Heterogeneous Cross-Company Defect Prediction by Unified Metric Representation and CCA-Based Transfer Learning

Xiaoyuan Jing[1,2,*], Fei Wu[1,2], Xiwei Dong[1,2], Fumin Qi[1], Baowen Xu[3,1,*]

[1]State Key Laboratory of Software Engineering, School of Computer, Wuhan University, China

[2]School of Automation, Nanjing University of Posts and Telecommunications, China

[3]Department of Computer Science and Technology, Nanjing University, China

[*]Corresponding author: jingxy_2000@126.com, bwxu@nju.edu.cn

ABSTRACT

Cross-company defect prediction (CCDP) learns a prediction model by using training data from one or multiple projects of a source company and then applies the model to the target company data. Existing CCDP methods are based on the assumption that the data of source and target companies should have the same software metrics. However, for CCDP, the source and target company data is usually heterogeneous, namely the metrics used and the size of metric set are different in the data of two companies. We call CCDP in this scenario as heterogeneous CCDP (HCCDP) task. In this paper, we aim to provide an effective solution for HCCDP. We propose a unified metric representation (UMR) for the data of source and target companies. The UMR consists of three types of metrics, i.e., the common metrics of the source and target companies, source-company specific metrics and target-company specific metrics. To construct UMR for source company data, the target-company specific metrics are set as zeros, while for UMR of the target company data, the source-company specific metrics are set as zeros. Based on the unified metric representation, we for the first time introduce canonical correlation analysis (CCA), an effective transfer learning method, into CCDP to make the data distributions of source and target companies similar. Experiments on 14 public heterogeneous datasets from four companies indicate that: 1) for HCCDP with partially different metrics, our approach significantly outperforms state-of-the-art CCDP methods; 2) for HCCDP with totally different metrics, our approach obtains comparable prediction performances in contrast with within-project prediction results. The proposed approach is effective for HCCDP.

Categories and Subject Descriptors

D.2.9 [**Software Engineering**]: Management-*Software quality assurance (SQA),* D.2.5 [**Software Engineering**]: Testing and Debugging-*Code inspections and walk-throughs*.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.

ESEC/FSE'15, August 30 – September 4, 2015, Bergamo, Italy
© 2015 ACM. 978-1-4503-3675-8/15/08...$15.00
http://dx.doi.org/10.1145/2786805.2786813

General Terms

Experimentation, Measurement, Reliability, Verification

Keywords

Heterogeneous cross-company defect prediction (HCCDP), common metrics, company-specific metrics, unified metric representation, canonical correlation analysis (CCA).

1. INTRODUCTION

Software defect prediction (SDP) is one of the most important research topics in software engineering, which has attracted a lot of attention from both academic and industrial communities [1-5]. Most prior studies on this issue attempt to train predictors based on historical data to detect the defect proneness of new software modules within the same company [6-11], which is called within-company defect prediction (WCDP). However, for a new company or companies with limited historical data, there is not enough training defect data to build a prediction model. In this case, it is hard to perform within-company defect prediction.

Fortunately, there are many open source defect datasets available, such as the PROMISE repository [12]. A potential way of predicting defects in projects without historical data is to make use of these public data sets. In recent years, several cross-company defect prediction (CCDP) methods have been developed [13-15], such as nearest-neighbor filter (NN-filter) [16], transfer Naive Bayes (TNB) [17], double transfer boosting (DTB) [18], CLIFF+MORPH [19], etc. They learn prediction model by using sufficient training data from existing source projects and then apply the model to the target project.

1.1 Motivation

Existing CCDP methods are based on the assumption that the data of source and target companies should have the same software metrics. In fact, since different companies might select different programming languages, develop software modules according to different user requirements, and test software modules from different aspects, there usually exist different metrics in the data of different companies. Table 1 tabulates the numbers of metrics in defect data of companies including NASA [12, 20-21], SOFTLAB [12], ReLink [22-23] and AEEEM [23-24]. Table 2 shows the numbers of common metrics between projects of these four companies. Figure 1 illustrates the detailed metrics used in defect data of four companies. In the figure, we outline the common metrics shared by two different companies with rectangle boxes in different colors. Specifically, the red rectangle box outlines the common metrics of NASA and

Figure 1. List of the metrics used in defect data of four companies.

SOFTLAB, the green rectangle box outlines the common metrics of NASA and ReLink, and the blue rectangle box outlines the common metrics of SOFTLAB and ReLink. It is noted that only 40 metrics of AEEEM are listed in the figure. From Tables 1, 2 and Figure 1, we can see that the types of metrics and the sizes of metric sets are varied in these companies.

Table 1. Number of metrics in projects of four companies

Company	NASA	SOFTLAB	ReLink	AEEEM
Number of metrics	38	29	26	61

Table 2. Number of common metrics between projects of different companies

Company A∩ Company B	NASA∩SOFTLAB	NASA∩ReLink	NASA∩AEEEM
Number	28	3	0
Company A∩ Company B	SOFTLAB∩ReLink	SOFTLAB∩AEEEM	ReLink∩ AEEEM
Number	3	0	0

If we want to detect the defect proneness of modules in NASA by regarding the defect data from SOFTLAB, ReLink or AEEEM as training data, existing CCDP methods can only make use of the common metrics shared by NASA and SOFTLAB or ReLink. However, the size of common metric set across different companies may be very small (like the common metric set between NASA and ReLink), and the metrics except the common metrics might have favorable discriminant ability. Since no common metrics exist in NASA and AEEEM, existing CCDP methods cannot make use of defect data in AEEEM. We call CCDP in this scenario as heterogeneous CCDP (HCCDP).

In this paper, we answer the following three research questions:

RQ 1: How to design an effective approach for HCCDP?

RQ 2: Is the heterogeneous cross-company defect data helpful for cross-company defect prediction?

RQ 3: If the heterogeneous cross-company defect data can help prediction, when it helps the most?

1.2 Contribution

The contributions of our study are summarized as the following two points:

1. Focusing on heterogeneous CCDP (HCDDP), we propose a unified metric representation (UMR) for the data of source and target companies. The UMR consists of three types of metrics including the common metrics of the source and target companies, source-company specific metrics, and target-company specific metrics. We set the target-company specific metrics as zeros when constructing the UMR for source company data, and set the source-company specific metrics as zeros when constructing the UMR for target company data.

2. Based on the unified metric representation, we for the first time introduce the transfer learning method named canonical correlation analysis (CCA) [25] into CCDP for making the data distribution of target company similar to that of source company. CCA is an effective machine learning method, which can maximize the correlation of source and target data.

We call the proposed approach for HCCDP as CCA+. We conduct experiments on 14 public datasets from four companies including NASA [12, 20-21], SOFTLAB [12], ReLink [22-23] and AEEEM [23-24]. Experimental results demonstrate that the proposed approach can obtain desirable prediction results for HCCDP.

1.3 Organization

The rest of this paper is organized as follows: Section 2 reviews the related work. Section 3 describes the proposed CCA+ approach. Experimental results are reported in Section 4 and conclusions are drawn in Section 5.

2. RELATED WORK

In this section, we briefly review existing cross-company defect prediction (CCDP) methods, cross-project defect prediction (CPDP) methods, and canonical correlation analysis (CCA) and transfer learning methods.

2.1 Cross-company Defect Prediction (CCDP) Methods

CCDP refers to using data from other companies to build defect predictors. There exist two mainstream ways for CCDP. The first one is to find the best suitable training data for the target modules. Turhan et al. [16] found that defect predictors built on all available cross-company data dramatically increase defect detection ability, but with unacceptably high false alarm rates. They reckoned that the irrelevancies in the cross-company data lead to the false alarms and presented a nearest-neighbor filter (NN-filter) method to select training data close to within-company data. Peters et al. [13] developed the Peters filter to select training data for CCDP.

