-based reasoning PSO (CBR-PSO) [27], a differential evolution algorithm based ABE

(DEABE) [30], a bee colony guided analogy-based estimation (BABE) [31], and SVR-FS [32]. The results of different techniques on the Desharnais dataset are summarized in Table 6. According to the results obtained, using the PSO-SA approach for feature weighting leads to better results compared to the other approaches except for DEABE and BABE for all the performance measures. MMRE of the BABE model is less than other techniques. MMRE of DEABE is similar to the PSO-SA approach. In the case of PRED (0.25), the DEABE model with PRED (0.25) = 0.81 achieved the best result, and the PSO-SA approach with PRED (0.25) = 0.50 has also an acceptable result.

Table 5. Improvements of different ABE models for the Desharnais dataset.

Methods	MMRE		MDMRE		PRED (0.25)	
	Training (%)	Testing (%)	Training (%)	Testing (%)	Training (%)	Testing (%)
WEue - ABE	34.60	3.77	25.00	17.48	40.37	12.36
WMan - ABE	36.52	6.36	27.47	-9.86	37.89	12.54
WMink - ABE	42.39	22.90	35.51	24.12	67.82	82.39

Table 6. Results of comparisons for the Desharnais dataset.

Methods		MMRE		MDMRE		PRED(0.25)	
		Training	Testing	Training	Testing	Training	Testing
PSO-SA	WEue	0.38	0.38	0.22	0.20	0.55	0.38
	WMan	0.36	0.38	0.21	0.24	0.58	0.50
	WMink	0.39	0.49	0.21	0.34	0.57	0.38
PSO	WEue	0.40	0.40	0.25	0.20	0.50	0.54
	WMan	0.38	0.40	0.22	0.27	0.55	0.46
	WMink	0.52	0.79	0.26	0.34	0.50	0.38
GA	WEue	0.41	0.40	0.22	0.26	0.53	0.50
	WMan	0.36	0.41	0.23	0.27	0.54	0.42
	WMink	0.45	0.54	0.24	0.76	0.51	0.31
CBR-PSO, 2018 [27]		0.46	0.58	0.29	0.40	0.46	0.35
SVR-FS, 2019 [32]		N/A	0.46	N/A	0.38	N/A	0.22
DEABE, 2020 [30]		N/A	0.37	N/A	0.31	N/A	0.81
BABE, 2020 [31]		0.06	0.09	N/A	N/A	0.29	0.47

The performance measures for the training and testing phases of the different approaches are compared in Figures 6 and 7, respectively.
Figure 6 shows the accuracy of the different models for the Desharnais dataset at the training phase. Among all the techniques, BABE is the best based on the MMMRE measure. In terms of MDMRE and PRED (0.25), the PSO-SA approach clearly outperforms the other techniques, and the PSO approach is in the next ranking. Accuracies of the different models for the Desharnais dataset at the testing phase are depicted in Figure 7. The superiority of the DEABE model is quite evident according to the PRED (0.25) measure, and the PSO-SA and PSO approaches are in the next rankings, respectively. In terms of MMRE and MDMRE, the PSO-SA approach clearly outperforms the other techniques but there is not much difference between the PSO-SA approach and DEABE at the testing phase.

6. Conclusions

In this work, a hybrid approach for feature weighting in the analogy-based software development effort estimation was proposed. We investigated the ABE model with three widely used similarity functions (Euclidean, Manhattan, and Minkowski) based on the feature weights optimized by PSO-SA. Two well-known datasets (Desharnais and Albrecht) were used in order to evaluate the proposed model. The performance of the PSO-SA approach was compared with those of the two weighting approaches frequently cited in the literature (PSO and GA). Moreover, the proposed model was compared with the reported results of some existing techniques in the literature based on the MMRE, MDMRE, and PRED (0.25) performance measures.
The Implementation results show, according to the performance measures, that the combined PSO-SA approach improved the efficiency of estimation at both the training and testing phases. The PSO-SA weighting approach for the ABE model with the Manhattan and Minkowski similarity functions, mean adaptation function and KNN = 2 in Albrecht dataset, and the PSO-SA weighting approach for the ABE model with the Euclidean and Manhattan similarity functions,

mean adaptation function and KNN = 2, 3 in the Desharnais dataset led to the best results and the most accurate estimates compared to the other approaches. These findings confirm that PSO-SA is a suitable approach for feature weighting with the ABE model. Further studies can examine other global and local optimization algorithms and their combinations for feature weighting, and the application of other optimization techniques, neural networks, fuzzy techniques, and ensemble learning and their combinations.

