ISSN 1330-3651 (Print), ISSN 1848-6339 (Online) https://doi.org/10.17559/TV-20220421110027
Original scientific paper

A Cross-project Defect Prediction Model Using Feature Transfer and Ensemble Learning

Fuping ZENG, Wanting LIN*, Ying XING, Lu SUN, Bin YANG

Abstract: Cross-project defect prediction (CPDP) trains the prediction models with existing data from other projects (the source projects) and uses the trained model to predict the target projects. To solve two major problems in CPDP, namely, variability in data distribution and class imbalance, in this paper we raise a CPDP model combining feature transfer and ensemble learning, with two stages of feature transfer and the classification. The feature transfer method is based on Pearson correlation coefficient, which reduces the dimension of feature space and the difference of feature distribution between items. The class imbalance is solved by SMOTE and Voting on both algorithm and data levels. The experimental results on 20 source-target projects show that our method can yield significant improvement on CPDP.

Keywords: cross-project defect prediction; ensemble learning; machine learning; transfer learning

1 INTRODUCTION

With the expansion of software scale and the continuous improvement of complexity, the quality of software has caught a focus of attention. How to inchoately mine the defective module, particularly in the early development process becomes an urgent problem to be solved. Therefore, before software testing, it is crucial to use some attributes of software to establish a model to judge whether a specific software module contains defects or not, providing decision support for resource allocation of software testing.

As an important direction of empirical software engineering, software defect prediction (SDP) designs metrics that have a strong correlation with software defects that can measure the program modules and then use statistics or machine learning technology to build defect prediction models, realizing the prediction of defect tendency, defect severity and defect number distribution of target software modules [1] by mining and analysing the historical data accumulated 15 in the process of software development.

Most of the early researches focused on the same project SDP, using part of the marked program modules from the same project to build the prediction model, predicting the defects of the remaining modules in the project. At present, the establishment of defect prediction model has been more comprehensive, and the defect prediction results in software projects are also more satisfactory. However, there is still a long way to go for defect prediction technology to be put into practice, and a series of problems need to be solved, including the lack of datasets at the initial stage of the project, datasets denoising, selection of defect prediction features, optimization of datasets, optimization of prediction model and so on [2]. SDP of the same project often needs sufficient historical data for training. But in practical application, lacking historical data is a common problem, especially in the new software. Therefore, researchers began to pay attention to the study of CPDP [3]. The establishment of CPDP model is helpful to solve the problem of lack of datasets at the initial stage of the project, reducing the time and cost, and extracting features, obtaining the feature information which is ignored by the target project from all kinds of similar projects.

However, CPDP technology is still in its infancy, and the performance of existing CPDP methods is usually weaker than that of the same project. This is because in most cases, the value distribution of features of different projects is significantly different, and there is a common class imbalance problem in the defect prediction datasets [4]. Besides, based on the traditional defect prediction process, the prediction accuracy of cross-project defect prediction (CPDP) is often very low. This is because most of the machine learning models used in defect prediction are based on traditional statistical learning and probability classification methods, so the target projects and training projects are required to have the same statistical characteristics [5]. In the scenario of CPDP, the difference between the target project and the source projects will lead to a big discount in statistical learning results. Therefore, it is necessary to propose solutions to reduce project differences, including the selective selection of highly correlated datasets, extraction of distribution rules to help data migration, etc.

To solve the above problems, researchers have tried many traditional methods to improve the prediction performance of CPDP methods. However, with the development of software, the shortcomings of traditional algorithms such as poor data preprocessing and low efficiency of high-dimensional feature space operation are more and more obvious. Therefore, researchers shift their research focus to using a variety of machine learning methods for CPDP [6].

In this paper, we propose a CPDP model based on feature transfer and ensemble learning. In our model, we divide CPDP into two phases: feature transfer and classification. We propose corresponding solutions for each stage. Through these schemes, the prediction effect of the prediction model can be improved.

Specifically, the main contributions of this paper are as follows.

(1) We propose a feature transfer method based on Pearson correlation coefficient, which transfers the features of source and target items, reduces the dimension of feature space and the difference of feature distribution between items.

(2) From the data and algorithm level, the impact of class imbalance is alleviated. SMOTE method is used to oversample the data, and the Votingmethod is used to integrate multiple single classifiers.

(3) Experiments were carried out on 20 groups of open-source target project pairs, and the experimental results were discussed.

The rest of the paper is structured as follows. We survey the related work in Section 2. Detail the proposed approach in Section 3. Describe our three 70 experiments in Section 4, and give the analytical results of our experiments in Section 5. Disclose the threats to the validity of our research in Section 6. Section 7 concludes the paper and highlights directions for future work.

2 RELATED WORK

Three research routes are most relevant to the work described in this paper. Firstly, the research status of CPDP is discussed. Secondly, the related work of feature transfer method is briefly introduced. Finally, from the data level and algorithm level, class imbalance problem has been roundly studied.

2.1 Transfer Learning

Transfer learning can solve the problem of data distribution difference in defect prediction [7, 8]. According to whether the metric meta space between the tested project and the source project is the same, homogeneous transfer learning and heterogeneous transfer learning are defined as two main types of transfer learning, among which homogeneous transfer can be divided into instance-based transfer learning and feature-based transfer learning [9]. Among the three categories mentioned above, feature-based transfer learning is one of the focuses of researchers, which is also the research topic of this paper.

Feature-based transfer learning mainly transforms the metric set of source projects and tested projects into the same space through feature transformation.

The most commonly used defect prediction model has transferred component analysis (TCA). Cao et al. [10] used transfer principal component analysis (TCA) to transform the training dataset data so that the training dataset has a similar data distribution to the test dataset. Liu et al. [11] proposed an algorithm based on special points and transfer component analysis (SPTr-RMMEDA), which used some special points and their neighborhood solutions to generate a transitive learning prediction model and some new initial solutions to generate the next population when change occurs. Chen et al. [12] proposed a complete and partial feature transfer learning structure to complete the task of pest detection, classification and counting.

2.2 Class Imbalance

Considering the uneven distribution of defect data, the number of defective modules is far less than that of non-defective modules, which is called the class imbalance problem. Flawless instances dominate the data samples, and the learning classifier will favour flawless instances [13], resulting in overfitting and reducing the prediction effect. Class imbalance learning is a hot research field in machine learning. There are many types of research in SDP, and some class imbalance learning methods have been formed. The class imbalance adjustment methods for defect prediction can be roughly divided into data level and algorithm level.

2.3 Class Imbalance

CPDP [14-17] can be described as the SDP model based on other projects (source project), in order to fit the data-lacking status of current projects (target project). However, different data distribution between the source project and target project makes it difficult for the model based on the source project to have good prediction performance on the target project. Turhan et al. [18] proposed a Burak filtering method based on K-NN. Peters et al. [19] believed that the data in the source project contains more information, so the Peters filtering method was proposed. Xia et al. [15] proposed a two-stage framework Hydra, which introduced genetic algorithm and ensemble learning to obtain the common information between the source project and the target project.

The existing methods have improved the performance of CPDP models but still have some weaknesses. The first is that the influence of different feature distribution between source and target projects still plays a role in the performance of prediction models, leading the bad predictive results on the target projects. We decide that we can try to use effective feature transfer methods to reduce the differences between source and target projects. The second one is the existing methods only alleviate the influence of class imbalance data from one aspect, which we consider to solve this problem from several aspects may have a good effect.

