Computer Science and Information Systems 20(3):1157–1177 https://doi.org/10.2298/CSIS220830036C

A Novel Multi-objective Learning-to-rank Method for Software Defect Prediction

Yiji Chen, Lianglin Cao, and Li Song

Jiujiang University
Jiujiang 332005, China
chenyiji1984@jju.edu.cn
charlies_navy@aliyun.com
songli@jju.edu.cn

Abstract. Search-Based Software Engineering (SBSE) is one of the techniques used for software defect prediction (SDP), in which search-based optimization algorithms are used to identify the optimal solution to construct a prediction model. As we know, the ranking methods of SBSE are used to solve insufficient sample problems, and the feature selection approaches of SBSE are employed to enhance the prediction model's performance with curse-of-dimensionality or class imbalance problems. However, it is ignored that there may be a complex problem in the process of building prediction models consisting of the above problems. To address the complex problem, two multi-objective learning-to-rank methods are proposed, which are used to search for the optimal linear classifier model and reduce redundant and irrelevant features. To evaluate the performance of the proposed methods, excessive experiments have been conducted on 11 software programs selected from the NASA repository and AEEEM repository. Friedman's rank test results show that the proposed method using NSGA-II outperforms other state-of-the-art single-objective methods for software defect prediction.

Keywords: Search-Based Software Engineering, software defect prediction, multi-objective optimization algorithm, ranking method.

1. Introduction

Data quality [17], software metric, and classification algorithm are the main factors in constructing a promising prediction model [27]. Thereby, many problems need to be addressed in the process of construction. Machine learning algorithm is the most popular classification algorithm for SDP tasks, and it has a minimum requirement for the number of training samples [8]. Because of the expensive efforts, the samples are difficult to be collected in some systems, thus causing insufficient samples. Where engineers and researchers use more metrics to collect information from software features, redundant and irrelevant features can affect the efficiency of predictive models, leading to the curse of dimensionality problems. In addition, the number of normal samples is much more than the number of defective samples called the class imbalance problem. For this problem, Yu et al. [33] propose two extended resampling strategies that effectively handle imbalanced defect data to predict the number of defects.

Since SBSE was proposed by Harman [14], it has attracted more scholars and addressed the above problems in software defect prediction. A learning-to-rank method fits a linear classifier model to allow the data with insufficient samples problem. A feature selection method solves the curse-of-dimensionality and class imbalance problem. Especially, a search-based optimization algorithm is employed to search optimal feature subsets [25],[13], [24]. Although these approaches can address a single problem for SDP, there are few methods for a complex problem which is consisted of the above problems.

Even though ensemble predictors like Random Forest, k-Nearest Neighbors, Support Vector Machine, etc., are state-of-the-art for SDP tasks, these do not obtain a promising performance on NASA datasets. In our previous work [15], the experimental results show that the performance of these set predictors without the feature selection methods is quite the same as that of random predictors. In other words, obtaining a promising SDP model on NASA datasets is easier with other assist methods like feature selection. In addition, some metrics like AUC [12], F-Measure, etc., are always used to evaluate the performance of SDP models. However, fault-percentile-average (FPA) is frequently employed to estimate the performance of SDP models in ranking tasks. In this study, two multi-objective optimization algorithms are employed to obtain promising regression predictors in SDP ranking tasks, optimizing the parameters of regression predictors and searching for an optimal feature subset. FPA is used to evaluate the performance of the proposed methods in solving a complex problem, including insufficient samples, curse-of-dimensionality, and class imbalance.

The remainder of this paper is organized as follows: some related work is introduced in Section 2. The details of the SBSE multi-objective ranking method are described in Section 3. Section 4 shows the experimental studies conducted on NASA datasets. Finally, conclusions and future work are drawn in Section 5.

2. Literature Review

Software defect prediction is an important technique that can guide engineers to assign limited resources to focus on probable defect-proneness [6]. SBSE is one of the most popular approaches to improving the performance of SDP. Generally speaking, the ranking method and feature selection using SBSE effectively solve different problems in SDP tasks.

2.1. SBSE Ranking Method

The classification and ranking models are the two most frequently used prediction models. The goal of the ranking model is to predict an order of software modules based on the predicted scores of each module, and the goal of the classification model is to predict whether a software module has a defect or not. Where there is an insufficient sample problem, the ranking model is more efficient than the classification model in predicting the defect-prone software modules. The ranking method is used to build a ranking model, including the point-wise, pair-wise, and list-wise approaches. The SBSE ranking method, called Learning-To-Rank (LTR), is one of the list-wise approaches that optimizes performance measures to obtain a ranking model.

Yang et al. [29] first use CoDE to obtain the coefficients of the linear models instead of classification models by least squares (LS) and use FPA to evaluate the ranking performance. The experimental results on five data sets show that the proposed method is competitive with others ranking methods employing machine learning algorithms. Based on this investigation, they improve their studies that a feature selection method InfoGain is used to select a subset of metrics for the ranking task. Their empirical studies demonstrate that the feature selection method is beneficial to obtain a linear model based on data sets with large metrics [30]. However, the performance of the proposed LTR method depends too much on the effectiveness of InfoGain.

Buchari et al. [4] propose a novel LTR approach using the Chaotic Gausspian Particle Swarm Optimization algorithm (CGPSO) to optimize the ranking performance. Compared with the LTR using CoDE, it improves the performance of FPA on most of the eleven data sets. Peng et al. [20] propose the NABC algorithm to search the optimal coefficients of the linear models. It also obtains a better FPA than the LTR using CoDE. An empirical study of the LTR method [32] compares 23 ranking algorithms on 41 data sets from the PROMISE repository. In the comparison, the LTR method obtains the best ranking performance. Li et al. [16] consider the SDP problem as multiple goals to be optimized so that the revised NSGA-II is used to optimize the ranking performance FPA and prediction accuracy for the ranking task. The experimental results indicate that multiple-objective optimization algorithms can further improve the performance of the LTR.

Based on these investigations, one can use the LTR method to improve the ranking performance for SDP with insufficient samples. However, more research needs to be done on addressing SDP alongside other problems using the LTR method.

2.2. SBSE Feature Selection Method

Various software metrics have been proposed to provide vital information for constructing SDP models. Moser et al. [18] propose that adopting change metrics is more beneficial for predictive performance than static code attributes. Bell et al. [3] propose to increase the developer metrics to improve the prediction performance further. Choudhary et al. [7] propose that mixed metrics are the best choice the more metrics used in the dataset, the more serious the dimension problem. The SBSE feature selection method is an effective approach to reducing redundant and irrelevant features. Additionally, it can be used to search for an optimal feature subset to address class imbalance problems.

Balogun et al. [2] propose a study of performance analysis of feature selection methods using exhaustive and heuristic search. The performance of 7 search methods is evaluated using four classification algorithms on 5 data sets from the NASA repository. Although using search methods for feature selection can effectively improve prediction accuracy, the impact of the class imbalance problem on prediction performance cannot be evaluated.

