A Systematic Literature Review and Meta-analysis on Cross Project Defect Prediction

Seyedrebvar Hosseini, Burak Turhan, *Member, IEEE*, and Dimuthu Gunarathna

Abstract—**Background:** Cross project defect prediction (CPDP) recently gained considerable attention, yet there are no systematic efforts to analyse existing empirical evidence. **Objective:** To synthesise literature to understand the state-of-the-art in CPDP with respect to metrics, models, data approaches, datasets and associated performances. Further, we aim to assess the performance of CPDP vs. within project DP models. **Method:** We conducted a systematic literature review. Results from primary studies are synthesised (thematic, meta-analysis) to answer research questions. **Results:** We identified 30 primary studies passing quality assessment. Performance measures, except precision, vary with the choice of metrics. Recall, precision, f-measure, and AUC are the most common measures. Models based on Nearest-Neighbour and Decision Tree tend to perform well in CPDP, whereas the popular naïve Bayes yield average performance. Performance of ensembles varies greatly across f-measure and AUC. Data approaches address CPDP challenges using row/column processing, which improve CPDP in terms of recall at the cost of precision. This is observed in multiple occasions including the meta-analysis of CPDP vs. WPDP. NASA and Jureczko datasets seem to favour CPDP over WPDP more frequently. **Conclusion:** CPDP is still a challenge and requires more research before trustworthy applications can take place. We provide guidelines for further research.

Index Terms—Defect Prediction, Fault Prediction, Cross Project, Systematic Literature Review, Meta-analysis, Within Project

1 Introduction

In the realm of software quality control, defect prediction in general and cross project defect prediction in particular, have attracted a lot of attention. The principle of defect prediction is to learn a model (supervised or unsupervised) from a corpus of data (e.g., static code features, churn data, defect information) and apply the model to new and unseen data. The training data can be from the same project, i.e., within project defect prediction (WPDP) or (the majority or the entirety) from other projects, i.e., cross project defect prediction (CPDP). The goals of defect prediction are to identify the faulty units of code, to estimate the number of remaining defects in the system, to track and locate faulty changes, classes, functions or statements, for optimal use of the available quality assurance resources. Defect prediction, however, is not just limited to prediction making process. Specifically, studying defect prediction is a reasonable choice for training young researchers in software analytics, and defect prediction concepts can be applied to many other areas in software engineering [98], [99]. While these *in vitro* benefits are acknowledged by the community, the limited amount of *in vivo* applications is the major source of criticism for defect prediction research [97].

The lack of, and the difficulty of collecting and organising, defect related data is one of the reasons why companies do not consider using defect prediction in practice [84] - this is also why researchers usually conduct *in vitro* studies. In such circumstances, CPDP is a viable option for companies such that it requires minimum effort for data collection, which can be a long and tedious process for the alternative case of setting up local repositories, from their side. Another reason for using cross project data as Kitchenham et al. asserted (originally for effort estimation, but true for defect prediction as well), is the change of practices in companies over time which could make the local data collection less important, because the existing practices might not be representative anymore [80]. On the other hand, the key premise of CPDP is to learn from data from a set of projects and then to apply resulting models to another set of projects [82], [84]. Therefore, in presence of relevant data from other projects, including open source ecosystems, CPDP has a great practical potential for *in vivo* applications. The prediction outcomes with data from other projects can be of great value as even a tiny decrease in the bug rates (through early and automated detection of defects) can lead to significant financial savings in terms of quality assurance costs [84], as opposed to exponential growth in repair costs and damages [81], [84] as a result of failure to discover bugs in a timely manner.

Examples of *in vivo* CPDP studies for commercial/proprietary systems already exist in the literature, revealing the potential of cross project defect prediction. For example, Turhan et al. [83], [84] and Misirli et al. [91] were able to use the data collected from very different domains for predicting defects across projects, e.g., aerospace to telecommunication and white-goods, and reported reduced inspection efforts by 72% [91]. Their results, even though not

- *S. Hosseini and D. Gunarathna are with the M3S, Faculty of Information Technology and Electrical Engineering, University of Oulu, Oulu, Finland.*
  E-mail: rebvar@oulu.fi, dimuthugunarathna@gmail.com
- *B. Turhan is with the Department of Computer Science, Brunel University London, London, UK.*
  E-mail: burak.turhan@brunel.ac.uk

Manuscript received April 19, 20XX; revised August 26, 20XX.

as accurate as the within project predictions, demonstrated the applicability of CPDP in practical ways. In addition, experiments by Zimmerman et al. [82] revealed extensive insights as well as challenges for CPDP. They investigated CPDP in a large scale study on 12 projects from both open and closed source domains. Their 622 pairwise predictions resulted in only 21 successful cases (precision, recall and accuracy all higher than or equal to 0.75), i.e., 3.4% success rate. Their experiments demonstrated that CPDP is not necessarily bi-directional/symmetrical. Particularly, the predictor trained with Mozilla Firefox was able to predict defects of Internet Explorer with acceptable performance, while the results did not hold when the prediction direction was reversed. More broadly, and despite the low overall success rate, they observed that the open source projects were more successful in predicting closed source software than the other way around [82].

The increased interest toward CPDP also resulted in a growing number of publications on the topic. However, there is no up-to-date comprehensive picture of the current state of CPDP research. Studies in this area do not always agree in their conclusions, especially when it comes to the performance in comparison with WPDP. While some studies favour WPDP (e.g [S15], [S39]), others make a case in favour of CPDP (e.g [S1], [S3], [S5], [S11], [S23], [S40], [S41]). Moreover, the role of the numerous proposed approaches in these disagreements is not clear due to varying settings across studies. In other words, it is not possible to have a clear view of the current status of research in CPDP (and vs. WPDP) based on individual and isolated studies, unless they are systematically synthesised.

