

# APPENDICES for “DSSDPP: Data Selection and Sampling based Domain Programming Predictor for Cross-project Defect Prediction”

Zhiqiang Li, Hongyu Zhang, Xiao-Yuan Jing, Juanying Xie, Min Guo, Jie Ren

## APPENDIX A

### EFFECT OF MMD WITH DIFFERENT PERCENTAGES OF TARGET DATA FOR THE SELECTION OF SOURCE

An important question is how much target project data is required to select source for MMD-based source data selection algorithm [1]. To evaluate the effect of MMD with different percentages of target data for the selection of source data, we compare the prediction performance of the selected sources by using MMD with different percentages of target data. Specifically, the used percentage of target data ranges from 10 to 90 percent with step length 10 percent. For this propose, we use DPP to conduct cross-project experiments and the Scott-Knott effect size difference (ESD) test [2], [3] to statistically compare the results.

Fig. 1 shows the Scott-Knott ESD test of MMD with different percentages of target data for selecting the source across the 22 projects. As shown in the figure, we make the following observations:

① There are two different groups in terms of MCC according to the test results. The MMD with 10%, 30% and 40% of target data are in the first group, while MMD with the other percentages of target data are in the second group.

② For AUC, MMD with 10%, 20%, 60%, 70%, and 80% of target data rank the first and MMD with the other percentages of target data are in the last group.

Based on the above observations, MMD can use a small amount of target data (e.g., 10% data) to select source data, since it is able to produce better prediction results among MMD with other percentages of target data in our experimental settings. It does not need all the target data to

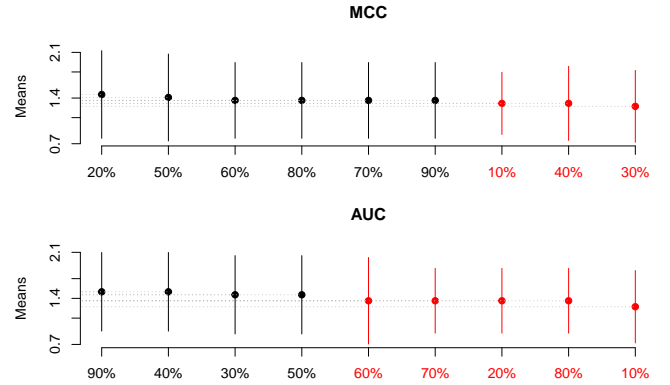


Fig. 1. Comparison of MMD with different percentages of target data for selecting the source across 22 projects in terms of *MCC* and *AUC*. Different colors represents different groups with statistical significance (black < red).

be used for selecting the source. This demonstrates that it is feasible for MMD to select source data for projects with limited historical defect data. Hence, we use MMD with 10% of target data to select source data in the following experiments.

## APPENDIX B

### DETAILED SOURCE DATA SELECTION RESULTS

In this section, we report the detailed selection results of project-level based source data selection algorithms including Bellwether [4], DCNNS [5], VC [6], CORR [7], [8], and MMD. The latter four algorithms can select multiple sources for CPDP. We call DCNNS, VC, CORR, and MMD that with the selection of multiple source projects as mDCNNS, mVC, mCORR, and mMMD, respectively.

#### B.1 Selection Results of Single-source Scenario

Table 1 shows the selected single source project of each data selection algorithm from the studied 22 projects. As shown in the table, we can observe that different source data selection algorithms have different selection results

- Zhiqiang Li, Juanying Xie, Min Guo and Jie Ren are with the School of Computer Science, Shaanxi Normal University, Xi'an 710119, China, and also with the Engineering Laboratory of Teaching Information Technology, Shaanxi Province, China. E-mail: lizq@snnu.edu.cn, xiejuany@snnu.edu.cn, guomin@snnu.edu.cn, and renjie@snnu.edu.cn. Corresponding author: Zhiqiang Li.
- Hongyu Zhang is with the School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW 2308, Australia. E-mail: hongyu.zhang@newcastle.edu.au.
- Xiao-Yuan Jing is with the School of Computer Science, Wuhan University, Wuhan 430072, China. E-mail: jingxy\_2000@126.com.

TABLE 1  
The selected single source of each data selection algorithm from the given 22 projects.

Dataset	Target	Bellwether	DCNNS	VC	CORR	MMD
NASA	CM1	PC1	MC2	KC3	PC3	KC3
	JM1	PC1	PC1	PC3	KC3	PC1
	KC3	PC1	MC2	MW1	PC1	MW1
	MC1	PC1	CM1	KC3	MW1	PC1
	MC2	PC1	MW1	PC4	KC3	MW1
	MW1	PC1	MC2	KC3	MC1	KC3
	PC1	PC3	MC2	MC2	KC3	CM1
	PC2	PC1	MC2	KC3	PC3	CM1
	PC3	PC1	KC3	MC2	PC1	PC4
	PC4	PC1	KC3	MC2	PC1	MC2
AEEEM	EQ	LC	LC	ML	ML	PDE
	JDT	LC	EQ	PDE	PDE	EQ
	LC	EQ	EQ	ML	PDE	PDE
	ML	LC	EQ	EQ	LC	JDT
	PDE	LC	EQ	EQ	LC	EQ
ReLink	Apache	ZXing	Safe	Safe	Safe	Safe
	Safe	ZXing	Apache	Apache	ZXing	Apache
	ZXing	Apache	Safe	Safe	Safe	Safe
Eclipse	R-2.0	R-3.0	R-2.1	R-2.1	R-2.1	R-2.1
	R-2.1	R-2.0	R-2.0	R-3.0	R-2.0	R-2.0
	R-3.0	R-2.0	R-2.0	R-2.1	R-2.0	R-2.1