3 METHODOLOGY

In this paper, we divide the CPDP problem into two stages. Stage one is the feature transfer stage, and stage two is the classification stage. We use different solutions at these two stages to improve the overall CPDP model performance. In the feature transfer phase, we use the Pearson correlation coefficient method to select the features to reduce the feature space dimension and make the distribution of features similar. In the classification phase, we use the SMOTE method and the Voting method to alleviate the class imbalance problem from both data and algorithm level. In 3.1 and 3.2, we describe the two-stage solutions in detail.

Fig. 1 shows the overall process of this method.

Firstly, we merge the data of the source project and the target project, removing the class label. The second step is to calculate the Pearson correlation coefficient between features on the merged dataset to determine the degree of correlation between features. The third step is to select features with small correlation by sorting the correlation degree. Overrate source and target project datasets using these features. The fourth step, train the Voting method on the filtered source project dataset. The fifth step is to apply the trained classifier to the filtered target project dataset for prediction. Step 6, output the classification results.

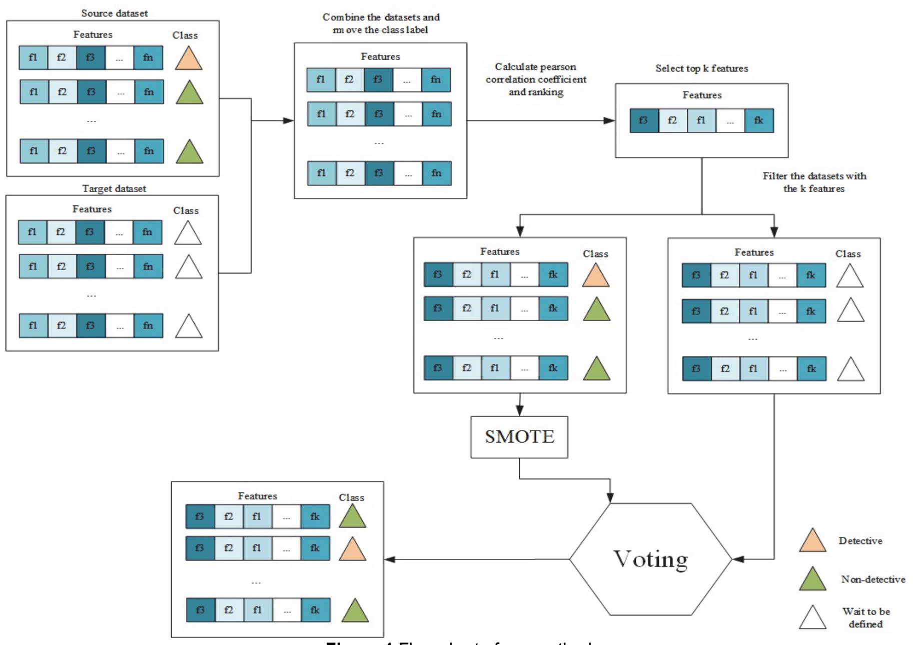
Source dataset
Features
Class
Target dataset
Combine the datasets and rm ove the class label
Calculate pearson correlation coefficient and ranking
Select top k features
Filter the datasets with the k features
SMOTE
Voting
Detective
Non-detective
Wait to be defined

Figure 1 Flow chart of our method

3.1 Feature Transfer by Pearson Correlation Coefficient

The main goal of this stage is to reduce the dimension of feature space through feature transfer and make the feature distribution of source and target projects semblable.

In order to alleviate the differences in distribution of source and target projects, we decide to choose the features which have similar distribution in both projects. We use the distance between the samples to measure whether the features are related or not. The Euclidean distance is usually used to measure the distance. But in the defect data sets, spatial dimension of the features is always high. The Euclidean distance is not very accurate in this data set. To fit the high dimension space, we choose the Pearson correlation coefficient to measure the distance of samples in this paper.

The source project dataset D_s and the target project dataset D_t are merged into a new dataset after removing the class label. The Pearson correlation coefficient between each feature is calculated in turn, and the features are sorted according to the coefficient from small to large. The strong correlation features are removed by sorting results, and finally, k features are obtained. The k features are used to filter D_s and D_t. The filtered datasets D_{sf} and D_{tf} are the input projects in the classification stage.

The formula of Pearson correlation coefficient r is as Eq. (1):

$$r = \frac{\sum_{i=1}^{n}\left(fX_i - f^{-}X\right)\left(fY_i - f^{-}Y\right)}{\sqrt{\sum_{i=1}^{n}\left(fX_i - f^{-}X\right)^2}\sqrt{\sum_{i=1}^{n}\left(fY_i - f^{-}Y\right)^2}} \tag{1}$$

Among the formula, fX_i and fY_i are the values of feature fX and feature fY in the i sample, fX^{-} and fY^{-} are the total average values of feature fX and fY.

Through the calculation of the Pearson correlation coefficient, the correlation degree is sorted from large to small, and the purpose of reducing the feature space dimension is achieved by removing the strong correlation characteristics.

By reducing the feature dimension, the efficiency of the given task can be improved, and the low-dimensional feature set can be selected from the initial high-dimensional feature set to optimize the feature space of the target project and the source project according to certain evaluation criteria, to achieve the purpose of similar feature distribution. The strong correlation feature belongs to the repetitive feature, which is redundant for the prediction model, so it can be removed. Moreover, reducing the dimension can reduce the impact of redundant data on feature distribution, thereby improving the similarity of feature distribution between source and target projects.

3.2 Feature Transfer by Pearson Correlation Coefficient

The class imbalance problem, which is considered as a major factor affecting the performance of CPDP model. In this paper, we alleviate the negative impact of class imbalance from both data and algorithm levels.

At the data level, the class imbalance problem can be solved by sampling method. The purpose of sampling is to balance the majority and minority classes. As undersampling has a disadvantage that may lead to the loss of information, we use SMOTE method in this paper. SMOTE method is based on KNN, oversampling the source project dataset. In the source project dataset of CPDP, the minority class is the non-defective class. The SMOTE method adds new non-defective samples to the source project dataset by calculating the K-nearest neighbor of a non-defective sample to achieve the purpose of increasing the number of non-defective samples. The problem of class imbalance is alleviated from the data level.

At the algorithm level, we use the Voting method to integrate four base classifiers, J48, Random Forest, REPTree and Naive Bayes.

Voting is a type of ensemble learning method. The Voting method integrates multiple based classifiers and adopts the strategy of minority obeys majority in the prediction process. The existing researches prove that Random Forest and Naïve Bayes have good performances as a single classification in CPDP while J48's and REPTree's performances are mediocre [20-23]. In Voting method, these four classifications can correct each other as strong and weak classifications. This mechanism can help classifying samples correctly. Compared with a single learner, the Voting method is not easily affected by class imbalance data and has stronger generalization, which is suitable for class imbalance in CPDP problems.

In the previous stage, we obtained Dsf and Dtf. We conduct SMOTE oversampling on the source project dataset to obtain the over-sampled source project dataset Dsfsmote. Dsfsmote is input into the Voting-based ensemble learning classifier for training. The trained classifier is applied to Dtf, and the prediction results Dtfpre is finally output. The process of Voting is shown in Fig. 2.