The second mainstream way for CCDP is to design effective machine learning algorithms with high generalization ability for constructing the defect predictor. Ma et al. [17] presented a cross-company prediction algorithm named transfer Naive Bayes (TNB), which estimates the distribution of the test data, transfers cross-company data information into the weights of the training data, and then builds defect prediction model on these weighted data. Recently, Chen et al. [18] developed the double transfer boosting (DTB) method for CCDP, which firstly uses data gravitation for reshaping the whole distribution of cross-company data to fit within-company data, and then designs the transfer boosting algorithm to remove negative samples in cross-company data with a small ratio of labeled within-company data.

However, existing CCDP methods are based on the restrictive assumption that the software metrics used in source and target company data should be the same. They do not apply to the scenario of heterogeneous CCDP.

2.2 Cross-project Defect Prediction (CPDP) Methods

Cross-project defect prediction (CPDP) refers to predicting defects in a project using prediction models trained from historical data of other projects [26-28]. When we only use one project in a specific company for training the prediction model, CCDP can be considered as CPDP.

Over the past recent years, we have witnessed lots of interest in developing new CPDP methods. Using 12 applications, Zimmermann et al. [29] performed 622 cross-project predictions. The results indicate that CPDP is a serious challenge, i.e., simply using models from projects in the same domain or with the same process does not lead to accurate predictions. Careful selection of training data and the characteristics of data and process play important roles in successful CPDP. He et al. [30] investigated CPDP by focusing on training data selection. They showed that the prediction results were related to the distributional attributes of datasets, which is useful for training data selection. Rahman et al. [31] introduced the measure called area under the cost effectiveness curve (AUCEC) to investigate the feasibility of CPDP and drew a conclusion that CPDP is no worse than within-project prediction in terms of AUCEC. Turhan et al. [32] introduced a mixed model for CPDP, which uses both the within- and cross-project data as training data. They concluded that the performance of the mixed model could be comparable to that of within-project prediction. Recently, Nam et al. [23] applied transfer component analysis (TCA) to CPDP, which is a feature-based transfer learning method. Furthermore, they extended TCA to TCA+ by using the normalization techniques to preprocess data, which exhibits good performance for defect prediction. Ryu et al. [33] developed the value-cognitive boosting with support vector machine method for class-imbalance issue of CPDP.

Existing CPDP methods are also based on the assumption that the data of source and target companies should have the same software metrics. If there exist partially different metrics between the source and target projects, existing CPDP methods can only make use of common metrics. When no common metrics exist between source and target projects, existing CPDP methods cannot be used for defect prediction.

2.3 Canonical Correlation Analysis (CCA) and Transfer Learning Methods

Canonical correlation analysis (CCA) [25] is a powerful tool in multivariate data analysis to find the correlation between two sets of variables. The two sets of variables can be associated with two different objects or two different views of the same object. CCA aims to learn a pair of projective transformations corresponding to the two sets of variables such that the projected variables are maximally correlated.

CCA has been applied in many areas, such as signal processing [34], pattern classification [35] and multi-view feature learning [36-37]. Recently, CCA has been used for transfer learning. Wu et al. [38] addressed the heterogeneous transfer discriminant analysis of canonical correlations (HTDCC) method for cross-view action recognition, which learns a discriminative common feature space for linking source and target views to transfer knowledge between them. Zhang and Shi [39] presented the cross-domain CCA algorithm, which attempts to learn a semantic space of multi-view correspondences from different domains and transfer the knowledge by using dimensionality reduction in a multi-view way. Yeh et al. [40] employed CCA to derive a joint feature space for associating cross-domain data and developed a new support vector machine (SVM) algorithm that incorporates the domain adaptation ability observed in the derived subspace for cross-domain pattern classification application.

The difference between our approach and the above CCA-based methods is that we for the first time introduce CCA into the field of cross-company software defect prediction, and propose a novel metric representation for the heterogeneous source and target data, with which we propose the CCA+ approach.

3. OUR APPROACH

To answer the *RQ 1 "How to design an effective approach for HCCDP"*, we propose the CCA+ approach for heterogeneous CCDP. Our approach includes two parts: unified metric representation for heterogeneous source and target data, and CCA for transfer learning.

3.1 Unified Metric Representation for Heterogeneous Source and Target Data

To effectively utilize the heterogeneous data from two domains, Li et al. [41] introduced a common subspace for the source and target data so that the heterogeneous data can be compared. Specifically, for any source sample x^s and target sample x^t, the feature mapping functions φ_s and φ_t are defined as: $\varphi_s\left(x^s\right)=\left[Px^s;x^s;0_{d_t}\right]$ and $\varphi_t\left(x^t\right)=\left[Qx^t;0_{d_s};x^t\right]$, where P and Q are two projective matrices, d_s and d_t separately denote the

dimensionalities of source and target data. Promising results have been shown in [41].

Inspired by [41], we design a unified metric representation (UMR) for the heterogeneous source and target defect data. Assume that $X_S = \{x_S^1, x_S^2, \cdots, x_S^N\}$ and $X_T = \{x_T^1, x_T^2, \cdots, x_T^M\}$ separately denote the source and target company data, where x_S^i denotes the i^{th} module in X_S, N and M represent the numbers of modules in X_S and X_T, respectively. A module in the source company can be represented as $x_S^i = [a_S^{i1}; a_S^{i2}; \cdots; a_S^{id_s}]$ and a module in the target company can be represented as $x_T^i = [a_T^{i1}; a_T^{i2}; \cdots; a_T^{id_t}]$. Here, a_S^{ij} represents the metric value corresponding to the j^{th} metric of x_S^i, d_s and d_t are the numbers of metrics in source and target data, respectively. Usually, the metrics used in X_S and X_T are different and $d_s \neq d_t$.

Considering the large difference in values of different metrics, we firstly employ the z-score normalization [42] (without using the pooled standard deviation) to preprocess data, which is similar to the N2 normalization in [23]. Note that the normalization is applicable to either source or target company data. We then search the common metrics from the metrics used in X_S and X_T. We select row vectors that are associated with the common metrics from X_S and X_T to construct $X_S^C \in \mathbb{R}^{d_c \times N}$ and $X_T^C \in \mathbb{R}^{d_c \times M}$. It is noted that the k^{th} rows in X_S^C and X_T^C correspond to the same common metric. To make heterogeneous data from source and target companies can be compared, we define the **unified metric representation (UMR)** as follows:

$$\bar{X}_S = \begin{bmatrix} X_S^C \\ X_S^s \\ 0_{(d_t - d_c) \times N} \end{bmatrix} \text{ and } \bar{X}_T = \begin{bmatrix} X_T^C \\ 0_{(d_s - d_c) \times M} \\ X_T^s \end{bmatrix}, \quad (1)$$

where X_S^s is the data in X_S containing source-company specific metrics (that are metrics except the common metrics in X_S) and X_T^s is the data in X_T containing target-company specific metrics. After obtaining the unified metric representation, defect data from two companies can be readily compared. Figure 2 illustrates the construction of UMR for heterogeneous source and target data. It is noted that when there exist no common metrics in the data from two companies, the UMR can be defined as:

$$\bar{X}_S = \begin{bmatrix} X_S \\ 0_{d_t \times N} \end{bmatrix} \text{ and } \bar{X}_T = \begin{bmatrix} 0_{d_s \times M} \\ X_T \end{bmatrix}. \quad (2)$$

3.2 CCA for Transfer Learning

Based on the obtained UMR for the heterogeneous source and target company data, we can employ the effective transfer learning method CCA to make the distributions of source and target company data similar. CCA is presented to find a common space for data from two domains such that the correlation between the projected data in the space is maximized.

