Hindawi
Scientific Programming
Volume 2023, Article ID 5585130, 14 pages
https://doi.org/10.1155/2023/5585130

Research Article

A Software Defect Prediction Approach Based on Hybrid Feature Dimensionality Reduction

Shenggang Zhang, Shujuan Jiang, and Yue Yan

School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221116, China

Correspondence should be addressed to Shujuan Jiang; shjjiang@cumt.edu.cn

Received 26 November 2022; Revised 11 June 2023; Accepted 20 June 2023; Published 20 July 2023

Academic Editor: Dongpo Xu

Software defect prediction (SDP) is designed to assist software testing, which can reasonably allocate test resources to reduce costs and improve development efficiency. In order to improve the prediction performance, researchers have designed many defect-related features for SDP. However, feature redundancy (FR) and irrelevance caused by the increasing dimensions of data will greatly degrade the performance of defect prediction. In order to solve the problems, researchers have proposed various data dimensionality reduction methods. These methods can be simply divided into two categories of methods: feature selection and feature extraction. However, the two categories of methods have their own advantages and limitation. In this paper, we propose a Hybrid Feature Dimensionality Reduction Approach (HFDRA) for SDP, which combines the two different kinds of methods, to improve the performance of SDP. HFDRA approach can be divided into two stages: feature selection and feature extraction. First, HFDRA divides the original features into several feature subsets through a clustering algorithm in the feature selection stage. Then, in the feature extraction stage, kernel principal component analysis (KPCA) is used to reduce the dimensionality of each feature subset. Finally, the reduced-dimensional data is used to build the prediction model. In the empirical study, we use 22 projects from AEEEM, SOFTLAB, MORP, and ReLink as experiment object. In this paper, we first compare our approach with seven baseline methods and three state-of-the-art methods. Then, we analyze the relationship between FR and prediction performance. Experiment results show that our approach outperforms the state-of-the-art data dimensionality reduction methods for defect prediction.

1. Introduction

Software testing is an essential stage in the entire software development and project maintenance process. With limited resources in the software development process, it is extremely challenging to test each module in detail. Software defect prediction (SDP) can discover hidden defects in software modules in advance and then reasonably allocate test resources. It can improve test efficiency and reduce software development costs [1, 2]. At present, most of the research works are to predict whether the modules in the software project contain defects, which can be regarded as a binary classification problem.

In order to improve the accuracy of SDP, a variety of artificially defined features have been proposed, such as CK metrics [3], process metrics [4], and network metrics [5]. In recent years, more and more researchers attempt to automatically extract the required valuable features from the source code in the field of software engineering [6, 7]. However, there likely exist redundant or irrelevant features among the extracted features, which may affect the performance of defect prediction. At present, researchers have proposed a variety of dimensionality reduction algorithm, through feature selection or feature extraction, to eliminate redundant or irrelevant features [8]. Feature selection chooses the best feature subset from the original feature set [9], while feature extraction attempts to map the original feature space into a suitable low-dimension feature space, so as to eliminate the redundant and irrelavent features and improve the prediction performance [10]. The two different types of methods have their own advantages and characteristics. Therefore, we propose a hypothesis that the effectiveness of reducing redundancy and irrelevant features can be improved by combining these two methods. Thus, this paper proposes a Hybrid Feature Dimensionality Reduction Approach (HFDRA) for SDP, which can be divided into feature selection and feature extraction two stages. In the feature selection stages, the original feature set is divided into multiple feature subsets by a clustering algorithm. In the feature extraction stages, each of feature subsets are reduced by a feature extraction algorithm. Finally, the reduced-dimensional data is used to train the defect prediction model. Compared with feature selection method, we did not remove any features in the first stage, which will inevitably not cause information loss. We use feature extraction method to reduce the dimension of data for each cluster in feature extraction stage. Compared with feature extraction method, because the features in each cluster are highly similar to each other, data dimensionality reduction for each cluster will not cause too much information loss. In summary, HFDRA approach can avoid the loss of information as much as possible in the process of data dimensionality reduction, and has better dimension reduction effect compared with feature selection and feature extraction two kinds of methods.

In the experiments, we use 22 projects from four databases named AEEEM, SOFTLAB, MORP, and ReLink to conduct experiments. In order to study the effectiveness of HFDRA approach, seven baseline methods and three state-of-the-art methods are selected as comparison methods. In addition, we analyze the relationship between feature redundancy (FR) and prediction performance. The three evaluation metrics area under the curve (AUC), F1, and Matthews correlation coefficient (MCC) are used to evaluate defect prediction performance.

In summary, the main contributions of this paper are as follows:

(1) Different data dimensionality reduction methods have different advantages and limitations. Therefore, this paper proposes a SDP approach—HFDRA, which combines feature selection and feature extraction two types of methods, to solve the FR or irrelevance problem. The approach contains two stages: feature selection and feature extraction.

(2) Through detailed empirical research on 22 projects from four defect databases, the effectiveness of the approach proposed in this paper is verified.

The rest of the paper is organized as follows. We introduce my motivation in Section 2. In Section 3, we elaborate the proposed HFDRA approach. The experimental data, strategies, evaluation metrics, results, and analysis are presented in Section 4. We briefly review related work in Section 5. We conclude our work and point out potential future research work in Section 6.

2. Motivation

Generally speaking, feature selection and feature extraction are two kinds of methods to solve problems of FR and irrelevance. Feature selection refers to selecting the best feature subset among all features, which means that the original feature space does not change. However, feature extraction refers to the generation of new features space with fewer dimensions by combination or mapping, which means that the original feature space must be changed. Feature extraction is usually independent of specific learning model, while some feature selection methods are closely related to specific learning model. Feature extraction can not only reduce dimension, but also achieve the purpose of increasing instances density and noise reduction [11]. If the two kind methods have the same effect on prediction performance, there is no need for a hybrid approach. To confirm the need for a hybrid approach, we analyze how the two kinds of methods perform on different projects.

To do this, we use nine projects in MORPH dataset as experimental objects. Support Vector Machines (SVM) implemented with *scikit-learn* library is selected as the classifier [12]. We use AUC, F1, and MCC three evaluation metrics, which are commonly used in SDP, to evaluate the performance of defect prediction. We use information gain (IG) [13] as feature selection method and kernel principal component analysis (KPCA) [14] as feature extraction method. The number of feature dimensions are reduced to 2/3 of its original number. The experimental results of different methods on nine projects are shown in Table 1.

Table 1: Experimental results of two kinds of dimensionality reduction methods.

Projects	Metrics	ALL	IG	KPCA
Ant	AUC	0.924	0.924	0.944
	F1	0.939	0.939	0.960
	MCC	0.854	0.854	0.906
Arc	AUC	0.772	0.772	0.736
	F1	0.782	0.787	0.756
	MCC	0.547	0.551	0.480
Camel	AUC	0.964	0.978	0.957
	F1	0.961	0.976	0.954
	MCC	0.927	0.955	0.912
Poi	AUC	0.776	0.790	0.790
	F1	0.720	0.735	0.735
	MCC	0.532	0.559	0.559
Redktor	AUC	0.801	0.777	0.869
	F1	0.778	0.745	0.852
	MCC	0.598	0.557	0.733
Tomcat	AUC	0.850	0.853	0.850
	F1	0.841	0.853	0.850
	MCC	0.699	0.714	0.708
Velocity	AUC	0.782	0.780	0.780
	F1	0.787	0.794	0.794
	MCC	0.563	0.558	0.558
Xalan	AUC	0.807	0.817	0.838
	F1	0.789	0.804	0.828
	MCC	0.615	0.633	0.675
Xerces	AUC	0.789	0.544	0.795
	F1	0.795	0.788	0.792
	MCC	0.611	0.601	0.596
AVE	AUC	0.829	0.830	0.840
	F1	0.821	0.824	0.836
	MCC	0.661	0.665	0.681

Table 1 shows project name (Column 1), evaluation merits (Column 2), baseline method (ALL) prediction performance (Column 3), IG method prediction performance (Column 4), and KPCA method prediction performance (Column 5). The baseline method (ALL) refers to use the raw data without any data processing. From Table 1, what stands out in the table is that both mean value of IG and KPCA two methods exceed ALL method. This implies that both feature selection and feature extraction methods can improve the prediction performance. Further observation shows although the mean values of KPCA are the best, IG and KPCA have different performances in different projects. Specifically, the feature selection method (IG) performs the best on four out of nine projects selected in this paper, while the feature extraction method (KPCA) performs the best on the other projects, and the two methods have the same performance only on one project. It is a disappointing result that the prediction performance of ALL method is better than IG or KPCA in some projects under some evaluation metrics. We think there may be two reasons for this result. First, randomness may bias the results; second, in the original feature space of some projects, the two types of instances may be well distinguished, while IG and KPCA methods make the boundary between the two types of samples blurred, resulting in a certain degree of decline in prediction performance. However, IG or KPCA is still better compared with ALL in most projects. These results suggest that the two kinds of methods have their own advantages in different projects from the whole. Inspired by this, this paper proposes a hybrid dimensionality reduction method to improve the performance of defect prediction by combining the two types of data dimensionality reduction methods.