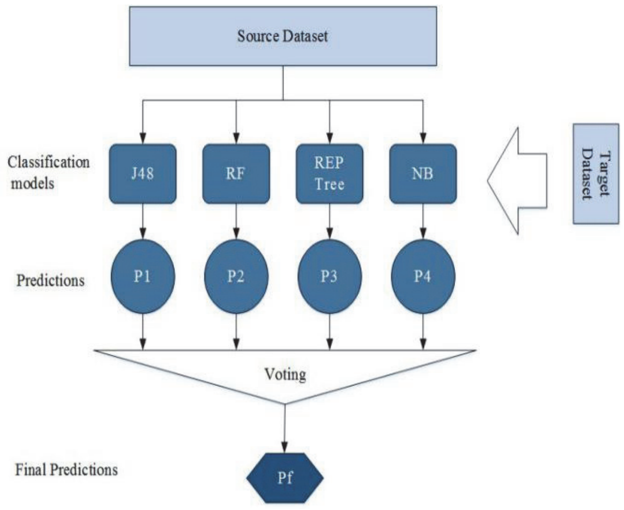

Figure 2 An example of oversampling with SMOTE

4 EXPERIMENTAL SETUP

Three research questions are raised from different perspectives to discuss the effectiveness of the proposed hybrid machine learning method. And we designed experiments to answer the research questions.

4.1 Research Question

For our method proposed in this paper, we proposed research questions on the effectiveness of the method from three aspects. For each problem, we designed experiments to analyze and validate.

To verify the effectiveness of our method, we proposed three research questions:

- RQ1: Is there a difference between using Voting and other single classifiers?
- RQ2: What are the differences between using and not using feature transfer based on the Pearson correlation coefficient?
- RQ3: How does the proposed method perform compared with the existing classical CPDP methods?

To answer the three research questions above, we designed three groups of experiments.

4.2 Experimental Design

For RQ1, in the classification phase of our method in this paper, we used the Voting method, integrating four classifiers, J48, RandomForest (RF), REPTree and NaiveBayes (NB), which have been verified to have good classification results in CPDP. We used the Voting method and J48, RF, REPTree and NB method to classify the data processed in the feature transfer stage of this paper to observe the difference between the four classifiers after Voting integration and the use of a single classifier.

For RQ2, we compare three types of CPDP models with different feature transfer methods. The first model (None) does nothing on feature transfer step. The second model (Pearson) does feature transfer by the Pearson correlation coefficient. The last model (Clustering) does feature transfer by clustering based on the Euclidean distance. The classification steps of these three models use the same Voting method. The goal of this experiment is to observe weather the model can be improved by the feature transfer based on Pearson correlation coefficient or not.

For RQ3, we compared our method with four classical CPDP methods, including transfer component analysis (TCA) [24, 25], Peters filter [19], Burak filter [18], and ALTRA [26].

The experimental environment is python 3.7, Weka and Pycharm, and the main programming language is Python. The processor is the Core i7 processor, and the memory is 12G. The operating system is Windows 10.

4.3 Dataset

FAEEEM is a public dataset collected by D'Ambros et al. [27]. The dataset contains data from several projects, which include multiple data characteristics and corresponding values. The experiments used five projects in the AEEEM dataset, namely, Equinox (EQ), Eclipse JDT Core (JDT), Apache Lucene (LC), Mylyn (ML) and Eclipse PDE UI (PDE). Tab. 1 shows the details of these five projects.

Table 1 Details of projects in the AEEEM dataset

Project	Number of samples	Number of features	Number of defective samples	Defective rate
EQ	324	61	29	39.81%
JDT	997	61	206	20.66%
LC	691	61	64	9.26%
ML	1862	61	245	13.16%
PDE	1497	61	209	13.96%

4.3 Evaluation Metrics

In this paper, the evaluation metrics of the CPDP model are *F*1-*measure*, Matthews correlation coefficient (*MCC*), and area under curve (*AUC*). The confusion matrix is useful for evaluating the two-class problem. F1-measure, MCCand AUC could be obtained based on True Positive (*TP*), False Positive (*FP*), True Negative (*TN*), and False Negative (*FN*). In this paper, *TP* means that the defective sample is correctly classified as defective. *FP* means that the defective sample is wrongly classified as non-defective. *FN* means that the defective sample is wrongly classified as non-defective. *TN* means that the non-defective sample is correctly classified as non-defective.

(1) $F1$-*measure*:

Precision (P) is the ratio of the real defective samples to samples that are correctly classified as defective. It is calculated by Eq. (2).

$$P = \frac{TP}{TP + FP} \tag{2}$$

Recall (R) is the ratio of the samples which are correctly classified as defective to the real defective samples. It can be calculated by Eq. (3).

$$R = \frac{TP}{TP + FN} \tag{3}$$

$F1$-*measure* is the harmonic mean between Precision and Recall. The higher the $F1$-*measure*, the more ideal the prediction model. $F1$-*measure* is calculated by Eq. (4).

In $F1$-*measure*, precision and recall play down the same significance which means $F1$-*measure* can fairly weigh these two evaluation indexes and combine them into a new index. A good model can achieve a high $F1$-*meausre*.

$$F1 - measure = 2 \times \frac{P \times R}{P + R} \tag{4}$$

(2) *MCC*:

The *MCC* measure takes into account all items of the confusion matrix, which can be calculated by Eq. (5).

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}} \tag{5}$$

MCC measure is the geometric mean of the regression coefficients of the problem and its dual. *MCC* can reflect the correlation between predicted and actual results. When the results of the CPDP model are totally different from the truth, *MCC* is less than 0.

(3) AUC:

AUC is unaffected by the class imbalance problem as well as being independent of the prediction threshold.

A better method can reach a higher AUC. When the value of AUC is between 0.5 and 1, the method has predictive value. When the value of AUC is less than 0.5, the predictive capability of the method is worse than random predictions.

Besides, Friedman test is also adopted to prove the difference between the methods in comparison.

5 RESULTS AND ANALYSIS

5.1 Answering RQ1

To observe the difference between using the Voting classifier and single classifier in the classification stage, we used the Voting classifier and four single classifiers in the CPDP model on the AEEEM dataset after feature transfer in one-to-one mode. These four single classifiers were J48, RandomForest, REPTree and NaiveBayes, respectively.