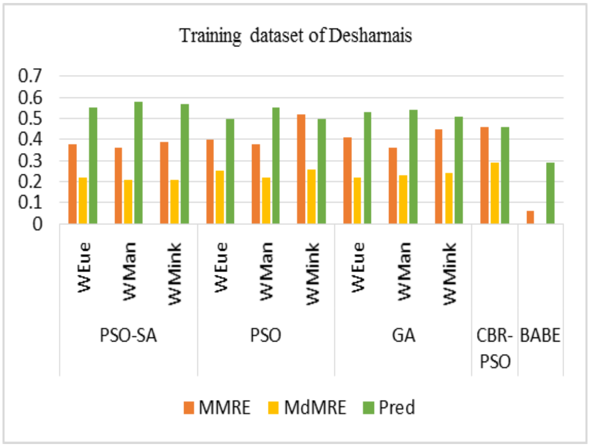

Figure 6. Accuracy comparisons for the Desharnais training dataset.

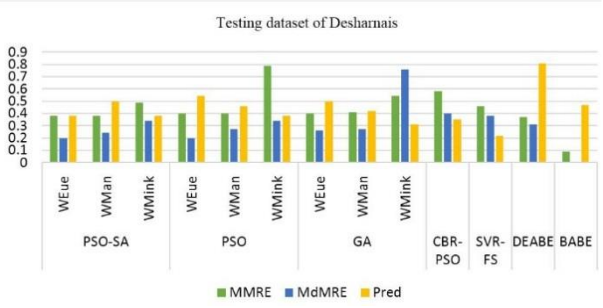

Figure 7. Accuracy comparisons for the Desharnais testing dataset.

References

[1] M. Shepperd and C. Schofield, “Estimating software project effort using analogies,” *IEEE Transactions on Software Engineering*, vol. 23, no. 11, pp. 736-743, 1997.

[2] I. Guyon and A. Elisseeff “An introduction to variable and feature selection,” *Journal of Machine Learning Research*, vol. 3, pp. 1157-1182, 2003.

[3] D. B. Skalak, “Prototype and feature selection by sampling and random mutation hill climbing algorithms,” *In 11th International Machine Learning Conference*, *ICML-94*, *Morgan Kau_mann*, pp. 293-301, 1994.

[4] Z. Shahpar et al., “Improvement of effort estimation accuracy in software projects using a feature selection approach,” *Journal of Advances in Computer Engineering and Technology*, vol. 2, pp. 31-38, 2016.

[5] E. Papatheocharous et al., “Feature Subset Selection for Software Cost Modelling and Estimation,” *Engineering Intelligent Systems*, vol. 18, 2010.

[6] B. B. Sigweni, “An Investigation of Feature Weighting Algorithms and Validation Techniques using Blind Analysis for Analogy-based Estimation,” Ph.D. dissertation, Brunel Univ., London, 2016.

[7] A. Kolcz and Y. Wen-tau “Raising the baseline for high-precision text classifiers,” *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 400-409, 2007.

[8] Y. Saeys, I. Inza and P. Larranaga, “A review of feature selection techniques in bioinformatics,” *Bioinformatics*, vol. 23, no. 19, pp. 2507-2517, 2007.

[9] D. Koller and M. Sahami “Toward optimal feature selection,” Tech. Rep. TR-1996-77, Stanford InfoLab. 1996.

[10] J. W. Keung and B. Kitchenham, “Optimizing Project Feature Weights for Analogy-based Software Cost Estimation using the Mantel Correlation,” *IEEE, 14th Asia-Pacific Software Engineering Conference*, *Aichi*, pp. 222-229, 2007.

[11] J. W. Keung, A. Kitchenham and D. R. Jeffery, “Analogy-X: Providing Statistical Inference to Analogy-Based Software Cost Estimation,” *IEEE Transactions on Software Engineering*, vol. 34, no. 4, pp. 471-484, 2008.