3. Our Approach

In order to introduce the SDP approach proposed in this paper, we first need to define the correlation between features and defect classes (FC-correlation) and the correlation among features (FF-correlation).

Definition 1 (FC-correlation): The correlation between any feature f_i and defect classes (defective or nondefective) is defined as the FC-correlation of f_i, denoted as FC (f_i).

Definition 2 (FF-correlation): The correlation between any two features f_i and f_j is defined as the FF-correlation of f_i and f_j, denoted as FF (f_i, f_j).

The value range of FC(f_i) is [0, 1], the larger the value is, the more relevant the feature f_i is to defect class. If FC(f_i) = 0, it means that there is no correlation between the feature f_i and the defect class. Otherwise, if FC(f_i) = 1, it means that there is a complete correlation between the feature f_i and the defect class. Similarly, the value range of FF(f_i, f_j) is also [0, 1]. Larger value of FF(f_i, f_j) indicates more relevance between the two features.

3.1. FC-Correlation and FF-Correlation Measure. FC-correlation is used to measure the correlation between any features and defect classes. Researchers have proposed a variety of methods to calculate FC-correlation. These methods can be roughly divided into three categories: based on statistical theory, information entropy, and cases [15]. According to the previous research work, the methods based on information entropy can achieve better prediction performance. Liu et al. [16] considered IG, CS, and RF as FC-correlation measure. They found that IG can achieve relatively better prediction performance by empirical research. Therefore, IG is used to measure the correlation between any features and defects classes in this paper. IG is calculated as follows:

$$IG(Y|X) = H(Y) + \sum_{x\in X} P(x) \sum_{y\in Y} P(y|x) \log_2 P(y|x). \quad (1)$$

$P(x)$ represents the prior probability of variable X and $P(y|x)$ represents the posterior probability when the value of random variable X is x and the value of random variable Y is y. The calculation formula of $H(y)$ is defined as follows:

$$H(y) = -\sum_{y\in Y} P(y) \log_2 P(y). \quad (2)$$

FF-correlation represents the degree of correlation between any two features. There also exists many researches on calculating FF-correlation. The proposed FF-correlation calculation methods were similar to the previous described FC-correlation calculation methods. Maximum information coefficient (MIC) is optimized on the basis of MI (mutual information). MIC can measure not only the linear functional relationship between variables, but also the complex nonlinear functional relationship between variables. Considering the complexity of the relationship between any two features, this paper utilizes MIC to measure the correlation between any two features according to the suggestion of Tong et al. [17].

3.2. KPCA. PCA is a commonly used dimensionality reduction method. However, it is powerless for nonlinear complex data. KPCA solves the above problems of PCA by introducing the kernel function. The main principle of KPCA is to map all data into a high-dimensional space so that the data is linearly separable in this space, and then use PCA to reduce the dimension in this high-dimensional space.

Given a dataset $X = x_1, x_2, \ldots, x_n$, where $x_i = [x_{i1}, x_{i2}, \ldots, x_{ik}]$ denotes a instance and x_{ij} denotes jth feature of x_i. Suppose the dataset X is mapped to a new space $\varphi(X)$ by a nonlinear function φ and the new space denoted by F. Each instance $\varphi(x_i)$ in the new sample space F needs to be centralized:

$$\varphi(x_i) = \varphi(x_i) - \frac{1}{n}\sum_{j=1}^{n} \varphi(x_j). \quad (3)$$

The covariance matrix C of the mapped data $\varphi(X)$ is:

$$\varphi(X) = \frac{1}{n}\sum_{i=1}^{n} \varphi(x_i)\varphi(x_i)^T. \quad (4)$$

We diagonalize the covariance matrix C as the same as PCA, which can be viewed as solving the eigenvalue problem of the covariance matrix:

$$\sum_{i=1}^{n} \varphi(x_i)\varphi(x_i)^T p = \lambda p. \tag{5}$$

We divide both sides of Equation (5) by λ to get:

$$p = \frac{1}{\lambda}\sum_{i=1}^{n} (\varphi(x_i)|\varphi(x_i)^T p|). \tag{6}$$

Equation (6) indicates that there exits cofficients a_1, a_2, …, a_n that linearly express the eigenvectors p with $\varphi(x_1), \varphi(x_2), \ldots, \varphi(x_n)$:

$$p = \sum_{i=1}^{n} a_i \varphi(x_i). \tag{7}$$

Since the specific form of the nonlinear function φ cannot be obtained, we introduce kernel function $\kappa(x_i, x_j) = \varphi(x_i)\varphi(x_j)^T$. We can get the following equation by substituting Equation (7) and kernel function into Equation (5) and simplification.

$$K\alpha = \lambda\alpha. \tag{8}$$

In Equation (8), K is the kernel matrix corresponding to the kernel function κ, where $K_{ij} = \kappa(x_i, x_j)$. The reduced dimension data can be obtained by solving all nonzero eigenvalues in Equation (8) finally.

Considering the complex nonlinear relationship among features, KPCA is used to reduce the dimension of the feature in this paper.

3.3. HFDRA Method. The overall framework of the approach proposed in this paper is shown in Figure 1. Circles of the different colors represent different types of features, that is, features are independent of each other, whereas circles of the same color are the opposite in Figure 1. Triangles represent the features processed by the feature extraction method. HFDRA approach can be divided into two stages: feature selection and feature extraction. First, in the feature selection stage, all features are automatically divided into several clusters by k-medoids clustering algorithm [18] according to FC-correlation vector and FF-correlation matrix. The correlation among features in each cluster is higher, that is, the redundancy is higher. The correlation between different clusters is lower, conversely, the redundancy is lower. Then, in the feature extraction stage, we use the feature extraction algorithm—KPCA to reduce the dimension of each cluster of features, so as to keep the necessary key information and remove useless or redundant information to improve the classification ability of features. Finally, the data of each cluster is combined to get the final data. In order to more clearly illustrate our approach, there is dataset D = {[[2, 3, 6, 8, 9, 12], [4–6, 8–10], [4, 5, 8, 9, 12]]} that contains three instances. Each instance in dataset D is represented by six features. In the feature selection stage, we use k-medoids clustering algorithm to divide six features into two clusters. The clustering result C = {[[2, 6], [9, 10], [4, 5]], [[3, 8, 9, 12], [4–6, 8], [8, 9, 12]]}, which means that the first two features are classified into one category and the last four features are classified into another category. Assume that the number of features needs to be reduced from six to four, which means that 2/3 of the features are retained. Therefore, each cluster in C also needs to retain 2/3 of the features. In the feature extraction stage, each cluster in C is reduced to the specified dimension by using the KPCA. The reduced dimension results of all clusters are {[[−0.41], [0.83], [−0.41]], [[−0.41, 0.71, 0.00], [0.82, −0.00, 0.00], [−0.41, −0.71, 0.00]]}. Finally, the reduced dimension data of different clusters are combined to get the final training data that is [[−0.41,−0.41, 0.71, 0.00], [0.83, 0.82, −0.00, 0.00], [−0.41, −0.41, −0.71, 0.00]].