CCA seeks to obtain two projection directions w_S and w_T, one for each company data, to maximize the following linear correlation coefficient:

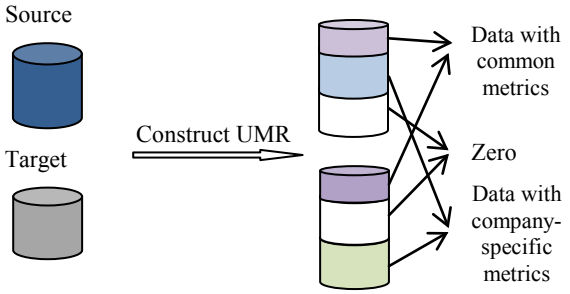

Figure 2. Illustration of UMR construction for heterogeneous source and target data.

$$\frac{cov(w_S^T \bar{X}_S, w_T^T \bar{X}_T)}{\sqrt{var(w_S^T \bar{X}_S) var(w_T^T \bar{X}_T)}} = \frac{w_S^T C_{ST} w_T}{\sqrt{(w_S^T C_{SS} w_S)(w_T^T C_{TT} w_T)}}, \quad (3)$$

where $cov(\cdot)$ denotes the covariance function, $var(\cdot)$ denotes the auto-variance function, $(\cdot)^T$ refers to the transpose of a vector or a matrix. With the projection directions w_S and w_T, we can separately project $\bar{X}_S$ and $\bar{X}_T$ into a common space, where the projected samples $w_S^T \bar{X}_S$ and $w_T^T \bar{X}_T$ are maximally correlated, that is, their distributions can be made to be similar. **This is why CCA can be used for CCDP.** C_{SS} and C_{TT} denote the within-company covariance matrices of $\bar{X}_S$ and $\bar{X}_T$, respectively. C_{ST} refers to the cross-company covariance matrix of $\bar{X}_S$ and $\bar{X}_T$. C_{SS}, C_{TT} and C_{ST} are separately defined as:

$$C_{SS} = \frac{1}{N} \sum_{i=1}^{N} (\bar{x}_S^i - m_S)(\bar{x}_S^i - m_S)^T, \quad (4)$$

$$C_{TT} = \frac{1}{M} \sum_{i=1}^{M} (\bar{x}_T^i - m_T)(\bar{x}_T^i - m_T)^T, \quad (5)$$

$$C_{ST} = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} (\bar{x}_S^i - m_S)(\bar{x}_T^j - m_T)^T, \quad (6)$$

where $\bar{x}_S^i$ denotes the i^{th} module vector with the unified metric representation in $\bar{X}_S$, m_S and m_T are the mean modules of $\bar{X}_S$ and $\bar{X}_T$:

$$m_S = \frac{1}{N} \sum_{i=1}^{N} \bar{x}_S^i \text{ and } m_T = \frac{1}{M} \sum_{i=1}^{M} \bar{x}_T^i. \quad (7)$$

Since Formula (3) is invariant with respect to scaling of w_S and w_T, the objective function of CCA can be defined as follows:

$$\begin{aligned} &\max_{w_S, w_T} w_S^T C_{ST} w_T \\ &s.t.\ w_S^T C_{SS} w_S = 1,\ w_T^T C_{TT} w_T = 1 \end{aligned}. \quad (8)$$

Formula (8) can be solved by generalized eigenvalue problem as follows:

$$\begin{bmatrix} & C_{ST} \\ C_{ST} & \end{bmatrix} \begin{bmatrix} w_S \\ w_T \end{bmatrix} = \lambda \begin{bmatrix} C_{SS} & \\ & C_{TT} \end{bmatrix} \begin{bmatrix} w_S \\ w_T \end{bmatrix}. \quad (9)$$

λ is the generalized eigenvalue corresponding to the generalized eigenvector $\begin{bmatrix} w_S \\ w_T \end{bmatrix}$. Suppose that we get p pairs of projective vectors (w_S, w_T) corresponding to the largest eigenvalues, we can

Table 3. Details of dataset used in the experiment

Company	Project	Description	Number of metrics	Number of total modules	Number of defective modules	Percentage of defective modules
NASA	CM1	Spacecraft instrument	37	327	42	12.84%
	MW1	A zero gravity experiment	37	253	27	10.67%
	PC1	Flight software	37	705	61	8.65%
SOFTLAB	AR3	Embedded controller	29	63	8	12.70%
	AR4	Embedded controller	29	107	20	18.69%
	AR5	Embedded controller	29	36	8	22.22%
ReLink	Apache HTTP Server (Apache)	Web server	26	194	98	50.52%
	OpenIntents Safe (Safe)	Security	26	56	22	39.29%
	ZXing	Bar-code reader library	26	399	118	29.57%
AEEEM	Equinox (EQ)	OSGi framework	61	324	129	39.81%
	Eclipse JDT Core (JDT)	Development	61	997	206	20.66%
	Apache Lucene (LC)	Text search engine library	61	691	64	9.26%
	Mylyn (ML)	Task management	61	1862	245	13.16%
	Eclipse PDE UI (PDE)	Development	61	1497	209	13.96%

construct the projective transformation $W_S = [w_{S1}, \cdots, w_{SP}]$ and $W_T = [w_{T1}, \cdots, w_{TP}]$.

When obtaining the projected samples $W_S^T \bar{X}_S$ and $W_T^T \bar{X}_T$, we use the nearest neighbor (NN) classifier [43] with the Euclidean distance for prediction. Specifically, for each projected target sample, we predict label (defective or defect-free) for it with the label of the projected source sample that is nearest to it according to Euclidean distance. Algorithm 1 realizes the proposed CCA+ approach.

Algorithm 1 CCA+ Approach

Require: Source company data X_S, target company data X_T and the class labels of X_S.

Output: Class labels for X_T.

1. Use the z-score normalization to preprocess X_S and X_T.
2. Search the common metrics from the heterogeneous X_S and X_T, and construct the unified metric representation as Formula (1) or (2) to obtain $\bar{X}_S$ and $\bar{X}_T$.
3. Construct the covariance matrices C_{SS}, C_{TT} and C_{ST}.
4. Obtain the projective transformations W_S and W_T by using Formula (9).
5. Based on the obtained $W_S^T \bar{X}_S$ and $W_T^T \bar{X}_T$, use the NN classifier with the Euclidean distance for defect prediction.

4. EXPERIMENTS

In this section, we evaluate the proposed CCA+ approach for heterogeneous CCDP empirically. We firstly introduce the benchmark datasets and three commonly used evaluation measures. Then, we perform experiment of HCCDP with partially different metrics, followed by the experiment of HCCDP with totally different metrics.

4.1 Data Set

In the experiment, we employ 14 publicly available and commonly used datasets (projects) from four different companies including NASA [12, 20-21], SOFTLAB [12], ReLink [22-23] and AEEEM [23-24] as the test data. Table 3 tabulates the details about the datasets we used and Figure 1 illustrates the detailed metrics used in these companies.

Each dataset in NASA represents a NASA software system or subsystem, which contains the corresponding defect-marking data and various static code metrics. The NASA data was collected from across the United States over a period of five years from numerous NASA contractors working at different geographical centers [16]. Static code metrics of NASA datasets include size, readability, complexity and etc., which are closely related to software quality.

Turkish software company (SOFTLAB) contains three datasets, i.e., AR3, AR4 and AR5, which are controller software for a washing machine, a dishwasher and a refrigerator, respectively. The used datasets from SOFTLAB and those from NASA are obtained from PROMISE repository [12]. There exist 28 common metrics between these two companies. Although the defect data of these two companies are from the same repository, these companies are very different from each other.

ReLink was collected by Wu et al. [23] and the defect information in ReLink has been manually verified and corrected. ReLink has 26 complexity metrics, which are widely used in defect prediction [23]. Among three used datasets, the size of each one (the number of modules) ranges from 56 to 399, while the number of attributes is fixed to 26.

The AEEEM data set was collected by D'Ambros et al. [24]. AEEEM consists of 61 metrics: 17 source code metrics, 5 previous-defect metrics, 5 entropy-of-change metrics, 17 entropy-of-source-code metrics, and 17 churn-of-source code metrics [24]. In particular, AEEEM includes linearly decayed entropy (LDHH) and weighted churn (WCHU). Both LDHH and WCHU have been verified as informative defect predictors [24]. Figure 1 lists part of the metrics used in AEEEM.