Table 2 Fl-measure, MCC, AUG of Voting, J48, RF, REPTtee, NB

*F*l-*measure*						*MCC*			*AUG*						
	Voting	J48	RF	REPTree	NB	Voting	J48	RF	REPTree	NB	Voting	J48	RF	REPTree	NB
JDT-EQ	0.875	0.558	0.598	0.623	0.590	0.613	0.271	0.277	0.293	0.288	0.944	0.559	0.722	0.602	0.628
LC-EQ	0.642	0.601	0.574	0.562	0.639	0.303	0.261	0.261	0.263	0.327	0.559	0.467	0.661	0.689	0.754
ML-EQ	0.611	0.553	0.562	0.606	0.625	0.291	0.203	0.263	0.237	0.316	0.494	0.506	0.539	0.634	0.747
PDE-EQ	0.646	0.613	0.550	0.613	0.630	0.333	0.265	0.165	0.303	0.339	0.618	0.644	0.649	0.766	0.701
EQ-JDT	0.680	0.438	0.474	0.474	0.775	0.307	0.194	0.193	0.193	0.426	0.784	0.557	0.728	0.728	0.809
LC-JDT	0.785	0.898	0.941	0.850	0.820	0.333	0.696	0.823	0.577	0.438	0.809	0.874	0.966	0.842	0.803
ML-JDT	0.793	0.801	0.787	0.723	0.823	0.372	0.368	0.321	0.176	0.442	0.793	0.595	0.729	0.557	0.799
PDE-JDT	0.832	0.783	0.824	0.778	0.818	0.474	0.328	0.446	0.307	0.451	0.801	0.656	0.812	0.766	0.814
EQ-LC	0.769	0.598	0.652	0.553	0.786	0.271	0.159	0.196	0.175	0.284	0.776	0.706	0.745	0.650	0.781
JDT-LC	0.895	0.854	0.942	0.884	0.819	0.675	0.561	0.823	0.649	0.440	0.949	0.863	0.968	0.885	0.695
ML-LC	0.895	0.853	0.880	0.855	0.882	0.336	0.213	0.247	0.183	0.310	0.722	0.531	0.681	0.584	0.782
PDE-LC	0.888	0.844	0.814	0.654	0.833	0.317	0.242	0.229	0.164	0.283	0.760	0.699	0.624	0.687	0.765
EQ-ML	0.710	0.523	0.558	0.486	0.721	0.162	0.065	0.102	0.064	0.193	0.631	0.612	0.585	0.537	0.653
JDT-ML	0.819	0.823	0.831	0.792	0.796	0.253	0.165	0.228	0.242	0.228	0.707	0.549	0.699	0.667	0.500
LC-ML	0.833	0.838	0.839	0.834	0.825	0.205	0.242	0.235	0.208	0.238	0.674	0.606	0.659	0.580	0.673
PDE-ML	0.818	0.827	0.821	0.823	0.783	0.233	0.216	0.226	0.191	0.227	0.668	0.568	0.671	0.565	0.667
EQ-PDE	0.666	0.152	0.335	0.147	0.779	0.146	0.042	0.123	0.058	0.244	0.667	0.582	0.584	0.516	0.728
JDT-PDE	0.816	0.789	0.821	0.785	0.828	0.262	0.241	0.274	0.245	0.258	0.703	0.594	0.718	0.660	0.718
LC-PDE	0.833	0.781	0.798	0.752	0.807	0.258	0.140	0.155	0.158	0.245	0.700	0.538	0.670	0.638	0.599
ML-PDE	0.828	0.810	0.822	0.810	0.818	0.254	0.173	0.215	0.173	0.241	0.651	0.587	0.637	0.587	0.684
Average	0.782	0.697	0.721	0.680	0.770	0.320	0.252	0.290	0.243	0.311	0.721	0.615	0.702	0.657	0.715

Tab. 2 shows the values of F1-measure, MCC, AUC of the above five classifiers in 20 pairs of source-target projects as well as the average values. And the box plot and bar plot are shown in Fig. 3 and Fig. 4.

From the overall effect, the average value of F1-measure by the Voting method was higher than that by J48, RF, REPTree and NB method alone, which were 12.2%, 8.4%, 14.9% and 1.5% higher, respectively. In terms of MCC values, the CPDP model using Voting produced higher average MCC values than the CPDP model using J48, RF, REPTree and NB alone, indicating better prediction performance. As shown in Tab. 2, the average AUC values of CPDP model with Voting were higher than those with J48, RF, REPTree and NB, which were 17.2%, 2.6%, 9.7% and 0.8% higher, respectively.

From an individual point of view, Voting achieved a higher F1-measure than other separate classifiers in 11 of 20 pairs of experiments shown in bold in Tab. 2, indicating its relatively good prediction performance. The MCC values of the CPDP model with Voting and J48, RF, REPTree and NB were all greater than 0, which meant that their performances were better than random prediction.

It could be seen from the box plot that Voting and NB were more stable than other classifiers, and the fluctuation of RF was relatively large. Based on those results, we could deduce that the CPDP model using Voting performs better.

Tab. 3 and Fig. 4 show the average order values of the Friedman test using Voting and using J48, RF, REPTree and NB separately. We assumed that all algorithms perform the same operation. When α = 0.05 and the critical value of Friedman test was 2.310, the Friedman test results Tf value of *F*1-*measure*, *MCC* and *AUC* were all greater than the critical value, so this assumption was denied. Therefore, there were significant differences in the effectiveness between Voting and using J48, RF, REPTree and NB alone.

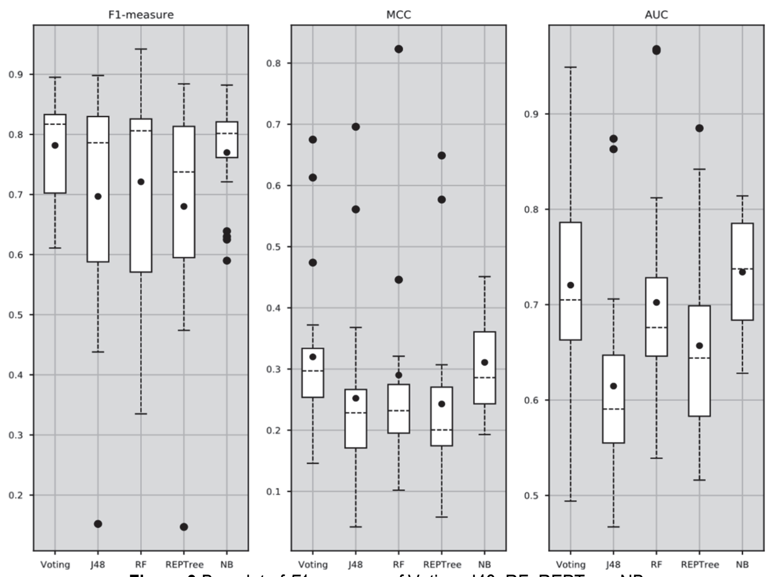

Figure 3 Box plot of *F*1-*measure* of Voting, J48, RF, REPTree, NB

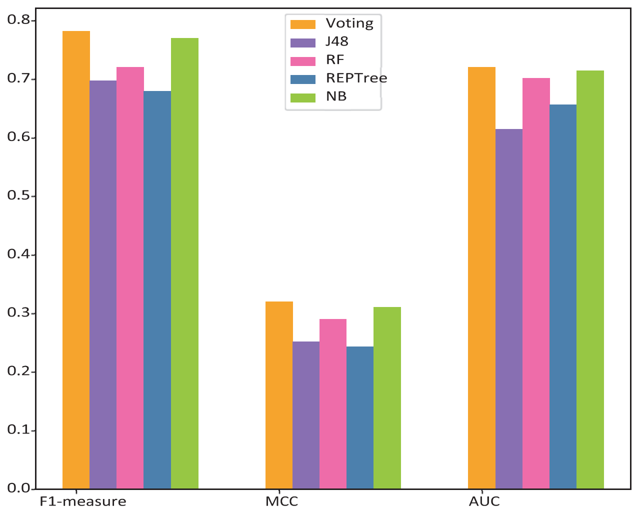

Figure 4 Bar plot of *F*1-*measure* of Voting, J48, RF, REPTree, NB

Table 3 Friedman test of Voting, J48, RF, REPTree, NB

	Voting	J48	RF	NB	
*F*1-*measure*	0.720	0.615	0.702	0.734	15.400
MCC	0.320	0.252	0.290	0.311	16.932
AUC	0.720	0.615	0.702	0.734	15.400

The above results proved that Voting had higher average values of F1measure, MCC and AUC than J48, RF, REPTree and NB. This proved that whether using Voting or not had different effects on the performance of the overall prediction model. It also showed that there were significant differences between Voting and other classifiers. CPDP model with Voting had a better effect than with other single classifiers.