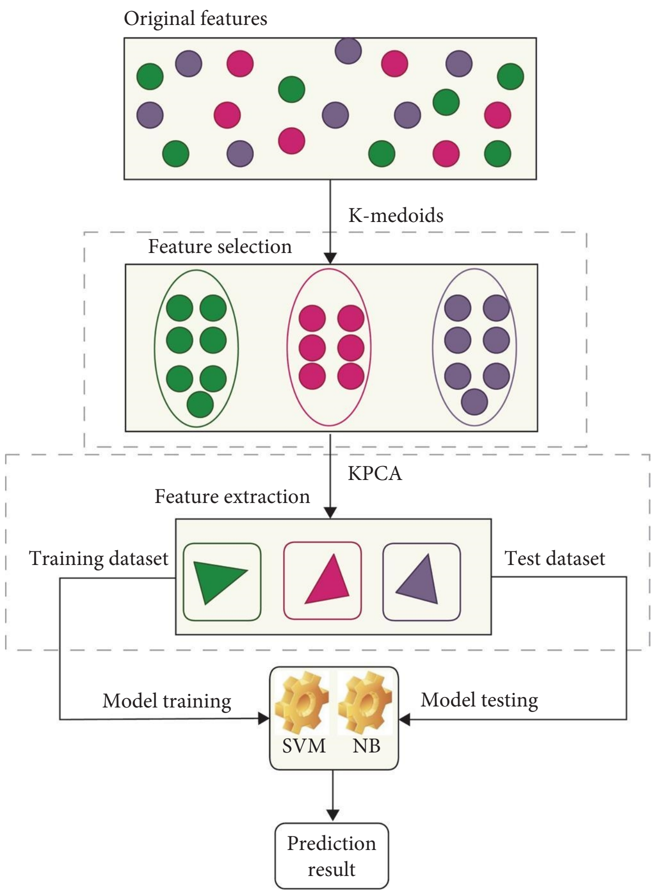


Figure 1: Overall framework of the HFDRA approach.

The details of HFDRA approach proposed in this paper are shown in Algorithm 1. The input of HFDRA method is the original data, FC-correlation measurement method, FF-correlation measurement method, the number of clusters and dimensions after dimensionality reduction, the output is the data after dimensionality reduction. Algorithm 1 can be divided into feature selection (Lines 1–9) and feature extraction (Lines 10–13) two stages. Next, we will illustrate the two stages of HFDRA approach in detail.

```
Input:
Original dataset, FF-Correlation Measurement method: MIC, FC-Correlation measurement method: IG, Dimensions after dimensionality reduction: $w$, Number of clusters: $k$;
Output:
Data after dimensionality reduction: $R$
/* Feature selection stage */
1.  For $i = 1$ to $n$ do
2.      For $j = 1$ to $n$ do
3.          Use the MIC to calculate the correlation $\mathrm{FF}(f_i, f_j)$ and save it in the matrix $F_{ij}$.
4.      end for
5.  end for
6.  For $h = 1$ to $n$ do
7.      Use IG to calculate the $\mathrm{FC}(f_h)$ and save it to the vector $S_h$.
8.  end for
9.  According to the information of the matrix $F$ and the vector $C$, the K-Medoids algorithm is used to divide the feature set into $k$ clusters.
/* Feature dimensionality reduction stage */
10. Initialize the matrix $R$
11. For $q = 1$ to $k$ do
12.     Use KPCA to reduce the dimensionality of the data subset of cluster $C_q$ to $\left\lceil \frac{|C_q| \times W}{n} \right\rceil$, and add the reduced dimensionality result to $R$.
13. End for
14. Return $R$
```

Algorithm 1: HFDRA algorithm.

In the feature selection stage, FF-correlation matrix (Lines 1–5) and FC-correlation vector (Lines 6–8) are needed to be calculated first. Then, we use the clustering algorithm to take similar features into one cluster, while dissimilar features are put into different clusters (Line 9). The goal of clustering algorithm is that the features within same cluster are "similar" to each other, while the features from different clusters are 'dissimilar'. In this paper, k-medoids algorithm, a variant of k-means algorithm, is used as the clustering algorithm. The details of the algorithm are shown in Algorithm 2. The input of the algorithm is the FC-correlation vector, FF-correlation matrix, the number of clusters and the maximum number of iterations. The output of the algorithm is the clustering results of all features. The process of k-medoids clustering algorithm is as follows: first, we find out k features and initialize the centroid of k clusters (Line 1), then assign all features to k clusters according to the FF-correlation matrix (Line 3), after that readjust the centroid of each cluster (Line 4), finally repeat the above process until the maximum number of iterations is reached or k centroids is no longer changed (Line 2).

In Step 1 of Algorithm 2, we heuristic selected k-initialize centroids compared with the original k-medoids algorithm. Generally, if the k-initialize centroids are not selected properly, the convergence speed of the algorithm will slow down or the clustering effect is not ideal. In order to choose the appropriate k-initialize centroids, this paper heuristically selects the first k features most related to the defect class as the initialization centroids of k clusters according to the FC-correlation vector.

```
Input:
Original dataset, FC-Correlation vector: $F$, FF-Correlation matrix: $S$, The number of clusters: $k$, The maximum number of iterations: $N$
Output:
Clustering results of $k$ clusters;
1.  Select $k$ features as the initial centroids of $k$ clusters
2.  While $k$ centroids no longer change or reach the specified number of iterations do
3.      each feature is assigned to the centroids with the highest correlation.
4.      Update the centroids of $k$ clusters
5.  end while
6.  return Clustering results of $k$ clusters
```

Algorithm 2: K-Medoids algorithm.

In Step 4 of Algorithm 2, we use FF-correlation instead of Euclidean distance to measure the relevance among features compared with the original k-medoids algorithm. In addition, in order to maximize the internal correlation within each clusters, we calculate the sum of FF-correlation of each features in every clusters and choose the features with the largest correlation with other features in the cluster as the new centroid. Correlation calculation is defined as follows:

$$v_{c,i} = \sum_{1 \le j \le m, j \ne i} F\left(v_{c,i}, v_{c,j}\right). \tag{9}$$

$V_{c,i}$ represents the sum of the correlation among the *ith* feature and other features in the *cth* cluster, while $F(x, y)$ represents the correlation value between the feature x and the feature y.

After clustering all features, the similarity of features in the same cluster is the largest, while the similarity between different clusters is the smallest. At the same time, higher similarity of features means more redundant features or irrelevant features. Next, we employ KPCA to reduce the dimension of each cluster so as to remove redundant or irrelevant features and improve the classification ability of features in the feature extraction stage. Because the size of each cluster is different after clustering, this paper adaptively reduces to the required dimension according to the size of each cluster. For example, a clustering result is $P = \{C_1, C_2, \ldots, C_q\}$, the dimension of each cluster C_i in P after features dimensionality reduction is $\left| \frac{|c_i| \times m}{M} \right|$. Where $|C_i|$ is the number of features contained in cluster C_i, m is dimension after features dimensionality reduction and M is the number of original features. Finally, we combine the data of

Table 2: Experimental data.

Dataset	Project	#Feature	#Instances	#Defect	Defect (%)
AEEEM	EQ	61	324	129	39.81
	JDT	61	997	206	20.66
	LC	61	691	64	9.26
	ML	61	1,862	245	13.16
	PDE	61	1,497	209	13.96
SOFTLAB	AR1	29	121	9	7.44
	AR3	29	63	8	12.70
	AR4	29	107	20	18.69
	AR5	29	36	8	22.22
	AR6	29	101	15	14.85
MORPH	Ant	20	125	20	16
	Camel	20	339	13	3.83
	Poi	20	237	141	59.49
	Tomcat	20	858	77	8.97
	Velocity	20	196	147	75
	Xalan	20	723	110	15.21
	Xerces	20	440	71	16.14
	Arc	20	234	27	11.54
	Redktor	20	176	27	15.34
ReLink	Apache	26	194	98	50.52
	Safe	26	56	22	39.29
	ZXing	26	399	118	29.57

dimensionality reduction of each cluster to get the final needed data for training the prediction model.

4. Experiment

To verify the effectiveness of our proposed approach, we first design the following three research questions.

(RQ1) How does the HFDRA approach compare to baseline methods?

(RQ2) How does the HFDRA approach compare to other state-of-the-art methods?

(RQ3) What is the relationship between FR and predicted performance?

4.1. Experimental Data. The data used in this paper come from 22 projects in AEEEM, SOFTLAB, MORPH, and ReLink four defect databases, which have been widely used by many researchers in this field [19–22]. In order to improve the quality of the experimental data, we use Shepperd et al. [19] method to process all experiment data. Table 2 shows the details of the data used in this paper. The Table 2 shows dataset name (Column 1), project name (Column 2), number of features (Column 3), number of instances (Column 4), number of defective instances (Column 5), and defect rate (Column 6). It should be noted that when the defect rate is smaller, the data is more unbalanced.