Turabieh et al. [26] provide a novel feature selection algorithm with a layered recurrent neural network for software fault prediction. Genetic algorithm (GA), particle swarm optimization (PSO), and ant colony optimization (ACO) algorithms are randomly selected to search for an optimal feature subset in each iteration. The results of performing 5-fold cross-validation experiments on 19 data sets selected from the PROMISE repository using 20 metrics show that the proposed method improves the performance measure AUC of the classification models.

Proposed by Balogun et al. [1], the performance of 13 SBSE feature selection methods is verified on 7 data sets. It can be concluded that feature selection based on the meta-heuristic search methods outperforms others. An empirical study is shown by Nguyen et al. [19], because of the efficiency of swarm-based intelligence algorithms. These have been embedded in feature selection to search for an optimal feature subset. Rostami et al [21] showed more feature selection applications using swarm intelligence algorithms.

Based on the above investigations, the SBSE ranking methods enhance the performance of ranking models with insufficient samples problem. The SBSE feature selection approaches improve the efficiency of the training process to obtain a classification model by reducing redundant and irrelevant features. However, few studies consider the complex problem, which contains insufficient samples, class imbalance, and curse-of-dimensionality. Thereby, the issue is still an open question to be addressed. In this paper, we propose a novel multi-objective learning-to-rank method using two multi-objective optimization algorithms, which optimizes two objectives consisting of the optimal coefficients of the linear model and the optimal features subsets. Besides, the 10-fold cross-validation approach is used to verify the class imbalance problems.

3. Proposed Methodology

In this section, MSFFA [5] and NSGA-II [10] are employed to optimize the performance of the ranking model based on some datasets with a complex problem. Minimizing the size of feature subsets and maximizing FPA values are the two goals to achieve for complex problems. Additionally, the 10-fold cross-validation method is employed for the class imbalance problem. The 10-fold cross-validation makes little sense because splitting the datasets into ten folds does not affect the class imbalance problem and may only deteriorate it further. In this case, if the SDP model achieves promising performance, it can be shown that the adopted method solves well the class imbalance problem of SDP.

3.1. Optimization Algorithms

FA is an efficient swarm intelligence algorithm proposed by Yang [31]. In the search space, due to the self-learning and self-organizing capability of the population, it can effectively search for optimal solutions to objective functions by evaluating the fitness of each firefly location. The pseudo-code of FA is expressed in Alg. 1.

To enhance the efficiency of FA, two strategies were proposed in our previous work, including the multi-swarm strategy and the free strategy. While the multi-swarm strategy reduces the redundant attractions, the free strategy guides the population to adaptively change its state to balance the search ability between exploration and exploitation.

Each individual can update its position followed two status. In each interaction, where the weakness firefly moves according to the FA rules shown in Equ. 1, the brightness firefly follows free rules expressed in Equ. 2.

$$X_i(t+1) = x_i(t) + \beta_0 e^{(-\gamma r_{ij}^2)}(x_j(t) - x_i(t)) + \alpha\varepsilon \tag{1}$$

Where γ stands for the light absorption coefficient, β_0 expresses the attractiveness when γ = 0, r_{ij} represents the Euclidean distance between the firefly i and the firefly j, and $x_i(t)$ indicates the position of firefly i in t^{th} iteration. α is a control parameter and ε is a random vector in [0,1].

$$x_i(t+1) = x_i(t) + \mu(\frac{t+1}{t})^{times_i}(x_i(t) - x_r) \tag{2}$$

Algorithm 1 Firefly Algorithm

```
Initialize population and generate $N$ fireflies $i$, $i = 1, 2, \ldots, N$, the maximum evaluations MaxFEs;
while FEs $\leq$ MaxFEs do
    for i=1 to $N$ do
        for j=1 to $i$ do
            if fitness(i) $>$ fitness(j) then
                Update the location of the firefly $i$ and evaluate the fitness(i);
            end if
        end for
    end for
    Rank the fitness for all fireflies and find the best solution;
    t=t+1;
end while
Output the optimal solution;
```

$x_i(t)$ represents the position of the current firefly, x_r is the position of a firefly random selected in the entire population, t is the t^{th} iteration, μ is a random value between $[0, 1]$, $times_i$ is the number of times that i has moved, the new position is $x_i(t+1)$.

NSGA-II is the other multi-objective algorithm used in this study. We use distribution indexes of NSGA-II for crossover and mutation operators as 20, the crossover probability is 0.9 and the mutation probability is 0.05. The pseudo-code of NSGA-II is expressed in Alg.2

Algorithm 2 NSGA-II

```
Initialize: Set population size (number of solutions) to $N$, and randomly generate $N$ solutions that compose population $P_0$. Sort the solutions in $P_0$ using fast non-dominated sorting, and compute the non-dominated rank value of each solution.
Evaluate the fitness of the multi-objective function for each solution, and sort the solutions in $P_0$, and compute the rank value of each solution. Set the generation number $t$ = 0;
while $t \leq t_{max}$ do
    Use binary tournament selection to select individuals from Pt for crossover and mutation to generate the offspring population $Q_t$;
    Combine solutions in $P_t$ and $Q_t$ to get $R_t$ = $P_t \bigcup Q_t$;
    Sort $R_t$ based on non-domination rank value and crowding distance, and select $N$ elitist individuals to compose the new population $P_{t+1}$;
    t=t+1;
end while
Return Pareto-optimal solutions in $P_{t+1}$;
```

3.2. Learning-to-rank method Using Multi-objective Optimization Algorithm

Once the features of a software module X are extracted by d metrics, it can be expressed as

$$X = (x_1, x_2, \cdots, x_d) \tag{3}$$

The task of the proposed method is to predict the defect number of the module, which can be denoted as $f(X)$. We study a simple linear model which is good and realistic for SDP proposed by Weyuker et al [28]:

$$f(X) = \Sigma_{i=1}^{d} a_i x_i \tag{4}$$

If the parameters a_i is fixed, the prediction model is obtained.

Considering insufficient samples, we use optimization algorithms instead of machine learning algorithms to optimize the parameters to obtain prediction models. Obtaining an ordered list according to $f(x)$ is not a good choice. We use FPA to evaluate the obtained prediction models proposed by [28]. Different from other learning-to-rank methods [29][30][20]), we add another task of the proposed method is to select more essential features to improve the efficiency of the obtained prediction models. Once the performance of the obtained prediction model is enhanced by reducing the redundant and irrelevant features, the proposed method can also address the curse-of-dimensionality problem or class imbalance problem.

A similar study proposed by Yang et al.[30] uses filter-based feature selection methods before training a prediction model. We know those wrapper-based feature selection methods are more competitive than filter-based feature selection methods. We employ a wrapper-based feature selection method using optimization algorithms to optimize the goals of SDP for the feature selection task.

From the above goals of SDP for the ranking task, a multi-objective optimization algorithm is used to obtain a promising prediction model addressing insufficient samples, class imbalance, and curse-of-dimensionality. Therefore, two fitness functions are designed to evaluate the performance of the prediction models obtained by the multi-objective optimization algorithm. First, FPA is used to evaluate the ranking performance. The equation of FPA is defined as Equ.5.

$$f_{FPA} = \frac{1}{k}\Sigma_{m=1}^{k}\frac{1}{n}\Sigma_{i=k-m+1}^{k} n_i \tag{5}$$

Setting k as the modules number, f_1, f_2, ..., f_k listed in an increasing order of predicted defect number, n_i as the actual defect number of the modules i, and n = n_1 + n_2 + ... + n_k as the total number of defects. FPA is an average of the proportions of actual defects in the top i predicted modules, the larger FPA, the better the performance.