for a given target project. Taking the PC1 target in NASA dataset as an example, the Bellwether algorithm selects PC3 as the source, both DCNNS and VC select the MC2 project as the source, CORR selects KC3 while MMD selects the CM1 project. This is because their algorithm ideas are different to make them get different selection results.

## B.2 Selection Results of Multi-source Scenario

Table 2 shows the selected multiple sources of each data selection algorithms from the used 22 projects. From this table, we can see that each data selection algorithm can find one or multiple similar sources for a given target project. It demonstrates that the distribution of distance values based automatic selection scheme is effective. Taking the mMMD algorithm as an example, according to the given thresholds, there are 13 out of 22 targets with the selection of two source projects at least. The other algorithms can also select similar sources for the given target project. Hence, it is feasible to find appropriate thresholds by observing the distribution of generated distance values of each source selection algorithm.

## APPENDIX C COMPARISON WITH DIFFERENT CLASSIFIERS

Domain programming predictor (DPP) is the main part of DSSDPP, which can be viewed as a transfer classifier [9]. DPP is non-parametric and discriminative. To evaluate the prediction performance of DPP on CPDP, we compare the results of DPP with five well-established classifiers [3], [10], [11] including logistic regression (LR), k-nearest neighbor (KNN), support vector machine (SVM), naive Bayes (NB), and random forest (RF). These classifiers for CPDP are referred to as CPDP\_LR, CPDP\_KNN, CPDP\_SVM, CPDP\_NB, and CPDP\_RF, respectively.

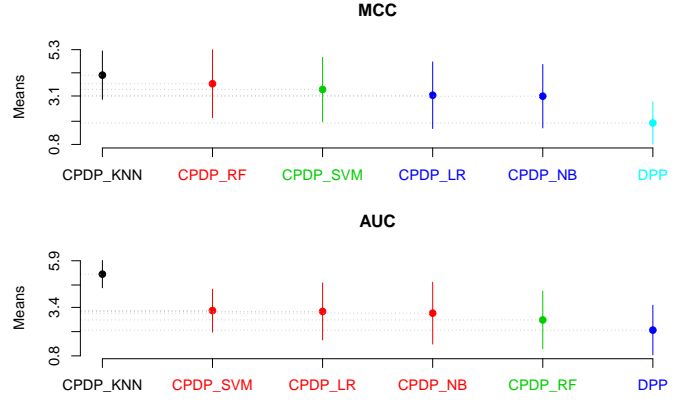


Fig. 2. Comparison of studied classifiers against each other with Scott-Knott ESD test in terms of *MCC* and *AUC*. Different colors denote different groups with statistical significance (black < red < green < blue < cyan).

Recent work [3], [12] points out that classifiers with default parameter settings have a great effect on the prediction performance, which usually leads to suboptimal results. Thus, the parameter settings of the classifiers should be carefully optimized. In our experimental settings, the source data is used to train the model and the target data is used for prediction. In order to conduct parameter optimization for the used classifiers, we assume that there is a small amount of data in the target project. Specifically, we use the source data as the training set and randomly select 10% of target data as the validation set to tune the classifier's parameters. We repeat the above process 30 times. According to the overall best performance measure value (e.g., *MCC*) across all 22 projects from 30 random trials, we can get the optimal parameter values.

For these five used classifiers, only LR does not require parameter settings according to the study [3]. Hence, we refer to the candidate parameter settings of the above work as the examined parameters of the other four classifiers, which are shown in Table 3. Since we use multiple different classifiers, we carry out the Scott-Knott ESD test to rank and compare them.

Fig. 2 shows the results of studied classifiers with Scott-Knott ESD test in terms of *MCC* and *AUC* across 22 target projects with 30 repetition times. In this test, different colors denote different groups with statistical significance. The lower the average rank is, the better the classifier is. As shown in the figure, we have the following observations:

① In terms of *MCC*, there are five groups {CPDP\_KNN}, {CPDP\_RF}, {CPDP\_SVM}, {CPDP\_LR, CPDP\_NB}, and {DPP}. The transfer DPP predictor leads to the best performance among all classifiers with statistical significance since it ranks the first.

② Similarly, DPP performs better *AUC* values than the other used classifiers with statistical significance since it lies in the top rank group.

The above observations indicate the superiority of DPP in comparison with the used classifiers for CPDP. The reason of DPP achieving better results is that it learns discriminative transferable knowledge from the labeled source data to the unlabeled target data by exploring their intra structures. While the other studied classifiers only use the source data

TABLE 2  
The selected multiple sources of each data selection algorithm from the given 22 projects.

