Supplementary Experiments and Results Analysis

This file provides detailed experimental data and analysis results about the supplementary experiments.

Section I. Applying Trust Score to Cross-Version SDP

The cross-version SDP is a crucial branch in SDP research. It is often treated as a specialized form of cross-project SDP, taking into account the issue of inconsistent distribution of defect data across different versions (also known as concept drift or data drift) [1-3]. According to whether the training and testing data come from the same source, SDP is typically categorized into within-project SDP and cross-project SDP. Cross-project SDP focuses on leveraging defect data from source projects to predict defects in target projects. The challenge arises from the inconsistent data distribution between the source project and the target project, violating the assumptions of traditional machine learning and making it more akin to a transfer learning problem [4-6]. Owing to the adjusted trust score and the trust score not addressing trustworthiness estimation under data drift, the findings of this paper consequently cannot be entirely generalized to the cross-version SDP scenario.

In this section, we conducted preliminary experiments on the application of TS and ATS in the cross-version SDP scenario on a subset of datasets, thereby addressing RQ1-RQ3.

For this purpose, we selected datasets from the PROMISE corpora that have different versions. As the datasets used in this paper are all the latest versions from the PROMISE corpora, we collected datasets from the version just before the latest one. These earlier versions were employed to train ML classifiers, while the latest versions served as the test set. Here we briefly outline the findings obtained from the extended experiment.

RQ1

The experimental results for RQ1 are as shown in Tables 1 and 2. From Tables 1 and 2, it can be observed that in the cross-version SDP scenario, whether identifying correct predictions or incorrect predictions, TS-D is superior to both TS-U and TS. Meanwhile, TS-U and TS have their respective strengths and weaknesses. This observation slightly differs from our previous conclusions.

The discrepancy may arise from the nature of the within-project scenario.

In the within-project SDP scenario, where the sample distributions in the training and testing sets are consistent, the impact of data filtering operations is limited. Furthermore, due to the insufficient quality of data filtering, it may negatively impact the estimation capability of the trust score. Consequently, TS outperforms TS-D and TS-U.

In the cross-version SDP scenario, the distributions of datasets from different versions are inconsistent. In such a context, data filtering can reduce this distribution difference between training set and testing set, thus enhancing the capability of trust score estimation. Moreover, it is important to note that the utilization of initial filtering does not guarantee an enhancement in estimation capability. e.g., TS-U did not exhibit a significant improvement over TS. This also indicates that it is necessary to design a more appropriate data filtering strategy in the cross-version SDP scenario when employing the trust score to estimate the trustworthiness of a prediction.

RQ2

The experimental results for RQ2 are as shown in Tables 3 and 4. From Tables 3 and 4, it can be observed that in the cross-version SDP scenario, whether identifying correct predictions or incorrect predictions, the performance of ATS is slightly better than TS, and both ATS and TS are superior to CS. This result is consistent with our previous findings. Both ATS and TS outperform the raw confidence score, indicating that the trust score can effectively evaluate the trustworthiness of SDP models, even in the

cross-version SDP scenario.

The comparison between ATS and TS indicates that incorporating the confidence score into the trust score does not consistently yield substantial enhancements. Due to the presence of data drift issues, the model's raw confidence score may fail to reflect the randomness or variance inherent in its training process. This could lead to a compromised effect of the raw confidence score on adjusting the trust score. However, it is also noted that while TS and ATS generally outperform CS in cross-version SDP scenarios, exceptions exist. In certain cases, TS may underperform compared to CS, the information provided by CS can adjust the estimation of TS. This adjustment contributes to the overall superiority of ATS compared to TS.

RQ3

The experimental results for RQ3 are as shown in Table 5.

From Table 5, it can be observed that in the cross-version SDP scenario, the application of a reject option based on ATS can improve the performance of defect prediction models in general. This result is consistent with our previous findings.

By comparing the Matthews Correlation Coefficient (MCC) values of the defect prediction model before and after the introduction of the reject option, it is evident that while in some cases the incorporation of the reject option may lead to a decline in predictive performance, in most cases, the reject option indeed enhances the model's defect prediction performance. This also indicates that the ATS demonstrates commendable performance in identifying incorrect predictions, and formulating a reject option based on ATS is indeed feasible, even in cross-version SDP scenarios.

Table 1 The experimental results of three versions of trust scores when identifying correct predictions in cross-version scenario

Datasets	LR			SVM			NB			DT			RF			MLP		
	TS	TS-D	TS-U	TS	TS-D	TS-U	TS	TS-D	TS-U	TS	TS-D	TS-U	TS	TS-D	TS-U	TS	TS-D	TS-U
ant-1.6 → ant-1.7	0.380	**0.395**	0.326	0.407	**0.417**	0.351	0.380	**0.395**	0.326	0.394	**0.405**	0.344	0.415	**0.423**	0.359	0.380	**0.395**	0.326
camel-1.4 → camel-1.6	0.251	0.252	**0.274**	0.374	0.345	**0.379**	0.251	0.252	**0.274**	0.293	0.280	**0.314**	0.255	0.255	**0.277**	0.251	0.252	**0.274**
ivy-1.4 → ivy-2.0	0.201	**0.207**	0.142	**0.276**	0.259	0.236	0.201	**0.207**	0.142	**0.235**	0.227	0.178	0.201	**0.207**	0.142	0.201	**0.207**	0.142
lucene-2.2 → lucene-2.4	0.693	**0.709**	0.635	0.688	**0.703**	0.626	0.693	**0.709**	0.635	0.677	**0.697**	0.608	0.712	**0.728**	0.657	0.693	**0.709**	0.635
poi-2.5 → poi-3.0	0.710	**0.725**	0.682	0.714	**0.728**	0.691	0.710	**0.725**	0.682	0.721	**0.735**	0.679	0.710	**0.726**	0.683	0.710	**0.725**	0.682
prop-5 → prop-6	0.094	**0.138**	0.101	**0.278**	0.272	0.232	0.094	**0.138**	0.101	0.096	**0.139**	0.105	0.094	**0.138**	0.101	0.094	**0.138**	0.101
synapse-1.1 → synapse-1.2	0.439	**0.466**	0.441	0.439	**0.466**	0.441	0.439	**0.466**	0.441	0.472	**0.489**	0.470	0.439	**0.466**	0.441	0.439	**0.466**	0.441
velocity-1.5 → velocity-1.6	0.105	0.104	**0.108**	0.124	0.126	**0.143**	0.105	0.104	**0.108**	0.105	0.104	**0.108**	0.105	0.104	**0.108**	0.105	0.104	**0.108**
xerces-1.3 → xerces-1.4	0.386	**0.403**	0.371	0.698	**0.721**	0.691	0.386	**0.403**	0.371	0.328	**0.340**	0.319	0.398	**0.414**	0.380	0.386	**0.403**	0.371
average	0.362	**0.378**	0.342	0.444	**0.449**	0.421	0.362	**0.378**	0.342	0.369	**0.380**	0.347	0.370	**0.384**	0.350	0.362	**0.378**	0.342