Next, The ratio of selected features to total features is another fitness function shown in Equ.6.

$$f_{ratio} = \frac{S}{T} \tag{6}$$

S represents the number of selected features, T as the total number of features. The solution representation is defined that the solution is encoded as a binary vector of length equal to the total number of features shown in Fig. 1

In Fig. 1, N indicates the total number of features, and R is the selection threshold. x_i is a vector between 0 to 1, which indicates the index of the i^{th} feature. So that one optimization algorithm can search the index of the features to compose an optimal feature subset. If the searched value x_i is greater than R, the i^{th} feature is selected so that y_i is set to 1. Otherwise, where the searched value x_j is less than R, the j^{th} feature is not selected

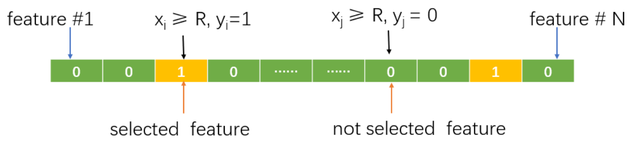


Fig. 1. An example of a feature selection solution.

that y_j is set to 0. R is set to 0.5 in this paper. Therefore, the smaller the ratio of selected features, the more effective the ranking model is.

Additionally, the last task of the proposed method is to verify the class imbalance problem, in which the 10-fold cross-validation method is employed. The 10-fold cross-validation makes little sense because splitting the datasets into ten folds does not affect the class imbalance problem and may only deteriorate it further. In this case, if the SDP model achieves promising performance, it can be shown that the adopted method solves well for the class imbalance problem of SDP.

3.3. Proposed Multi-Objective learning-to-rank Algorithm

This study uses MSFFA and NSGA-II as search techniques to obtain a set of Pareto-optimal solutions for prediction models. The procedure of the proposed algorithm is shown in Fig. 2

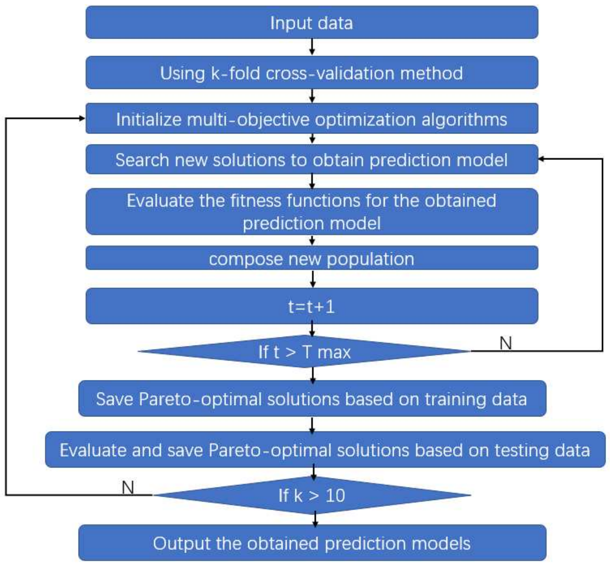

Input data
Using k-fold cross-validation method
Initialize multi-objective optimization algorithms
Search new solutions to obtain prediction model
Evaluate the fitness functions for the obtained prediction model
compose new population
t=t+1
If t > T max
N
Save Pareto-optimal solutions based on training data
Evaluate and save Pareto-optimal solutions based on testing data
N
If k > 10
Output the obtained prediction models

Fig. 2. The procedure of the proposed algorithm

First, the 10-fold cross-validation method is used in the search process to obtain a fair prediction model. Then, MSFFA and NSGA-II are employed to search for better solutions for both fitness functions. Last, when the termination condition is reached, a set of Pareto-optimal solutions are searched and saved. The average value of the best-level solutions is computed and saved in experimental results. The pseudo-code of the proposed algorithm is shown in Alg.3.

Algorithm 3 Proposed Multi-Objective learning-to-rank Algorithm

```
Initialize: load data, set data-length to $D$, set population size (number of solutions) to $N$, and randomly generate $N$ solutions that compose population $P_0$.
Using k-fold cross-validation method;
for k = 1: to 10 do
    Evaluate the fitness of the multi-objective function for each solution, and sort the solutions in $P_0$, and compute the rank value of each solution. Set the generation number $t$ = 0;
    while $FEs \leq MaxFEs$ do
        Use MSFFA (NSGA-II) to search new solutions that compose population $Q_t$;
        Evaluate the fitness of the multi-objective function for each solution in $Q_t$;
        Combine solutions in $P_t$ and $Q_t$ to get $R_t$ = $P_t \bigcup Q_t$;
        Sort $R_t$ based on non-domination rank value and crowding distance, and select $N$ elitist individuals to compose the new population $P_{t+1}$;
        t=t+1;
    end while
    Return Pareto-optimal solutions in $P_{t+1}$;
end for
```

4. Experimental Result and Discussion

In the experiments, 11 software programs selected from NASA [22] and AEEEM [9] are used to verify the performance of the proposed method for a complex problem. More details of the programs are shown in Tab.1.

Table 1. The details of the programs selected from NASA datasets and AEEEM datasets

	program	features	modules	defective modules	ratio
NASA	PC1	40	1107	76	0.074
	MC1	38	9466	68	0.007
	MW1	39	403	31	0.077
	JM1	21	10878	2102	0.193
	CM1	37	327	42	0.128
	KC1	21	2107	325	0.154
AEEEM	Eclipse JDT Core	15	997	463	0.464
	Eclipse PDE UI	15	1562	401	0.257
	Equinox framework	15	439	279	0.636
	Mylyn	15	2196	677	0.308
	Apache Lucene	15	691	103	0.149

From Tab.1, it can be seen that not only a few samples can be used to train prediction models, but also more metrics are used to extract software features for most software programs. Besides, the number of defective modules is far more than that of normal modules. Therefore, a complex problem may exist in most of the programs, that the proposed approach's performance can be evaluated by employing these software programs in this study.

To investigate the efficiency of the proposed approach, four single objective learning-to-rank algorithms are compared to build a ranking model for SDP, including CoDE, NABC, PSO proposed by D'Ambros et al.[11], and BSO proposed by Shi et al.[23]. A maximum of 1000 individual evaluations is the termination iteration condition for all algorithms. To verify the performance of the proposed multi-objective ranking methods, the 10-fold cross-validation method is employed to build ranking models on all datasets. For each dataset, the samples are divided into ten equal parts, and one of them is selected as the testing set in turn, and the remaining part is used as the training set to train a ranking model.

All experiments are performed on a computer with Intel Core i7-8700 CPUS, MATLAB language, and Windows 10 platform chosen for building the experimental environment.