Dataset	Target	mDCNNS	mVC	mCORR	mMMD
NASA	CM1	MC2 MW1 KC3	KC3 MW1 MC2 PC2 PC3	PC3 PC1 KC3 MC1	KC3 MC2 MW1
	JM1	PC1 PC2 CM1 PC3 PC4 MC1	PC3	KC3 PC1 PC4 MC1 PC3 MW1	PC1 MC1 PC3 PC2 PC4 CM1
	KC3	MC2 MW1 CM1	MW1 MC2 MC1 PC3 CM1 PC1 PC2	PC1	MW1 MC2
	MC1	CM1 KC3 MW1 MC2 PC1	KC3 PC3 MW1 MC2	MW1 PC3 JM1 PC4 PC1 PC2 KC3	PC1 CM1 PC4 MC2 KC3 PC3
	MC2	MW1 KC3 CM1	PC4 KC3 MW1 PC5 PC1	KC3	MW1
	MW1	MC2 KC3 CM1	KC3 PC3 MC2 PC4 PC2 PC5	MC1 PC3	KC3
	PC1	MC2 MW1 KC3 CM1	MC2 MW1	KC3 PC3 PC2	CM1 KC3 MC2 PC4 MW1
	PC2	MC2 MW1 KC3 CM1 PC1	KC3 MC2 MW1	PC3 PC1 KC3	CM1 PC4 PC1 MC2 KC3
	PC3	KC3 MW1 MC2 CM1 PC1	MC2 KC3	PC1 KC3 JM1 MW1 PC2 PC4	PC4 PC1 CM1
	PC4	KC3 MW1 MC2 CM1 PC1	MC2 KC3 MW1 CM1	PC1 KC3 PC3 MC1 CM1 MW1	MC2 KC3 CM1 MW1 PC1
	PC5	CM1 KC3 MW1 MC2 PC1 PC2	MC2	CM1 PC4 MC1 KC3 PC1	PC3 PC2 JM1 PC4 PC1 MC1
AEEEM	EQ	LC	ML	ML	PDE
	JDT	EQ LC	PDE EQ ML	PDE ML	EQ
	LC	EQ JDT	ML	PDE	PDE
	ML	EQ LC JDT	EQ JDT	LC EQ	JDT LC PDE EQ
	PDE	EQ LC JDT ML	EQ ML JDT	LC EQ	EQ JDT ML LC
ReLink	Apache	Safe	Safe	Safe Zxing	Safe
	Safe	Apache	Apache Zxing	Apache	Apache
	ZXing	Safe Apache	Safe	Safe	Safe Apache
Eclipse	R-2.0	R-2.1	R-2.1	R-2.1	R-2.1
	R-2.1	R-2.0	R-3.0	R-2.0	R-2.0
	R-3.0	R-2.0 R-2.1	R-2.1	R-2.0	R-2.1 R-2.0

TABLE 3  
Parameters of the used classifiers.

Classifier	Parameter name	Parameter description	Candidate parameter values
KNN	n_neighbors	the number of neighbors	{1, 5, 9, 13, 17}
SVM	$\sigma$	the width of Gaussian kernels	{0.1, 0.3, 0.5, 0.7, 0.9}
NB	distribution type	the type of prior distribution	{‘normal’, ‘kernel’}
RF	n_trees	the number of trees	{10, 20, 30, 40, 50}

to build models and predict the target data, these models do not reduce the distribution difference between the source and target data. As a conclusion, DPP leads to the best prediction performance against the studied five classifiers with statistical significance on two evaluation measures in our experimental settings. This demonstrates the effectiveness of DPP for CPDP.

## APPENDIX D DETAILED COMPARISON RESULTS OF SOURCE SELECTION ALGORITHMS

To empirically compare the prediction performance of the project-level based source data selection algorithms, we use DPP to conduct cross-project prediction experiments since it leads to better prediction performance than the other well-established classifiers (see Appendix C). ALL [13] and UM [14], [15] are two methods that use all available projects in the same dataset as the source. Tables 4 and 5 show the median *MCC* and *AUC* values of each source data selection

algorithm on 22 target projects with 30 repetition times, respectively. The overall median results (i.e., median of those  $22 \times 30 = 660$  CPDP values) are also reported in the second to the last row of the table (denoted as “Median”). The average rank of each algorithm with Scott-Knott ESD test is listed as the last row of the table (denoted as “AR”, the numbers in brackets indicate the ranking of algorithms based on this test). The lower the AR value, the better the algorithm.

From these tables, we have the following observations:

① In terms of the overall results across 22 target projects from “Median”, CORR, Bellwether, and mVC achieve better *MCC* results, UM and Bellwether achieve better *AUC* results among all the algorithms.

② In terms of the AR values, CORR, Bellwether, and mCORR rank in the first place among all the algorithms, which demonstrates that they achieve better performance with statistic significance according to *MCC*. Similarly, mMMD, CORR, mCORR, and UM achieve better prediction performance with statistic significance according to *AUC*.