4.2 Evaluation Measures

In the experiment, we employ three commonly used evaluation measures to evaluate the performance of defect prediction models,

including recall rate, false positive rate and F-measure. These measures can be defined by using A, B, C and D in Table 4. Here, A, B, C and D are the number of defective modules that are predicted as defective, the number of defective modules that are predicted as defect-free, the number of defect-free modules that are predicted as defective, and the number of defect-free modules that are predicted as defect-free, respectively.

Table 4. Four kinds of defect prediction results

	Predict as defective	Predict as defect-free
Defective modules	*A*	*B*
Defect-free modules	*C*	*D*

The recall rate is defined as $A/(A+B)$. It denotes the ratio of the number of defective modules that are correctly classified as defective to the total number of defective modules. This measure is very important for SDP, because prediction models intend to find out defective modules as much as possible.

The false positive rate is defined as $C/(C+D)$. It denotes the ratio of the number of defect-free modules that are wrongly classified as defective to the total number of defect-free modules.

For SDP, the prediction precision of a model denotes the ratio of the number of defective modules that are correctly classified as defective to the number of modules that are classified as defective. The prediction precision evaluates the correct degree of prediction model and is defined as $A/(A+C)$. Obviously, a good prediction model desires to achieve high value of recall rate and precision. However, there exists trade-off between the recall rate and precision. Therefore, a comprehensive measure of recall rate and precision is necessary. F-measure is the harmonic mean of recall rate and precision, which is defined as:

$$F\text{-}measure = \frac{2 \times recall \times precision}{recall + precision}.$$

All the above evaluation measures range from 0 to 1. Obviously, an ideal defect prediction model should hold high values of recall rate and F-measure, and low value of false positive rate. In the experiment, we evaluate the performances of all defect prediction models in terms of recall (Pd), false positive (Pf) and F-measure values. It is noted that we do not specially report results with respect to the precision measure since it has been included in the comprehensive F-measure.

4.3 Heterogeneous CCDP with Partially Different Metrics

4.3.1 Compared Methods and Experimental Setting

To validate the effectiveness of the proposed CCA+ approach for heterogeneous CCDP where the metrics of source and target data are partially different, we compare CCA+ with two state-of-the-art cross-company defect prediction methods, namely NN-filter [16] and TNB [17], and a state-of-the-art cross-project defect prediction method, namely TCA+ [23]. In these compared methods, NN-filter attempts to select suitable training samples to learn predictors. And for NN-filter, each target sample selects 5 nearest neighbors to construct the training set. TCA+ and TNB employ effective machine learning methods for prediction.

We design the following two experiments to evaluate our approach:

(1) One-to-one heterogeneous CCDP. We conduct cross-company prediction using all modules in only one project as the source company data, which can also be called cross-project defect prediction. For example, AR4=>CM1, CM1=>Apache, etc. Here, the left side of "=>" denotes the source company data and the right side of "=>" represents the target company data.

(2) Many-to-one heterogeneous CCDP. We conduct cross-company prediction using all modules in multiple projects as the source company data. For example, {CM1,MW1,PC1}=>AR3, {CM1,MW1,PC1}=>Apache, etc.

For both experiments, we observe the prediction results when the number of common metrics is large and when the number of common metrics is very small. Note that all the compared methods can only use the common metrics in the source and target companies. The evaluation of heterogeneous CCDP does not involve any randomness, because all modules in a project or multiple projects from source company constitute the training set and all modules in a project from the target company constitute the test set.

4.3.2 One-to-one Heterogeneous CCDP

Table 5 shows the Pd and Pf values of one-to-one heterogeneous CCDP when 28 common metrics exist in source and target data. "M" denotes the measure. Table 6 tabulates the corresponding F-measure values. In these tables, the numbers presented with boldface denote the best results in the corresponding prediction scenes. From Tables 5 and 6, we can see that CCA+ can obtain better Pd and Pf values in most prediction scenes as compared with other competing methods, and it always obtains the best F-measure values. The reason is that our approach uses all the metrics rather than only using the common metrics, and the company-specific metrics usually contain some useful discriminant information.

Table 5. Pd and Pf values of one-to-one heterogeneous CCDP with 28 common metrics

Source=>Target	M	TCA+	NN-filter	TNB	CCA+
AR4=>CM1	Pd	0.59	0.15	0.76	**0.78**
	Pf	0.40	**0.02**	0.52	0.03
CM1=>AR4	Pd	0.60	0.58	**0.74**	0.70
	Pf	0.32	0.09	0.65	**0.01**
AR4=>MW1	Pd	0.58	0.64	0.51	**0.96**
	Pf	0.08	0.09	0.13	**0.06**
MW1=>AR4	Pd	0.32	**0.75**	0.50	0.60
	Pf	0.10	0.18	0.38	**0.02**
AR4=>PC1	Pd	0.47	0.40	0.65	**0.85**
	Pf	0.23	0.15	0.21	**0.04**
PC1=>AR4	Pd	0.30	0.60	0.50	**0.60**
	Pf	0.06	0.23	0.36	**0.00**
CM1=>AR3	Pd	**0.75**	0.37	0.50	0.50
	Pf	0.40	0.08	0.41	**0.01**
CM1=>AR5	Pd	0.37	0.25	0.50	**0.62**
	Pf	0.17	0.05	0.43	**0.03**
PC1=>AR3	Pd	0.37	0.75	0.75	**0.75**
	Pf	0.16	0.10	0.21	**0.01**
PC1=>AR5	Pd	0.37	**1.00**	0.50	0.62
	Pf	0.03	0.25	0.37	**0.00**
Average	Pd	0.47	0.55	0.59	**0.70**
	Pf	0.20	0.12	0.36	**0.02**

Table 6. F-measure values of one-to-one heterogeneous CCDP with 28 common metrics

Source=>Target	TCA+	NN-filter	TNB	CCA+
AR4=>CM1	0.27	0.23	0.28	**0.78**
CM1=>AR4	0.40	0.60	0.32	**0.80**
AR4=>MW1	0.43	0.52	0.38	**0.77**
MW1=>AR4	0.38	0.58	0.31	**0.70**
AR4=>PC1	0.23	0.27	0.33	**0.74**
PC1=>AR4	0.37	0.54	0.32	**0.75**
CM1=>AR3	0.33	0.40	0.22	**0.61**
CM1=>AR5	0.37	0.28	0.33	**0.71**
PC1=>AR3	0.30	0.61	0.46	**0.80**
PC1=>AR5	0.50	0.50	0.36	**0.76**
Average	0.36	0.45	0.33	**0.74**
Ranksum	155.0	154.0	155.0	

Table 7. Pd and Pf values of one-to-one heterogeneous CCDP with 3 common metrics

Source=>Target	M	TCA+	NN-filter	TNB	CCA+
CM1=>Apache	Pd	0.60	0.60	0.52	**0.68**
	Pf	0.35	0.28	0.41	**0.10**
Apache=>CM1	Pd	0.35	0.16	0.26	**0.55**
	Pf	0.23	0.15	**0.15**	0.27
PC1=>Safe	Pd	0.13	0.54	0.61	**0.83**
	Pf	**0.08**	0.20	0.55	0.25
Safe=>PC1	Pd	0.54	0.12	0.70	**0.72**
	Pf	0.38	0.10	0.47	**0.08**
AR4=>ZXing	Pd	0.47	0.11	0.23	**0.48**
	Pf	0.36	**0.16**	0.21	0.23
ZXing=>AR4	Pd	0.20	0.26	0.27	**0.75**
	Pf	0.13	**0.07**	0.24	0.24
AR3=>Apache	Pd	0.17	0.75	0.53	**0.92**
	Pf	**0.07**	0.62	0.47	0.50
Apache=>AR3	Pd	0.37	0.25	0.12	**0.75**
	Pf	0.14	0.17	**0.10**	0.30
MW1=>ZXing	Pd	0.38	0.36	0.23	**0.48**
	Pf	0.25	0.25	0.21	**0.21**
ZXing=>MW1	Pd	0.55	0.45	0.35	**0.61**
	Pf	0.31	0.13	0.34	**0.11**
Average	Pd	0.38	0.36	0.38	**0.67**
	Pf	0.23	**0.21**	0.31	0.22

To statistically analyze the F-measure results given in Table 6, we perform the Wilcoxon rank-sum test [44-45], which is one of non-parameter statistical significance test for comparison of two methods, at a confidence level of 95%, and the rank sum values are shown in the last low of Table 6. According to the critical value [46], the proposed approach makes a significant difference in comparison with other methods. In following experiments (Tables 8, 10 and 12), we also conduct the statistical test and the test results indicate the significant difference exists.