Table 2 The experimental results of three versions of trust scores when identifying incorrect predictions in cross-version scenario

Datasets	LR			SVM			NB			DT			RF			MLP		
	TS	TS-D	TS-U	TS	TS-D	TS-U	TS	TS-D	TS-U	TS	TS-D	TS-U	TS	TS-D	TS-U	TS	TS-D	TS-U
ant-1.6 → ant-1.7	0.901	**0.907**	0.868	0.899	**0.904**	0.865	0.901	**0.907**	0.868	0.900	**0.904**	0.866	0.897	**0.900**	0.863	0.901	**0.907**	0.868
camel-1.4 → camel-1.6	0.858	0.860	**0.873**	0.844	0.837	**0.861**	0.858	0.860	**0.873**	0.856	0.855	**0.871**	0.858	0.859	**0.873**	0.858	0.860	**0.873**
ivy-1.4 → ivy-2.0	0.947	**0.948**	0.927	**0.945**	0.944	0.930	0.947	**0.948**	0.927	**0.946**	0.945	0.927	0.947	**0.948**	0.927	0.947	**0.948**	0.927
lucene-2.2 → lucene-2.4	0.502	**0.506**	0.441	0.513	**0.517**	0.451	0.502	**0.506**	0.441	0.530	**0.537**	0.477	0.470	**0.474**	0.409	0.502	**0.506**	0.441
poi-2.5 → poi-3.0	0.517	**0.529**	0.479	0.510	**0.522**	0.472	0.517	**0.529**	0.479	0.506	**0.519**	0.466	0.521	**0.533**	0.481	0.517	**0.529**	0.479
prop-5 → prop-6	0.903	**0.933**	0.902	0.900	**0.921**	0.888	0.903	**0.933**	0.902	0.901	**0.932**	0.901	0.903	**0.933**	0.902	0.903	**0.933**	0.902
synapse-1.1 → synapse-1.2	0.796	**0.806**	0.789	0.796	**0.806**	0.789	0.796	**0.806**	0.789	0.788	**0.800**	0.786	0.796	**0.806**	0.789	0.796	**0.806**	0.789
velocity-1.5 → velocity-1.6	0.926	**0.929**	0.922	0.785	0.789	**0.815**	0.926	**0.929**	0.922	0.926	**0.929**	0.922	0.926	**0.929**	0.922	0.926	**0.929**	0.922
xerces-1.3 → xerces-1.4	0.698	**0.721**	0.691	0.386	**0.403**	0.371	0.698	**0.721**	0.691	0.723	**0.749**	0.717	0.697	**0.720**	0.688	0.698	**0.721**	0.691
average	0.783	**0.793**	0.766	0.731	**0.738**	0.716	0.783	**0.793**	0.766	0.786	**0.797**	0.770	0.779	**0.789**	0.762	0.783	**0.793**	0.766

Table 3 The comparison results of the adjusted trust score and the baselines when identifying correct predictions

Datasets	LR			SVM			NB			DT			RF			MLP		
	ATS	TS	CS	ATS	TS	CS	ATS	TS	CS	ATS	TS	CS	ATS	TS	CS	ATS	TS	CS
ant-1.6 → ant-1.7	0.892	**0.893**	0.818	**0.886**	**0.886**	0.844	**0.894**	0.892	0.791	0.885	**0.888**	0.759	**0.899**	0.895	0.878	**0.894**	0.892	0.817
camel-1.4 → camel-1.6	0.815	**0.819**	0.644	0.849	**0.855**	0.697	**0.850**	**0.850**	0.723	0.843	**0.844**	0.755	**0.861**	0.857	0.852	0.802	**0.804**	0.688
ivy-1.4 → ivy-2.0	**0.942**	**0.942**	0.823	0.945	**0.946**	0.784	**0.943**	**0.943**	0.838	0.939	**0.940**	0.830	**0.951**	**0.951**	0.938	**0.912**	0.911	0.798
lucene-2.2 → lucene-2.4	**0.705**	0.681	0.736	**0.716**	0.707	0.712	0.698	0.695	**0.731**	**0.654**	**0.654**	0.566	**0.683**	0.680	0.633	0.744	0.728	**0.772**
poi-2.5 → poi-3.0	**0.740**	0.737	0.676	0.721	**0.728**	0.585	0.632	0.630	**0.662**	0.707	**0.711**	0.598	0.709	**0.718**	0.616	**0.725**	**0.725**	0.652
prop-5 → prop-6	**0.906**	0.900	0.898	**0.894**	0.888	0.864	**0.896**	0.892	0.830	**0.873**	0.871	0.779	0.909	0.900	**0.914**	0.885	**0.890**	0.829
synapse-1.1 → synapse-1.2	**0.787**	0.780	0.712	**0.800**	0.788	0.791	0.799	**0.804**	0.652	**0.796**	0.794	0.696	0.809	0.804	**0.812**	**0.707**	**0.707**	0.524
velocity-1.5 → velocity-1.6	0.290	0.289	**0.380**	0.275	**0.299**	0.301	0.470	0.466	**0.487**	0.315	0.295	**0.487**	0.271	0.258	**0.413**	0.188	0.192	**0.207**
xerces-1.3 → xerces-1.4	0.719	**0.726**	0.609	**0.721**	**0.721**	0.574	0.752	0.750	**0.773**	**0.717**	0.711	0.679	0.721	0.710	**0.763**	**0.591**	0.583	0.516
average	**0.755**	0.752	0.700	0.756	**0.758**	0.684	**0.771**	0.769	0.721	**0.748**	0.745	0.683	0.757	0.752	**0.758**	**0.717**	0.715	0.645

Table 4 The comparison results of the adjusted trust score and the baselines when identifying incorrect predictions