The average of training FPA and the average of testing FPA are recorded in Table.2 and Table.3. From Table.2, one can see that the proposed multi-objective ranking methods obtain the best performance of training FPA on 4 out of 11 data sets, and PSO obtains the best performance of training FPA on the remaining data sets. In the Table.3, it shows that the proposed methods provide the best testing FPA to ranking models on 7 out of 11 data sets, and NABC receives the best performance of testing FPA on the remaining testing data sets.

Table 2. The performance of FPA using CoDE, NABC, PSO, BSO, NSGA-II, and MOMSFFA in training data sets

	Single-objective methods				Proposed multi-objective methods	
	CoDE	NABC	PSO	BSO	NSGA-II	MOMSFFA
PC1	0.5433	0.5422	0.5452	0.5443	**0.5457**	0.5413
MC1	0.8967	0.8956	**0.9001**	0.8971	0.8993	0.8945
MW1	0.5141	0.5133	0.5156	0.5141	**0.5157**	0.5127
JM1	0.7414	0.7378	0.7600	0.7426	**0.7604**	0.7169
CM1	0.7334	0.7316	0.7355	0.7336	**0.7358**	0.7321
KC1	0.7981	0.7986	**0.8015**	0.7980	0.7979	0.7927
jdt	0.8150	0.8200	**0.8251**	0.8210	0.7971	0.7885
pde	0.7590	0.7668	**0.7787**	0.7692	0.7575	0.7430
luce	0.8392	0.8494	**0.8618**	0.8468	0.8198	0.8110
equ	0.7799	0.7832	**0.7911**	0.7797	0.7880	0.7807
mylyn	0.7499	0.7744	**0.7951**	0.7581	0.6701	0.6021

Interestingly, both the best training FPA and the best testing FPA are obtained by the proposed multi-objective ranking methods on six software programs selected from the NASA repository. In other words, there may be a complex problem in these programs that multi-objective ranking methods can improve the performance of FPA and reduce the redundant features to enhance the efficiency of ranking models. Based on five programs selected from the AEEEM repository, the single-objective ranking methods are superior to the proposed multi-objective ranking methods. It is to say that reducing the number of features is not beneficial to improve the performance of FPA on AEEEM data sets.

Table 3. The performance of FPA using MOMSFFA, CoDE, NABC, PSO, BSO, and NSGA-II in testing data sets

	Single-objective methods				Proposed multi-objective methods	
	CoDE	NABC	PSO	BSO	NSGA-II	MOMSFFA
PC1	0.5439	0.5455	0.4875	0.5461	0.5479	**0.5529**
MC1	0.8914	0.8892	0.5789	0.8882	**0.8951**	0.8832
MW1	0.5265	0.5247	0.4970	0.5253	0.5280	**0.5310**
JM1	0.7213	0.7194	0.5972	0.7067	**0.7401**	0.7034
CM1	0.7335	0.7316	0.6448	0.7324	**0.7411**	0.7344
KC1	0.7931	0.7930	0.6291	0.7905	**0.7986**	0.7964
jdt	0.7992	**0.8082**	0.7537	0.7973	0.7771	0.7485
pde	0.7316	**0.7474**	0.7121	0.7438	0.7447	0.7284
luce	0.7982	**0.8098**	0.7662	0.7831	0.7802	0.7750
equ	0.7668	0.7652	0.6335	0.7682	**0.7863**	0.7752
mylyn	0.7460	**0.7567**	0.6874	0.6905	0.6674	0.6131

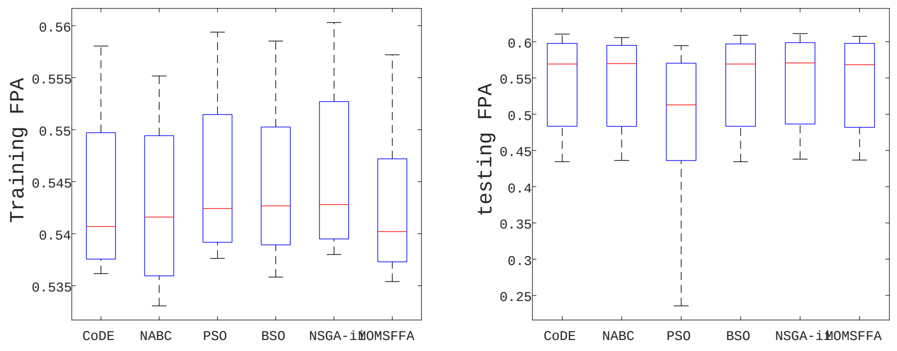
Additionally, the Friedman test is used in the significance test and employed in the rank-sum test. In the proposed study, it is used to obtain the rank order of the competitors' performance in SDP ranking tasks. Based on the numerical results of training FPA and testing FPA, the average rankings of the competitors are shown in Table.4. One can see that the proposed NSGA-II ranking method obtains the best performance to build ranking models for SDP. Although the proposed MOMSFFA ranking method is inferior to all single-objective ranking methods, the ranking models using MOMSFFA are more stable than those with single-objective ranking methods.

Table 4. Average Rankings of the algorithms based on the experimental results of FPA using Friedman test

Algorithm	Ranking
CoDE	3.4318
NABC	3.3636
PSO	3.5455
BSO	3.3864
NSGA-II	2.5909
MOMSFFA	4.6818

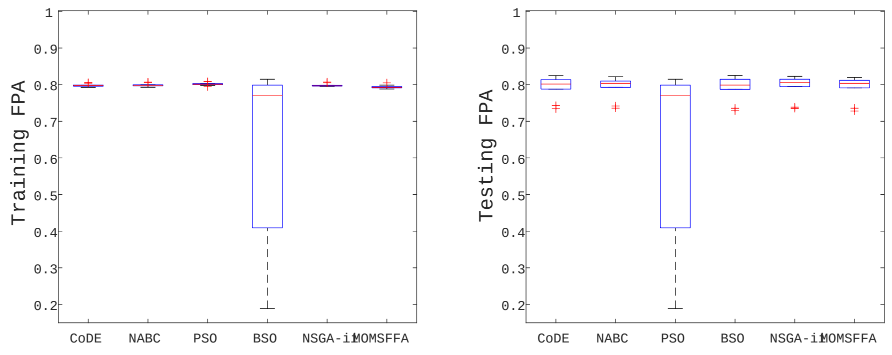
To analyze the performance of these competitors in-depth, an intuitive comparison of competitors in terms of FPA results can be seen that boxplots of the training FPA and the testing FPA based on each program are shown in Fig.3− 13.