The aforementioned reasons indicate a need, and motivate us, to conduct a systematic literature review (SLR) and a formal meta-analysis to summarise the empirical evidence on CPDP literature.

1.1 Prior Research

To the best of our knowledge, there does not exist any SLRs and formal meta-analysis on CPDP, however, three secondary studies have already been performed on defect prediction [1], [3], [5]. In this section, we discuss these studies in comparison with our review to highlight our differences.

The first one by Fenton and Neil [5] covered defect prediction studies up to 1999 and provided a critique of the field with an emphasis on good and bad practices in statistical data analysis, though their review did not follow a systematic approach. In 2009, Catal and Diri [3] conducted a systematic review of software fault prediction studies. While Catal and Diri's review provided a conceptual classification of the frequency of metrics, methods and datasets used, it lacks a (qualitative or quantitative) synthesis of primary studies, hence one should consider it more of a systematic mapping study than a literature review [2]. The most recent and comprehensive review by Hall et al. [1] has investigated how model performance is affected by the context, independent variables and the modelling techniques in defect prediction with a synthesis of existing work from 2000 to 2010.

All three previous reviews have performed their analysis in the broader area of defect prediction and none have specifically addressed CPDP. Many CPDP publications are not included in earlier reviews, given the field's relatively short history. For example, of the 30 primary studies that passed quality assessment in this review, only 4 were published before 2010, which is the last year covered by the latest secondary study. Given the practical implications discussed in the earlier section, this certainly points to a niche in research, justifying our motivation to conduct this SLR.

Further, CPDP poses unique learning challenges as opposed to WPDP. Specifically, CPDP is an instance of transfer learning problem in the broader area of machine learning where additional challenges associated with learning from different/heterogeneous data sources, i.e., under dataset shift [13], becomes the main focus. As a result, rather than mere applications of existing techniques, data manipulation approaches become of utmost interest and importance in CPDP settings. Due to their nature of inquiry, none of the earlier secondary studies provided a view under the lens of different data manipulation or learning approaches specific to CPDP settings.

Finally, no formal meta-analysis has been considered by previous SLRs due to the lack of a 'control' group to benchmark with. In our case, synthesis of studies that compare CPDP vs. WPDP, i.e., comparing the experimental group CPDP with the control group WPDP, via a meta-analysis is legitimate. Therefore, to facilitate the understanding and use of CPDP, it is necessary to systematically summarise the empirical evidence from existing studies in contrast with WPDP. We should caution the reader that, different than the systematic literature review part, our meta-analysis covers only those studies that benchmark CPDP vs. WPDP, i.e., we did not include pure CPDP or WPDP studies in the meta-analysis.

In conducting and reporting our review, we follow the reporting structure of Hall et al. [1] while setting the scope on CPDP rather than defect prediction in general. We extend their approach by conducting meta-analysis using both fixed and random effect models to investigate the relative performances of CPDP and WPDP.

1.2 Research Goals

The objective of this study is to summarise, analyse and assess the empirical evidence regarding metrics, modelling techniques, different approaches and performance evaluation criteria in the context of CPDP. Further, we want to explore the relative performance of CPDP vs. Within Project Defect Prediction (WPDP) models.

We defined five research questions to achieve these goals. Table 1 shows those research questions and the motivation behind each question.

1.3 Contributions

This SLR contributes to the existing literature in the following ways:

- We identified a set of 46 primary studies related to CPDP published until 2015. Community can use this set as a starting point to conduct further research on CPDP.

TABLE 1
Research Questions

Research Question	Motivation
RQ1. Which independent variables are used in CPDP and how are their performances?	Different independent variables (metrics) have been used in the context of CPDP. Product, process and object-oriented (OO) metrics are the most popular among them. One of the goals of this study is to synthesise the metrics that are being used in CPDP based on the current knowledge. Also, the performances of the models that use such metrics are compared, since although these comparisons are sometimes performed within the context of individual studies, there does not exist any comprehensive study that investigates which set of metrics performs the best in CPDP.
RQ2. Which modelling techniques are used in CPDP and how are their performances?	Different modelling techniques (Machine Learning and Regression) are used in the context of CPDP. Logistic Regression (LR), naïve Bayes (NB), Bayesian Networks (BN), Decision Tree (DTree) and Random Forest (RF) are some of the popular techniques. The goal here is to determine which modelling techniques are frequently used in CPDP and how they relatively perform.
RQ3. Which evaluation criteria are used to measure the performance of CPDP?	Most of the studies have used measures such as precision, recall and f-measure to report the performance of the CPDP models. The aim of this question is to determine how CPDP models are evaluated.
RQ4. Which approaches in CPDP yield better performances?	Filtering, feature selection, and transfer learning are some of the techniques used by researchers to achieve better CPDP performance. The aim of this question is to find and report such approaches.
RQ5. How is the performance of CPDP models compared with that of WPDP models?	The results of some studies suggest that CPDP models are able to achieve performance similar to that of WPDP models and even in some cases they outperform WPDP [S5], [S40], [S41]. On the other hand, some other studies present the evidence for the claim that CPDP under-performs WPDP [S20], [S39]. We investigate this question as there is no clear consensus about the performance of CPDP models compared with that of WPDP.