Datasets	LR			SVM			NB			DT			RF			MLP		
	ATS	TS	CS	ATS	TS	CS	ATS	TS	CS	ATS	TS	CS	ATS	TS	CS	ATS	TS	CS
ant-1.6 → ant-1.7	**0.397**	0.389	0.334	**0.464**	0.455	0.421	0.358	**0.359**	0.289	0.437	**0.444**	0.270	**0.350**	0.347	0.299	**0.425**	0.422	0.341
camel-1.4 → camel-1.6	0.565	**0.569**	0.369	0.335	**0.349**	0.224	0.402	**0.404**	0.300	0.420	**0.425**	0.284	**0.298**	0.288	0.283	0.615	**0.617**	0.473
ivy-1.4 → ivy-2.0	0.443	**0.447**	0.256	0.264	**0.275**	0.163	**0.362**	0.361	0.249	0.430	**0.433**	0.228	0.246	**0.248**	0.208	0.600	**0.601**	0.418
lucene-2.2 → lucene-2.4	**0.492**	0.478	0.440	**0.452**	0.442	0.450	**0.493**	0.492	0.474	**0.581**	0.565	0.495	0.475	**0.483**	0.413	**0.435**	0.417	0.439
poi-2.5 → poi-3.0	**0.460**	0.454	0.435	0.445	**0.454**	0.319	**0.662**	0.650	0.680	0.517	**0.530**	0.385	0.464	**0.474**	0.347	**0.491**	0.489	0.367
prop-5 → prop-6	**0.297**	0.288	0.210	0.440	0.432	**0.442**	**0.374**	0.360	0.318	0.514	**0.515**	0.354	**0.210**	0.194	0.200	**0.428**	0.423	0.361
synapse-1.1 → synapse-1.2	**0.543**	0.528	0.497	**0.508**	0.499	0.490	**0.468**	0.463	0.359	**0.470**	0.454	0.368	**0.424**	0.416	0.421	**0.663**	0.659	0.464
velocity-1.5 → velocity-1.6	**0.375**	0.368	0.489	0.336	**0.364**	0.347	0.152	0.150	**0.236**	0.387	0.368	**0.586**	0.455	0.440	**0.570**	0.688	**0.697**	0.664
xerces-1.3 → xerces-1.4	0.482	**0.487**	0.369	**0.515**	**0.515**	0.421	0.370	0.364	**0.389**	0.395	**0.398**	0.384	**0.434**	0.428	0.427	0.641	0.626	**0.677**
average	**0.450**	0.445	0.378	0.418	**0.420**	0.364	**0.405**	0.400	0.366	**0.461**	0.459	0.373	**0.373**	0.369	0.352	**0.554**	0.550	0.467

Table 5 The performance of defect prediction models with and without the reject option based on the adjusted trust score

Datasets	LR			SVM			NB			DT			RF			MLP		
	before	after	imp	before	after	imp	before	after	imp	before	after	imp	before	after	imp	before	after	imp
ant-1.6 → ant-1.7	0.425	0.470	**10.668**	0.412	0.434	**5.458**	0.400	0.437	**9.150**	0.292	0.327	**11.846**	0.402	0.440	**9.489**	0.406	0.448	**10.469**
camel-1.4 → camel-1.6	0.157	0.201	**28.380**	0.136	0.166	**21.888**	0.147	0.201	**37.289**	0.191	0.236	**23.990**	0.317	0.313	-1.223	0.146	0.201	**37.394**
ivy-1.4 → ivy-2.0	0.332	0.320	-3.635	0.039	-0.019	-147.999	0.280	0.258	-7.825	0.074	0.046	-37.959	0.212	0.000	-100.000	0.258	0.279	**8.181**
lucene-2.2 → lucene-2.4	0.242	0.301	**24.446**	0.280	0.319	**13.872**	0.311	0.334	**7.337**	0.055	0.104	**89.155**	0.182	0.227	**24.973**	0.327	0.361	**10.611**
poi-2.5 → poi-3.0	0.267	0.277	**3.802**	0.268	0.273	**2.090**	0.191	0.178	-7.113	0.208	0.207	-0.570	0.272	0.274	**0.895**	0.230	0.239	**4.128**
prop-5 → prop-6	0.073	0.062	-14.001	0.044	0.112	**155.653**	0.041	0.106	**157.584**	-0.047	-0.023	-51.596	0.019	-0.014	-173.605	0.021	0.084	**305.735**
synapse-1.1 → synapse-1.2	0.175	0.225	**28.246**	0.212	0.243	**14.617**	0.295	0.346	**17.107**	0.167	0.248	**48.904**	0.220	0.328	**48.875**	0.171	0.222	**30.258**
velocity-1.5 → velocity-1.6	0.141	0.147	**4.080**	0.050	0.073	**46.090**	0.276	0.271	-1.893	0.093	0.103	**11.168**	0.112	0.119	**6.237**	0.103	0.104	**0.636**
xerces-1.3 → xerces-1.4	0.194	0.172	-11.527	0.151	0.148	-2.055	0.252	0.277	**9.974**	0.159	0.140	-11.912	0.130	0.143	**10.028**	0.137	0.167	**21.787**
average	0.223	0.242	**8.463**	0.177	0.194	**9.962**	0.244	0.267	**9.750**	0.132	0.154	**16.582**	0.207	0.203	-1.855	0.200	0.234	**17.125**

Section II. Applying Trust Score to Sophisticated Models

In this section, we have implemented four models: Multi-Layer Perceptron (MLP), Gradient Boosting (GB), TabNet, and DeepJIT. Among them, Gradient Boosting (GB) is an ensemble learning algorithm distinct from RF, while MLP and TabNet are two representative deep learning algorithms suitable for tabular data (as the datasets used in this paper are based on code or process metrics, making them tabular data). GB and MLP have frequently been utilized as baseline classifiers in many prior defect prediction studies. TabNet was employed as a baseline classifier for deep learning methods in a recent JIT defect prediction paper published in 2023 [7]. DeepJIT [8] is a more recent deep learning model for Just-In-Time (JIT) defect prediction.

DeepJIT

Firstly, we reviewed the DeepJIT paper [8] and, utilizing the open-source code and model files provided in it, replicated the experiments on the two datasets, openstack and qt, that they offered. The download links for the source code, experimental data and pretrained models of this paper are as follows:

https://drive.google.com/drive/folders/1sMl2-LbVi3__56kMeS8YSUW7qaKrL2sQ?usp=sharing

https://zenodo.org/record/3965246#.XyEDVnUzY5k

https://github.com/soarsmu/DeepJIT

Since our proposed method requires calculating the distance between training and test samples in a high-dimensional space, the samples must be represented as high-dimensional vectors. Hence, input samples need to be structured. The DeepJIT method, which combines commit messages and code changes, involves dealing with unstructured data. Therefore, DeepJIT employs CNNs to encode the commit message and code changes separately and then uses a fully connected network for feature fusion. Due to the reasons mentioned above, when applying our method to DeepJIT, it is

necessary to structure the input samples. Considering that DeepJIT employs a fully connected network in its final stage, with the input layer having 512 nodes, the input vectors are 512-dimensional. To reduce the dimensionality, we utilized Principal Component Analysis (PCA) to transform 512-dimensional vectors into 20-dimensional vectors. Based on this reduction, we then computed both the trust score and the adjusted trust score. Next, we will analyze the results of RQ1 and RQ2 separately.

RQ1

As shown in Fig. 1, when identifying correct predictions, the precision curves of the confidence score do not consistently rise with increasing percentiles and even show a downward trend. In contrast, the curves of trust scores, including TS, TS-D, and TS-U, exhibit a more consistent upward trend as the percentile level increases. Likewise, when identifying incorrect predictions, the precision curves of TS, TS-D and TS-U tend to increase monotonically with the percentile level, unlike the confidence score. In contrast to the original paper where TS consistently outperforms TS-D and TS-U in identifying both correct and incorrect predictions, in this case, the distinctions between TS, TS-U, and TS-D are less marked.