③ In the single-source scenario, according to the AR values, CORR overall achieves the best performance, followed by Bellwether and MMD. All of them outperform DCNNS and VC with statistic significance.

④ In the multi-source scenario, according to the AR values, mCORR achieve the best *MCC* performance, mMMD and mCORR and the best *AUC* performance with statistic significance.

⑤ Compared to the selection algorithms with single source, according to the AR values, only DCNNS that selects multiple sources gets improvement in *MCC*, while MMD, DCNNS and VC get improvement in *AUC*. Hence, not

TABLE 4  
The median *MCC* values of each source data selection algorithm with DPP on 22 target projects.

Target	ALL	UM	Bellwether	DCNNS	VC	CORR	MMD	mDCNNS	mVC	mCORR	mMMD
CM1	0.201	0.203	0.211	0.212	0.196	0.211	0.196	0.178	0.203	0.185	0.178
JM1	0.220	0.213	0.241	0.241	0.231	0.233	0.241	0.230	0.231	0.238	0.230
KC3	0.208	0.165	0.258	0.198	0.216	0.258	0.216	0.222	0.241	0.258	0.230
MC1	0.120	0.092	0.147	0.132	0.132	0.126	0.147	0.133	0.119	0.121	0.122
MC2	0.328	0.303	0.260	0.248	0.240	0.280	0.248	0.278	0.328	0.280	0.248
MW1	0.222	0.216	0.275	0.218	0.231	0.272	0.231	0.235	0.222	0.244	0.231
PC1	0.173	0.159	0.184	0.170	0.170	0.189	0.176	0.175	0.170	0.187	0.182
PC2	0.138	0.150	0.167	0.130	0.165	0.141	0.162	0.161	0.159	0.150	0.164
PC3	0.184	0.240	0.175	0.170	0.187	0.175	0.245	0.177	0.173	0.176	0.230
PC4	0.146	0.178	0.144	0.182	0.148	0.144	0.148	0.131	0.129	0.154	0.131
PC5	0.255	0.289	0.229	0.247	0.295	0.247	0.281	0.215	0.295	0.256	0.254
EQ	0.379	0.389	0.382	0.382	0.371	0.371	0.371	0.382	0.371	0.371	0.371
JDT	0.492	0.391	0.488	0.473	0.510	0.510	0.473	0.502	0.490	0.493	0.473
LC	0.291	0.293	0.282	0.282	0.296	0.303	0.303	0.308	0.296	0.303	0.303
ML	0.258	0.204	0.271	0.201	0.201	0.271	0.260	0.264	0.255	0.261	0.258
PDE	0.271	0.252	0.273	0.267	0.267	0.273	0.267	0.271	0.275	0.270	0.271
Apache	0.361	0.459	0.355	0.359	0.359	0.359	0.359	0.359	0.359	0.361	0.359
Safe	0.412	0.446	0.414	0.394	0.394	0.394	0.394	0.394	0.412	0.394	0.394
ZXing	0.190	0.159	0.188	0.179	0.179	0.179	0.179	0.190	0.179	0.179	0.190
R-2.0	0.411	0.410	0.399	0.423	0.423	0.423	0.423	0.423	0.423	0.423	0.423
R-2.1	0.310	0.303	0.325	0.325	0.303	0.325	0.325	0.325	0.303	0.325	0.325
R-3.0	0.381	0.397	0.371	0.371	0.379	0.371	0.379	0.381	0.379	0.371	0.381
Median	0.257	0.246	0.265	0.243	0.243	0.267	0.259	0.256	0.267	0.262	0.254
AR	3.364(3)	3.682(4)	2.545(1)	3.682(4)	3.227(3)	2.545(1)	2.909(2)	3.091(3)	3.227(3)	2.682(1)	3.364(3)

TABLE 5  
The median *AUC* values of each source data selection algorithm with DPP on 22 target projects.