AEEEM dataset is provided by D'ambros et al. [20]. The dataset includes five different projects: equinox framework (EQ), eclipse JDT core (JDT), Apache Lucene (LC), mylyn (ML), and eclipse PDE UI (PDE). Each project is represented by 61 features. The MORPH dataset was collected and provided by Jureczko and Madeyski from PROMISE database [21]. Each project contains 20 features. SOFTLAB defect database contains AR1, AR3, AR4, AR5, and AR6 five projects, each of which contains 29 features. ReLink dataset is provided by Wu et al. [22]. It contains Apache, safe, and ZXing three projects, each project contains 26 features. We use zero-mean normalization method to preprocess the experimental data to improve the prediction performance. Besides, in order to solve the class imbalance problem, the synthetic minority over-sampling technique (SMOTE) is used in this paper. SMOTE is a technique for solving class imbalance problems by generating minority class samples. It is implemented by invoking the *imbalanced-learn* toolkits in this paper.

Table 3: The confusion matrix.

	Predicted as defective	Predicted as nondefective
Actual defective	TP	FN
Actual nondefective	FP	TN

4.2. Evaluation Metrics

4.2.1. Predictive Performance Evaluation Metrics. First, we need to give the confusion matrix to facilitate the introduction of the evaluation metrics used in this paper, a typical confusion matrix for binary tasks is shown in Table 3.

The elements in the confusion matrix represent all possible prediction results. The definition of all elements in the confusion matrix is as follows:

TP: defective instance is correctly predicted as defective instance,

FN: defective instance is wrongly predicted as nondefective instance,

FP: nondefective instance is wrongly predicted as defective instance,

TN: nondefective instance is correctly predicted as nondefective instance.

According to the confusion matrix, *precision* and *recall* can be define as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{10}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{11}$$

In this paper, AUC, F1, and MCC three evaluation metrics are used to evaluate the prediction performance of models [23].

AUC is based on ROC (receiver operating characteristic curve) curve to evaluate the prediction performance of different defect prediction models. ROC curve comprehensively considers different classification thresholds when evaluating the performance of classifiers. In this paper, we use the tools provided in the *scikit-learn* toolkits to calculate AUC value of the predictive model.

F1 value is an evaluation metric used to measure the comprehensive performance of *precision* and *recall*, and its calculation is given in Equation (12). This evaluation metric is appropriate for datasets with class-unbalanced problem. We calculate F1 value of prediction model by invoking the *scikit-learn* toolkits in this paper.

$$\mathrm{F1} = 2 \times \frac{\mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}. \tag{12}$$

MCC takes all elements of confusion matrix into account, which is also a comprehensive evaluation metrics. The calculation of MCC is given in Equation (13). This evaluation metrics is also suitable for datasets with class imbalance problem. We also use the tool provided in *scikit-learn* toolkits to calculate MCC value of predictive model.

$$\mathrm{MCC} = \frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{TP} \times \mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP}) \times (\mathrm{TP}+\mathrm{FN}) \times (\mathrm{TN}+\mathrm{FP}) \times (\mathrm{TN}+\mathrm{FN})}}. \tag{13}$$

4.2.2. Feature Redundancy (FR). In order to test whether HFDRA approach can eliminate the irrelevant and useless features from the original features and improve the prediction performance, this paper defines a new evaluation metric: FR. The calculation of FR is given in Equations (14) and (15).

$$\mathrm{FR}(T) = \frac{1}{|T|^2} \sum_{i=1}^{|T|} \sum_{j=1}^{|T|} w_{i,j} \times F(f_i, f_j), \tag{14}$$

$$w_{i,j} = \begin{cases} 1, \text{if } F(f_i, f_j) \geq 0.5 \text{ and } i \neq j \\ 0, \text{otherwise} \end{cases}. \tag{15}$$

$|T|$ denotes the feature dimension of project, and $F(f_i, f_j)$ denotes the correlation between the feature f_i and the feature f_j. In this paper, we think that if the correlation value between the two features is greater than 0.5, then there is redundancy between the two features, otherwise there is no redundancy. $1/|T|^2$ is the normalization factor, so that the range of FR of project is [0, 1].

4.3. Experimental Design

4.3.1. Classifier and Dimension Reduction Models. For the experiment of this paper, we use two different classifier to build defect prediction model: Naive Bayes (NB) and support vector machine (SVM). In this paper, we implement NB and SVM classifiers by invoking *scikit-learn* toolkit with default parameter settings.

KPCA is also implemented by invoking the *scikit-learn* toolkits in this paper. We set the kernel function as radial basis function (RBF), other parameters are the default parameters.

4.3.2. Comparison Methods. For Question 1, in order to verify the effectiveness of HFDRA method, we choose four different kinds of methods as baseline methods: (1) ALL, (2) IG and MIC, (3) KPCA, (4) CFS, FCBF, and ReliefF.

All means that no feature dimensionality reduction is performed, that is, only the original data is used to build the defect prediction model. In this paper, this method is used as a baseline method, which can study whether HFDRA can improve the performance of defect prediction.

IG and MIC are two feature selection methods based on ranking. These two methods selects a specified number of feature subsets after features ranking according to IG and MIC two correlation measurement method, respectively. The HFDRA contains two stages. In Stage 1, IG measurement method is utilized to measure the correlation between features and defect classes, and the MIC measurement method to measure the correlation among features. This paper uses IG and MIC as baseline methods to study whether the measurement method used in Stage 1 is effective.

KPCA method refers to the use of KPCA method to reduce the feature dimension of data. In this paper, KPCA as a baselines method can be used to study whether the dimensionality reduction method used in Stage 2 is effective.

CFS [24], FCBF [25], and ReliefF [26] are three typical feature selection methods. Specifically, CFS feature selection method not only considers the correlation between features and defect classes, but also considers the correlation among features. CFS uses best first search strategy to find the best feature subset. The FCBF feature selection method also considers two correlations as the same as CFS. Since this method only evaluates one feature at one time, it does not need pair wise correlation analysis. According to suggestion of Yu et al. [27], we set the correlation threshold as $\lfloor \log_2 M \rfloor$, where M is the number of original features. ReliefF algorithm is an improved algorithm based on ReliefF algorithm. The algorithm first randomly selects a sample s each time, and then finds k nearest neighbor samples the same type as s and k nearest neighbor samples the different type as s, finally updates the weight of each feature. The purpose of comparing the three baseline methods with the HFDRA approach is to study the effectiveness of the approach proposed in this paper.

In addition, we also compare HFDRA with KPWE [28], BPDET [29], and FECAR [16] three state-of-the-art methods in Question 2. KPWE applied KPCA to eliminate redundant or irrelevant features. BPDET utilized staked denoising autoencoders (SDA) to eliminate redundant or irrelevant features. Both KPCA and SDA belong to the feature extraction methods. However, FECAR exploited the feature selection method to eliminate redundant or irrelevant features. We can further study the effectiveness of HFDRA method by comparing it with the three state-of-the-art methods.

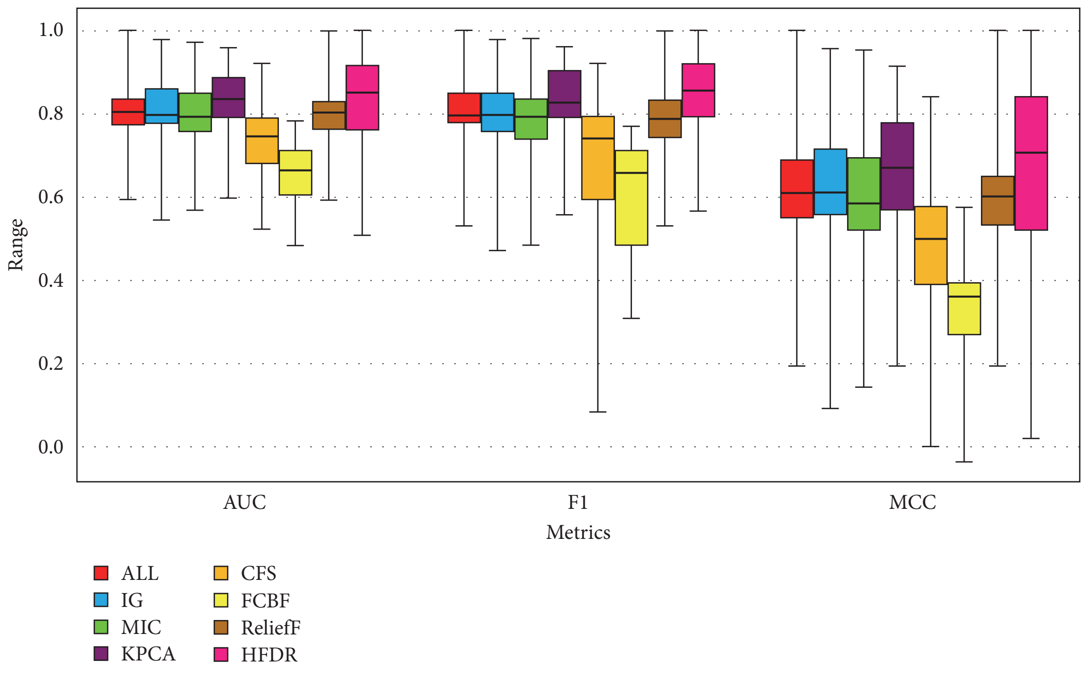


Figure 2: Box-plot of prediction performance of four different dimension number after dimensionality reduction under SVM classifier.