- We further provide a subset of 30 primary studies which fulfilled the quality assessment criteria indicative of a baseline rigour in their reporting. These studies can act as a reliable basis for further comparisons and benchmarks.
- We present a comprehensive qualitative and quantitative synthesis reflecting the state-of-the-art in CPDP with data extracted from those 30 high-rigour studies. Our synthesis covers the following themes: metrics, prediction models, different approaches and performance evaluation in CPDP research.
- We present a meta-analysis to assess the relative performances of CPDP vs. WPDP grouped per study, dataset and prediction model. To increase the validity of the meta-analysis, we use both fixed and random effects analysis and account for the differences within and between different studies.
- We provide guidelines and recommendations based on our findings to support further research in the area.

2 Research Methodology

To achieve our research goals, we conducted a Systematic Literature Review (SLR) following the guideline provided by Kitchenham and Charters [36]. In addition to this guideline, we followed the structure of Hall et al. [1] for conducting the review and presenting the results. Particularly, we used Hall et al.'s set of criteria (with slight modifications) for inclusion/exclusion and quality assessment of defect prediction studies with customisation for cross project defect prediction context. All steps of our research are documented and the related data are available online for further validation, exploration and replication. Scripts for the data analysis part of this SLR are also available online as part of the replication package for our SLR [100].

2.1 Selection of Primary Studies

A set of search strings were identified based on the research questions. This set was refined by checking the title and keywords of the relevant papers that were already known to us. After considering the alternative spellings and synonyms for the identified search strings, they were combined using logical ANDs and ORs to create the complete search term. We used the following search string:

("cross-project" OR "cross project' OR "multi-project" OR "multi project" OR "multiple project" OR "cross-company" OR "cross company" OR "mixed-project" OR "mix-project" OR "mixed project" OR "mix project") AND (defect* OR fault* OR bug* OR error*) AND (predict* OR estimat*) AND software

This search term was modified to suit the specific requirements of different electronic databases. The complete list of customised search terms is available in the online appendix. We used ACM Digital Library, IEEExplore, ISI Web of Science, Scopus and Google Scholar to search for candidate primary studies. Search was performed on the full text of the papers, if allowed by the digital library, i.e., ACM Digital Library, IEEExplore, Google Scholar. For Scopus and ISI Web of Science, the search was conducted on title, abstract, and keywords. We conducted our search on June 6th, 2015 and identified studies published up until that date. After applying inclusion/exclusion criteria to digital library search results (explained in the next section), we used the resulting set as input for the snowballing process, recommended by Wohlin [28], to identify additional studies (with potential hits to papers published after the initial search date). As per the guidelines provided by Wohlin, we applied backward and forward snowballing iteratively until no further papers for inclusion were detected [28].

2.2 Inclusion and Exclusion Criteria

Studies must report empirical CPDP research (learning and prediction) to be included. These could involve experiments, case studies, tool development and evaluation, novel model developments, replications, new approaches to overcome

TABLE 2
Inclusion/Exclusion Criteria

Inclusion Criteria
1) The paper must be an empirical study about cross project defect prediction - It must include models trained on data from a set of projects and tested on different set of projects for the purpose of predicting defects in a software unit. 2) The prediction outcome must contain defect related information, including but not limited to defect labels (classification) or defect numbers (regression). 3) The paper must be a peer reviewed full research paper published in a conference proceedings or a journal.
Exclusion Criteria
1) Studies focusing on topics other than CPDP such as pure WPDP, test-suite reduction, test-case prioritisation, code reviews, inspections, recommendation systems, reliability modelling, and correlational studies or recommendation systems with no or limited prediction focus. 2) Conference version of a study that has an extended journal version. 3) Grey Literature or not in English.

specific CPDP challenges, or proposing new datasets or metrics, as long as a prediction and its evaluation is involved. We consider predictions that are trained and tested using data from different projects as cross project defect prediction. Prediction outcome must include defect related information, for example, defect labels, number of defects, etc. Small proportion of within project data included in the training process, i.e., mixed data predictions are also acceptable as long as the focus of the study is to learn from other projects. In addition, papers must be in English and peer reviewed full conference/journal papers to be included. This criteria is important due to use of Google Scholar, which possibly could return low quality studies, non-published manuscripts and grey literature. Only the most comprehensive or the most recent version of repeated studies are included [1], i.e., if a conference paper was extended as a journal paper, only the extended journal version was included. Examples are [85] by Turhan et al. and [86] by Canfora et al. which are the earlier (excluded) conference versions of their more recent (included) journal papers, i.e., [S13] and [S11] respectively.

Correlational studies that investigate associations between metrics and defect related information, as well as recommendations with no prediction, or prediction not as their main focus, are excluded. For example, Menzies et al. [51] mine rules in the form of (e.g., KLOC=1, NOC=2) from instances in local (vs. global) clusters that minimise the defect counts. The results reported in that paper are not predictions, but rather should be read as "Modules with KLOC=1 are more likely to be non-faulty, so if you can bring all your code base to satisfy KLOC=1 rule, your defects will be reduced". Reported results are coverage of defects by the rules in terms of the percentage of total defects in modules that satisfy the rule over all defects. Hence, no defect prediction involved. Further, studies are excluded when they focus on tasks other than defect prediction, such as test-suite reduction, test-case prioritisation, code reviews, inspections, recommendation systems, and reliability modelling. Studies which perform only WPDP predictions and grey literature, e.g., theses, technical reports, practitioner blogs, are excluded.