Target	ALL	UM	Bellwether	DCNNS	VC	CORR	MMD	mDCNNS	mVC	mCORR	mMMD
CM1	0.678	0.706	0.691	0.683	0.651	0.715	0.651	0.671	0.710	0.705	0.671
JM1	0.660	0.661	0.665	0.665	0.667	0.663	0.665	0.667	0.667	0.667	0.667
KC3	0.673	0.673	0.710	0.670	0.665	0.710	0.665	0.668	0.691	0.710	0.679
MC1	0.720	0.699	0.744	0.715	0.747	0.719	0.744	0.725	0.745	0.715	0.755
MC2	0.673	0.687	0.639	0.651	0.603	0.637	0.651	0.669	0.657	0.637	0.651
MW1	0.698	0.727	0.730	0.679	0.710	0.724	0.710	0.699	0.681	0.727	0.710
PC1	0.693	0.742	0.747	0.694	0.694	0.723	0.728	0.697	0.693	0.731	0.719
PC2	0.786	0.787	0.785	0.790	0.789	0.806	0.792	0.790	0.788	0.796	0.793
PC3	0.712	0.738	0.736	0.723	0.714	0.736	0.746	0.726	0.716	0.718	0.742
PC4	0.632	0.665	0.664	0.671	0.637	0.664	0.637	0.644	0.626	0.665	0.644
PC5	0.665	0.712	0.654	0.667	0.685	0.667	0.689	0.656	0.685	0.668	0.665
EQ	0.819	0.818	0.826	0.826	0.798	0.798	0.820	0.826	0.798	0.798	0.820
JDT	0.811	0.799	0.815	0.819	0.819	0.819	0.819	0.814	0.812	0.812	0.819
LC	0.792	0.789	0.790	0.790	0.794	0.798	0.798	0.796	0.794	0.798	0.798
ML	0.674	0.665	0.660	0.651	0.651	0.660	0.669	0.675	0.680	0.678	0.674
PDE	0.738	0.732	0.733	0.726	0.726	0.733	0.726	0.738	0.738	0.739	0.738
Apache	0.684	0.752	0.684	0.710	0.710	0.710	0.710	0.710	0.710	0.684	0.710
Safe	0.784	0.801	0.780	0.806	0.806	0.806	0.806	0.806	0.784	0.806	0.806
ZXing	0.635	0.650	0.640	0.626	0.626	0.626	0.626	0.636	0.626	0.626	0.636
R-2.0	0.774	0.794	0.772	0.777	0.777	0.777	0.777	0.777	0.777	0.777	0.777
R-2.1	0.718	0.732	0.719	0.719	0.716	0.719	0.719	0.719	0.716	0.719	0.719
R-3.0	0.752	0.774	0.752	0.752	0.752	0.752	0.752	0.752	0.752	0.752	0.752
Median	0.708	0.732	0.730	0.711	0.713	0.723	0.724	0.716	0.714	0.720	0.725
AR	4.182(4)	2.773(1)	3.136(2)	3.636(3)	3.995(4)	2.773(1)	3.045(2)	3.273(2)	3.364(2)	2.773(1)	2.727(1)

all data selection algorithms with multiple sources can be improved in these two performance measures.

⑥ With respect to ALL and UM that use all the non-target projects in the same dataset, they did not produce superior prediction performance as compared to the source data selection algorithms. On the contrary, their performance is inferior to some of the source data selection algorithms with statistical significance, especially for ALL. It demonstrates that the prediction results of simply using more source projects are not as good as the well-chosen sources in our experimental settings. This further shows the

necessity of source data selection.

Based on the above observations, on the whole, CORR achieves better *MCC* and *AUC* results with statistical significance among all the source data selection algorithms both in the single and multiple scenarios. It is effective for CORR to conduct the source data selection at the project level. Hence, future CPDP research can use CORR to select suitable source data for target project to further improve prediction performance. In terms of the proposed MMD-based source data selection algorithm, it is a simple and unsupervised method with comparable prediction performance to the state-of-the-

art. Hence, it is feasible for MMD to select source data from a practical perspective, which can be used as an alternative option to the state-of-the-art. In the following experiments, we will use CORR to select single or multiple projects as the source for the target data.

## APPENDIX E

### DETAILED COMPARISON RESULTS OF DATA SAMPLING ALGORITHMS

To examine the effectiveness of the proposed MPOS algorithm on the prediction performance of DSSDPP, we compare the results of DSSDPP with MPOS and other three widely used data sampling algorithms including random under-sampling (RUS), random over-sampling (ROS) and synthetic minority oversampling technique (SMOTE) [16]. Besides, we also compare the prediction results of DSSDPP without sampling that only uses DPP.

Table 6 and 7 show the median MCC and AUC values of DSSDPP with NO, RUS, ROS, SMOTE, and MPOS on 22 projects with 30 repeats, respectively. The overall median values, the *N/S/M/L* results with Cliff's  $\delta$  effect size test [17], and the average rank (AR) with Scott-Knott ESD test of each data sampling algorithm are also reported in these tables. From these tables, we have the following observations:

① In term of the overall median values across 22 projects, there are no apparent differences among these data samplings according to MCC and AUC.

② With respect to the *N/S/M/L* results, DSSDPP with sampling algorithms lead to better performance in less than half of the projects with statistical difference when compared to DPP.

③ Considering that the AR results of each sample algorithm with the Scott-Knott ESD test (the numbers in brackets indicate the ranking of algorithms based on this test), DSSDPP with MPOS yields the best performance than DSSDPP with RUS, ROS, SMOTE and no sampling with statistical significance in terms of MCC. With respect to the AUC values, DSSDPP with MPOS, RUS, ROS, and SMOTE achieve similar prediction performance with statistical significance since they are in the same group. All of them outperform DPP that is without sampling.

Based on the above observations, DSSDPP with MPOS algorithm produces better or comparable prediction results with statistical significance among all the algorithms in terms of MCC and AUC two performance measures. This demonstrates that the MPOS sampling algorithm should be used in DSSDPP for cross-project prediction. Compared to DPP, the results of DSSDPP with data sampling did not produce superior performance in our experimental settings, which implies that it is a challenge task to handle the class imbalance problem in the context of CPDP. This is because there exists large data distribution difference between source and target projects, and the algorithm is sampled over source projects might not generalize well over target projects. The prediction performance of learned model based on the source projects cannot be significantly improved. Hence, future CPDP research should pay more attention to this problem for further improvement of performance of the methods.