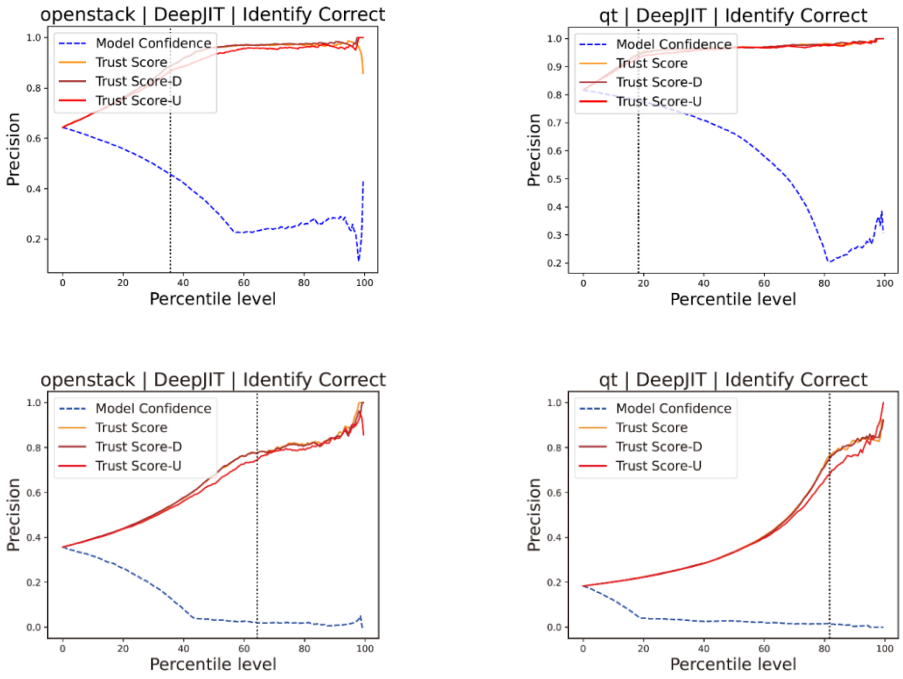

Fig. 1 Precision-percentile curves of identifying correct (incorrect) predictions

To further quantify and compare the effectiveness of the three trust score methods, we calculated the area under the precision-percentile curve (AUC), when identifying correct (incorrect) predictions. The experimental results are as follows.

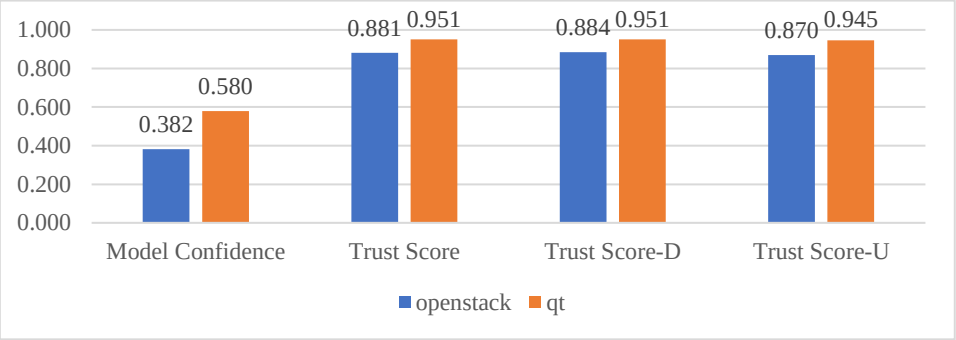

Fig. 2 The AUC values for the above precision-percentile curves (identify correct predictions)

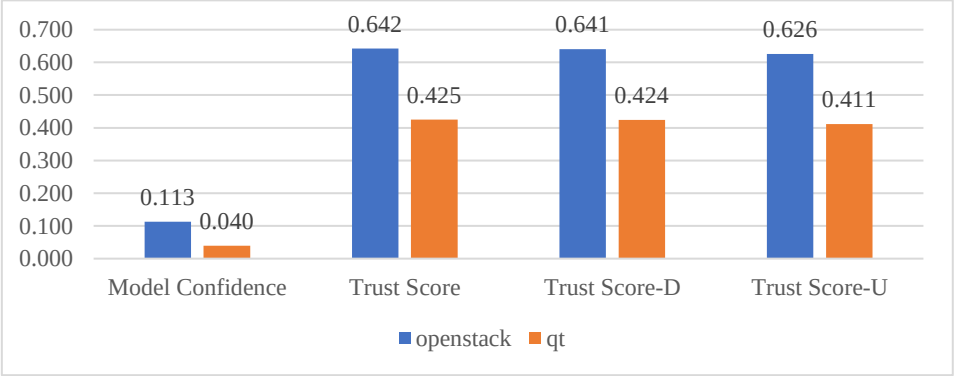

Fig. 3 The AUC values for the above precision-percentile curves (identify incorrect predictions)

The data from both figures clearly show that in identifying both correct and incorrect predictions, TS performs comparably to TS-D and slightly better than TS-U, while significantly outperforming the model confidence. This trend aligns well with the original paper, reinforcing the conclusion that different filtering methods distinctly impact the trust score's ability to estimate trustworthiness, with density-based filtering surpassing uncertainty-based filtering. Inappropriate filtering can negatively impact the precision of the trust score.

However, we must acknowledge that the differences between TS and TS-D, TS-U are relatively minor, significantly less pronounced than those presented in our paper. Upon analyzing these datasets, we found that the sample sizes of openstack and qt are

substantially larger than those of the benchmark datasets used in our paper. The openstack dataset contains 11,973 training samples, and the qt dataset comprises 23,133, significantly exceeding the size of datasets in our paper, where most include only 200-500 samples, even when accounting for test samples. This results in a considerably less pronounced impact of filtering on the trust score in these larger datasets.

RQ2

Upon thorough examination of the code and intermediate outputs, we found that the DeepJIT method in the paper focuses on maximizing AUC values during optimization and training, and its performance is also evaluated based on AUC. As noted by Moussa and Sarro [9], AUC is a threshold-independent measure, which means it evaluates the entire ranking of model predictions without depending on a specific classification threshold. This presents a challenge in applying the confidence score to adjust the trust score. Notably, the outputs of DeepJIT are not centered around the default threshold of 0.5, differing from the setup of our study. In our experiments, we used the default threshold of 0.5 for classification – predictions above 0.5 were classified as defective, and those below as non-defective, which was the rationale behind setting our adjustment function in the form of exp(x-0.5). Consequently, we adjusted this approach by sliding the threshold to obtain an optimized threshold (opt_threshold) based on the model's test outputs, and revised the adjustment function to the form of exp(x-opt_threshold). Subsequently, we obtained the following precision-percentile curves.

Fig. 4 reveals that when identifying both correct and incorrect predictions, the precision curves of the confidence score do not consistently rise with increasing percentiles; in fact, they sometimes trend downwards. In contrast, ATS and TS show a more uniform increase along the percentile levels, compared with CS. However, the comparison between ATS and TS fails to establish ATS's superiority in estimating trustworthiness, which is different from our earlier conclusions. Previously, we deduced that ATS slightly outperformed TS, positing that the integration of the confidence score into the trust score, thereby refining trustworthiness estimation.