The inclusion and exclusion criteria, also summarised in Table 2, were applied and piloted by two researchers starting with the assessment of 15 randomly selected primary studies from the initial set of identified papers. The reliability of the inclusion/exclusion decisions was measured using pairwise inter-rater reliability with Cohen's Kappa statistic [29]. The agreement rate in the pilot study was "moderate" (0.59). The pilot study helped us to develop a collective understanding of the inclusion and exclusion criteria. Then, the assessment was performed for the full list of the identified studies. The agreement rate in this case was "substantial" (0.69). Disagreements were resolved after discussions between the two researchers and in case a consensus was not reached the third author was consulted as a tie breaker.

2.3 Selection Results

Figure 1 illustrates the overview of the study selection process. In total, 1889 studies were detected based on our defined search terms. After discarding the duplicate studies, 962 of them were left for further assessment. Applying the inclusion and exclusion criteria to the title and abstract of each paper, decreased the pool of the papers for full-text reading from 962 to 41 studies. These studies were read in full and 29 fulfilled the inclusion criteria. Finally, 5 and 12 relevant primary studies were found using backward and forward snowballing, respectively, resulting in 46 studies to be included in this SLR. One might notice the high number of studies discovered through snowballing method in comparison with the total number of primary studies. The reason for observing such results is due to the high number of studies published in 2015 after the search was performed. Specifically, 8 out of 12 papers discovered through forward snowballing are published after the search date ([S37]–[S44]), and during the pilot and protocol refinement phases.

2.4 Quality Assessment

The quality of each primary study was assessed based on a quality checklist. The quality checklist questions were developed according to the suggestions given by Kitchenham and Charters [36] and Hall et al. [1]. In fact, for the most part, the checklist customised for defect prediction and defined by Hall et al. [1] was used with slight modifications. The goal of the quality assessment step is to extract the primary studies with sufficient information suitable for analysis and answering the defined research questions.

The assessment process was piloted by applying the assessment criteria to a set of five randomly selected papers. Two researchers were involved in the pilot study and the assessment process was refined according to the results of the pilot study. The resulting six stages of assessment criteria are provided below.

Stage 1: Prediction. To be eligible, a study must report the process of building a cross project defect prediction model. Further, the prediction model must be trained and tested on different (project) data and the labels of the test data must not be used during the training of the model.

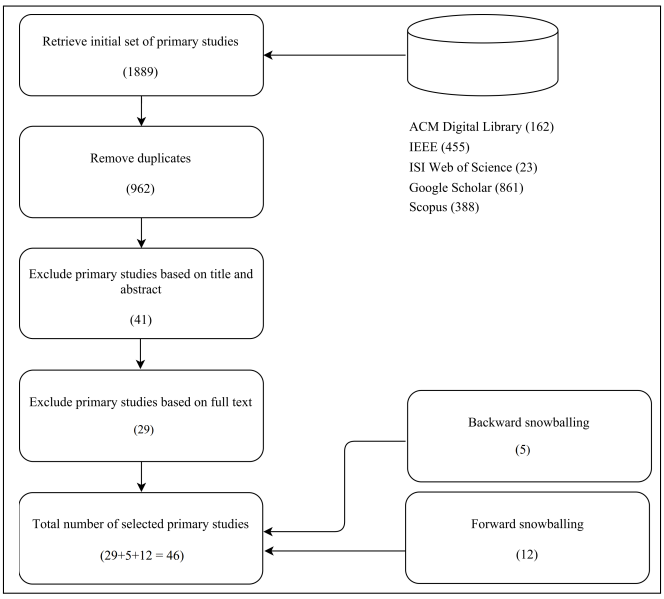


Fig. 1. Selection of the primary studies

To summarise, the purpose of this stage is to double check whether a paper reports a cross project prediction model or not. If so, the details of the model building process will be extracted in Stage 3.

Stage 2: Context. Sufficient context data must be reported in a study as they are important for interpreting the findings. Except for the maturity, similar contextual criteria used by the Hall et al. [1] were utilised in this study. Maturity was ignored as many of the studies are using the NASA datasets for which no maturity information could be found from MDP or PROMISE repositories. The details of this stage are presented in Table 3.

Stage 3: Prediction Model Details. The details of the prediction model are crucial for answering research questions RQ1 and RQ2. Independent and dependent variables, their granularity, and the modelling technique(s) used to build the model(s) must be reported clearly. Table 4 provides details of this criteria.

Stage 4: Approach. To answer RQ4, it is necessary for a study to clearly report utilised CP approaches. The CP approach (for example NN-filtering, EM-clustering, transfer learning, etc.) must be described in detail if a specific approach is taken into consideration by a study.

Stage 5: Data. The data and how they are acquired must be reported in the study. The criteria for data are essential for the reliability of the prediction models. Table 5 denotes the criteria applied in this stage.

Stage 6: Predictive performance. Assessing the performance of the prediction model is vital for validating the model usefulness and making comparisons. Suitable performance measures must be reported for both Categorical and Continuous models. Note that analysis units can be package, class, file, module, etc. To be able to answer RQ4, the study must satisfy the criteria presented in Table 6.

The quality assessment checklist was independently applied to all 46 primary studies by two researchers. All disagreements on the quality assessment results were discussed and a consensus was reached eventually. Few cases where agreement could not be reached were sent to a third researcher for further investigation. 11 studies did not meet the quality assessment criteria (failed in one or more steps) and this phase resulted in 35 studies to proceed with the data extraction phase. Table 7 shows the studies that failed at each quality assessment stage. Of the five studies failed in stage 6, three did not report the predictive performance values clearly. Further, five studies which passed the quality assessment criteria were not included in the data extraction process. These papers and the reason why they are excluded can be found in Table 8. Therefore, 30 studies were included for the data extraction phase. An overview of these studies will be presented in later sections.