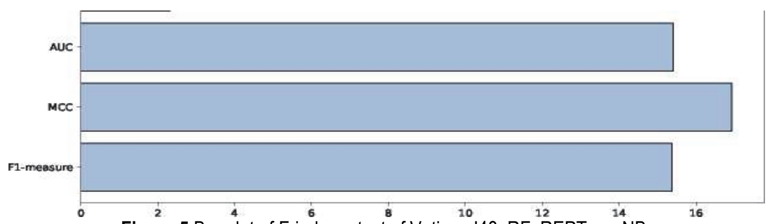

Figure 5 Bar plot of Friedman test of Voting, J48, RF, REPTree, NBr

5.2 Answering RQ2

We conducted one-on-one experiments on the AEEEM dataset using Voting as a unified classifier with good results in the previous experiment to verify the effect of feature transfer based on Pearson coefficient on CPDP performance. Before the classifier training, the training sets were over-sampled by SMOTE.

Tab. 4 shows the values of None, P-Voting and C-Voting on 20 pairs of source-target projects, and the box plot and the bar plot are shown in Fig. 6 and Fig. 7.

From the view of overall effect, the average *F*1-*measure* of P-Voting is increased by 10.2%, 10.5% than None and C-Voting, and the average MCC value is also higher than that of two CPDP models. The three CPDP models have good performance in AUC. The average AUC of CPDP is increased by 0.2% by feature transfer. From the individual point of view, the CPDP model using feature transfer based on Pearson correlation coefficient achieves a higher F1-measure in 14 of 20 pairs of experiments, as shown in bold in Tab. 4. The MCC values of the CPDP model with feature transfer based on Pearson correlation coefficient are all greater than 0, which indicates that the performance of the model is better than random prediction.

Table 4 *F*1-*measure*, *MCC*, *AUC* of None, P-Voting and C-Voting

	*F*1-*measure*			*MCC*			*AUC*		
	None	P-Voting	C-Voting	None	P-Voting	C-Voting	None	P-Voting	C-Voting
JDT-EQ	0.590	0.875	0.578	0.288	0.613	0.291	0.644	0.944	0.609
LC-EQ	0.621	0.642	0.598	0.323	0.303	0.277	0.826	0.559	0.763
ML-EQ	0.644	0.611	0.537	0.348	0.291	0.251	0.798	0.494	0.641
PDE-EQ	0.656	0.646	0.590	0.347	0.333	0.288	0.716	0.618	0.686
EQ-JDT	0.390	0.680	0.492	0.207	0.307	0.222	0.774	0.784	0.712
LC-JDT	0.788	0.785	0.771	0.337	0.333	0.407	0.783	0.809	0.808
ML-JDT	0.822	0.793	0.667	0.447	0.372	0.309	0.802	0.793	0.791
PDE-JDT	0.780	0.832	0.816	0.407	0.474	0.441	0.813	0.801	0.791
EQ-LC	0.551	0.769	0.757	0.183	0.271	0.193	0.795	0.776	0.731
JDT-LC	0.899	0.895	0.879	0.406	0.675	0.199	0.751	0.949	0.686
ML-LC	0.887	0.895	0.871	0.302	0.336	0.228	0.660	0.722	0.647
PDE-LC	0.866	0.888	0.888	0.293	0.317	0.332	0.801	0.760	0.734
EQ-ML	0.338	0.710	0.555	−0.041	0.162	0.028	0.577	0.631	0.568
JDT-ML	0.794	0.819	0.687	0.131	0.253	0.074	0.604	0.707	0.557
LC-ML	0.830	0.833	0.782	0.237	0.205	0.190	0.622	0.674	0.593
PDE-ML	0.759	0.818	0.823	0.130	0.233	0.230	0.578	0.668	0.681
EQ-PDE	0.500	0.666	0.424	0.134	0.146	0.118	0.660	0.667	0.658
JDT-PDE	0.830	0.816	0.825	0.251	0.262	0.208	0.732	0.703	0.703
LC-PDE	0.816	0.833	0.821	0.198	0.258	0.233	0.723	0.700	0.725
ML-PDE	0.823	0.828	0.805	0.265	0.254	0.094	0.725	0.651	0.683
Average	0.709	0.782	0.708	0.260	0.320	0.231	0.719	0.721	0.688

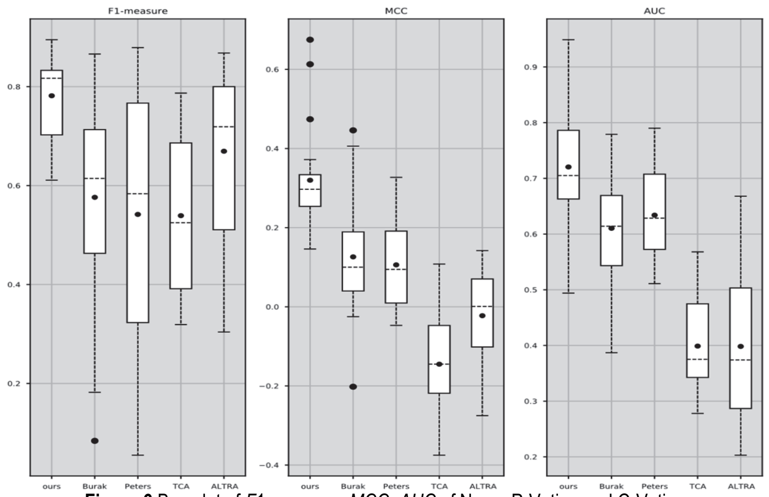

Figure 6 Box plot of *F*1-*measure*, *MCC*, *AUC* of None, P-Voting and C-Voting

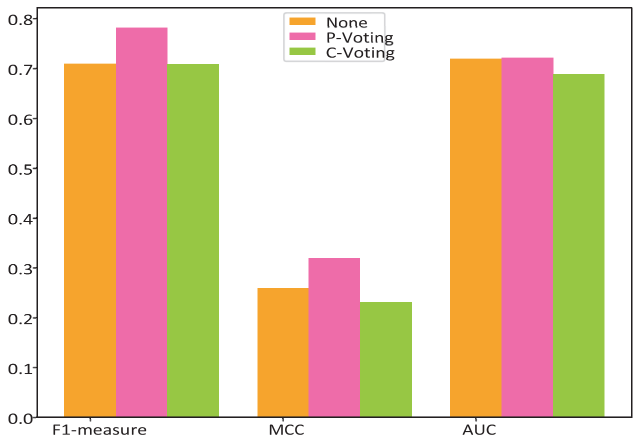

Figure 7 Bar plot of *F*1-*measure*, *MCC*, *AUC* of None, P-Voting and C-Voting

It could be seen from the box plot that P-Votings are more stable than the other two methods. Tab. 5 and Fig. 7 showed the mean order values of the Friedman test for CPDP with or without feature transfer. We assumed that all methods perform the same operation. When $\alpha = 0.05$, the critical value of Friedman test was 3.245, then Friedman test results *Tf* of *F*1-*measure*, *MCC* and *AUC* were greater than the critical value.