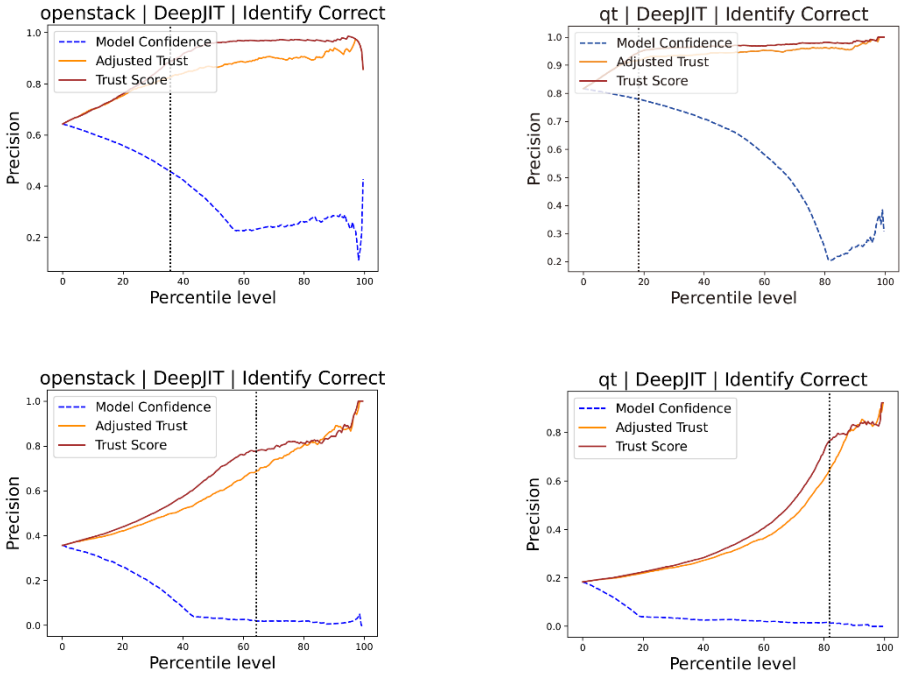

Fig. 4 Precision-percentile curves of identifying correct (incorrect) predictions with ATS, TS, CS

Our understanding of this difference is that the predictive outputs of DeepJIT are low, and their numerical distribution deviates significantly from 0.5. To verify this statement, we have taken the openstack dataset as an example and drawn a boxplot of the predictive outputs of the DeepJIT model on the test set:

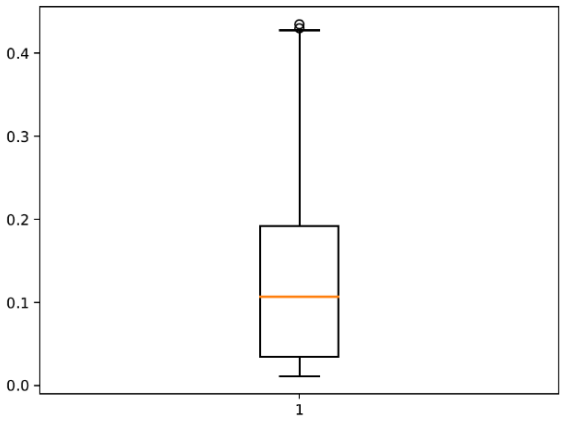

Fig. 5 The distribution of prediction outputs of DeepJIT

Although the predictive performance seems promising based on the ranking of its probability output values (i.e., the AUC value), it may not be satisfactory in terms of

the MCC metric since it is challenging to find the optimal classification threshold. This difficulty also leads to the ineffectiveness of adjusting based on the proposed formulas exp(x-0.5) or exp(x-threshold) in our paper. In such a context, the effectiveness of TS actually surpasses that of ATS. This could be viewed as a limitation inherent to the ATS method, which requires that the outputs of the prediction model remain largely unbiased. Otherwise, the adjustment function we propose would require a redesign, as its effectiveness often cannot surpass TS under these conditions. Nevertheless, it is important to note that both ATS and TS consistently outperform CS. This indicates that even in deep learning models, the more critical factor causing uncertainty or unreliability in model outputs is a lack of knowledge or data to inform the model's predictions (epistemic uncertainty), rather than the intrinsic randomness within the learning process (aleatoric uncertainty).

Adding Three Models: MLP, GB and TabNet

Here we briefly outline the findings obtained from the extended experiment. Subsequently, we provide the complete experimental data.

RQ1

The two tables 6 and 7 present a comparison of the three trust scores in terms of the average AUC values when identifying correct and incorrect predictions, respectively. The results show that the TS method outperforms the other two methods across most datasets, both for identifying correct and incorrect predictions. This trend aligns with our previous conclusions, indicating that our response to RQ1 possesses a certain degree of generalizability. Even within some sophisticated models such as tree ensemble algorithms and deep learning methods, as well as within some updated datasets, the performance of TS consistently surpasses that of TS-D and TS-U.

RQ2

The two tables 8 and 9 present a comparison of the three trust scores in terms of the average AUC values when identifying correct and incorrect predictions, respectively.

The results show that, whether for identifying correct predictions or for identifying incorrect predictions, ATS and TS can achieve higher AUC values than CS in most datasets. Meanwhile, the AUC values of ATS are always slightly higher than those of TS. This trend also aligns with our previous conclusions, validating the extensibility of our response to RQ2 within some sophisticated models and more updated datasets.

RQ3

The table 10 presents the test performance results with and without the reject option based on ATS values. The results show that, the average test performance of three models is improved after introducing the reject option. This trend also aligns with our previous conclusions, validating the extensibility of our response to RQ3 within some sophisticated models and more updated datasets.

The supplementary experiments above further validate the generalizability of the methods and conclusions proposed in our paper by expanding sophisticated models.

Table 6 The results of three versions of trust scores when identifying correct predictions