Tables 7 and 8 show the Pd, Pf, and F-measure values of one-to-one heterogeneous CCDP when only three common metrics exist in defect data from two companies. We can see that when very few common metrics exist in source and target data, three compared methods have unsatisfactory performances. However, CCA+ can still achieve "normal" prediction results. Generally, the average F-measure of CCA+ in Table 8 is significantly inferior to that in Table 6. **The reason is that** larger size of common metrics in fact means more useful information can be explored and the large common metric set can relieve the stress of good prediction model learning in aspect of metric difference.

4.3.3 Many-to-one Heterogeneous CCDP

In this subsection, we perform the many-to-one heterogeneous CCDP experiments. Since TCA+ [23] is designed for cross-project defect prediction, we just take the NN-filter and TNB methods as compared methods. We firstly report the prediction results when the source and target company data have 28 common metrics. The Pd, Pf and F-measure values are shown in Tables 9 and 10. In general, as compared with the results in Tables 5 and 6, all the compared methods gain some improvement with respect to three used evaluation measures, especially Pd and F-measure. In addition, our CCA+ always outperforms the other compared methods in terms of F-measure.

Table 8. F-measure values of one-to-one heterogeneous CCDP with 3 common metrics

Source=>Target	TCA+	NN-filter	TNB	CCA+
CM1=>Apache	0.61	0.64	0.54	**0.76**
Apache=>CM1	0.24	0.13	0.22	**0.32**
PC1=>Safe	0.21	0.59	0.48	**0.73**
Safe=>PC1	0.19	0.10	0.20	**0.56**
AR4=>ZXing	0.40	0.14	0.26	**0.47**
ZXing=>AR4	0.22	0.32	0.22	**0.52**
AR3=>Apache	0.27	0.63	0.53	**0.76**
Apache=>AR3	0.31	0.21	0.13	**0.38**
MW1=>ZXing	0.38	0.36	0.26	**0.48**
ZXing=>MW1	0.26	0.35	0.15	**0.47**
Average	0.31	0.34	0.30	**0.55**
Ranksum	144.5	131.5	138.5	

Table 9. Pd and Pf values of many-to-one heterogeneous CCDP with 28 common metrics

Source=>Target	M	NN-filter	TNB	CCA+
{CM1,MW1,PC1}=>AR3	Pd	**1.00**	0.87	0.75
	Pf	0.11	0.12	**0.00**
{CM1,MW1,PC1}=>AR4	Pd	0.50	**1.00**	0.80
	Pf	0.07	0.96	**0.00**
{CM1,MW1,PC1}=>AR5	Pd	0.62	0.50	**0.75**
	Pf	0.07	0.37	**0.03**
{AR3,AR4,AR5}=>CM1	Pd	**0.85**	0.78	0.83
	Pf	0.08	0.43	**0.03**
{AR3,AR4,AR5}=>MW1	Pd	0.58	0.62	**0.85**
	Pf	0.07	0.17	**0.01**
{AR3,AR4,AR5}=>PC1	Pd	0.40	0.70	**0.90**
	Pf	0.13	0.19	**0.02**
Average	Pd	0.65	0.74	**0.81**
	Pf	0.09	0.37	**0.02**

Table 10. F-measure values of many-to-one heterogeneous CCDP with 28 common metrics

Source=>Target	NN-filter	TNB	CCA+
{CM1,MW1,PC1}=>AR3	0.76	0.63	**0.85**
{CM1,MW1,PC1}=>AR4	0.57	0.32	**0.88**
{CM1,MW1,PC1}=>AR5	0.66	0.36	**0.80**
{AR3,AR4,AR5}=>CM1	0.69	0.54	**0.81**
{AR3,AR4,AR5}=>MW1	0.54	0.40	**0.86**
{AR3,AR4,AR5}=>PC1	0.29	0.37	**0.85**
Average	0.59	0.44	**0.84**
Ranksum	57.0	57.0	

We also report the prediction results (Pd, Pf and F-measure values) corresponding to many-to-one heterogeneous CCDP with three common metrics existing in source and target companies, as shown in Tables 11 and 12. We can see that our approach performs the best in terms of F-measure, and sufficient source

company data can improve the prediction performances as compared with the prediction results in Tables 7 and 8.

Table 11. Pd and Pf values of many-to-one heterogeneous CCDP with 3 common metrics

Source=>Target	M	NN-filter	TNB	CCA+
{CM1,MW1,PC1}=>Apache	Pd	0.68	0.48	**0.81**
	Pf	0.38	0.27	**0.16**
{Apache,Safe,ZXing} =>CM1	Pd	0.21	0.57	**0.92**
	Pf	0.14	0.30	**0.08**
{CM1,MW1,PC1}=>Safe	Pd	0.51	0.45	**0.81**
	Pf	**0.05**	0.21	0.23
{Apache,Safe,ZXing}=>PC1	Pd	0.32	0.52	**0.83**
	Pf	0.29	0.26	**0.06**
{AR3,AR4,AR5}=>ZXing	Pd	0.13	0.46	**0.97**
	Pf	**0.06**	0.15	0.22
{Apache,Safe,ZXing}=>AR4	Pd	0.40	0.32	**0.65**
	Pf	0.10	0.15	**0.06**
{AR3,AR4,AR5}=>Apache	Pd	0.71	0.42	**0.94**
	Pf	0.48	0.12	**0.01**
{Apache,Safe,ZXing}=>AR3	Pd	0.37	0.37	**0.87**
	Pf	**0.23**	0.38	0.36
{CM1,MW1,PC1}=>ZXing	Pd	0.40	0.48	**0.91**
	Pf	**0.13**	0.25	0.19
{Apache,Safe,ZXing} =>MW1	Pd	0.45	0.47	**0.92**
	Pf	**0.10**	0.32	0.12
Average	Pd	0.41	0.45	**0.86**
	Pf	0.19	0.24	**0.15**

Table 12. F-measure values of many-to-one heterogeneous CCDP with 3 common metrics

Source=>Target	NN-filter	TNB	CCA+
{CM1,MW1,PC1}=>Apache	0.66	0.55	**0.82**
{Apache,Safe,ZXing}=>CM1	0.18	0.31	**0.75**
{CM1,MW1,PC1}=>Safe	0.62	0.51	**0.75**
{Apache,Safe,ZXing}=>PC1	0.14	0.24	**0.66**
{AR3,AR4,AR5}=>ZXing	0.20	0.46	**0.77**
{Apache,Safe,ZXing}=>AR4	0.43	0.30	**0.66**
{AR3,AR4,AR5}=>Apache	0.65	0.54	**0.96**
{Apache,Safe,ZXing}=>AR3	0.25	0.18	**0.40**
{CM1,MW1,PC1}=>ZXing	0.46	0.46	**0.77**
{Apache,Safe,ZXing}=>MW1	0.38	0.21	**0.61**
Average	0.40	0.37	**0.72**
Ranksum	146.0	150.0	

4.4 Heterogeneous CCDP with Totally Different Metrics

4.4.1 Experimental Setting

For heterogeneous CCDP where source and target companies have totally different metrics, existing CCDP methods cannot be used for prediction. In this part, we perform within-project (Target=>Target) prediction experiments and use the within-project prediction results as references, which is similar to the performance comparison strategy in [23]. Specifically, we employ our previously proposed cost-sensitive discriminative dictionary learning (CDDL) [11] method to conduct within-project defect prediction, which is a state-of-the-art within-project prediction method. Here, 50% modules in the target project are randomly selected for training and the remained modules are used for testing. The random selection process for training and test data may be biased and may affect the prediction performance. Thus, we repeat this process 20 times and report the average prediction results.