2.5 Data Extraction

The data extraction form was designed to collect the data from primary studies and consequently, answering the research questions. The data were extracted from all studies that have passed the quality assessment criteria. The data extraction process was piloted on a sample of five randomly selected primary studies to assess the completeness and usability of the data extraction form, as well as to guarantee the consistency between the researchers. Two researchers were involved in the pilot study and the structure of the data extraction form was finalised accordingly. The studies that had passed the quality assessment criteria were divided between the researchers to extract data from each set of papers independently. The extracted data for each study were held in tables, i.e., an Excel sheet per study. All the extracted data were inspected by a second researcher to ensure the correctness of the extracted values. In addition, we implemented automated scripts to check the consistency of the entries in the data tables and the text of the primary studies. Finally, the results were compiled into a single Excel file for the data analysis phase.

The data extraction form consists of three parts; namely context data, qualitative data, and quantitative data.

Context Data. Context of the study includes the details of the study such as the domain of the applications, project/dataset names, project size, and sources of data. The "aim of the study" field holds extra information about the CPDP related challenges addressed in the study with the explicit goal descriptions extracted from each (such as "difference in the data distribution" or "class imbalance problem").

Qualitative Data. For each primary study, a short summary of the main findings and conclusions, reported by the author(s), were extracted.

Quantitative Data. The quantitative data were extracted for categorical and continuous cases separately based on the granularity of the dependent variable. For those studies reporting both categorical and continuous results, only the set that presented the majority of the results was extracted. For both categorical and continuous models, dependent variable granularity, independent variable(s), performance evaluation criteria, modelling technique(s), data set(s), and performance values are extracted. In terms of notations, used in the form, W and C are used respectively to represent the type of the prediction model within (usually as benchmark), and cross (usually multiple models, one or

TABLE 3
Context Criteria

Criteria	Definition
Source of the data	The source must be reported. If the study has used publicly available datasets, the names and versions of the datasets must be reported.
Size	Size of the system being studied must be given. It could be in terms of KLOC, #classes, #Instances or an indication to get an idea about the size of the system.
Application domain	If the model used data from different systems, the application domain should be reported for the majority.
Programming language	The programming language(s) of the system being studied must be given.

TABLE 4
Prediction Model Details Criteria

Criteria	Definition
Are the Independent variables (metrics) clearly reported?	Independent variables used to build the model must be clearly defined (for example, Size and Complexity, Object Oriented or Process metrics).
Is the dependent variable clearly reported?	The model can predict whether the code unit is faulty or non-faulty (i.e., categorical dependent variable) or predict the number of defects in a code unit (i.e., continuous dependent variable).
Is the granularity of the dependent variable reported?	The granularity of the dependent variable (method, class, package, file, etc.) must be reported.
Is the modelling technique clearly reported?	Statistical or machine learning modelling technique (e.g., naïve Bayes, Logistic Regression, DTree, Support Vector Machine, K-nearest neighbour etc.) approaches must be clearly stated.

TABLE 5
Data Criteria

Criteria	Definition
Is the fault data extraction process clear?	If the study has used private datasets, it is necessary to mention how data was extracted (for example from the Version Control System of a software project). If the data is obtained from the publicly available repositories (like PROMISE repository), the details of the datasets and the source(s) must be presented.
Is the metric extraction (Independent variables) process adequately described?	The process by which independent variables (metrics) are extracted must be stated (e.g tools used and metric collection process). In the case of using publicly available datasets, reference for the data must be presented.

TABLE 6
Predictive Performance Criteria

Criteria	Definition
Is the performance evaluation criteria used clearly reported?	The performance measure of the model must be reported. (for example the performance of categorical models can be measured by confusion matrix or AUC and those for continuous models can be reported in terms of regression error measures).
Has the predictive performance values been clearly reported?	The predictive performance values must be clearly presented in terms of raw performance numbers, means or medians.

TABLE 7
Results of applying the quality assessment criteria

Quality assessment stage	Excluded studies
Stage 1: Prediction	
Stage 2: Context	[S6], [S32]
Stage 3: Model building	[S28], [S36]
Stage 4: CP approach	[S45]
Stage 5: Data	
Stage 6: Predictive performance	[S9], [S10], [S22], [S33], [S34]
Failed in more than one stages	[S12]

TABLE 8
Additional excluded studies and the reasons for their exclusion

Paper identifier	Reason for Exclusion
[S17], [S19], [S21]	The performance values are given only in the plots. The authors were contacted, but no data was received.
[S29], [S30]	The source of the papers is arXiv. These two studies were kept in the list until the analysis phase started. By that time they were excluded from the list as they were not published in a peer-reviewed source.

more of which are the proposed approach by the study) reported in the primary studies. The within/cross data extraction process is similar to the Kitchenham et al.'s practice in their two systematic review studies where they evaluate within versus cross company effort estimation performances [79], [80]. The performance values are recorded in terms of original, average, and median values. In order to have a more extensive analysis, additional performance values are calculated whenever possible (e.g., F-Score from precision and recall). We used the guidelines provided by Hall et al. and Bowes et al. [1], [42] for this purpose. These additional values are estimates for the exact values whenever the median and mean values are reported. The dataset (training and/or test) used for the prediction (e.g Ant-1.7 $\rightarrow$ Camel-1.6) are extracted whenever possible. Moreover, the CP data approach and other additional information related to the performance measures are also extracted.