Datasets	GB			MLP			TabNet		
	TS	TS-D	TS-U	TS	TS-D	TS-U	TS	TS-D	TS-U
CM1	**0.896**	0.892	0.893	0.834	0.828	**0.835**	**0.796**	0.794	0.794
JM1	**0.822**	0.723	0.808	**0.816**	0.720	0.801	**0.766**	0.684	0.745
KC1	0.819	0.778	**0.823**	**0.808**	0.767	0.808	0.807	0.777	**0.808**
KC3	**0.883**	0.878	0.842	**0.830**	0.823	0.788	**0.771**	0.767	0.721
MC2	0.716	**0.718**	0.682	**0.695**	0.693	0.669	0.620	**0.621**	0.586
MW1	**0.917**	0.915	**0.917**	**0.858**	0.853	0.854	**0.769**	0.762	0.761
PC1	**0.951**	0.944	0.946	**0.936**	0.927	0.931	**0.827**	0.819	0.818
PC3	**0.920**	0.909	0.914	**0.916**	0.907	0.910	**0.925**	0.919	0.920
PC4	**0.947**	0.937	0.944	**0.941**	0.933	0.935	**0.949**	0.942	0.943
PC5	**0.843**	0.803	0.838	**0.832**	0.789	0.825	**0.838**	0.804	0.832
ant-1.7	**0.899**	0.860	0.886	**0.891**	0.855	0.878	**0.832**	0.814	0.814
camel-1.6	**0.860**	0.830	0.855	**0.836**	0.806	0.827	**0.852**	0.831	0.846
ivy-2.0	**0.937**	0.933	0.927	**0.913**	0.910	0.902	**0.832**	0.828	0.821
lucene-2.4	0.766	**0.776**	0.771	0.734	**0.750**	0.730	0.655	**0.665**	0.648
poi-3.0	0.898	0.896	**0.906**	0.873	0.872	**0.876**	0.799	0.800	**0.803**
prop-6	**0.913**	0.909	0.882	**0.886**	0.881	0.852	0.830	**0.831**	0.796
synapse-1.2	**0.849**	0.841	0.827	**0.792**	0.787	0.765	0.742	**0.745**	0.709
tomcat	**0.952**	0.940	0.944	**0.945**	0.934	0.936	**0.952**	0.945	0.943
velocity-1.6	**0.846**	0.834	0.845	**0.772**	0.761	0.773	**0.737**	0.727	0.729
xerces-1.4	0.963	0.962	**0.964**	**0.924**	0.921	0.923	**0.824**	0.823	0.820
Alfresco	**0.888**	0.863	0.854	**0.882**	0.858	0.845	**0.835**	0.816	0.789
owncloudandroid	**0.915**	0.905	0.901	**0.908**	0.894	0.894	**0.616**	0.604	0.590
androidWalpaper	**0.889**	0.883	0.871	**0.819**	0.810	0.802	**0.835**	0.827	0.817
reddit	**0.917**	0.915	0.909	0.876	**0.878**	0.866	0.897	**0.899**	0.892
chatSecure	**0.856**	0.848	0.837	**0.819**	0.811	0.795	**0.583**	0.580	0.538
anySoftKeyboard	**0.855**	0.840	0.840	**0.838**	0.826	0.820	**0.754**	0.742	0.730
kiwis	**0.862**	0.854	0.851	**0.847**	**0.837**	0.833	**0.681**	0.671	0.661
Pageturner	**0.943**	0.942	0.935	0.834	**0.835**	0.822	**0.895**	**0.895**	0.882
aFall	**0.756**	0.749	0.751	**0.745**	0.740	0.737	**0.716**	0.714	0.703
facebook	**0.855**	0.850	0.817	**0.819**	0.818	0.779	0.825	**0.827**	0.791
Apg	**0.838**	0.829	0.830	**0.830**	0.823	0.820	**0.703**	0.698	0.685
androidSync	**0.842**	0.840	0.819	0.782	**0.789**	0.761	0.814	**0.819**	0.794
average	**0.876**	0.862	0.863	**0.845**	0.832	0.831	**0.790**	0.781	0.773

Table 7 The results of three versions of trust scores when identifying incorrect predictions

Datasets	GB			MLP			TabNet		
	TS	TS-D	TS-U	TS	TS-D	TS-U	TS	TS-D	TS-U
CM1	0.454	0.440	**0.455**	0.583	0.574	**0.585**	**0.624**	0.619	0.622
JM1	**0.471**	0.423	0.444	**0.507**	0.460	0.478	**0.606**	0.566	0.582
KC1	0.423	0.402	**0.424**	**0.490**	0.470	**0.490**	**0.489**	0.466	0.488
KC3	**0.333**	0.316	0.269	**0.524**	0.515	0.455	**0.592**	0.585	0.533
MC2	0.390	**0.396**	0.333	0.410	**0.421**	0.362	**0.542**	**0.542**	0.502
MW1	**0.309**	0.294	0.290	**0.492**	0.486	0.471	**0.710**	0.705	0.691
PC1	**0.389**	0.365	0.377	**0.529**	0.509	0.517	**0.640**	0.633	0.632
PC3	**0.483**	0.471	0.477	**0.504**	0.488	0.496	**0.433**	0.417	0.420
PC4	**0.261**	0.238	0.254	**0.340**	0.319	0.331	**0.313**	0.289	0.301
PC5	**0.426**	0.406	0.422	0.478	0.457	**0.481**	**0.462**	0.442	0.457
ant-1.7	**0.369**	0.355	0.353	**0.425**	0.412	0.405	**0.603**	0.591	0.582
camel-1.6	**0.471**	0.425	0.451	**0.549**	0.510	0.526	**0.489**	0.446	0.458
ivy-2.0	**0.423**	0.412	0.398	**0.547**	0.540	0.527	**0.648**	0.647	0.633
lucene-2.4	0.417	**0.424**	0.411	0.492	**0.498**	0.488	0.623	**0.634**	0.620
poi-3.0	**0.304**	0.301	0.319	0.445	0.446	**0.447**	0.661	**0.662**	0.656
prop-6	**0.399**	0.381	0.364	**0.573**	0.566	0.548	**0.610**	0.605	0.577
synapse-1.2	0.337	**0.339**	0.303	0.519	**0.526**	0.479	**0.634**	0.633	0.602
tomcat	**0.411**	0.383	0.381	**0.467**	0.444	0.438	**0.408**	0.384	0.381
velocity-1.6	**0.409**	0.388	0.373	**0.567**	0.552	0.545	**0.618**	0.610	0.605
xerces-1.4	0.125	**0.126**	0.120	**0.480**	0.477	0.472	**0.719**	0.717	0.703
Alfresco	**0.360**	0.329	0.257	**0.413**	0.385	0.300	**0.582**	0.561	0.474
owncloudandroid	**0.386**	0.370	0.334	**0.441**	0.422	0.396	**0.830**	0.827	0.796
androidWalpaper	**0.515**	0.500	0.501	**0.645**	0.638	0.624	**0.594**	0.582	0.574
reddit	0.323	**0.330**	0.309	0.422	**0.430**	0.411	0.475	**0.487**	0.452
chatSecure	**0.374**	0.360	0.336	**0.520**	0.510	0.478	**0.791**	0.790	0.763
anySoftKeyboard	**0.406**	0.390	0.369	**0.497**	0.480	0.454	**0.623**	0.612	0.586
kiwis	**0.407**	0.391	0.392	**0.478**	0.462	0.460	**0.732**	0.725	0.716
Pageturner	**0.323**	0.321	0.296	**0.459**	**0.459**	0.444	0.545	**0.548**	0.525
aFall	**0.347**	0.341	0.332	**0.405**	0.403	0.381	**0.463**	0.460	0.432
facebook	0.374	**0.380**	0.344	0.517	**0.523**	0.487	0.502	**0.514**	0.463
Apg	**0.349**	0.341	0.339	**0.426**	0.413	0.409	**0.669**	0.665	0.652
androidSync	0.368	**0.380**	0.343	0.492	**0.505**	0.471	0.506	**0.518**	0.485
average	**0.379**	0.366	0.355	**0.489**	0.478	0.464	**0.585**	0.577	0.561