TABLE 6  
The median MCC results of DSSDPP with each sampling algorithm on 22 target projects.

Target	DPP	DSSDPP_RUS	DSSDPP_ROS	DSSDPP_SMOTE	DSSDPP_MPOS
CM1	0.216	0.187(N)	0.191(N)	0.192(N)	0.206(N)
JM1	0.194	0.214(L)	0.225(L)	0.218(L)	0.226(L)
KC3	0.273	0.211(N)	0.108(N)	0.180(N)	0.190(N)
MC1	0.120	0.095(N)	0.130(L)	0.134(L)	0.123(S)
MC2	0.231	0.291(L)	0.294(L)	0.254(S)	0.239(N)
MW1	0.236	0.086(N)	0.199(N)	0.216(N)	0.212(N)
PC1	0.192	0.208(L)	0.184(N)	0.201(S)	0.203(M)
PC2	0.167	0.198(L)	0.201(L)	0.165(N)	0.188(L)
PC3	0.293	0.348(L)	0.322(L)	0.329(L)	0.318(L)
PC4	0.188	0.269(L)	0.173(N)	0.247(L)	0.223(L)
PC5	0.246	0.258(L)	0.268(L)	0.265(L)	0.251(S)
EQ	0.353	0.423(L)	0.427(L)	0.391(L)	0.403(L)
JDT	0.500	0.42(N)	0.445(N)	0.432(N)	0.434(N)
LC	0.307	0.282(N)	0.269(N)	0.196(N)	0.269(N)
ML	0.269	0.169(N)	0.125(N)	0.190(N)	0.185(N)
PDE	0.272	0.264(N)	0.266(N)	0.260(N)	0.275(N)
Apache	0.386	0.382(N)	0.404(M)	0.399(M)	0.404(M)
Safe	0.439	0.439(N)	0.439(N)	0.439(N)	0.439(N)
ZXing	0.170	0.201(L)	0.179(S)	0.188(M)	0.179(S)
R-2.0	0.456	0.441(N)	0.448(N)	0.456(N)	0.478(L)
R-2.1	0.326	0.326(N)	0.326(N)	0.326(N)	0.326(N)
R-3.0	0.376	0.376(N)	0.376(N)	0.376(N)	0.376(N)
Median	0.267	0.264	0.260	0.247	0.249
N/S/M/L	-	13/0/0/9	13/1/1/7	12/2/2/6	11/3/2/6
AR	2.636(2)	2.500(2)	2.455(2)	2.500(2)	2.318(1)

TABLE 7  
The median AUC results of DSSDPP with each sampling algorithm on 22 target projects.

Target	DPP	DSSDPP_RUS	DSSDPP_ROS	DSSDPP_SMOTE	DSSDPP_MPOS
CM1	0.680	0.709(L)	0.694(L)	0.731(L)	0.701(L)
JM1	0.611	0.656(L)	0.663(L)	0.659(L)	0.671(L)
KC3	0.768	0.748(N)	0.689(N)	0.738(N)	0.714(N)
MC1	0.717	0.633(N)	0.710(N)	0.716(N)	0.693(N)
MC2	0.597	0.608(S)	0.647(L)	0.600(N)	0.644(L)
MW1	0.693	0.561(N)	0.747(L)	0.696(S)	0.742(L)
PC1	0.674	0.728(L)	0.736(L)	0.740(L)	0.766(L)
PC2	0.834	0.856(L)	0.832(N)	0.830(N)	0.833(N)
PC3	0.796	0.810(L)	0.800(M)	0.794(N)	0.783(N)
PC4	0.723	0.739(L)	0.685(N)	0.736(L)	0.712(N)
PC5	0.661	0.713(L)	0.700(L)	0.704(L)	0.703(L)
EQ	0.730	0.795(L)	0.753(L)	0.787(L)	0.777(L)
JDT	0.818	0.801(N)	0.815(N)	0.795(N)	0.812(N)
LC	0.801	0.791(N)	0.790(N)	0.750(N)	0.794(N)
ML	0.623	0.636(L)	0.607(N)	0.651(L)	0.636(L)
PDE	0.739	0.747(L)	0.744(M)	0.721(N)	0.740(N)
Apache	0.775	0.750(N)	0.773(N)	0.769(N)	0.774(N)
Safe	0.852	0.852(N)	0.852(N)	0.852(N)	0.852(N)
ZXing	0.664	0.653(N)	0.665(N)	0.662(N)	0.666(S)
R-2.0	0.793	0.794(S)	0.788(N)	0.797(S)	0.794(N)
R-2.1	0.712	0.712(N)	0.712(N)	0.712(N)	0.712(N)
R-3.0	0.744	0.744(N)	0.744(N)	0.744(N)	0.744(N)
Median	0.728	0.738	0.736	0.735	0.738
N/S/M/L	-	10/2/0/10	13/0/2/7	13/2/0/7	13/1/0/8
AR	2.682(2)	2.227(1)	2.455(1)	2.364(1)	2.273(1)

## APPENDIX F

### EXTENSIBILITY OF DSSDPP IN MULTI-SOURCE SCENARIO

To check the extensibility of DSSDPP to existing CPDP methods in multi-source scenario, we combine these methods by replacing their classifiers with DSSDPP for CPDP.