[12] J. Wen, S. Li. and L. Tang, “Improve Analogy-based Software Effort Estimation using Principal Components Analysis and Correlation Weighting,” *16th Asia-Pacific Software Engineering Conference*, pp. 179-186, 2009.

[13] E. Khatibi and V. Khatibi, “Model to estimation the software development effort based on in-depth analysis of project attributes,” *The Institution of Engineering and Technology*, vol. 9, pp. 109-118, 2015.

[14] J. Li and G. Ruhe, “Software effort estimation by analogy using attribute selection based on rough set analysis,” *International Journal of Software Engineering and Knowledge Engineering*, vol. 18, no. 1, pp. 1-23, 2008.

[15] J.Li and G. Ruhe, “Analysis of attribute weighting heuristics for analogy based software effort estimation method AQUA+,” *Empirical Software Engineering*, vol. 13, no. 1, pp. 63-96, 2008.

[16] Y. F. Li, M. Xie, and T. N. Goh, “A study of project selection and feature weighting for analogy based software cost estimation,” *The Journal of Systems and Software*, vol. 82, no. 2, pp. 241-252, 2009.

[17] Y. F. Li, M. Xie, and T. N. Goh, “A study of genetic algorithm for project selection for analogy based software cost estimation,” *IEEE, International Conference on Industrial Engineering and Engineering Management, Singapore*, pp. 1256-1260, 2007.

[18] S. J. Huang and N. H. Chiu, “Optimization of Analogy Weights by Genetic Algorithm for Software Effort Estimation,” *Information and Software technology*, vol. 48, pp. 1034-1045, 2006.

[19] Y. F. Li, M. Xie, and T. N. Goh, “Optimization of feature weights and number of neighbors for Analogy based cost Estimation in software project management,” *IEEE International Conference on Industrial Engineering and Engineering Management, Singapore*, pp. 1542-1546, 2008.

[20] A. L. I. Oliveira et al., “GA-based method for feature selection and parameters optimization for machine learning regression applied to software effort estimation,” *Information and Software Technology*, vol. 52, no. 11, pp. 1155-1166, 2010.

[21] V. Khatibi et al., “A hybrid method for increasing the accuracy of software development effort estimation,” *Scientific Research and Essays*, vol. 6, no. 30, pp. 6382-6382, 2011.

[22] P. Reddy, C. Hari and S. Rao “Multi- Objective Particle Swarm Optimization for Software Cost Estimation,” *International Journal of Computer Applications*, vol. 32, no. 3, pp. 13-17, 2011.

[23] M. Azzeh et al., “Pareto efficient multi-objective optimization for local tuning of analogy-based estimation,” *Springer, Neural Comput & Applic, November*, vol. 27, no. 8, pp. 2241-2265, 2015.

[24] R. D. A. Araujo, S. Soares and A. L. I. Oliveria, “Hybrid morphological methodology for software development cost estimation,” *Expert Systems with Applications*, vol. 39, pp. 6129–6139, 2012.

[25] V. Khatibiet et al., “A flexible method to estimate the software development effort based on the classification of projects and localization of comparisons,” *Empirical Software Engineering*, vol. 19, pp. 857-884, 2014.

[26] V. Khatibi et al., “A pso-based model to increase the accuracy of software development effort estimation,” *Software Quality Journal*, vol. 21, pp. 501-526, 2013.

[27] D. Wu, J. Li and C. Bao, “Case-based reasoning with optimized weight derived by particle swarm optimization for software effort estimation,” *soft computing*, vol. 22, 5299–5310, 2018.

[28] T. R. Benala and R. Mall, “DABE: Differential in Analogy-Based Software Development Effort Estimation,” *Swarm and Evolutionary Computation*, vol. 38, pp. 158-172, 2017.

[29] S. Ranichandra “Optimization Non-Orthogonal space distance using ACO in software cost estimation,” *Mukt shabd journal*, vol. IX, pp. 1592-1604, 2020.

[30] A. Khatibi, “An Intelligent Model to Predict the Development Time and Budget of Software Projects,” *Int. J. Non-linear Anal. Appl*. vol. 11, no. 2, pp. 85-102, 2020.