To comprehensively investigate the performance of the proposed multi-objective ranking methods, the solutions of reducing features based on all data sets are recorded in Table.5. One can see that the proposed multi-objective ranking methods reduce features on all data sets. Thereby, the efficiency of building the ranking model is improved. Considering the performance of FPA, it can be concluded that the proposed multi-objective ranking method can address the complex problem of most of the programs selected from the NASA repository. The complex problem may not exist in the programs selected from the AEEEM repository that the single objective ranking methods are superior to the proposed approaches to improve the performance FPA of ranking models. However, sometimes



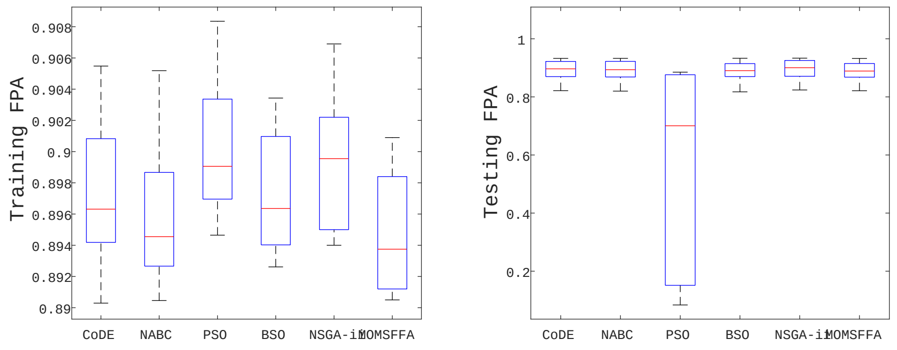
All methods achieve similar FPA performance. Using NSGA-II is slightly better than other methods in the training set and employing MOMSFFA slightly better than the other approaches in the testing set. The ranking model obtained by PSO shows performance degradation in testing set. In other words, compared with single-objective ranking methods, the proposed multi-objective ranking methods are beneficial to build a ranking model based on PC1.

Fig. 3. The performance FPA of ranking models based on program PC1 using CoDE, NABC, PSO, BSO, NSGA-II, and MOMSFFA



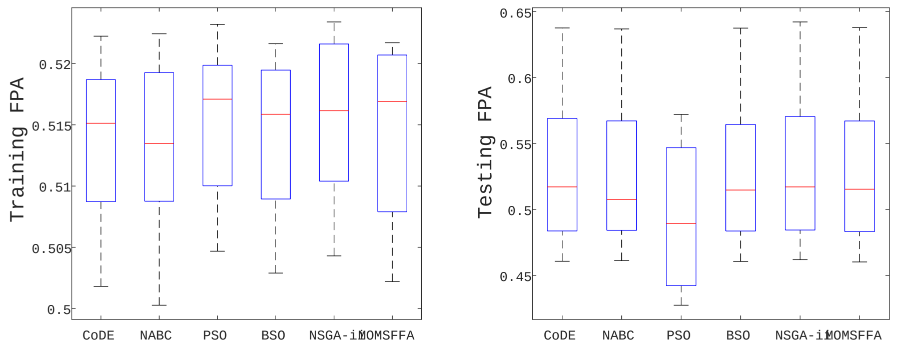
The proposed multi-objective ranking methods are slightly better than CoDE and NABC both in the training FPA and testing FPA, significantly better than BSO in training FPA, and superior to PSO in testing FPA. In other words, compared with single-objective ranking methods, the proposed multi-objective ranking methods are beneficial to build a ranking model based on KC1.

Fig. 4. The performance FPA of ranking models based on program KC1 using CoDE, NABC, PSO, BSO, NSGA-II, and MOMSFFA



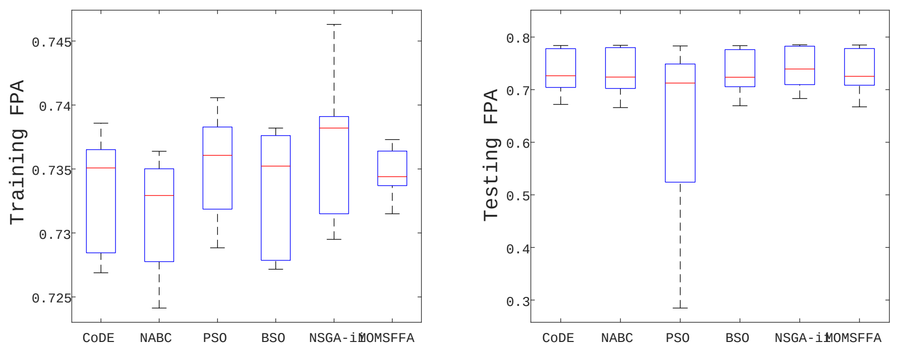
The proposed NSGA-II ranking method obtains the best performance both in the training FPA and testing FPA. Although the performance of the proposed MOMSFFA ranking method is inferior to that of the single objective ranking methods in training FPA, it is similar to these in the testing FPA. In other words, compared with single-objective ranking methods, the proposed ranking method using NSGA-II is beneficial to build a ranking model based on MC1.

Fig. 5. The performance FPA of ranking models based on program MC1 using CoDE, NABC, PSO, BSO, NSGA-II, and MOMSFFA



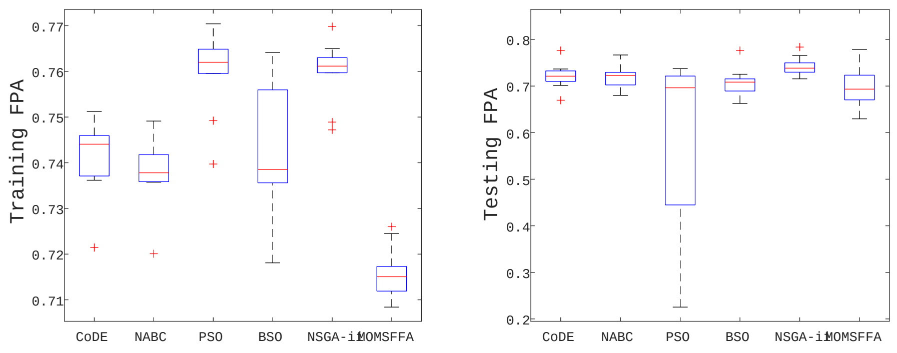
The proposed multi-objective ranking methods are superior to CoDE, NABC, and BSO both in the training FPA and testing FPA. Although the performance of the proposed MOMSFFA ranking method is similar to that of PSO in training FPA, it is significantly better than PSO in testing FPA. In other words, compared with single-objective ranking methods, the proposed MOMSFFA ranking method is beneficial to build a ranking model based on MW1.

Fig. 6. The performance FPA of ranking models based on program MW1 using CoDE, NABC, PSO, BSO, NSGA-II, and MOMSFFA



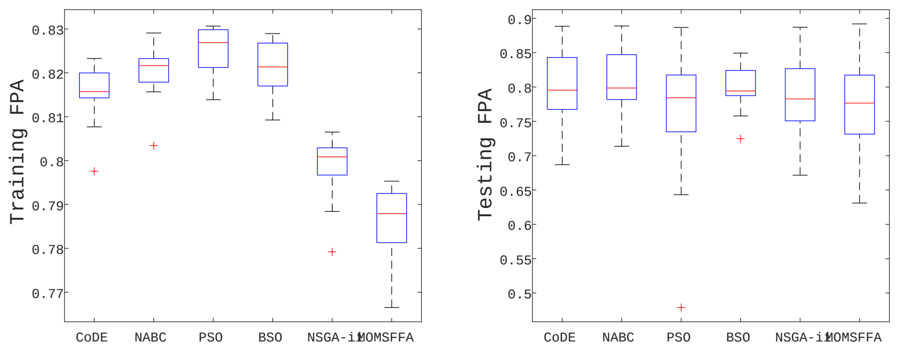
The proposed NSGA-II ranking method obtains the best performance in training FPA and in testing FPA. It can be concluded that compared with single-objective ranking methods, the proposed NSGA-II ranking method is beneficial to build a ranking model based on CM1.