Table 8 The results of the ATS and two baselines when identifying correct predictions

Datasets	GB			MLP			TabNet		
	ATS	TS	CS	ATS	TS	CS	ATS	TS	CS
CM1	**0.900**	0.896	0.882	**0.838**	0.834	0.764	0.794	**0.796**	0.612
JM1	**0.828**	0.822	0.795	**0.819**	0.816	0.725	**0.767**	0.766	0.640
KC1	**0.823**	0.819	0.798	**0.810**	0.808	0.742	**0.809**	0.807	0.729
KC3	**0.884**	0.883	0.839	**0.830**	**0.830**	0.688	0.770	**0.771**	0.582
MC2	0.719	0.716	**0.760**	**0.698**	0.695	0.674	0.615	**0.620**	0.547
MW1	**0.918**	0.917	0.901	**0.860**	0.858	0.749	**0.769**	**0.769**	0.501
PC1	**0.953**	0.951	0.943	**0.938**	0.936	0.884	**0.827**	**0.827**	0.587
PC3	**0.922**	0.920	0.905	**0.919**	0.916	0.893	**0.927**	0.925	0.908
PC4	0.954	0.947	**0.969**	**0.947**	0.941	0.941	**0.952**	0.949	0.946
PC5	**0.850**	0.843	0.848	**0.837**	0.832	0.814	**0.843**	0.838	0.810
ant-1.7	**0.900**	0.899	0.877	**0.892**	0.891	0.848	0.829	**0.832**	0.645
camel-1.6	**0.865**	0.860	0.835	**0.841**	0.836	0.770	**0.856**	0.852	0.805
ivy-2.0	**0.937**	**0.937**	0.911	**0.913**	**0.913**	0.795	0.830	**0.832**	0.605
lucene-2.4	0.781	0.766	**0.793**	**0.749**	0.734	0.730	0.654	**0.655**	0.520
poi-3.0	0.901	0.898	**0.909**	**0.873**	**0.873**	0.772	0.795	**0.799**	0.578
prop-6	**0.913**	**0.913**	0.875	**0.887**	0.886	0.770	0.829	**0.830**	0.635
synapse-1.2	**0.849**	**0.849**	0.830	**0.793**	0.792	0.722	0.737	**0.742**	0.545
tomcat	**0.953**	0.952	0.921	**0.946**	0.945	0.844	**0.953**	0.952	0.914
velocity-1.6	**0.852**	0.846	0.831	**0.780**	0.772	0.687	**0.738**	0.737	0.586
xerces-1.4	**0.964**	0.963	0.971	**0.927**	0.924	0.902	0.823	**0.824**	0.570
Alfresco	**0.890**	0.888	0.859	**0.883**	0.882	0.809	**0.837**	0.835	0.710
owncloudandroid	**0.920**	0.915	0.912	**0.910**	0.908	0.811	**0.622**	0.616	0.505
androidWalpaper	**0.889**	**0.889**	0.836	**0.819**	**0.819**	0.637	0.832	**0.835**	0.591
reddit	**0.919**	0.917	0.918	0.875	**0.876**	0.772	0.896	**0.897**	0.726
chatSecure	**0.864**	0.856	0.867	**0.821**	0.819	0.735	**0.591**	0.583	0.545
anySoftKeyboard	**0.862**	0.855	0.859	**0.841**	0.838	0.762	**0.756**	0.754	0.644
kiwis	**0.870**	0.862	0.861	**0.847**	**0.847**	0.710	**0.683**	0.681	0.565
Pageturner	**0.944**	0.943	0.924	0.833	**0.834**	0.712	0.894	**0.895**	0.659
aFall	0.771	0.756	**0.810**	**0.746**	0.745	0.712	0.710	**0.716**	0.594
facebook	0.859	0.855	**0.861**	**0.819**	**0.819**	0.721	0.824	**0.825**	0.674
Apg	0.850	0.838	**0.858**	**0.835**	0.830	0.779	**0.711**	0.703	0.663
androidSync	**0.845**	0.842	0.830	**0.787**	0.782	0.730	**0.818**	0.814	0.679
average	**0.880**	0.876	0.868	**0.847**	0.845	0.769	**0.790**	**0.790**	0.651

Table 9 The results of the ATS and two baselines when identifying incorrect predictions

Datasets	GB			MLP			TabNet		
	ATS	TS	CS	ATS	TS	CS	ATS	TS	CS
CM1	**0.458**	0.454	0.391	**0.585**	0.583	0.475	0.622	**0.624**	0.430
JM1	**0.477**	0.471	0.396	**0.511**	0.507	0.399	**0.612**	0.606	0.469
KC1	**0.429**	0.423	0.379	**0.495**	0.490	0.407	**0.495**	0.489	0.399
KC3	**0.339**	0.333	0.284	0.523	**0.524**	0.378	0.591	**0.592**	0.403
MC2	**0.393**	0.390	0.411	**0.412**	0.410	0.391	0.537	**0.542**	0.446
MW1	**0.312**	0.309	0.255	**0.496**	0.492	0.360	**0.710**	**0.710**	0.449
PC1	**0.392**	0.389	0.332	**0.533**	0.529	0.407	0.639	**0.640**	0.407
PC3	**0.485**	0.483	0.410	**0.507**	0.504	0.426	**0.437**	0.433	0.367
PC4	**0.272**	0.261	0.284	**0.355**	0.340	0.328	**0.319**	0.313	0.293
PC5	**0.436**	0.426	0.429	**0.488**	0.478	0.447	**0.472**	0.462	0.425
ant-1.7	**0.375**	0.369	0.329	**0.431**	0.425	0.352	0.598	**0.603**	0.410
camel-1.6	**0.477**	0.471	0.398	**0.556**	0.549	0.450	**0.496**	0.489	0.401
ivy-2.0	0.422	**0.423**	0.340	**0.548**	0.547	0.386	0.645	**0.648**	0.421
lucene-2.4	**0.428**	0.417	0.411	**0.504**	0.492	0.446	0.619	**0.623**	0.497
poi-3.0	**0.311**	0.304	0.314	**0.448**	0.445	0.325	0.657	**0.661**	0.427
prop-6	**0.400**	0.399	0.305	**0.574**	0.573	0.410	**0.610**	**0.610**	0.400
synapse-1.2	**0.340**	0.337	0.334	**0.524**	0.519	0.421	0.628	**0.634**	0.454
tomcat	**0.413**	0.411	0.308	**0.472**	0.467	0.325	**0.410**	0.408	0.315
velocity-1.6	**0.424**	0.409	0.384	**0.574**	0.567	0.457	0.615	**0.618**	0.478
xerces-1.4	0.130	0.125	**0.143**	**0.487**	0.480	0.379	0.716	**0.719**	0.431
Alfresco	**0.368**	0.360	0.307	**0.418**	0.413	0.322	**0.592**	0.582	0.414
owncloudandroid	**0.399**	0.386	0.358	**0.447**	0.441	0.330	**0.835**	0.830	0.666
androidWalpaper	**0.515**	**0.515**	0.407	**0.647**	0.645	0.446	0.590	**0.594**	0.368
reddit	0.325	0.323	**0.332**	**0.425**	0.422	0.357	0.472	**0.475**	0.322
chatSecure	**0.389**	0.374	0.371	**0.525**	0.520	0.412	**0.796**	0.791	0.711
anySoftKeyboard	**0.418**	0.406	0.389	**0.501**	0.497	0.403	**0.629**	0.623	0.489
kiwis	**0.419**	0.407	0.387	**0.478**	**0.478**	0.348	**0.738**	0.732	0.580
Pageturner	**0.329**	0.323	0.295	**0.459**	**0.459**	0.339	0.540	**0.545**	0.326
aFall	0.360	0.347	**0.413**	**0.409**	0.405	0.420	0.458	**0.463**	0.378
facebook	0.385	0.374	**0.398**	**0.521**	0.517	0.436	**0.503**	0.502	0.378
Apg	0.368	0.349	**0.377**	**0.437**	0.426	0.380	**0.679**	0.669	0.585
androidSync	**0.372**	0.368	0.336	**0.494**	0.492	0.445	**0.506**	**0.506**	0.367
average	**0.386**	0.379	0.350	**0.493**	0.489	0.394	**0.586**	0.585	0.435