Table 13. Pd and Pf values of within-project prediction and heterogeneous CCDP (one-to-one) with no common metrics

Source=>Target	M	CCA+	Within (Target=>Target)
JDT=>PC1	Pd	0.75	**0.86**
	Pf	**0.11**	0.29
PC1=>JDT	Pd	0.45	**0.69**
	Pf	**0.06**	0.17
JDT=>AR4	Pd	**0.70**	0.66
	Pf	**0.22**	0.23
AR4=>JDT	Pd	0.66	**0.69**
	Pf	**0.12**	0.17
JDT=>ZXing	Pd	0.61	**0.68**
	Pf	**0.41**	0.43
ZXing=>JDT	Pd	**0.80**	0.69
	Pf	**0.15**	0.17
ML=>PC1	Pd	0.73	**0.86**
	Pf	**0.08**	0.29
PC1=>ML	Pd	0.33	**0.58**
	Pf	**0.08**	0.20
ML=>AR4	Pd	0.53	**0.66**
	Pf	**0.06**	0.23
AR4=>ML	Pd	0.49	**0.58**
	Pf	**0.18**	0.20
ML=>ZXing	Pd	0.50	**0.68**
	Pf	**0.21**	0.43
ZXing=>ML	Pd	**0.60**	0.58
	Pf	**0.19**	0.20
PDE=>PC1	Pd	0.31	**0.86**
	Pf	**0.00**	0.29
PC1=>PDE	Pd	0.40	**0.77**
	Pf	**0.09**	0.33
PDE=>AR4	Pd	0.45	**0.66**
	Pf	**0.03**	0.23
AR4=>PDE	Pd	0.51	**0.77**
	Pf	**0.14**	0.33
PDE=>ZXing	Pd	0.67	**0.68**
	Pf	0.44	**0.43**
ZXing=>PDE	Pd	**0.85**	0.77
	Pf	**0.27**	0.33
EQ=>CM1	Pd	0.44	**0.74**
	Pf	**0.30**	0.37
CM1=>EQ	Pd	0.63	**0.75**
	Pf	**0.29**	0.37
LC=>Apache	Pd	0.17	**0.67**
	Pf	**0.00**	0.31
Apache=>LC	Pd	**0.95**	0.71
	Pf	0.44	**0.19**
Average	Pd	0.57	**0.71**
	Pf	**0.18**	0.28

It is noted that the proposed CCA+ approach is only evaluated once for each prediction scene, because the evaluation of heterogeneous CCDP with CCA+ does not involve any randomness. When we conduct HCCDP with CCA+, all modules in a project or multiple projects from source company constitute the training set and all modules in a project from the target company constitute the test set.

We also design two experiments including one-to-one heterogeneous CCDP and many-to-one heterogeneous CCDP to evaluate CCA+. Since the metrics used in AEEEM are totally different from those of other companies, we use the projects in AEEEM as the source or target data in all prediction cases.

Table 14. F-measure values of within-project prediction and heterogeneous CCDP (one-to-one) with no common metrics

Source=>Target	CCA+	Within (Target=>Target)
JDT=>PC1	**0.51**	0.41
PC1=>JDT	0.54	**0.59**
JDT=>AR4	**0.51**	0.49
AR4=>JDT	**0.62**	0.59
JDT=>ZXing	0.47	**0.50**
ZXing=>JDT	**0.67**	0.59
ML=>PC1	**0.55**	0.41
PC1=>ML	0.36	**0.40**
ML=>AR4	**0.59**	0.49
AR4=>ML	0.37	**0.40**
ML=>ZXing	0.49	**0.50**
ZXing=>ML	**0.42**	0.40
PDE=>PC1	**0.45**	0.41
PC1=>PDE	**0.40**	0.40
PDE=>AR4	**0.56**	0.49
AR4=>PDE	**0.42**	0.40
PDE=>ZXing	0.49	**0.50**
ZXing=>PDE	**0.47**	0.40
EQ=>CM1	0.25	**0.38**
CM1=>EQ	0.60	**0.65**
LC=>Apache	0.29	**0.68**
Apache=>LC	0.30	**0.44**
Average	0.47	**0.48**

Table 15. Pd and Pf values of within-project prediction and heterogeneous CCDP (many-to-one) with no common metrics

Source=>Target	M	CCA+	Within (Target=>Target)
{JDT,ML,PDE} =>PC1	Pd	0.66	**0.86**
	Pf	**0.05**	0.29
{CM1,MW1,PC 1}=>JDT	Pd	0.55	**0.69**
	Pf	**0.05**	0.17
{JDT,ML,PDE} =>AR4	Pd	0.52	**0.66**
	Pf	**0.16**	0.23
{AR3,AR4,AR5} =>JDT	Pd	0.57	**0.69**
	Pf	**0.04**	0.17
{JDT,ML,PDE} =>ZXing	Pd	0.63	**0.68**
	Pf	**0.38**	0.43
{Apache,Safe,Z Xing}=>JDT	Pd	**0.75**	0.69
	Pf	**0.10**	0.17
{CM1,MW1,PC 1}=>ML	Pd	0.55	**0.58**
	Pf	**0.19**	0.20
{AR3,AR4,AR5} =>ML	Pd	**0.60**	0.58
	Pf	0.21	**0.20**
{Apache,Safe,Z Xing}=>ML	Pd	**0.73**	0.58
	Pf	0.24	**0.20**
{CM1,MW1,PC 1}=>PDE	Pd	0.69	**0.77**
	Pf	**0.24**	0.33
{AR3,AR4,AR5} =>PDE	Pd	**0.81**	0.77
	Pf	**0.25**	0.33
{Apache,Safe,Z Xing}=>PDE	Pd	0.36	**0.77**
	Pf	**0.02**	0.33
{PDE,ML,EQ} =>CM1	Pd	0.49	**0.74**
	Pf	**0.00**	0.37
{CM1,MW1,PC 1}=>EQ	Pd	0.70	**0.75**
	Pf	**0.28**	0.37
{PDE,ML,LC} =>Apache	Pd	0.66	**0.67**
	Pf	**0.25**	0.31
{Apache,Safe,Z Xing}=>LC	Pd	**0.72**	0.71
	Pf	**0.18**	0.19
Average	Pd	0.62	**0.70**
	Pf	**0.17**	0.27

4.4.2 One-to-one Heterogeneous CCDP

Table 13 gives the Pd and Pf values of our approach for one-to-one heterogeneous CCDP where no common metric exists in the source and target data. For comparison, the Pd and Pf values of within-project (Target=>Target) defect prediction are also reported in this Table. Table 14 shows the corresponding F-measure values. We can see that CCA+ can obtain comparable results in contrast with the within-project prediction results.