Fig. 7. The performance FPA of ranking models based on program CM1 using CoDE, NABC, PSO, BSO, NSGA-II, and MOMSFFA



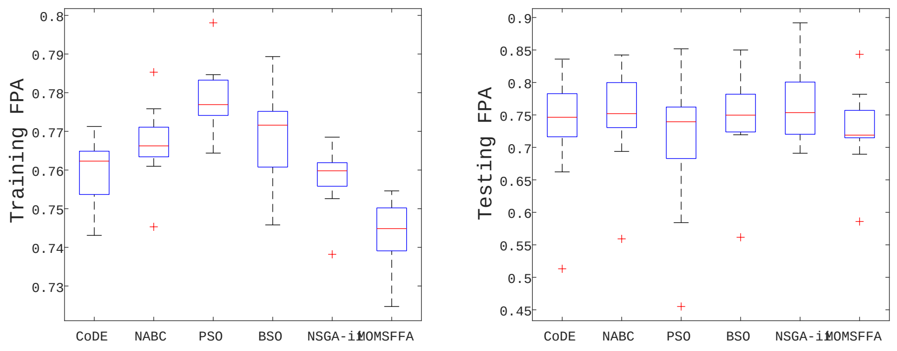
The proposed NSGA-II ranking method is superior to CoDE, NABC, and BSO both in the training FPA and testing FPA. Although the performance of the proposed NSGA-II ranking method is similar to that of PSO in training FPA, it is significantly better than PSO in the testing FPA. In other words, compared with single-objective ranking methods, the proposed NSGA-II ranking method is beneficial to build a ranking model based on JM1.

Fig. 8. The performance FPA of ranking models based on program JM1 using CoDE, NABC, PSO, BSO, NSGA-II, and MOMSFFA



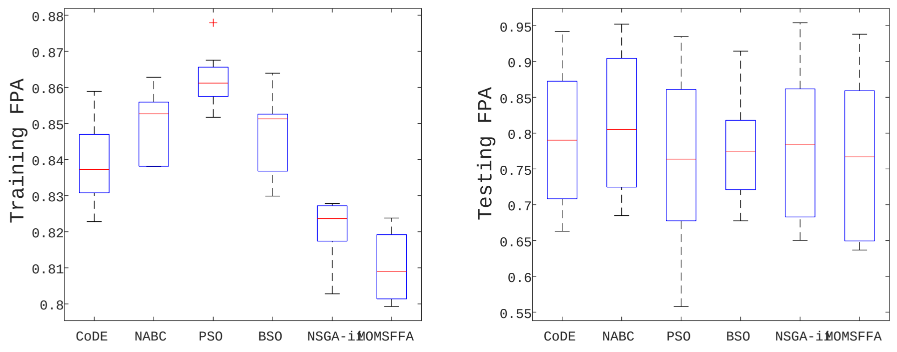
Where PSO obtains the best performance in the training FPA, NABC receives the best solutions in the testing FPA. Although the proposed multi-objective ranking methods are inferior to all single objective approaches in the training FPA and in the testing FPA, NSGA-II and MOMSFFA reduce the number of features from 15 to 4 and 2, respectively. In other words, where single objective methods can improve the ranking performance of FPA on JDT, multi-objective algorithms can enhance the efficiency of building ranking models on JDT.

Fig. 9. The performance FPA of ranking models based on program Eclipse JDT Core using CoDE, NABC, PSO, BSO, NSGA-II, and MOMSFFA



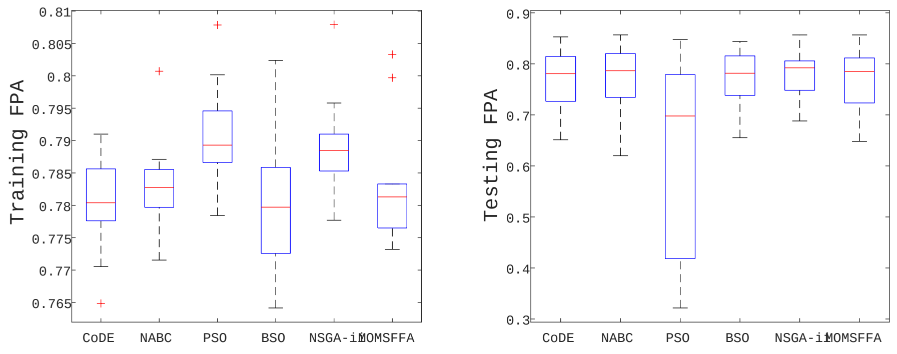
Where PSO obtains the best performance in training FPA, NSGA-II receives the best solutions in testing FPA. Considering the other objective that reducing the number of features, NSGA-II is the best choice to build a ranking model on PDE.

Fig. 10. The performance FPA of ranking models based on program Eclipse PDE UI using CoDE, NABC, PSO, BSO, NSGA-II, and MOMSFFA



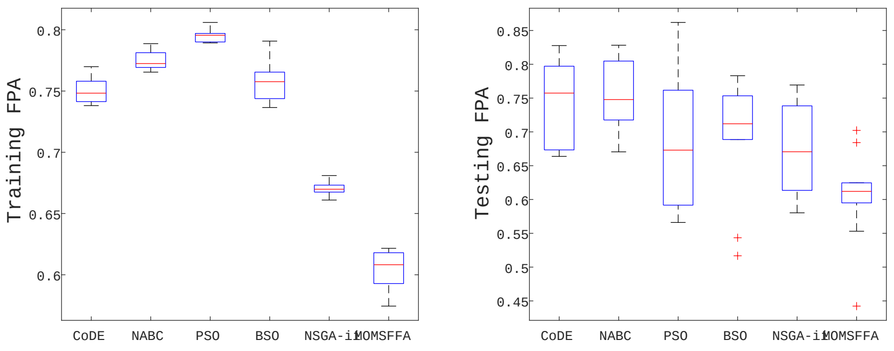
Where PSO obtains the best performance in training FPA, NABC receives the best solutions in testing FPA. The performance of the proposed multi-objective ranking methods is inferior to that of all single-objective approaches in training FPA but the performance of all methods is similar in testing FPA. In other words, the proposed multi-objective ranking methods are more stable than single-objective algorithms on Apache Lucene. Considering the other objective that mining the numbers of selected features, the proposed multi-objective ranking methods are beneficial to build ranking models on Apache Lucene.

Fig. 11. The performance FPA of ranking models based on program Apache Lucene using CoDE, NABC, PSO, BSO, NSGA-II, and MOMSFFA



Where PSO obtains the best performance in training FPA but provides the worse performance in testing FPA, NSGA-II receives similar solutions to PSO in training FPA but obtains the best solutions in testing FPA. Considering the other objective that mining the numbers of selected features, the proposed NSGA-II ranking method is beneficial to build ranking models on Equinox framework.