Table 5 Friedman test of prediction *F*1-*measure*, *MCC*, *AUC* of None, P-Voting and CVoting

	None	P-Voting	C-Voting	*Tf*
F1	0.709	0.782	0.708	15.441
MCC	0.260	0.320	0.231	15.531
AUC	0.719	0.720	0.688	15.448

The above results proved that CPDP with feature transfer based on Pearson correlation coefficient had better impacts on the performance of the overall prediction model than other two methods. Specifically, CPDP with feature transfer based on Pearson correlation had higher average values of F1-measure, 365 MCC, and AUC. And it could prove the existence of significant differences between these three methods.

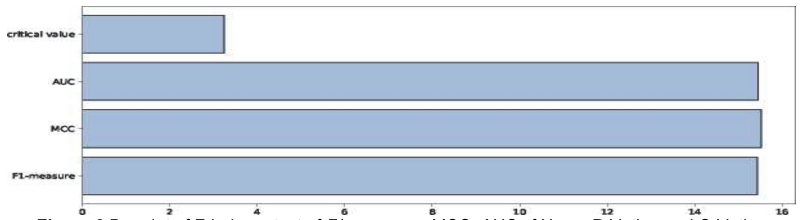

Figure 8 Bar plot of Friedman test of *F*1-*measure*, *MCC*, *AUC* of None, P-Voting and C-Voting

5.3 Answering RQ3

We compared our method with four classical CPDP methods of Peters filter (Peters), Burak filter (Burak), TCA and ALTRA.

Tab. 6 shows the *F*1-*measure*, *MCC*, *AUC* and average values of our method, Burak, Peters, TCA and ALTRA in 20 pairs of source-target projects. And the box plot and the bar plot are shown in Fig. 9 and Fig. 10.

From the overall effect, the average F1-measure of our method was 35.6%, 44.3%, 44.9% and 16.8% higher than that of Burak, Peters, TCA and ALTRA. The average MCC value of our method was also higher than that of the other four methods. The *AUC* of our method was 18.0%, 13.7%, 80.6% and 80.9% higher than that of Burak, Peters, TCA and ALTRA, respectively.

From an individual perspective, our method achieved a higher *F*1-*measure* in 17 of the 20 experiments, as shown in bold in Tab. 6. The *MCC* values of our method were all above 0, while there were values below 0 using the other four methods, indicating that the prediction results of this method were better than random prediction. As shown in Fig. 9 and Fig. 10, compared with the other four methods, our method had smaller fluctuation and was more stable.

Table 6 *F*1-*measure*, *MCC*, *AUG* of ours, Burak, Peters,TCA, ALTRA

	*F*1-*measure*					*MCC*					*AUC*				
	ours	Burak	Peters	TCA	ALTRA	ours	Burak	Peters	TCA	ALTRA	ours	Burak	Peters	TCA	ALTRA
JDT-EQ	0.875	0.687	0.587	0.375	0.448	0.613	0.446	0.192	-0.325	-0.079	0.944	0.779	0.712	0.348	0.266
LC-EQ	0.642	0.672	0.458	0.355	0.449	0.303	0.375	0.016	-0.361	-0.064	0.559	0.777	0.79	0.278	0.286
ML-EQ	0.611	0.433	0.529	0.511	0.304	0.291	-0.202	0.188	0.106	-0.275	0.494	0.387	0.592	0.489	0.253
PDE-EQ	0.646	0.473	0.509	0.346	0.415	0.333	0.119	0.121	-0.375	-0.233	0.618	0.416	0.786	0.282	0.203
EQ-JDT	0.680	0.421	0.255	0.435	0.526	0.307	0.222	0.104	-0.339	-0.109	0.784	0.746	0.617	0.291	0.388
LC-JDT	0.785	0.741	0.704	0.641	0.704	0.333	0.208	0.062	-0.132	0.062	0.809	0.64	0.649	0.406	0.271
ML-JDT	0.793	0.703	0.58	0.69	0.725	0.372	-0.003	0.209	-0.053	0.142	0.793	0.529	0.645	0.505	0.605
PDE-JDT	0.832	0.798	0.781	0.606	0.713	0.474	0.406	0.301	-0.175	0.051	0.801	0.746	0.706	0.389	0.668
EQ-LC	0.769	0.78	0.317	0.685	0.465	0.271	0.114	-0.01	-0.147	-0.152	0.776	0.594	0.578	0.326	0.297
JDT-LC	0.895	0.182	0.621	0.74	0.868	0.675	0.085	0.191	-0.149	0.091	0.949	0.665	0.713	0.377	0.441
ML-LC	0.895	0.61	0.835	0.528	0.862	0.336	0.016	-0.019	0.02	-0.017	0.722	0.535	0.588	0.551	0.341
PDE-LC	0.888	0.866	0.879	0.464	0.792	0.317	0.076	0.327	-0.143	0.078	0.760	0.462	0.737	0.362	0.605
EQ-ML	0.710	0.429	0.24	0.522	0.71	0.162	-0.012	0.085	-0.123	0.115	0.631	0.546	0.511	0.371	0.579
JDT-ML	0.819	0.619	0.325	0.787	0.751	0.253	0.086	-0.001	-0.012	-0.099	0.707	0.599	0.511	0.489	0.309
LC-ML	0.833	0.515	0.805	0.319	0.808	0.205	0.054	-0.026	-0.198	0.019	0.674	0.608	0.556	0.372	0.54
PDE-ML	0.818	0.828	0.762	0.357	0.806	0.233	0.169	0.272	-0.28	-0.018	0.668	0.64	0.673	0.307	0.287
EQ-PDE	0.666	0.084	0.076	0.616	0.644	0.146	-0.025	0.013	-0.183	-0.174	0.667	0.582	0.623	0.373	0.373
JDT-PDE	0.816	0.704	0.71	0.719	0.8	0.262	0.048	0.013	-0.109	0.077	0.703	0.682	0.513	0.423	0.388
LC-PDE	0.833	0.479	0.808	0.693	0.8	0.258	0.183	0.128	-0.03	0.068	0.700	0.62	0.634	0.47	0.375
ML-PDE	0.828	0.503	0.055	0.397	0.8	0.254	0.159	-0.047	0.108	0.068	0.651	0.654	0.545	0.568	0.491
Average	0.782	0.576	0.542	0.539	0.670	0.320	0.126	0.106	-0.145	-0.022	0.721	0.610	0.634	0.399	0.398