2.6 Data Analysis

There are many potential confounding factors that may challenge and affect the results of the synthesis performed in secondary studies. For example, in our case, these are different defect prediction models build for addressing different aspects of CPDP and evaluated in different contexts. It is useful to develop or follow heuristics to conduct data analysis based on assumptions to provide insights, as opposed to doing nothing. It is, after all, one of the goals of SLRs to find patterns in individual studies that hold across different, and potentially confounding, settings. For example, Kitchenham et al. reported on the merits of cross vs. within company effort estimation studies [80], where one acknowledges the contextual differences between included studies in interpreting their results. Similarly, Hall et al. combined data from individual defect prediction studies and summarised them around identified themes. Hence, as long the results are interpreted in the light of stated limitations, they would be valid in their scope.

Therefore, acknowledging the potential threats associated, we have followed the practices of Hall et al. [1] for data synthesis and analysis. Specifically, we collected and combined the qualitative and quantitative data provided by the primary studies as the basis for our data synthesis in answering our research questions on the different themes related with CPDP studies. Hence, we caution the reader that the results presented under each theme should be considered independent of others and not be taken as performance comparison between two specific models. Rather, our results for theme X should be interpreted with assuming that other factors (themes) do not affect the outcome (directly or as a result of interplay of many factors) or that their impact is uniformly distributed across different studies cancelling out a systematic bias. With this assumption, we report the performance of CPDP, whenever applicable, in relation to theme X, avoiding comparisons between specific models or techniques, but rather highlighting trends.

In addition, we have performed a meta-analysis of CPDP vs. WPDP, since we have dichotomies of treatments that can be compared, similar to the meta-analysis of test-driven development vs. other software development approaches by Rafique and Misić [92]. Studies included in our meta-analysis are a subset of primary studies. Specifically, our meta-analysis includes those studies that passed the quality assessment stage and if they reported results of experiments that compare CPDP with WPDP. Similar to Kitchenham et al., in comparing the two treatments we are only interested in those studies that compare the treatments of interest [80], i.e., CPDP vs. WPDP.

We did not replicate the proposed approaches/models in the studies, as this is out of the scope of an SLR. For example, to conduct an SLR in medicine, a research group do not replicate the medical trials reported in primary studies.

2.6.1 Performance Indicators

We mainly used violin plots [31] to illustrate the results of our analysis centred around performance indicators. A violin plot is similar to box plots, but more informative. A box plot only shows the summary statistics such as mean/median and inter-quartile ranges, while the violin plot shows the full distribution of the data. The violin plots were drawn only for categorical data. In the plots, the solid horizontal lines indicate the mean of the values for each case whereas the thin continuous lines represent the median. Each case has a label assigned to it representing the name for that particular case. For example, in the case of learning techniques, NB, LR, SVM, etc. were used as the labels for each technique in the plots[1]. Following each label, the number of data points that were used to generate that particular plot are noted in parenthesis. The plots in each case are sorted based on the medians, as the median is less sensitive to outliers. Like Hall et al. [1], we restricted each case to have at least three data points to contribute to the plots. The values reported by the continuous studies were in terms of dissimilar performance measures and hence not suitable for the violin plot.

There were no common set of performance measures used by all of the categorical studies. Accordingly, and due to the nature of our data, we did not perform any statistical tests for comparing the results of different CPDP studies or models. Instead, we grouped data from primary studies according to different themes described in the following sections. The violin plots simply show pooled performances in relation to an investigated theme without considering other confounding themes or factors that may affect the performance. Despite its limitations, this is useful to identify general patterns and potential research gaps [1]. Potential threats associated with this approach are addressed in Section 7. Therefore, the primary studies were synthesised in relation to the certain factors such as independent variables, modelling techniques, and CPDP data approaches whenever the same performance measures were used in the studies.

The categorical studies have presented their CPDP predictive performance values in different forms. To be exact, the studies have reported the values in terms of original, average, and median values. Original values refer to the predictions for which either no repetitions are required or not performed. Randomness and aggregating the results of multiple predictions in one value are common cases where median and average values are reported. Eight of the studies have reported average performance values ([S1], [S5], [S14], [S18], [S23], [S25], [S26], [S40]). Another set of eight studies have presented median performance values ([S3], [S11], [S13], [S20], [S27], [S37], [S43], [S44]). Thirteen studies have presented original values ([S1], [S2], [S4], [S7], [S8], [S15], [S16], [S24], [S35], [S39], [S41], [S42], [S46]). Study [S1] has used multiple forms (average, original) in its reporting. These distinctions are taken into consideration when generating the violin plots.

Original values contribute to the median and average plots as they are median and average of a sample with size one. No weighting has been considered to distinguish the original values from averages and medians. Further, due to the high number of plots, only the plots for median cases are presented in this paper. The online appendix provides additional plots for the case of average values.

1. Please see Appendix A for a full list of the abbreviations used throughout the paper

Based on the utilised performance measures, different sets of plots were generated for comparisons. Plots were drawn for f-measure, precision, recall, and AUC performance metrics. These values were either reported in the study or were calculated indirectly by us when possible. Note that some studies have reported their performance values in terms of multiple performance measures and they appear in multiple plots (e.g [S25], [S39], [S40] have reported AUC along with f-measure, precision, and recall).

In addition to the violin plots, we used tables for summarising and presenting qualitative results.