Fig. 12. The performance FPA of ranking models based on program Equinox framework using CoDE, NABC, PSO, BSO, NSGA-II, and MOMSFFA



Where PSO obtains the best performance in training FPA, NABC receives the best solutions in testing FPA. All single-objective approaches are superior to the proposed multi-objective ranking methods in training FPA and in testing FPA. In other words, single objective methods are beneficial to build ranking models to improve the performance APF on Mylyn.

Fig. 13. The performance FPA of ranking models based on program Mylyn using CoDE, NABC, PSO, BSO, NSGA-II, and MOMSFFA

feature selection is more important than using metrics that the proposed multi-objective ranking methods can be used to enhance the efficiency of building the ranking models on these datasets.

Additionally, The time cost of all competitors to build the ranking model is saved in Table.6. One can see thoes single-objective ranking methods are superior to multi-objective ranking methods on 8 programs. The proposed MOMSFFA ranking method is more efficient on KC1, jdt, and pde. Considering the similar solutions of all competitors to the complex problem of these 3 programs, MOMSFFA is the best choice to build ranking models.

From the above experimental results, the advantage of each optimization algorithm should be analyzed in the process of building ranking models. One can see that six styles of evolutionary algorithms are used to optimize the ranking performance based on 11 software programs. Although PSO performs advantages of global search ability and convergence to obtain better solutions to the training FPA on most of the AEEEM data sets, it receives the worst ranking performance of testing FPA. It is to say that it is not a good choice to use PSO to build ranking models based on these data sets. Considering the performance of ranking models obtained by CoDE, NABC, and BSO, where NABC has advantages of the global search ability to CoDE and BSO on AEEEM data sets, BSO performs the best global search ability than CoDE and NABC on NASA data sets, NABC obtains the best solutions on AEEEM data sets, and CoDE is superior to NABC and BSO on convergence for most of data sets. Compared with the above single-objective ranking methods, where NSGA-II has the advantage of global search ability on most NASA data sets, MOMSFFA not only searches stable solutions to build ranking models for PC1 and MW1 but also converges fast for KC1, jdt, and pde. It can be concluded that where CoDE is beneficial to reduce the time cost of building ranking models based on most

Table 5. The experimental results of selected features used to build ranking models on all data sets

	Single-objective methods				Proposed multi-objective methods	
	CoDE	NABC	PSO	BSO	NSGA-II	MOMSFFA
PC1	40	40	40	40	6	13
MC1	38	38	38	38	5	14
MW1	39	39	39	39	5	13
JM1	21	21	21	21	2	5
CM1	37	37	37	37	5	12
KC1	21	21	21	21	2	5
jdt	15	15	15	15	1	5
pde	15	15	15	15	2	4
luce	15	15	15	15	2	3
equ	15	15	15	15	1	4
mylyn	15	15	15	15	2	3

Table 6. The time cost using MOMSFFA, CoDE, NABC, PSO, BSO, and NSGA-II

	MOMSFFA	CoDE	NABC	PSO	BSO	NSGA-II
PC1	37.8	**36.2**	39.7	37.4	36.8	37.9
MC1	2135	**1671**	2260	1968	1840	1842
MW1	11.7	**7.1**	8.6	8.2	9.8	7.5
JM1	2640	**2164**	2796	2605	2565	2407
CM1	6.1	5.5	5.6	**5.3**	5.8	5.7
KC1	**100**	108	111	112	107	110
jdt	**27.9**	29.3	30.7	29.2	29.8	34
pde	**54.5**	58.1	61.5	58.3	58.5	62.6
luce	16.6	15.5	15.9	**15**	15.8	16.7
equ	6.1	5.6	5.4	**5.1**	5.9	5.5
mylyn	86.9	79.8	83.5	**77.8**	78.8	87.2

datasets, NABC is the best choice to build ranking models on AEEEM datasets, the proposed multi-objective algorithms obtain the best performance of global search ability on most of NASA data sets. The most important advantage of the proposed multi-objective ranking method is that it enhances the efficiency of ranking models by reducing redundant features. Thereby, the proposed multi-objective ranking methods can address the complex problem in NASA data sets.

The performance of an SDP model mainly depends on the quality of the datasets. In other words, where single-objective optimizers can obtain one goal in the SDP tasks, multi-objective optimizers can archive multi-goals in the SDP tasks. However, if the datasets have only one problem to solve, the performance of all the optimizers seems the same. It is to say that the more problems in the dataset, the greater the performance difference between the single-objective optimizer and the multi-objective optimizer.

5. Threats to Validity

In this section, we discuss three validity threats to the experimental results of our work. One threat to validity is that the proposed method may not obtain promising results on other datasets. The metrics have the most impact on the ranking task for different datasets. In other words, the metrics cause different effects on different datasets for SDP problems. Another threat to validity is that all the compared methods are single-objective evolutionary algorithms, which optimize a single goal on datasets for the ranking task. The performance of those methods depends on the quality of the datasets. Generally speaking, single-objective evolutionary algorithms obtain good performance on datasets with a single problem and worse results on datasets with complex problems. It is the reason that the proposed methods perform better than other competitors on NASA datasets and perform worse than other competitors on AEEEM datasets. The last threat to validity is that we use Friedman's rank test to statistically analyze the six competitors and use FPA as the evaluation measure. Our work can also use the Wilcoxon rank test and mean square error (MSE).

6. Conclusion

In this study, two multi-objective ranking methods are proposed to address a complex problem which is consisted of insufficient samples, curse-of-dimensionality, and class imbalance. Compared with four single objective algorithms based on 11 software programs, although learning-to-rank methods improve the ranking performance for SDP, curse-of-dimensionality and class imbalance are ignored in building ranking models for software defect prediction. Where the proposed multi-objective ranking methods are used to build ranking models, the ranking performance of these ranking models is enhanced by selecting the related feature subsets. In other words, the proposed multi-objective ranking methods can address the complex problem of SDP.

The main reason for the outperformance of the proposed methods on NASA datasets should be discussed. Where NASA uses 40 metrics to collect the information for programs, AEEEM employs only 15 metrics to dig the information for its software system. In other words, as the number of metrics increases, the relationship between them and the performance of prediction methods becomes more complex. From our experimental results, feature selection methods always enhance the performance of the SDP tasks, especially on datasets using a large number of metrics. However, the feature selection methods have little impact on the SDP tasks of AEEEM datasets. One can see that the proposed multi-objective ranking methods outperform single-objective ranking methods in addressing complex problems in datasets. It can be concluded that the optimized goal of reducing redundant and irrelevant features is effective for SDP tasks on datasets with a large number of metrics, which is missed in ranking tasks using single-objective optimization algorithms. In addition, NSGA-II and MSFFA win the best performance compared with other single-objective optimization algorithms for most SDP ranking tasks. It is to say that NSGA-II and MSFFA obtain a good balance between exploration and exploitation to solve complex problems in SDP ranking tasks.

In the future, the performance of the proposed methods will be verified by more publicly available datasets containing more features.

Acknowledgments. The author would like to thank those who gave help and guidance to this work. This work was funded by two projects, the National Natural Science Foundation of China (62266024) and the Science and Technology Foundation of Jiangxi Province (20202BABL202019).