Table 16. F-measure values of within-project prediction and heterogeneous CCDP (many-to-one) with no common metrics

Source=>Target	CCA+	Within (Target=>Target)
{JDT,ML,PDE}=>PC1	**0.59**	0.41
{CM1,MW1,PC1}=>JDT	**0.63**	0.59
{JDT,ML,PDE}=>AR4	**0.60**	0.49
{AR3,AR4,AR5}=>JDT	**0.66**	0.59
{JDT,ML,PDE}=>ZXing	0.49	**0.50**
{Apache,Safe,ZXing}=>JDT	**0.70**	0.59
{CM1,MW1,PC1}=>ML	0.39	**0.40**
{AR3,AR4,AR5}=>ML	**0.40**	0.40
{Apache,Safe,ZXing}=>ML	**0.44**	0.40
{CM1,MW1,PC1}=>PDE	**0.42**	0.40
{AR3,AR4,AR5}=>PDE	**0.47**	0.40
{Apache,Safe,ZXing}=>PDE	**0.49**	0.40
{PDE,ML,EQ}=>CM1	0.34	**0.38**
{CM1,MW1,PC1}=>EQ	**0.66**	0.65
{PDE,ML,LC}=>Apache	**0.69**	0.68
{Apache,Safe,ZXing}=>LC	0.41	**0.44**
Average	**0.52**	0.48

4.4.3 Many-to-one Heterogeneous CCDP

Tables 15 and 16 report the performances (Pd, Pf and F-measure values) of our approach for many-to-one heterogeneous CCDP when the source and target data has totally different metrics. Within-project (Target=>Target) prediction results are also shown in these two tables. It is obvious that the many-to-one prediction results of our approach are better than the one-to-one prediction results shown in Tables 13 and 14. In many-to-one heterogeneous CCDP, our approach can achieve better or comparable prediction results as compared with the within-project prediction results, which indicates the effectiveness of the proposed approach.

4.5 Answers to Research Questions

1. *RQ 2: Is the heterogeneous cross-company defect data helpful for cross-company defect prediction?*

   From the tables above, we can conclude that the heterogeneous cross-company defect data is helpful for defect prediction. For HCCDP with partially different metrics, existing CCDP methods can only make use of the common metrics and show unsatisfactory performances. Our approach makes full use of all the metrics of source and target companies, and show desirable prediction results. For HCCDP with totally different metrics, our approach can obtain comparable or even better prediction results as compared with within-project prediction results.

2. *RQ 3: If the heterogeneous cross-company defect data can help prediction, when it helps the most?*

   According to the prediction results in Tables 5-16, we can find that, in general, the results of HCCDP with partially different metrics are better than those of HCCDP with totally different metrics. In addition, the results of many-to-one HCCDP are better than those of one-to-one HCCDP. Specifically, the prediction performances of many-to-one HCCDP with 28 common metrics are the best. Therefore, we conclude that when there exist multiple projects in the source company and a large

number of common metrics between source and target company data, the heterogeneous cross-company defect data can help prediction most.

4.6 Further Experiment

4.6.1 Performance with Other Classifiers

In this subsection, we report the prediction performances of our approach with other classifiers including support vector machine (SVM), random forest (RF) and logistic regression (LR). Table 17 tabulates the results of CCA+ with SVM, RF, LR and NN classifiers in some representative heterogeneous CCDP cases. We can see that CCA+ obtains favorable prediction results with various classifiers and achieves the best prediction results with the NN classifier.

Table 17. F-measure values of CCA+ in some specific heterogeneous CCDP cases

Source=>Target	SVM	RF	LR	NN
AR4=>CM1	0.77	0.76	0.74	**0.78**
CM1=>Apache	0.73	0.70	**0.76**	**0.76**
{CM1,MW1,PC1}=>AR3	0.83	0.78	0.79	**0.85**
{CM1,MW1,PC1}=>Apache	0.80	0.75	0.77	**0.82**
JDT=>PC1	**0.51**	0.47	0.49	**0.51**
{JDT,ML,PDE}=>PC1	0.56	0.51	0.57	**0.59**
average	0.70	0.66	0.69	**0.72**

Table 18. MCC values of CCA+ in some specific heterogeneous CCDP cases

Source=>Target	TCA+	NN-filter	TNB	CCA+	Within (Target=>Target)
AR4=>CM1	0.19	0.23	0.25	**0.76**	—
CM1=>Apache	0.25	0.32	0.11	**0.59**	—
{CM1,MW1,PC1}=>AR3	—	**0.90**	0.75	0.77	—
{CM1,MW1,PC1}=>Apache	—	0.30	0.22	**0.65**	—
JDT=>PC1	—	—	—	**0.48**	0.43
{JDT,ML,PDE}=>PC1	—	—	—	**0.64**	0.58

4.6.2 Performance with the MCC Measure

Considering that the source data may be class-imbalanced, especially for one-to-one CCDP, for this part, we also measure the prediction performance of CCA+ using the comprehensive matthews correlation coefficient (MCC) measure [47]. MCC includes all A, B, C and D in the confusion matrix of Table 4, and can be regarded as taking the class-imbalance issue into consideration. MCC is defined as follows:

$$MCC = \frac{A*D - B*C}{\sqrt{(A+B)*(A+C)*(B+D)*(C+D)}}.$$

The MCC measure ranges from -1 to 1, and an ideal defect prediction model should hold high values of MCC. Table 18 reports the results of CCA+ with MCC as the measure in some representative heterogeneous CCDP cases. The heterogeneous CCDP cases in the table correspond to those in Tables 5-16, and we fill the table with "-" where the compared methods are not performed in some specific cases. According to Table 18, our CCA+ approach can obtain better results in terms of MCC as compared with other related methods in most cases.

4.7 Threats to Validity

Followings are several potential threats to the validity with respect to the experiments:

(1) Bias of evaluation measures. One bias is the measures we used to report the performance of defect prediction. Other measures, such as area under curve (AUC) and g-measure (harmonic mean of pd and 1-pf) are not used. They are also comprehensive measures. In this work, we employ the widely used pd, pf, F-measure and MCC indices to show the empirical evaluation of defect prediction.

(2) Comparison accuracy. The authors of the three compared methods do not provide the program codes. We carefully implement these methods by following their papers.

5. CONCLUSIONS

In this paper, we propose an effective solution for heterogeneous cross-company defect prediction (HCCDP) problem, which refers to the cross-company prediction scenario where source and target company data has different metrics. We present a novel metric representation. By effectively combining the common metrics, company-specific metrics and an appropriate number of zeros, we can obtain a unified metric representation for data from two different companies. Then, we for the first time introduce the transfer learning method CCA into CCDP, such that the data distributions of source and target data with the unified metric representation can be made similar. We call the proposed approach for heterogeneous CCDP as CCA+.

We conduct HCCDP experiments on the 14 widely-used open source projects from four companies. And we separately design the one-to-one and many-to-one experiments to evaluate the performances of the proposed approach. The experimental results of one-to-one prediction indicate that our approach is superior to state-of-the-art CCDP methods in terms of three widely-used measures. In many-to-one HCCDP experiments, our approach also shows desirable prediction effects that are comparable to within-project prediction. All in all, the proposed approach is an effective solution for heterogeneous cross-company defect prediction.

For the future work, we would like to employ more company data that contains both open source and commercial proprietary closed projects to validate the generalization ability of our approach. We will also evaluate the application of our solution for cross-company data containing some other static code metrics.

6. ACKNOWLEDGEMENTS

The authors want to thank the anonymous reviewers for their constructive comments and suggestions. The work described in this paper was supported by the National Nature Science Foundation of China under Projects No. 61272273, No. 61233011, No. 61472186, No. 91418202, No. 61472178 and No. 61170071, the China 973 Program under Project No. 2014CB340702, the China 863 Program under Project No. 2015AA016306.

7. REFERENCES

[1] S. Kim, H. Zhang, R. Wu, and L. Gong. Dealing with noise in defect prediction. In *Proceedings of the 33rd International Conference on Software Engineering (ICSE)*, pages 481-490, 2011.

[2] T. Hall, S. Beecham, D. Bowes, D. Gray, and S. Counsell. A systematic literature review on fault prediction performance in software engineering. *IEEE Transactions on Software Engineering*, 38(6):1276-1304, 2012.

[3] S. Shivaji, E. J. Whitehead, R. Akella, and S. Kim. Reducing features to improve code change-based bug prediction. *IEEE Transactions on Software Engineering*, 39(4):552-569, 2013.