4.4. Experimental Results and Analysis

4.4.1. Results and Analysis for RQ1. In order to study the effectiveness and details of the HFDRA approach, seven baseline methods are selected in this paper. For the introduction of seven baseline methods, please refer to Section 4.3.2. Through detailed empirical research, we first find that it can achieve relatively good prediction performance when the number of clusters is 1/2 of the number of original features and the dimension after dimensionality reduction is 2/3 of the number of original features. Therefore, we set the number of clusters of the HFDRA approach as 1/2 of the number of original features and the dimension after dimensionality reduction as 2/3 of the number of original feature. The prediction performance of all methods is evaluated by AUC, F1, and MCC three metrics. The average value of 10-fold cross-validation is used as the experimental result. The experimental results of the prediction performance of HFDRA and seven baselines methods are provided in Figures 2 and 3.

Figures 2 and 3 show the box-plot representation of the HFDRA approach and seven baseline methods using three evaluation metrics on 22 projects under SVM and NB two classifiers, respectively. Through the overall observation of Figures 2 and 3, we can find that the HFDRA approach is superior to all baseline methods, which verifies the effectiveness of method proposed in this paper. Figure 2 shows that the median and upper quartile of the HFDRA approach is higher than other baseline methods. Figure 3 presents that the median, upper quartile, and lower quartile of HFDRA approach are higher than other baseline methods in F1 evaluation metric. For AUC and MCC evaluation metrics, although the upper quartile of the HFDRA approach is lower than that of the KPCA method, the median and lower quartile of HFDRA approach are still higher than all baseline methods. In addition, the comparison of Figures 2 and 3 two results reveals that the prediction performance of the SVM classifier is better than that of NB classifier. Overall, these results indicate that the prediction performance of the HFDRA approach is better than that of the other seven baseline methods.

In order to further analyze the prediction results between the HFDRA approach and seven baseline methods, we first calculate the mean of each method on all projects. Then, we count the win/draw/loss information between the HFDRA approach and seven baseline methods. In addition, Wilcoxon [30] signed rank test at 95% significance level is also used for statistical analysis. Moreover, we also applied Cliff's delta (a nonparametric effect size measure) to measure the effect size, which has four level effect size as show in Table 4. The statistical test results are shown in Tables 5 and 6. We can observe that the mean values of the HFDRA approach on 22 projects is better than other methods no matter which classifier and evaluation metrics is used. For the statistical results of win/draw/loss information, we can find that the prediction performance of the HFDRA approach is still better than all baseline methods no matter what classifier and evaluation method is used. For Wilcoxon signed rank and Cohen's delta test, when SVM classifier is used, the prediction performance of the HFDRA approach shows significant improvements (p-value < 0.05) over CFS and FCBF methods with large size effect on three metrics, and has no significant improvements on other baseline methods. For the NB classifier, when AUC or MCC evaluation metrics are used, the prediction performance of the HFDRA approach shows significant improvements (p-value < 0.05) only over FCBF method with large size effect. However, when F1 evaluation metrics

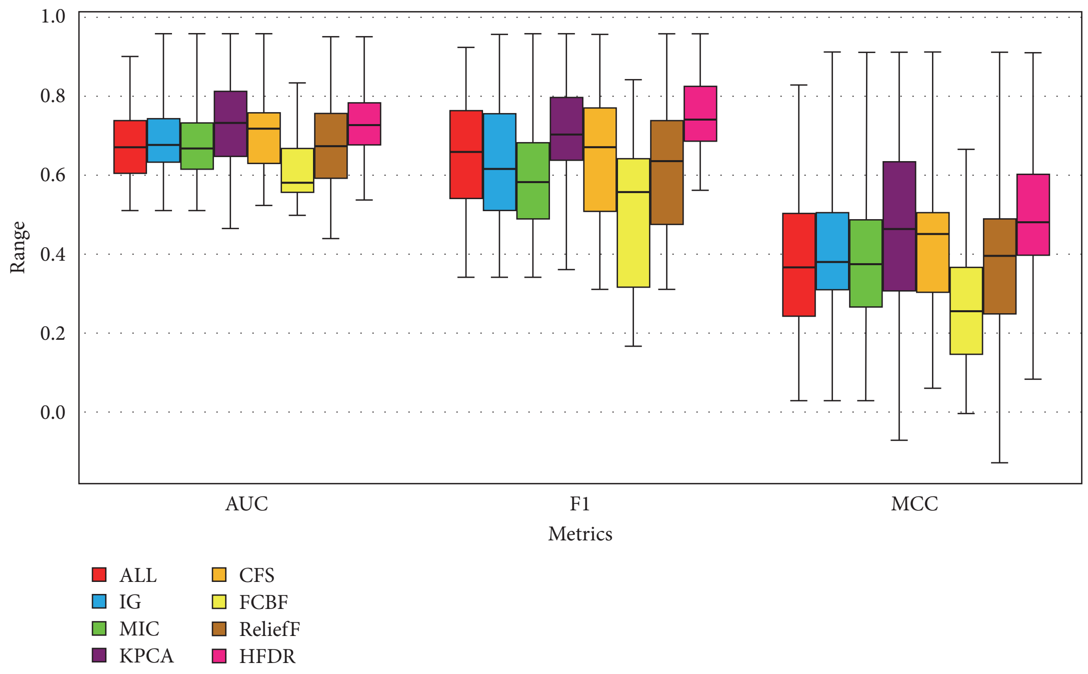


FIGURE 3: Box-plot of prediction performance of four different dimension number after dimensionality reduction under NB classifier.

TABLE 4: Cliff's delta and the effectiveness level.

Cliff's delta ($\|\delta\|$)	Effectiveness level
$0.000 \leq \|\delta\| < 0.147$	Negligible
$0.147 \leq \|\delta\| < 0.330$	Small
$0.330 \leq \|\delta\| < 0.474$	Medium
$0.474 \leq \|\delta\| < 1.000$	Large

TABLE 5: The statistical test results of prediction performance of the HFDRA method and other comparison methods under SVM classifier.

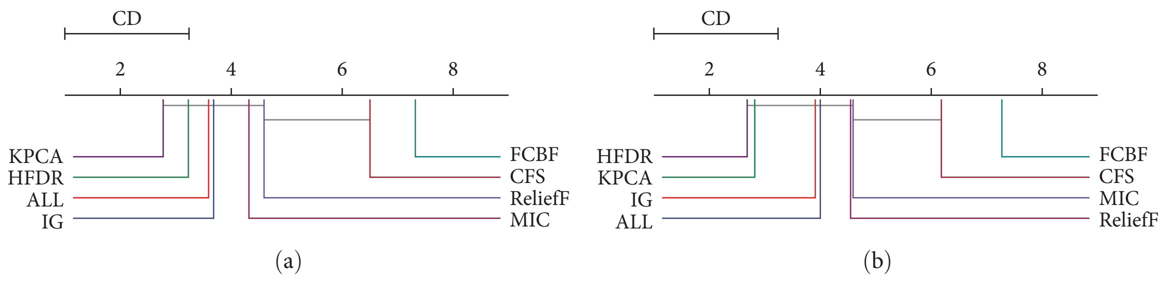
Measure		Methods							
		ALL	IG	MIC	KPCA	CFS	FCBF	ReliefF	HFDRA
AUC	AVE	0.811	0.807	0.808	0.836	0.727	0.654	0.805	0.835
	W/D/L	7/1/14	9/0/13	7/0/15	9/1/12	3/0/19	3/0/19	8/1/13	–
	p-value	0.434	0.387	0.392	0.950	0.004	0.000	0.335	–
	Cliff's	0.188	0.194	0.194	0.043	0.546	0.822	0.234	–
F1	AVE	0.803	0.795	0.794	0.831	0.666	0.603	0.795	0.842
	W/D/L	5/1/16	7/0/15	6/0/16	7/0/15	3/0/19	2/0/20	5/1/16	–
	p-value	0.234	0.175	0.161	0.726	0.001	0.000	0.149	–
	Cliff's	0.285	0.298	0.326	0.143	0.651	0.872	0.364	–
MCC	AVE	0.624	0.616	0.616	0.671	0.480	0.327	0.608	0.673
	W/D/L	6/1/15	8/0/14	7/0/15	10/1/11	3/0/19	3/0/19	8/1/13	–
	p-value	0.429	0.367	0.355	0.976	0.002	0.000	0.296	–
	Cliff's	0.219	0.198	0.186	0.064	0.558	0.802	0.267	–

is used, the prediction performance of HFDRA approach shows significant improvements (p-value < 0.05) over all baseline methods except KPCA method.