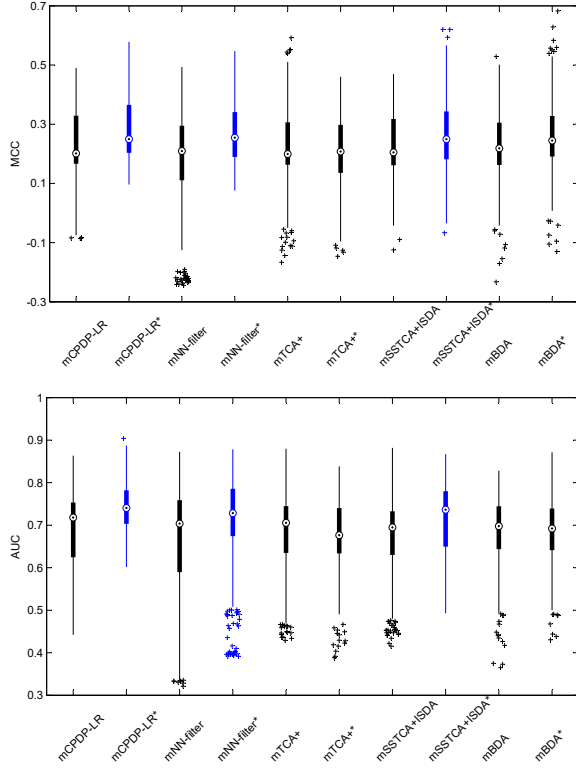


Fig. 3. Comparison of related CPDP methods with and without DPP on all 22 projects in terms of *MCC* and *AUC*. The blue color denotes *win*, black color denotes *tie*, and red color denotes *lose*.

TABLE 8

The Cliff's  $\delta$  values between the original and extended CPDP methods across 22 projects in terms of *MCC* and *AUC*.

Methods	<i>MCC</i>	<i>AUC</i>
mCPDP-LR versus mCPDP-LR*	0.274	0.288
mNN-filter versus mNN-filter*	0.264	0.225
mTCA+ versus mTCA+*	-0.100	-0.073
mSSTCA+ISDA versus mSSTCA+ISDA*	0.181	0.267
mBDA versus mBDA*	0.103	-0.056

Specifically, CPDP-LR [18], NN-filter [13], TCA+ [19], SSTCA+ISDA [20], and BDA [21] using multiple sources are referred to as mCPDP-LR, mNN-filter, mTCA+, mSSTCA+ISDA, and mBDA respectively. And these CPDP methods incorporating the domain programming predictor are referred to as mCPDP-LR\*, mNN-filter\*, mTCA+\*, mSSTCA+ISDA\*, and mBDA\*, respectively.

Fig. 3 shows the prediction results of mCPDP-LR, mNN-filter, mTCA+, mSSTCA+ISDA, and mBDA and their extended methods on 22 projects with 30 repetition times in terms of *MCC* and *AUC*. The boxes of the original CPDP methods are in black color, and there are three colors of boxes within the extended CPDP methods, which denote the significant difference between the original and extended CPDP methods.

- The **blue color** denotes that the extended method blue performs better than (*win*) the original one with statistical difference, according to the magnitude of the difference between these two methods is not trivial based on Cliff's  $\delta$  ( $|\delta| \geq 0.147$ ). The detailed results are shown in Table 8.
- The **black color** denotes that the extended method does

not perform better than (*tie*) the original one with statistical difference, according to the magnitude of the difference between these two methods is trivial ( $|\delta| < 0.147$ ).

- The **red color** denotes that the extended method performs worse than (*lose*) the original one with statistical difference, according to the magnitude of the difference between these two methods is not trivial.

From this figure, we have the following observations:

① In terms of the *MCC* measure, the prediction performance of mCPDP-LR\*, mNN-filter\*, and mSSTCA+ISDA\* outperform their original counterparts since all the Cliff's  $\delta$  values are larger than 0.147. This demonstrates mCPDP-LR, mNN-filter, and mSSTCA+ISDA obtain performance improvement by incorporating them with DSSDPP. For mTCA+\* and mBDA\*, their Cliff's  $\delta$  values are separately -0.100 and 0.103, which are lower than 0.147. This means that combining mTCA+, mBDA with DSSDPP do not improve their performance significantly. But their *MCC* results still slightly increase in comparison with their original counterparts.

② With respect to the *AUC* measure, only mCPDP-LR\*, mNN-filter\*, and mSSTCA+ISDA\* perform better than their original counterparts since all the Cliff's  $\delta$  values are larger than 0.147. On the contrary, mTCA+\* and mBDA\* do not produce better results than their original counterparts with statistical difference based on the Cliff's  $\delta$  values from Table 8. Actually, they seem to have similar prediction performance with their original counterparts in terms of the *AUC* results.