[4] T. Menzies, A. Butcher, D. Cok, A. Marcus, L. Layman, F. Shull, B. Turhan, T. Zimmermann. Local versus global lessons for defect prediction and effort estimation. *IEEE Transactions on Software Engineering*, 39(6):822-834, 2013.

[5] M. Shepperd, D. Bowes, and T. Hall. Researcher bias: The use of machine learning in software defect prediction. *IEEE Transactions on Software Engineering*, 40(6):603-616, 2014.

[6] K. Elish and M. Elish. Predicting defect-prone software modules using support vector machines. *Journal of Systems and Software*, 81(5):649-660, 2008.

[7] J. Zheng. Cost-sensitive boosting neural networks for software defect prediction. *Expert Systems with Applications*, 37(6):4537-4543, 2010.

[8] Z. B. Sun, Q. B. Song, and X. Y. Zhu. Using coding based ensemble learning to improve software defect prediction. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 42(6):1806-1817, 2012.

[9] S. Wang and X. Yao. Using class imbalance learning for software defect prediction. *IEEE Transactions on Reliability*, 62(2):434-443, 2013.

[10] M. Liu, L. Miao, and D. Zhang. Two-stage cost-sensitive learning for software defect prediction. *IEEE Transactions on Reliability*, 63(2):676-686, 2014.

[11] X. Y. Jing, S. Ying, Z. W. Zhang, S. S. Wu, and J. Liu. Dictionary learning based software defect prediction. In *Proceedings of the 36th International Conference on Software Engineering (ICSE)*, pages 414-423, 2014.

[12] G. Boetticher, T. Menzies, and T. Ostrand. The PROMISE repository of empirical software engineering data. http://promisedata.org/repository, 2007.

[13] F. Peters, T. Menzies, and A. Marcus. Better cross company defect prediction. *In 10th IEEE Working Conference on Mining Software Repositories (MSR)*, pages 409-418, 2013.

[14] P. Singh, S. Verma, and O. P. Vyas. Cross company and within company fault prediction using object oriented metrics. *International Journal of Computer Applications*, 74(8):5-11, 2013.

[15] F. Peters, T. Menzies, and L. Layman. LACE2: better privacy-preserving data sharing for cross project defect prediction. In *Proceedings of the 37th International Conference on Software Engineering (ICSE)*, article in press.

[16] B. Turhan, T. Menzies, A. B. Bener, and J. Di Stefano. On the relative value of cross-company and within-company data for defect prediction. *Empirical Software Engineering*, 14(5):540-578, 2009.

[17] Y. Ma, G. Luo, X. Zeng, and A. Chen. Transfer learning for cross-company software defect prediction. *Information and Software Technology*, 54(3):248-256, 2012.

[18] L. Chen, B. Fang, Z. Shang, and Y. Tang. Negative samples reduction in cross-company software defects prediction. *Information and Software Technology*, Article in Press, 2015.

[19] F. Peters, T. Menzies, L. Gong, and H. Zhang. Balancing privacy and utility in cross-company defect prediction. *IEEE Transactions on Software Engineering*, 39(8):1054-1068, 2013.

[20] T. Menzies, J. Greenwald, and A. Frank. Data mining static code attributes to learn defect predictors. *IEEE Transactions on Software Engineering*, 33(1):2-13, 2007.

[21] M. Shepperd, Q. Song, Z. Sun, and C. Mair. Data quality: some comments on the NASA software defect datasets. *IEEE Transactions on Software Engineering*, 39(9):1208-1215, 2013.

[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 (ESEC/FSE)*, pages 15-25, 2011.

[23] J. Nam, S. J. Pan, and S. Kim. Transfer defect learning. In *Proceedings of the 35th International Conference on Software Engineering (ICSE)*, pages 382-391, 2013.

[24] M. D'Ambros, M. Lanza, and R. Robbes. An extensive comparison of bug prediction approaches. In *7th IEEE Working Conference on Mining Software Repositories (MSR)*, pages 31-41, 2010.

[25] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12):2639-2664, 2004.

[26] A. Panichella, R. Oliveto, and A. De Lucia. Cross-project defect prediction models: L'Union fait la force. In *Software Evolution Week-IEEE Conference on Software Maintenance, Reengineering and Reverse Engineering (CSMR-WCRE)*, pages 164-173, 2014.

[27] G. Canfora, A. De Lucia, M. Di Penta, R. Oliveto, A. Panichella, and S. Panichella. Multi-objective cross-project defect prediction. In *IEEE 6th International Conference on Software Testing, Verification and Validation (ICST)*, pages 252-261, 2013.

[28] S. Herbold. Training data selection for cross-project defect prediction. In *Proceedings of the 9th International Conference on Predictive Models in Software Engineering (PROMISE)*, pages 6-16, 2013.

[29] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and B. Murphy. Cross-project defect prediction. In *Proceedings of the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE)*, pages 91-100, 2009.

[30] Z. He, F. Shu, Y. Yang, M. Li, and Q. Wang. An investigation on the feasibility of cross-project defect prediction. *Automated Software Engineering*, 19(2):167-199, 2012.

[31] F. Rahman, D. Posnett, and P. Devanbu. Recalling the imprecision of cross-project defect prediction. *In Proceedings of the ACM SIGSOFT 20th International*

Symposium on the Foundations of Software Engineering (FSE), pages 1-11, 2012.

[32] B. Turhan, A. T. Mısırlı, and A. Bener. Empirical evaluation of the effects of mixed project data on learning defect predictors. *Information and Software Technology*, 55(6):1101-1118, 2013.

[33] D. Ryu, O. Choi, and J. Baik. Value-cognitive boosting with a support vector machine for cross-project defect prediction. *Empirical Software Engineering*, Article in Press, 2014.

[34] Y. O. Li, T. Adali, W. Wang, and V. D. Calhoun. Joint blind source separation by multiset canonical correlation analysis. *IEEE Transactions on Signal Processing*, 57(10):3918-3929, 2009.

[35] T. K. Kim, J. Kittler, and R. Cipolla. Discriminative learning and recognition of image set classes using canonical correlations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1005-1018, 2007.

[36] P. Dhillon, D. P. Foster, and L. H. Ungar. Multi-view learning of word embeddings via CCA. In *Advances in Neural Information Processing Systems (NIPS)*, pages 199-207, 2011.

[37] X. Y. Jing, R. M. Hu, Y. P. Zhu, S. S. Wu, C. Liang, and J. Y. Yang. Intra-view and inter-view supervised correlation analysis for multi-view feature learning. In *28th AAAI Conference on Artificial Intelligence (AAAI)*, pages 1882-1889, 2014.

[38] X. Wu, H. Wang, C. Liu, and Y. Jia. Cross-view action recognition over heterogeneous feature spaces. In *International Conference on Computer Vision (ICCV)*, pages 609-616, 2013.

[39] B. Zhang and Z. Z. Shi. Classification of big velocity data via cross-domain canonical correlation analysis. In *International Conference on Big Data*, pages 493-498, 2013.

[40] Y. Yeh, C. Huang, and Y. Wang. Heterogeneous domain adaptation and classification by exploiting the correlation subspace. *IEEE Transactions on Image Processing*, 23(5):2009-2018, 2014.

[41] W. Li, L. Duan, D. Xu, and I. W. Tsang. Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(6):1134-1148, 2014.

[42] S. B. Kotsiantis, D. Kanellopoulos, and P. E. Pintelas. Data preprocessing for supervised leaning. *International Journal of Computer Science*, 1(2):111-117, 2006.

[43] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21-27, 1967.

[44] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 80-83, 1945.

[45] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1-30, 2006.

[46] R. C. Milton. An extended table of critical values for the Mann-Whitney (Wilcoxon) two-sample statistic. *Journal of the American Statistical Association*, 59(307):925-934, 1964.

[47] P. Baldi, S. Brunak, Y. Chauvin, C. A. Andersen, and H. Nielsen. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16(5):412-424, 2000.