Finally, we apply Friedman test with post hoc Nemenyi test [31] to compare the prediction performance of HFDRA approach and the seven baseline methods. Friedman test to determine whether there are significant statistical differences in the performance of multiple methods on multiple projects. If there are significant statistical differences, we use post hoc Nemenyi test to determine which algorithms performance is significantly different. The statistical test results are presented in Figures 4 and 5 which show that the prediction

Table 6: The statistical test results of prediction performance of the HFDRA method and other comparison methods under NB classifier.

Measure		Methods							
		ALL	IG	MIC	KPCA	CFS	FCBF	ReliefF	HFDRA
AUC	AVE	0.685	0.696	0.691	0.732	0.710	0.618	0.677	0.736
	W/D/L	3/0/19	4/0/18	4/2/16	10/1/11	8/0/14	2/0/20	2/2/18	–
	p-value	0.102	0.196	0.174	0.906	0.200	0.000	0.069	–
	Cliff's	0.296	0.265	0.308	0.006	0.223	0.599	0.331	–
F1	AVE	0.654	0.634	0.614	0.706	0.652	0.508	0.624	0.751
	W/D/L	2/0/20	2/0/20	2/0/20	7/0/15	5/0/17	2/0/20	2/1/19	–
	p-value	0.021	0.008	0.003	0.286	0.016	0.000	0.005	–
	Cliff's	0.362	0.444	0.533	0.143	0.401	0.703	0.444	–
MCC	AVE	0.397	0.423	0.404	0.471	0.438	0.279	0.385	0.501
	W/D/L	4/0/18	4/0/18	5/0/17	10/1/11	9/0/13	1/0/21	3/1/18	–
	p-value	0.087	0.201	0.139	0.674	0.193	0.000	0.068	–
	Cliff's	0.302	0.252	0.314	0.048	0.252	0.603	0.333	–



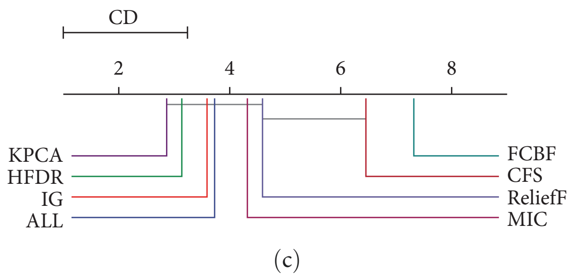


Figure 4: The rank of HFDRA method and comparison methods using post hoc Nemenyi test on (a) AUC, (b) F1, and (c) MCC three metrics under SVM classifier.

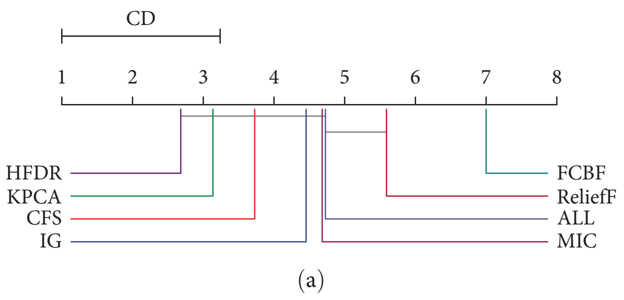
performance of HFDRA approach is only slightly weaker than the KPCA method on the SVM classifier. However, the HFDRA approach is preferable to all baseline methods on the NB classifier. Overall, the prediction performance of HFDRA approach is better than all baseline methods.

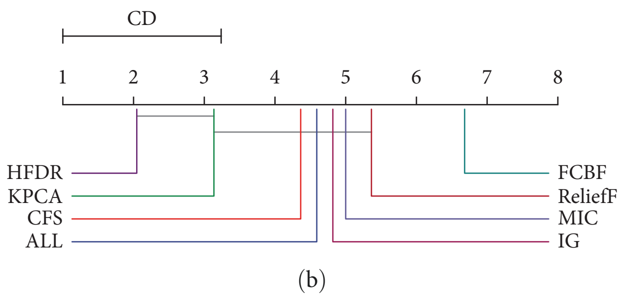
In summary, the HFDRA approach is superior to all baseline methods through the detailed analysis of the above experimental results. The evidence from experimental results suggests that HFDRA approach can reduce FR and improve the prediction performance.

4.4.2. Results and Analysis for RQ2. Question 2 aims to further study the effectiveness of HFDRA approach. Therefore, we choose three state-of-the-art methods to compare with HFDRA. Both FECAR and HFDRA selected SVM as the classifier to construct the prediction model. For KPWE and BPDET two methods, extreme learning machines (ELM) and ensemble classifiers were selected as classifier, respectively. The prediction performance of four methods is also evaluated by AUC, F1, and MCC three metrics. The average value of 10-fold cross-validation is also used as the experimental result on each project. Figure 6 presents the average value of four methods on 22 projects under AUC, F1, and MCC three evaluation metrics.

As shown in Figure 6, the prediction performance of HFDRA is better than the other three methods from the whole. The prediction performance difference between HFDRA and BPDET is very small, and even BPDET is slightly better than HFDRA under the F1 metric.

BPDET not only uses the deep learning method to extract valuable features, but also uses the ensemble learning classifier to construct a prediction model. However, deep learning models need to consider more experimental details to achieve the best predictive performance. In addition, we





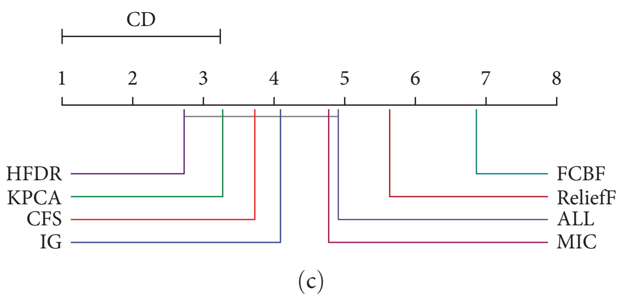


Figure 5: The rank of HFDRA method and comparison methods using post hoc Nemenyi test on (a) AUC, (b) F1, and (c) MCC three metrics under NB classifier.

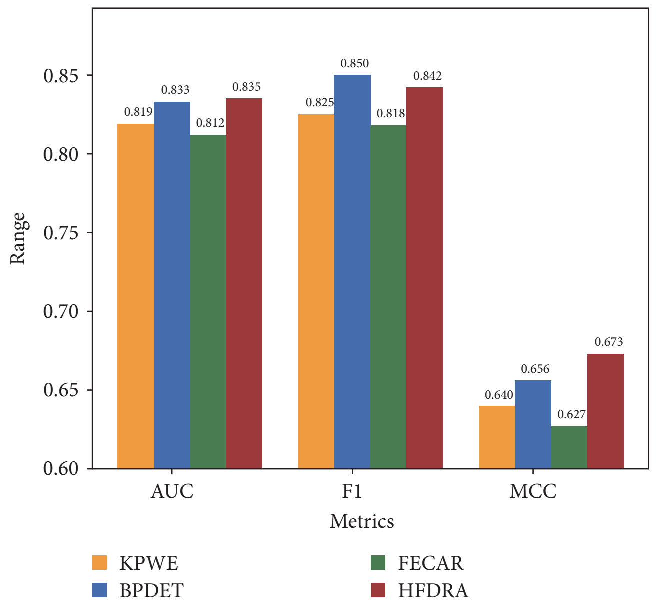


Figure 6: Bar chart of prediction performance of the HFDRA method and three comparison methods.

believe that if HFDRA approach uses ensemble learning method to construct the prediction model, it can further improve the prediction performance. By observing the prediction performance of KPCA in Table 5 and Figure 6, we find that ELM is slightly worse than SVM as a classifier. It suggests that ELM may not be suitable for SDP.