[31] A. M. Shah et al., “Ensembling artificial bee colony with analogy-Based Estimation to Improve software Development Effort Prediction,” *IEE Access*, vol. 8, pp. 58402-58415, 2020.

[32]A. Zakrani, M. Hain and A. Idri, “Improving Software Development effort estimating using Support Vector Regression and Feature Selection,” *IAES International Journal of Artificial Intelligence*, vol. 8, no. 4, pp. 399-410, 2019.

[33] E. Mendes, N. Mosley and S. Counsell, “A replicated assessment of the use of adaptation rules to improve web cost estimation,” *IEEE, International Symposium on In Empirical Software Engineering*, pp. 100-109, 2003.

[34] I. Angelis and I. Stamelos, “A Simulation Tool for Efficient Analogy Based Cost Estimation,” *Empirical Software Engineering*, vol. 5, no. 1, pp. 35-68, 2000.

[35] J. Keung, “Software development cost estimation using analogy: a review,” *IEEE, Software Engineering Conference, Australian*, pp. 327-336, 2009.

[36] S. K. Pal and S. C. K. Shiu, *Foundations of soft case-based reasoning*, John Wiley and Sons, New Jersey, 2004.

[37] J. Desharnais, “Statistical Analysis on the Productivity of Data Processing with Development Projects using the Function Point Technique,” Master's thesis, Quebec University, 1988.

[38] S. Beiranvand and M. A. Z. Chahooki, “Bridging the semantic gap for software effort estimation by hierarchical feature selection techniques,” *Journal of AI and Data Mining*, vol. 4, no. 2, pp. 157-168, 2016.

رویکرد ترکیبی PSO-SA برای وزن‌گذاری ویژگی‌ها در تخمین تلاش مبتنی بر قیاس پروژه‌های نرم-افزاری

زهرا شهپر[۱]، وحید خطیبی بردسیری[۲،*] و عمید خطیبی بردسیری[۲]

[۱] گروه مهندسی کامپیوتر، واحد کرمان، دانشگاه آزاد اسلامی، کرمان، ایران.

[۲] گروه مهندسی کامپیوتر، واحد بردسیر، دانشگاه آزاد اسلامی، بردسیر، ایران.

ارسال ۲۰۲۰/۱۰/۰۱؛ بازنگری ۲۰۲۱/۰۴/۲۶؛ پذیرش ۲۰۲۱/۰۶/۰۷

چکیده:

تخمین تلاش نرم افزار نقش مهمی در مدیریت پروژه های نرم‌افزاری دارد؛ و تخمین مبتنی بر قیاس رایج ترین مدل تخمین می‌باشد که تلاش پروژه نرم افزاری جدید را بر اساس شباهت آن با پروژه های پیشین تخمین می‌زند. شباهت پروژه ها با یکدیگر بر اساس مجموعه‌ای از ویژگی‌های پروژه ارزیابی می‌گردد که هر کدام از این ویژگی‌ها تاثیر متفاوتی در میزان شباهت بین پروژه ها و ویژگی تلاش دارند. این پژوهش به بررسی رویکرد ترکیبی PSO-SA برای وزن گذاری ویژگی‌های پروژه های نرم افزاری در تخمین مبتنی بر قیاس می‌پردازد. رویکرد پیشنهادی بر روی دو مجموعه داده معروف و شناخته شده از پروژه های نرم افزاری پیاده سازی و بررسی شده است. کارایی مدل پیشنهادی با الگوریم‌های بهینه سازی دیگر بر اساس معیارهای MMRE, MDMRE, PRED (0.25) مقایسه شده است. نتایج پژوهش نشان داد مدل‌های تخمین مبتنی بر قیاس وزن گذاری شده، تخمین تلاش-های دقیق تر و بهتری نسبت به مدل‌های تخمین مبتنی بر قیاس بدون وزن ایجاد می‌کنند؛ و رویکرد وزن گذاری ترکیبی PSO-SA عملکرد بهتر و دقیق تری در مقایسه با سایر رویکرد های وزن گذاری در هر دو مجموعه داده دارد.

کلمات کلیدی: تخمین تلاش نرم افزار، تخمین مبتنی بر قیاس، بهینه سازی وزن ویژگی، بهینه سازی ازدحام ذرات، شبیه سازی تبرید.