References

1. Balogun, A.O., Basri, S., Jadid, S.A., Mahamad, S., Al-momani, M.A., Bajeh, A.O., Alazzawi, A.K.: Search-based wrapper feature selection methods in software defect prediction: an empirical analysis. In: Computer Science On-line Conference. pp. 492–503. Springer (2020)
2. Balogun, A.O., Basri, S., Abdulkadir, S.J., Hashim, A.S.: Performance analysis of feature selection methods in software defect prediction: a search method approach. Applied Sciences 9(13), 2764 (2019)
3. Bell, R.M., Ostrand, T.J., Weyuker, E.J.: The limited impact of individual developer data on software defect prediction. Empirical Software Engineering 18(3), 478–505 (2013)
4. Buchari, M., Mardiyanto, S., Hendradjaya, B.: Implementation of chaotic gaussian particle swarm optimization for optimize learning-to-rank software defect prediction model construction. In: Journal of Physics: Conference Series. vol. 978, p. 012079. IOP Publishing (2018)
5. Cao, L., Ben, K., Peng, H.: Enhancing firefly algorithm with multiple swarm strategy. Journal of Intelligent & Fuzzy Systems 41(1), 99–112 (2021)
6. Chen, L., Fang, B., Shang, Z., Tang, Y.: Tackling class overlap and imbalance problems in software defect prediction. Software Quality Journal 26(1), 97–125 (2018)
7. Choudhary, G.R., Kumar, S., Kumar, K., Mishra, A., Catal, C.: Empirical analysis of change metrics for software fault prediction. Computers & Electrical Engineering 67, 15–24 (2018)
8. Cowlessur, S.K., Pattnaik, S., Pattanayak, B.K.: A review of machine learning techniques for software quality prediction. Advanced Computing and Intelligent Engineering pp. 537–549 (2020)
9. D'Ambros, M., Lanza, M., Robbes, R.: An extensive comparison of bug prediction approaches. In: 2010 7th IEEE working conference on mining software repositories (MSR 2010). pp. 31–41. IEEE (2010)
10. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE transactions on evolutionary computation 6(2), 182–197 (2002)

11. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science. pp. 39–43. Ieee (1995)
12. Fawcett, T.: An introduction to roc analysis. Pattern recognition letters 27(8), 861–874 (2006)
13. Hancer, E., Xue, B., Zhang, M., Karaboga, D., Akay, B.: Pareto front feature selection based on artificial bee colony optimization. Information Sciences 422, 462–479 (2018)
14. Harman, M.: The relationship between search based software engineering and predictive modeling. In: Proceedings of the 6th International Conference on Predictive Models in Software Engineering. pp. 1–13 (2010)
15. Li, J., Song, L., Cao, L.: An improved firefly algorithm with distance guided selection strategy and its application. Journal of Intelligent & Fuzzy Systems 43(1), 889–906 (2022)
16. Li, X., Yang, X., Su, J., Wen, W.: A multi-objective learning method for building sparse defect prediction models. In: 2020 IEEE 20th International Conference on Software Quality, Reliability and Security (QRS). pp. 204–211. IEEE (2020)
17. Mauša, G., Galinac-Grbac, T., Dalbelo-Bašić, B.: A systematic data collection procedure for software defect prediction. Computer Science and Information Systems 13(1), 173–197 (2016)
18. Moser, R., Pedrycz, W., Succi, G.: A comparative analysis of the efficiency of change metrics and static code attributes for defect prediction. In: Proceedings of the 30th international conference on Software engineering. pp. 181–190 (2008)
19. Nguyen, B.H., Xue, B., Zhang, M.: A survey on swarm intelligence approaches to feature selection in data mining. Swarm and Evolutionary Computation 54, 100663 (2020)
20. Peng, H., Deng, C., Wu, Z.: Best neighbor-guided artificial bee colony algorithm for continuous optimization problems. Soft computing 23(18), 8723–8740 (2019)
21. Rostami, M., Berahmand, K., Nasiri, E., Forouzandeh, S.: Review of swarm intelligence-based feature selection methods. Engineering Applications of Artificial Intelligence 100, 104210 (2021)
22. Shepperd, M., Song, Q., Sun, Z., Mair, C.: Data quality: Some comments on the nasa software defect datasets. IEEE Transactions on Software Engineering 39(9), 1208–1215 (2013)
23. Shi, Y.: Brain storm optimization algorithm. In: International conference in swarm intelligence. pp. 303–309. Springer (2011)
24. Song, X.F., Zhang, Y., Guo, Y.N., Sun, X.Y., Wang, Y.L.: Variable-size cooperative coevolutionary particle swarm optimization for feature selection on high-dimensional data. IEEE Transactions on Evolutionary Computation 24(5), 882–895 (2020)
25. Tran, B., Xue, B., Zhang, M.: Variable-length particle swarm optimization for feature selection on high-dimensional classification. IEEE Transactions on Evolutionary Computation 23(3), 473–487 (2018)
26. Turabieh, H., Mafarja, M., Li, X.: Iterated feature selection algorithms with layered recurrent neural network for software fault prediction. Expert systems with applications 122, 27–42 (2019)
27. Wang, S., Liu, T., Nam, J., Tan, L.: Deep semantic feature learning for software defect prediction. IEEE Transactions on Software Engineering 46(12), 1267–1293 (2018)
28. Weyuker, E.J., Ostrand, T.J., Bell, R.M.: Comparing the effectiveness of several modeling methods for fault prediction. Empirical Software Engineering 15(3), 277–295 (2010)
29. Yang, X., Tang, K., Yao, X.: A learning-to-rank algorithm for constructing defect prediction models. In: International Conference on Intelligent Data Engineering and Automated Learning. pp. 167–175. Springer (2012)
30. Yang, X., Tang, K., Yao, X.: A learning-to-rank approach to software defect prediction. IEEE Transactions on Reliability 64(1), 234–246 (2014)
31. Yang, X.S.: Firefly algorithms for multimodal optimization. In: International symposium on stochastic algorithms. pp. 169–178. Springer (2009)

32. Yu, X., Bennin, K.E., Liu, J., Keung, J.W., Yin, X., Xu, Z.: An empirical study of learning to rank techniques for effort-aware defect prediction. In: 2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER). pp. 298–309. IEEE (2019)
33. Yu, X., Liu, J., Yang, Z., Jia, X., Ling, Q., Ye, S.: Learning from imbalanced data for predicting the number of software defects. In: 2017 IEEE 28th International Symposium on Software Reliability Engineering (ISSRE). pp. 78–89 (2017)

Yiji Chen earned the Ph.D. degree in the WonKwang University, korea. He is a lecturer at Jiujiang University, China. His main research interests include software testing, big data marketing, and artificial intelligence.

Lianglin Cao earned the Ph.D. degree in the Naval University of Engineering, Wuhan, China. He is a lecturer at Jiujiang University, China. His main research interests include swarm intelligence algorithm, software quality assurance.

Li Song is a lecturer at Jiujiang University, China. His main research interests include swarm intelligence algorithm and artificial intelligence.

Received: August 30, 2022; Accepted: May 04, 2023.