In summary, the experimental results show that HFDRA approach is superior to the other three state-of-the-art methods in most metrics, which further suggests the effectiveness of HFDRA approach for SDP.

4.4.3. Results and Analysis for RQ3.

4.4.3. Results and Analysis for RQ3. Question 3 is to study whether the HFDRA approach can reduce FR and improve the prediction performance. The FR is calculated according to Equations (14) and (15). In addition, we study the relationship between features redundancy and prediction performance to further prove the effectiveness of the approach proposed in this paper. Figures 7 and 8 show the average value of AUC, F1, and MCC evaluation metrics of each method on 22 projects and the average FR of each method on 22 projects under SVM and NB classifiers, respectively. Tables 7 and 8 show the detailed experimental results of Figures 7 and 8.

By observing Figures 7 and 8, we can find that the change trend of each method's FR is inversely proportional to the change trend of each method's prediction performance, which implies that the prediction performance can be improved when the FR is reduced. By comparing between the features redundancy of the ALL method and the HFDRA method in Tables 7 and 8, we can find that the HFDRA method does reduce features redundancy. By observing the performance of the ALL approach and the HFDRA approach in the three evaluation metrics, we can find that the prediction performance of the HFDRA approach is better than that of the ALL method. These results suggest that the HFDRA approach can reduce the features redundancy and improve the prediction performance, which proves the effectiveness of the approach proposed in this paper. By comparing the experimental results of Question 1 and Question 2, what stands out is that the features redundancy of the HFDRA approach is lower than that of all comparison methods selected in this paper except the KPCA method, while the prediction performance of the HFDRA approach is the best of all methods. These findings suggest that reducing the FR can improve the prediction performance, but if the FR is too low, it may decrease the prediction performance in turn.

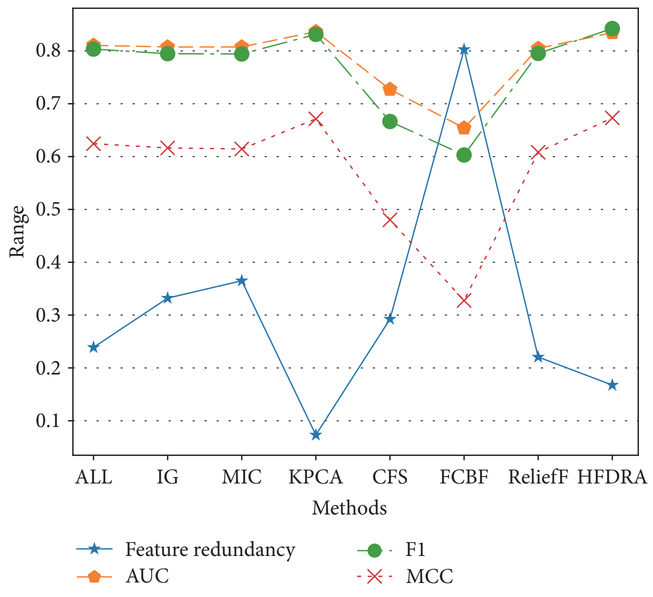


Figure 7: Line chart of prediction performance and feature redundancy of the HFDRA method and baseline methods under SVM classifier.

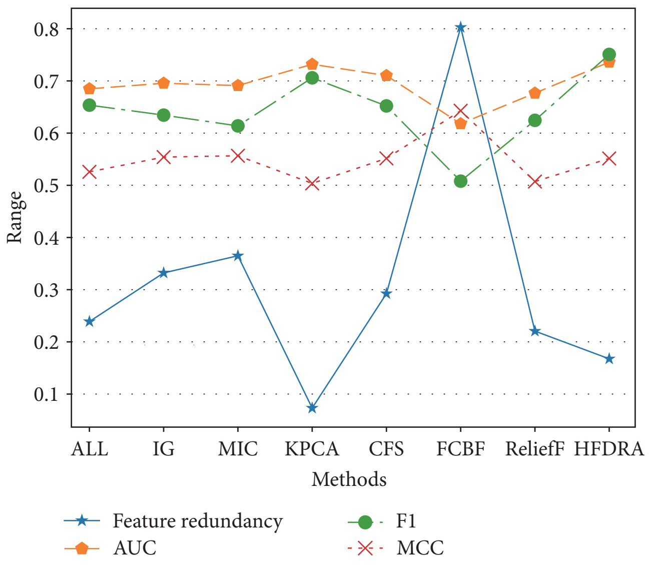


Figure 8: Line chart of prediction performance and feature redundancy of the HFDRA method and baseline methods under NB classifier.

Table 7: Prediction performance and feature redundancy of the HFDRA method and baseline methods under SVM classifier.

Measure	ALL	IG	MIC	KPCA	CFS	FCBF	ReliefF	HFDRA
FR	0.239	0.332	0.365	0.073	0.293	0.803	0.221	0.168
AUC	0.811	0.807	0.808	0.836	0.727	0.654	0.805	0.834
F1	0.803	0.794	0.794	0.831	0.666	0.603	0.795	0.842
MCC	0.624	0.616	0.614	0.671	0.480	0.328	0.608	0.673

Table 8: Prediction performance and feature redundancy of the HFDRA method and baseline methods under NB classifier.

Measure	ALL	IG	MIC	KPCA	CFS	FCBF	ReliefF	HFDRA
FR	0.239	0.332	0.365	0.073	0.293	0.803	0.221	0.168
AUC	0.685	0.696	0.691	0.732	0.710	0.618	0.677	0.736
F1	0.654	0.634	0.614	0.706	0.652	0.508	0.624	0.751
MCC	0.526	0.554	0.557	0.504	0.552	0.643	0.507	0.552

In summary, through the study of this problem, we can conclude that the approach proposed in this paper can reduce the features redundancy and improve the prediction performance, which further verifies the effectiveness of the approach proposed in this paper.

5. Related Works

In order to improve the performance of SDP, researchers extracted various features from the source code and related data. However, there would be problems such as FR or feature irrelevance with the increase of features. There was much research in the area of software defect prediction that was tried to solve the problems. These research methods can be divided into feature selection, feature extraction and other three categories.

As to research methods based on feature selection, Shivaji et al. [32] studied four filtering-based feature selection methods and two package-based feature selection methods. They iteratively deleted irrelevant features until a relatively good prediction performance is achieved. They found that when the number of features is reduced to 10% or even 1% of the original number, it could also get relatively good performance. Khoshgoftaar and Gao [33] used a wrapper-based attribute ranking technique to select a subset of features and the random undersampling technique to solve the class imbalance problem. Wahono and Herman [34] used the genetic algorithm to select the best feature subset and the bagging technique to solve the class imbalance problem. In contrast, our approach used Synthetic Minority Oversampling Technique (SMOTE) to solve the class imbalance problem. Furthermore, our approach not only included feature selection, but also used the feature extraction technology, which could reduce FR and improve feature classification ability.

As to research methods based on feature extraction, Xu et al. [28] proposed a software defect prediction based on kernel PCA (KPCA) and weighted learning. The method could be divided into two stages: in the first stage, KPCA was used to map the original features into a new feature space; in the second stage, the weighted learning method was used to solve the class imbalance problem. Tong et al. [35] proposed a new software defect prediction method by combining stacked denoising autoencoder and ensemble learning. They first used stacked denoising autoencoder to extract features from traditional software prediction features. Then, the acquired features were inputted into an improved ensemble learning model to solve the class imbalance problem and construct the prediction model. Unlike their approach, our approach used a combination of feature selection and feature extraction methods, which not only improves the effect of dimensionality reduction, but also avoids the choice of more hyperparameters.

As to other research methods, Liu et al. [16] proposed a new feature selection framework. This framework first used the clustering algorithm to divide the original feature into several subsets. Then, it selected a specified number of features from each cluster to form the final feature subset. Our approach, in contrast, used feature extraction technology for each cluster, which could not only reduce FR, but also improve feature classification ability.