RQ3 The performance of SDP models with and without the reject option based on the ATS

Datasets	GB			MLP			TabNet		
	before	after	imp	before	after	imp	before	after	imp
CM1	0.219	0.246	**12.145**	0.200	0.241	**20.679**	0.005	0.049	**818.843**
JM1	0.243	0.290	**19.131**	0.244	0.289	**18.622**	0.198	0.235	**18.796**
KC1	0.269	0.323	**20.166**	0.252	0.301	**19.501**	0.222	0.289	**29.890**
KC3	0.308	0.315	**2.159**	0.214	0.217	**1.211**	0.058	0.094	**62.702**
MC2	0.279	0.323	**15.868**	0.250	0.259	**3.415**	0.072	0.099	**38.841**
MW1	0.278	0.298	**6.927**	0.283	0.299	**5.575**	0.024	0.030	**25.819**
PC1	0.289	0.299	**3.604**	0.294	0.340	**15.583**	0.066	0.097	**47.012**
PC3	0.352	0.382	**8.731**	0.343	0.381	**11.080**	0.264	0.298	**12.952**
PC4	0.561	0.639	**13.852**	0.492	0.580	**17.909**	0.530	0.607	**14.369**
PC5	0.355	0.413	**16.240**	0.343	0.415	**20.957**	0.347	0.399	**14.996**
ant-1.7	0.438	0.494	**12.828**	0.402	0.459	**14.273**	0.115	0.160	**39.791**
camel-1.6	0.252	0.288	**14.131**	0.216	0.248	**15.076**	0.219	0.249	**13.982**
ivy-2.0	0.311	0.344	**10.684**	0.246	0.334	**35.843**	0.036	0.061	**71.676**
lucene-2.4	0.367	0.416	**13.521**	0.243	0.281	**15.462**	0.066	0.115	**74.847**
poi-3.0	0.596	0.618	**3.652**	0.479	0.534	**11.471**	0.183	0.249	**36.058**
prop-6	0.242	0.247	**1.980**	0.174	0.193	**11.310**	0.069	0.085	**23.587**
synapse-1.2	0.464	0.494	**6.646**	0.305	0.357	**17.192**	0.090	0.138	**54.192**
tomcat	0.334	0.350	**4.954**	0.296	0.365	**23.160**	0.220	0.259	**17.891**
velocity-1.6	0.413	0.464	**12.441**	0.193	0.256	**32.868**	0.109	0.183	**68.544**
xerces-1.4	0.805	0.846	**5.159**	0.517	0.640	**23.760**	0.204	0.285	**39.994**
Alfresco	0.435	0.517	**18.935**	0.371	0.456	**22.988**	0.310	0.376	**21.273**
owncloudandroid	0.477	0.552	**15.721**	0.398	0.480	**20.740**	0.160	0.212	**32.468**
androidWalpaper	0.258	0.235	-8.932	0.164	0.180	**9.594**	0.111	0.130	**17.549**
reddit	0.552	0.578	**4.700**	0.421	0.451	**7.175**	0.331	0.394	**19.285**
chatSecure	0.469	0.523	**11.474**	0.275	0.360	**30.989**	0.102	0.126	**24.146**
anySoftKeyboard	0.418	0.467	**11.706**	0.320	0.369	**15.400**	0.274	0.315	**14.996**
kiwis	0.381	0.438	**14.902**	0.265	0.332	**25.558**	0.214	0.253	**18.109**
Pageturner	0.426	0.423	-0.689	0.280	0.237	-15.322	0.098	0.140	**41.973**
aFall	0.385	0.412	**6.952**	0.268	0.307	**14.497**	0.178	0.201	**12.762**
facebook	0.437	0.493	**12.618**	0.245	0.309	**26.169**	0.197	0.258	**31.055**
Apg	0.465	0.521	**12.206**	0.373	0.431	**15.791**	0.256	0.306	**19.520**
androidSync	0.443	0.482	**8.968**	0.289	0.306	**5.813**	0.248	0.279	**12.321**
average	0.391	0.429	**9.793**	0.302	0.350	**16.073**	0.174	0.218	**55.945**

References

[1] Xu Z, Liu J, Luo X, et al. Cross-version defect prediction via hybrid active learning with kernel principal component analysis[C]//2018 IEEE 25th international conference on software analysis, evolution and reengineering (SANER). IEEE, 2018: 209-220.

[2] Zhao Y, Wang Y, Zhang Y, et al. ST-TLF: Cross-version defect prediction framework based transfer learning[J]. Information and Software Technology, 2022, 149: 106939.

[3] Amasaki S. Cross-version defect prediction: use historical data, cross-project data, or both?[J]. Empirical Software Engineering, 2020, 25: 1573-1595.

[4] Nam J, Pan S J, Kim S. Transfer defect learning[C]//2013 35th international conference on software engineering (ICSE). IEEE, 2013: 382-391.

[5] Ma Y, Luo G, Zeng X, et al. Transfer learning for cross-company software defect prediction[J]. Information and Software Technology, 2012, 54(3): 248-256.

[6] Liu C, Yang D, Xia X, et al. A two-phase transfer learning model for cross-project defect prediction[J]. Information and Software Technology, 2019, 107: 125-136.

[7] Hoang T, Dam H K, Kamei Y, et al. DeepJIT: an end-to-end deep learning framework for just-in-time defect prediction[C]//2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR). IEEE, 2019: 34-45.

[8] van Dinter R, Catal C, Giray G, et al. Just-in-time defect prediction for mobile applications: using shallow or deep learning?[J]. Software Quality Journal, 2023: 1-22.

[9] Moussa R, Sarro F. On the use of evaluation measures for defect prediction studies[C]//Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis. 2022: 101-113.