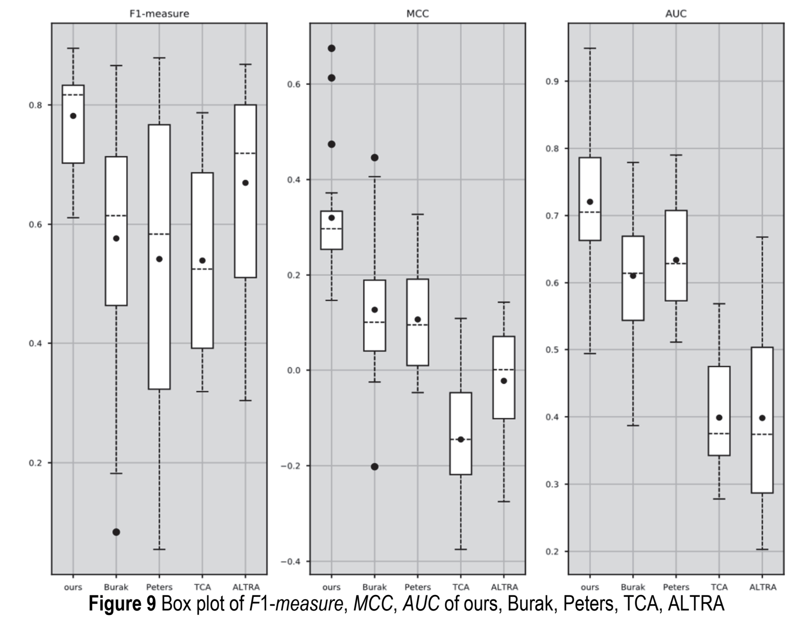

Figure 9 Box plot of *F*1-*measure*, *MCC*, *AUC* of ours, Burak, Peters, TCA, ALTRA

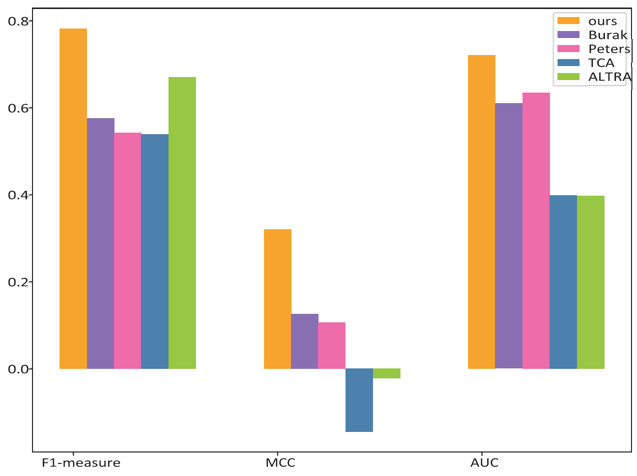

Figure 10 Bar plot of *F*1-*measure*, *MCC*, *AUC* of ours, Burak, Peters,TCA, ALTRA

Tab. 7 and Fig. 11 showed the average ordinal values of the Friedman test for our method, Burak, Peters, TCA and ALTRA. We assumed that all methods perform the same operation. When α = 0.05, the critical value of the Friedman test was 2.492, and Tf of F1-measure, MCC and AUC were greater than the critical value. To reject this assumption. There were significant differences in the effectiveness of our method, Burak, Peters, TCA and ALTRA methods.

Table 7 Friedman test of ours, Burak, Peters, TCA, ALTRA

	ours	Burak	Peters	TCA	ALTRA	T_f
*F*1-*measure*	0.782	0.576	0.542	0.539	0.670	15.417
MCC	0.320	0.126	0.106	-0.145	-0.022	15.536
AUC	0.720	0.610	0.634	0.399	0.398	15.441

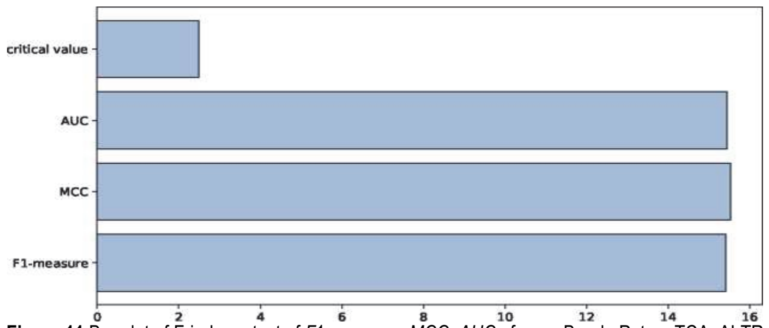

Figure 11 Bar plot of Friedman test of *F*1-*measure*, *MCC*, *AUC* of ours, Burak, Peters,TCA, ALTR

Compared with Burak, Peters, TCA and ALTRA, our method had higher average values in *F*1-*measure*, *MCC* and *AUC*. This proved that our method had a great improvement in performance compared with the existing four classical CPDP methods. And there were significant differences between these methods.

6 THREATS TO VALIDITY

External validity, internal validity and construct validity are the three main threats in this paper. The threats are illustrated as follows.

6.1 External Validity

The main threat to the external validity of this paper is that the dataset used in this paper contains five projects, and the application of our method on 31 400 other projects may produce better or worse results. Moreover, the dataset we use is open source, and the effect on other closed source datasets cannot be judged.

6.2 Internal Validity

The main threat to the internal validity of this paper comes from the number 405 of feature selections. The different number of features may bring the different effects of feature transfer. To solve this problem, we have tried many times to determine the relatively effective and stable number of features for the dataset features used in this paper.

6.3 Construct Validity

First, in this paper, the parameters of the classifier are unified using the default parameters in Weka. We may get better or worse effects for modifying classifier parameters. The selection of classifier parameters is a direction of our future work. As a hot spot in SDP, CPDP has put forward many methods for CPDP. However, CPDP has always been relatively low performance compared to SDP within the project, mainly for two reasons. The first is the existing enormous distribution differences between the projects, which leads to the reduction of the classifier effect. The second is the class imbalance problem in the project, that is, the number of modules with bugs is far less than that without bugs. However, in practice, there is also a need to validate predictive models against other evaluation metrics. More evaluation measures will be considered in our future work.

7 CONCLUSION AND FUTURE WORK

In this paper, we propose a CPDP method, which combines feature transfer and ensemble learning, aiming at two problems in CPDP that have a great influence on the prediction results: feature distribution variability and class imbalance. Three experiments are designed to analyze the effectiveness of our method from three aspects. The experiment was carried out on five projects from the AEEEM dataset. The evaluation indexes were *F*1-*measure*, *MCC* and *AUC*. In the first experiment, we use the Voting method and four single classifiers to predict the same dataset. The performances of the dataset with our feature transfer method and the dataset not without our feature transfer method were compared in the second experiment. In the third experiment, we compared our method with four classical CPDP methods, including TCA, Burak Filter, Peters Filter and ALTRA. The experimental results show that our method is effective in feature transfer and classification stages. Compared with the classical methods, our method has a significant improvement.

In this paper, we choose J48, Random Forest, REPTree and Naïve Bayes as the base classifiers in Voting as they have good performance in CPDP. The parameters of the classifier default from weka. In future work, we will study the parameter setting of the base classifier. And we can add more base classifiers such as support vector machine, linear regression, et al., in ensemble learning method, using the optimization algorithm to find the optimal combination of classifiers. We will work on studying the suitable combination of classifiers to improve the performance of CPDP model. And we will also consider the performance of other ensemble learning methods such as AdaBoost and GBDT, et al., to find out a combinatorial method that works well on CPDP.