6. Conclusion and Future Works

In this paper, we find that the two categories of data dimensionality reduction methods have own advantages in different projects. Therefore, this paper attempts to combine the two categories of methods to propose a hybrid feature dimensionality reduction approach for software defect prediction. We evaluate our HFDRA approach on 22 projects and compare it with seven baseline methods and three state-of-the-art methods. Experiment results show that HFDRA approach is better than all comparison methods on the three evaluation metrics, which proves the effectiveness of the approach proposed in this paper. In addition, we also study the relationship between features redundancy and prediction performance to further suggest the effectiveness of HFDRA approach. In the future, we will choose more comparison methods to conduct experiments on more projects to further verify the effectiveness of the approach proposed in this paper.

Data Availability

The AEEEM, SOFTLAB, MORPH, and ReLink three public datasets used to support the findings of this study. AEEEM dataset can be derived from https://bug.inf.usi.ch/. ReLink dataset is available at: http://www.cse.ust.hk/~scc/ReLink.htm.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China (grant no. 61673384) and Natural Science Foundation of Jiangsu Province (grant no. BK20181353).

References

[1] R. Özakıncı and A. Tarhan, "Early software defect prediction: a systematic map and review," *Journal of Systems and Software*, vol. 144, pp. 216–239, 2018.

[2] S. Zhang, S. Jiang, and Y. Yan, "A software defect prediction approach based on BiGAN Anomaly Detection," *Scientific Programming*, vol. 2022, Article ID 5024399, 13 pages, 2022.

[3] V. R. Basili, L. C. Briand, and W. L. Melo, "A validation of object-oriented design metrics as quality indicators," *IEEE Transactions on Software Engineering*, vol. 22, no. 10, pp. 751–761, 1996.

[4] N. Nagappan and T. Ball, "Use of relative code churn measures to predict system defect density," in *27th International Conference on Software Engineering*, vol. 2005, pp. 284–292, IEEE, St. Louis, MO, USA, 2005.

[5] C. Bird, N. Nagappan, B. Murphy, H. Gall, and P. Devanbu, "Putting it all together: using socio-technical networks to predict failures," in *2009 20th International Symposium on Software Reliability Engineering*, pp. 109–119, IEEE, Mysuru, India, 2009.

[6] S. Wang, T. Liu, J. Nam, and L. Tan, "Deep semantic feature learning for software defect prediction," *IEEE Transactions on Software Engineering*, vol. 46, no. 12, pp. 1267–1293, 2020.

[7] H. Wang, W. Zhuang, and X. Zhang, "Software defect prediction based on gated hierarchical LSTMs," *IEEE Transactions on Reliability*, vol. 70, no. 2, pp. 711–727, 2021.

[8] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, "A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction," *Journal of Applied Science and Technology Trends*, vol. 1, no. 2, pp. 56–70, 2020.

[9] S. Ding, H. Zhu, W. Jia, and C. Su, "A survey on feature extraction for pattern recognition," *Artificial Intelligence Review*, vol. 37, pp. 169–180, 2012.

[10] J. Cai, J. Luo, S. Wang, and S. Yang, "Feature selection in machine learning: a new perspective," *Neurocomputing*, vol. 300, pp. 70–79, 2018.

[11] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," in *2014 Science and Information Conference*, pp. 372–378, IEEE, London, UK, 2014.

[12] R. Duksan, O. Choi, and J. Baik, "Value-cognitive boosting with a support vector machine for cross-project defect prediction," *Empirical Software Engineering*, vol. 21, pp. 43–71, 2016.

[13] B. Azhagusundari and A. S. Thanamani, "Feature selection based on information gain," *International Journal of Innovative Technology and Exploring Engineering*, vol. 2, no. 2, pp. 18–21, 2013.

[14] B. Schölkopf, A. Smola, and K. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[15] K. Gao, T. M. Khoshgoftaar, H. Wang, and N. Seliya, "Choosing software metrics for defect prediction: an investigation on feature selection techniques," *Software: Practice and Experience*, vol. 41, no. 5, pp. 579–606, 2011.

[16] S. Liu, X. Chen, W. Liu, J. Chen, Q. Gu, and D. Chen, "FECAR: a feature selection framework for software defect prediction," in *2014 IEEE 38th Annual Computer Software and Applications Conference*, pp. 426–435, IEEE, Vasteras, Sweden, 2014.

[17] H. Tong, B. Liu, S. Wang, and Q. Li, "Transfer-learning oriented class imbalance learning for cross-project defect prediction," 2019.

[18] J. Deng, J. Guo, and Y. Wang, "A novel K-medoids clustering recommendation algorithm based on probability distribution for collaborative filtering," *Knowledge-Based Systems*, vol. 175, pp. 96–106, 2019.

[19] M. Shepperd, Q. Song, Z. Sun, and C. Mair, "Data quality: some comments on the NASA software defect datasets," *IEEE transactions on Software Engineering*, vol. 39, no. 9, pp. 1208–1215, 2013.

[20] M. D'Ambros, M. Lanza, and R. Robbes, "Evaluating defect prediction approaches: a benchmark and an extensive comparison," *Empirical Software Engineering*, vol. 17, pp. 531–577, 2012.

[21] M. Jureczko and L. Madeyski, "Towards identifying software project clusters with regard to defect prediction," in *6th International Conference on Predictive Models in Software Engineering*, pp. 1–10, Association for Computing Machinery, 2010.

[22] R. Wu, H. Zhang, S. Kim, and S. C. Cheung, "ReLink: Recovering links between bugs and changes," in *Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering*, pp. 15–25, Association for Computing Machinery, 2011.

[23] S. K. Pandey, R. B. Mishra, and A. K. Tripathi, "Machine learning based methods for software fault prediction: a survey," *Expert Systems with Applications*, vol. 44, Article ID 114595, 2021.

[24] A. O. Balogun, F. B. Lafenwa-Balogun, H. A. Mojeed et al., "Data sampling-based feature selection framework for software defect prediction," in *International Conference on Emerging Applications and Technologies for Industry 4.0 (EATI'2020)*, J. H. Abawajy, K. K. R. Choo, and H. Chiroma, Eds., vol. 254 of *Lecture Notes in Networks and Systems*, pp. 39–52, Springer, Cham, 2021.

[25] A. O. Balogun, S. Basri, S. A. Jadid et al., "Search-based wrapper feature selection methods in software defect prediction: an empirical analysis," in *Proceedings of the 9th Computer Science On-line Conference 2020: Intelligent Algorithms in Software Engineering*, R. Silhavy, Ed., pp. 492–503, Springer, Cham, Zlin, Czech Republic, August 2020.

[26] Y. Shao, B. Liu, S. Wang, and G. Li, "Software defect prediction based on correlation weighted class association rule mining," *Knowledge-Based Systems*, vol. 196, Article ID 105742, 2020.

[27] L. Yu and H. Liu, "Feature selection for high-dimensional data: a fast correlation-based filter solution," in *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, pp. 856–863, AAAI Press, Washington, DC, USA, 2003.

[28] Z. Xu, J. Liu, X. Luo et al., "Software defect prediction based on kernel PCA and weighted extreme learning machine," *Information and Software Technology*, vol. 106, pp. 182–200, 2019.

[29] S. K. Pandey, R. B. Mishra, and A. K. Tripathi, "BPDET: An effective software bug prediction model using deep representation and ensemble learning techniques," *Expert Systems with Applications*, vol. 144, Article ID 113085, 2020.

[30] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

[31] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, no. 1, pp. 1–30, 2006.

[32] I. H. Laradji, M. Alshayeb, and L. Ghouti, "Software defect prediction using ensemble learning on selected features," *Information and Software Technology*, vol. 58, pp. 388–402, 2015.

[33] T. M. Khoshgoftaar and K. Gao, "Feature selection with imbalanced data for software defect prediction," in *2009 International Conference on Machine Learning and Applications*, pp. 235–240, IEEE, Miami, FL, USA, 2009.

[34] R. S. Wahono and N. S. Herman, "Genetic feature selection for software defect prediction," *Advanced Science Letters*, vol. 20, no. 1, pp. 239–244, 2014.

[35] H. Tong, B. Liu, and S. Wang, "Software defect prediction using stacked denoising autoencoders and two-stage ensemble learning," *Information and Software Technology*, vol. 96, pp. 94–111, 2018.