2.6.2 Modelling Techniques

Different modelling techniques are extracted for each model reported in the primary studies. These modelling techniques are categorised as follows: Base learners (like naïve Bayes, Logistic Regression, etc.), optimisation techniques (e.g., Boosting [S18], [S26], [S42], [S44], Bagging [S42], Genetic Programming [S14], MO-Optimisation [S11], Meta-Learner [S5], [S8], [S42]), and other ensemble/composite algorithms (e.g., [S4], [S43]). We present them in tabular form and their performances with violin plots.

2.6.3 CPDP Approaches

CPDP approaches correspond to data processing approach by each study and are presented in tabular form and in violin plots. We discuss these in two categories: row (datum) processing and column (feature) processing. These include "filtering", "data transformation", "clustering", "mixed data", "feature selection", and "normalization" to name a few. Primary studies usually use a combination of these approaches. These approaches will be discussed in the next section.

When a study reports multiple models with different approaches, its relevant performance data rows are included in the category of both approaches when plotting violin plots. Likewise, if the proposed CPDP technique consists of multiple approaches itself, the performance data rows are included in the category of all the relevant approaches (e.g., DTB in [S18] is added to mixed data, filtering, re-weighting, and re-sampling as it uses all of them).

2.6.4 Meta-analysis and Forest Plots

Meta-analysis and, specifically, both fixed and random effect models are considered to compare the results of CPDP and WPDP studies. Forest plots are used to demonstrate the results of our meta-analysis visually. These comparisons are performed for precision, recall, f-measure, and AUC performance measures as they are among the top five frequently used measures in the studies. In addition to the traditional meta-analysis, where unit of analysis is each study, we performed two additional analyses by setting the unit of analysis to learners and datasets. These additional analyses can provide useful insights on how CPDP (vs. WPDP) behave on specific datasets and learners. Finally, we included both fixed and random effect models, not only to provide more validity and insight into the results, but also to reveal possible sources of bias as will be explained in Section 4

3 Results

To start with an overview of primary studies, the majority of the CPDP studies were published during the past five years. Figure 2 shows the frequency of the papers published each year. Figure 2 indicates that the interest toward CPDP is growing. 43% of the studies are published in 2015, indicating that more contemporary studies are included in this SLR. Regarding the publication sources, the proportion of primary studies published in journals (53%) are slightly higher than those published in the conferences (47%), which can be an indication of the maturity of research conducted in this area.

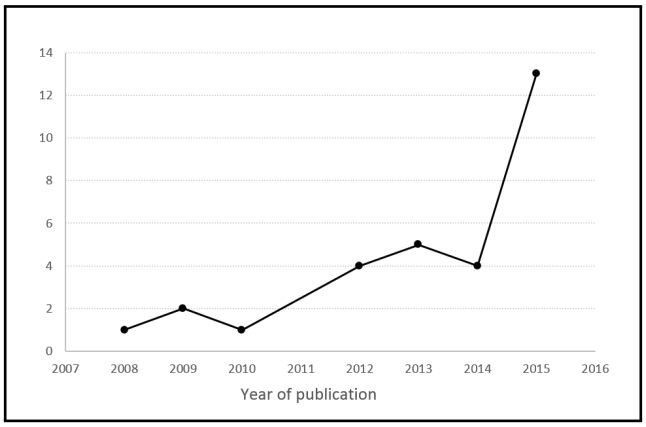


Fig. 2. The rate of interest toward CPDP

Out of the total 30 primary studies, 28 use categorical and the remaining two use continuous dependent variables. This paper presents the results of the categorical studies in depth (for details of the two continuous models, please see the online appendix). For abbreviations used in this section, please see Appendix A.

In the following we will present the results of our synthesis based on different themes. We start each subsection with a summary of the findings to provide an insight about the results and then these summaries are expanded and explained in detail.

3.1 Theme: Independent Variables

All performance measures vary in relation to the choice of set of independent variables. Combinations of OO and SCM as well as source code text metrics seem to provide good median performances in CPDP. Process metrics on the other hand, does not show the same premise. Table 9 provides a summary of independent variables used.

Many different sets of independent variables are used in the primary studies. Categorical and continuous model tables in the online appendix show the detailed list of independent variables used in individual studies. Table 9 summarises the metric sets used in the categorical models. As shown in the table, most studies use a combination of different metric sets. SCM+LOC (879 data rows) and OO+SCM+LOC (626 data rows), which are combinations of single metric sets, are the most common.

Five studies ([S13], [S24], [S37], [S40], [S41]) use multiple combinations for model building. The reason for