8 REFERENCES

[1] Menzies, T., Milton, Z., Turhan, B., Cukic, B., Jiang, Y., & Bener, A. (2010). Defect prediction from static code features: current results, limitations, new approaches. *Automated Software Engineering*, *17*(4), 375-407. https://doi.org/10.1007/s10515-010-0069-5

[2] He, Z., Shu, F., Yang, Y., Li, M., & Wang, Q. (2012). An investigation on the feasibility of cross-project defect prediction. *Automated Software Engineering*, *19*(2), 167-199. https://doi.org/10.1007/s10515-011-0090-3

[3] Saifudin, A., Trisetyarso, A., Suparta, W., Kang, C. H., Abbas, B. S., & Heryadi, Y. (2020). Feature Selection in Cross-Project Software Defect Prediction. *In GJournal of Physics: Conference Series*, *1569*(2), p. 022001). IOP Publishing. https://doi.org/10.1088/1742-6596/1569/2/022001

[4] Zhang, F., Zheng, Q., Zou, Y., & Hassan, A. E. (2016, May). Cross-project defect prediction using a connectivity-based unsupervised classifier. *In 2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*, 309-320. IEEE. https://doi.org/10.1145/2884781.2884839

[5] Li, N., Shepperd, M., & Guo, Y. (2020). A systematic review of unsupervised learning techniques for software defect prediction. *Information and Software Technology*, *122*, 106287. https://doi.org/10.1016/j.infsof.2020.106287

[6] Pendharkar, P. C. (2010). Exhaustive and heuristic search approaches for learning a software defect prediction model. *Engineering Applications of Artificial Intelligence*, *23*(1), 34-40. https://doi.org/10.1016/j.engappai.2009.10.001

[7] Ma, Y., Luo, G., Zeng, X., & Chen, A. (2012). Transfer learning for cross-company software defect prediction. *Information and Software Technology*, *54*(3), 248-256. https://doi.org/10.1016/j.infsof.2011.09.007

[8] Qing, H., Biwen, L., Beijun, S., & Xia, Y. (2015, November). Cross-project software defect prediction using feature-based transfer learning. *In Proceedings of the 7th Asia-Pacific Symposium on Internetware*, 74-82. https://doi.org/10.1145/2875913.2875944

[9] Pan, S. J. & Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, *22*(10), 1345-1359. https://doi.org/10.1109/TKDE.2009.191

[10] Cao, Q., Sun, Q., Cao, Q., & Tan, H. (2015, October). Software defect prediction via transfer learning based neural network. *In 2015 First International Conference on Reliability Systems Engineering (ICRSE)*, 1-10. IEEE. https://doi.org/10.1109/ICRSE.2015.7366475

[11] Liu, R., Li, N., Peng, L., & Wu, K. (2021, March). A Special Point and Transfer Component Analysis Based Dynamic Multi-objective Optimization Algorithm. *In International Conference on Evolutionary Multi-Criterion Optimization*, 218-231. Springer, Cham. https://doi.org/10.1007/978-3-030-72062-9_18

[12] Chen, Y. S., Hsu, C. S., & Lo, C. L. (2020). An entire-and-partial feature transfer learning approach for detecting the frequency of pest occurrence. IEEE Access, 8, 92490-92502. https://doi.org/10.1109/ACCESS.2020.2992520

[13] Moreno-Torres, J. G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N. V., & Herrera, F. (2012). A unifying view on dataset shift in classification. *Pattern recognition*, *45*(1), 521-530. https://doi.org/10.1016/j.patcog.2011.06.019

[14] Hosseini, S., Turhan, B., & Gunarathna, D. (2017). A systematic literature review and meta-analysis on cross project defect prediction. *IEEE Transactions on Software Engineering*, *45*(2), 111-147. https://doi.org/10.1109/TSE.2017.2770124

[15] Xia, X., Lo, D., Pan, S. J., Nagappan, N., & Wang, X. (2016). Hydra: Massively compositional model for cross-project defect prediction. *IEEE Transactions on software Engineering*, *42*(10), 977-998. https://doi.org/10.1109/TSE.2016.2543218

[16] Ni, C., Liu, W., Gu, Q., Chen, X., & Chen, D. (2017, July). FeSCH: a feature selection method using clusters of hybrid-data for cross-project defect prediction. *In 2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, *1*, 51-56. IEEE. https://doi.org/10.1109/COMPSAC.2017.127

[17] Krishna, R., Menzies, T., & Fu, W. (2016, August). Too much automation? The bellwether effect and its implications for transfer learning. *In Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, 122-131. https://doi.org/10.1145/2970276.2970339

[18] Turhan, B., Menzies, T., Bener, A. B., & Di Stefano, J. (2009). On the relative value of cross-company and within-company data for defect prediction. *Empirical Software Engineering*, *14*(5), 540-578. https://doi.org/10.1007/s10664-008-9103-7

[19] Peters, F., Menzies, T., & Marcus, A. (2013, May). Better cross company defect prediction. *In 2013 10th Working Conference on Mining Software Repositories (MSR)*, 409-418. IEEE. https://doi.org/10.1109/MSR.2013.6624057

[20] Gunarathna, D., Turhan, B., & Hosseini, S. (2016). A systematic literature review on cross-project defect prediction.

[21] Chen, J., Yang, Y., Hu, K., Xuan, Q., Liu, Y., & Yang, C. (2019). Multiview transfer learning for software defect prediction. *IEEE Access*, *7*, 8901-8916. https://doi.org/10.1109/ACCESS.2018.2890733

[22] Pant, M., Ray, K., Sharma, T. K., Rawat, S., & Bandyopadhyay, A. (2016). Soft computing: theories and applications. *Proc SoCTA*, *2*.

[23] Srivastava, S., Rani, S., Singh, S., Singh, S., & Vashisht, R. (2020). Heterogeneous Cross Project Defect Prediction in Software. *In Proceedings of the International Conference on Innovative Computing & Communications (ICICC)*. http://doi.org/10.2139/ssrn.3580671

[24] Pan, S. J., Tsang, I. W., Kwok, J. T., & Yang, Q. (2010). Domain adaptation via transfer component analysis. *IEEE transactions on neural networks*, *22*(2), 199-210. https://doi.org/10.1109/TNN.2010.2091281

[25] Chen, J., Hu, K., Yang, Y., Liu, Y., & Xuan, Q. (2020). Collective transfer learning for defect prediction. *Neurocomputing*, *416*, 103-116. https://doi.org/10.1016/j.neucom.2018.12.091

[26] Yuan, Z., Chen, X., Cui, Z., & Mu, Y. (2020). ALTRA: Cross-project software defect prediction via active learning and tradaboost. *IEEE Access*, *8*, 30037-30049. https://doi.org/10.1109/ACCESS.2020.2972644

[27] D'Ambros, M., Lanza, M., & Robbes, R. (2012). Evaluating defect prediction approaches: a benchmark and an extensive comparison. *Empirical Software Engineering*, *17*(4), 531-577. https://doi.org/10.1007/s10664-011-9173-9

Contact information:

Fuping ZENG, Lectural
School of Reliability and Systems Engineering,
Beihang University, Beijing100191, China
E-mail: zfp@buaa.edu.cn

Wanting LIN, Master
(Corresponding author)
School of Artificial Intelligence,
Beijing University of Posts and Telecommunications, Beijing 100876, China
E-mail: midawwood@gmail.com

Ying XING, Associate Professor
School of Artificial Intelligence,
Beijing University of Posts and Telecommunications, Beijing 100876, China
E-mail: xingying@bupt.edu.cn

Lu SUN, Master
School of Reliability and Systems Engineering,
Beihang University, Beijing 100191, China
E-mail: sunlu030@163.com

Bin YANG, PhD
Du Xiaoman (Beijing) Science Technology Co., Ltd., Beijing 100094, China
E-mail: researcher_yang@outlook.com