Based on the above observations, combining DSSDPP is beneficial to the prediction performance of mCPDP-LR, mNN-filter, and mSSTCA+ISDA for both *MCC* and *AUC*. For mTCA+ and mBDA, it has little effect on their prediction performance. In summary, most of the studied CPDP methods can improve (i.e., mCPDP-LR, mNN-filter, and mSSTCA+ISDA) or maintain (i.e., mTCA+ and mBDA) their prediction performance by incorporating DSSDPP, which shows the extensibility and effectiveness of DSSDPP in the multi-source scenario.

## REFERENCES

- [1] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. J. Smola, "A kernel method for the two-sample-problem," in *NIPS'07*. MIT Press, 2007, pp. 513–520.
- [2] C. Tantithamthavorn, S. McIntosh, A. Hassan, and K. Matsumoto, "An empirical comparison of model validation techniques for defect prediction models," *IEEE Transactions on Software Engineering*, vol. 43, no. 1, pp. 1–18, 2017.
- [3] C. Tantithamthavorn, S. McIntosh, A. E. Hassan, and K. Matsumoto, "The impact of automated parameter optimization on defect prediction models," *IEEE Transactions on Software Engineering*, vol. 45, no. 7, pp. 683–711, 2019.
- [4] R. Krishna and T. Menzies, "Bellwethers: A baseline method for transfer learning," *IEEE Transactions on Software Engineering*, vol. 45, no. 11, pp. 1081–1104, 2019.
- [5] S. Herbold, "Training data selection for cross-project defect prediction," in *PROMISE'13*. ACM, 2013, pp. 6–15.
- [6] Z. He, F. Peters, T. Menzies, and Y. Yang, "Learning from open-source projects: An empirical study on defect prediction," in *ESEM'13*. IEEE, 2013, pp. 45–54.
- [7] T. Fukushima, Y. Kamei, S. McIntosh, K. Yamashita, and N. Ubayashi, "An empirical study of just-in-time defect prediction using cross-project models," in *MSR'14*. ACM, 2014, pp. 172–181.

- [8] Y. Kamei, T. Fukushima, S. McIntosh, K. Yamashita, N. Ubayashi, and A. E. Hassan, "Studying just-in-time defect prediction using cross-project models," *Empirical Software Engineering*, vol. 21, no. 5, pp. 2072–2106, 2016.
- [9] J. Wang, Y. Chen, H. Yu, M. Huang, and Q. Yang, "Easy transfer learning by exploiting intra-domain structures," in *IEEE International Conference on Multimedia and Expo (ICME'19)*. IEEE, 2019, pp. 1210–1215.
- [10] S. Lessmann, B. Baesens, C. Mues, and S. Pietsch, "Benchmarking classification models for software defect prediction: A proposed framework and novel findings," *IEEE Transactions on Software Engineering*, vol. 34, no. 4, pp. 485–496, 2008.
- [11] B. Ghotra, S. McIntosh, and A. E. Hassan, "Revisiting the impact of classification techniques on the performance of defect prediction models," in *ICSE'15*. IEEE, 2015, pp. 789–800.
- [12] W. Fu, T. Menzies, and X. Shen, "Tuning for software analytics: Is it really necessary?" *Information and Software Technology*, vol. 76, pp. 135–146, 2016.
- [13] 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*, vol. 14, no. 5, pp. 540–578, 2009.
- [14] F. Zhang, A. Mockus, I. Keivanloo, and Y. Zou, "Towards building a universal defect prediction model," in *MSR'14*. ACM, 2014, pp. 182–191.
- [15] —, "Towards building a universal defect prediction model with rank transformed predictors," *Empirical Software Engineering*, vol. 21, no. 5, pp. 2107–2145, 2016.
- [16] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, no. 1, pp. 321–357, 2002.
- [17] J. Romano, J. D. Kromrey, J. Coraggio, and J. Skowronek, "Appropriate statistics for ordinal level data: Should we really be using t-test and cohen's d for evaluating group differences on the nsse and other surveys," in *annual meeting of the Florida Association of Institutional Research*, 2006, pp. 1–33.
- [18] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *The Journal of Machine Learning Research*, vol. 9, no. 9, pp. 1871–1874, 2008.
- [19] J. Nam, S. J. Pan, and S. Kim, "Transfer defect learning," in *ICSE'13*. IEEE, 2013, pp. 382–391.
- [20] X.-Y. Jing, F. Wu, X. Dong, and B. Xu, "An improved sda based defect prediction framework for both within-project and cross-project class-imbalance problems," *IEEE Transactions on Software Engineering*, vol. 43, no. 4, pp. 321–338, 2017.
- [21] Z. Xu, S. Pang, T. Zhang, X. Luo, J. Liu, Y. Tang, X. Yu, and L. Xue, "Cross project defect prediction via balanced distribution adaptation based transfer learning," *Journal of Computer Science and Technology*, vol. 34, no. 5, pp. 1039–1062, 2019.