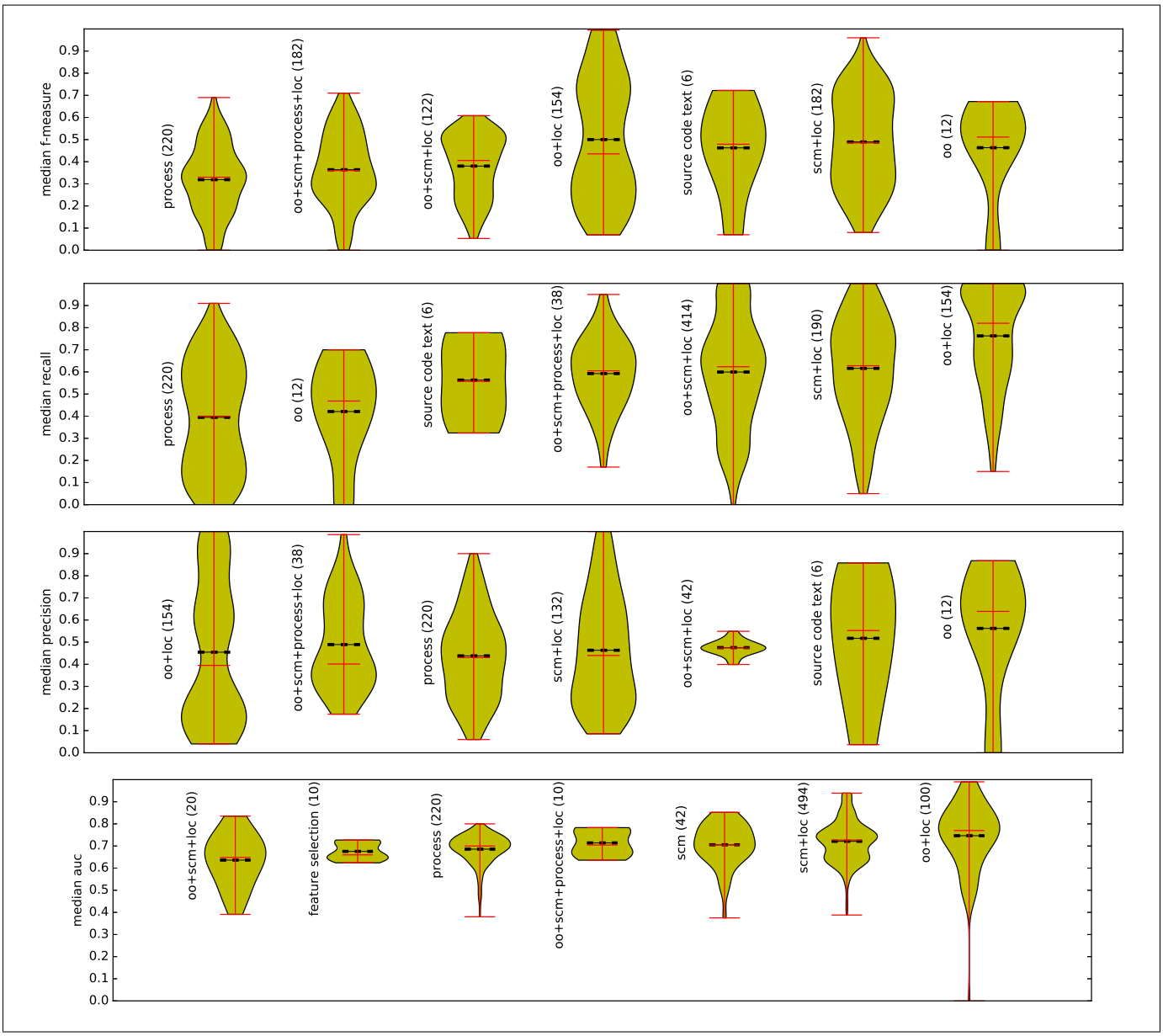


Fig. 3. Median performance of different metric combinations: f-measure, recall, precision and AUC

having multiple combinations is due to the availability of different set of features/metrics in the training and test data sets for each model. For example [S41] uses OO+SCM+Process+LOC for the models trained with the AEEEM datasets and tests the model on SOFTLAB and ReLink dataset suits for which a subset of SCM+LOC are available. Also [S37] uses feature matching as a part of their proposed approach, i.e., the heterogeneous defect prediction model that can make predictions using training and test datasets with different sets of metrics.

Five studies [S1], [S7], [S35], [S39], [S46] have used only one set of metrics (OO, source code text, SCM or process metrics). It is important to note that LOC is being used as a part of the combinations in 23 of the studies. Likewise, SCM and OO are used as a part of the metrics combinations in more than half of the studies (21 and 17 respectively). Lastly, two continuous model studies have mainly used Process and OO metrics when constructing prediction models.

3.1.1 *Performance in relation to independent variables*

Figure 3 presents the violin plots for the performance trends of the prediction models with respect to the independent variables used in each study. The violin plots in this figure are for median f-measure, precision, recall, and AUC values.

The plots show that the performance of the defect prediction models vary in relation to the independent variables. OO and SCM+LOC have the highest median values for f-measure, while Process metrics has the lowest. Also based on equal number of data points, SCM+LOC based models tend to perform better than OO+SCM+Process+LOC in terms of median f-measure. Even though Process metrics could be useful, their combination with other metric sets seems to decrease the performance of prediction models. The study by Kamei et al. [S39] which uses process met-

TABLE 9
Software metrics and studies

Type of metrics	Studies	#Predictions
Source code text	[S1]	12
OO	[S7], [S35]	24
SCM	[S40], [S46]	43
Process	[S39]	220
OO+LOC	[S2], [S8], [S11]	254
SCM+LOC	[S4], [S13]–[S16], [S20], [S24]–[S26], [S37], [S41], [S43]	879
OO+SCM+LOC	[S3], [S5], [S13], [S18], [S23], [S27], [S37], [S40], [S42], [S44]	626
OO+SCM+Process+LOC	[S24], [S37], [S41]	194
Multiple Combinations	[S13], [S24], [S37], [S40], [S41]	

rics has contributed to these low performance values. OO and OO+LOC as well as SCM+LOC have good performance individually but not when combined, i.e., in Figure 3 OO+SCM+LOC appears to the left of all three of them. One contributor to the good performance of OO+LOC in this case could be its good recall performance. It has the highest median recall value while having arguably the worst precision. Source code text has a good performance and it favours precision more than recall in comparison to the other counterparts. In most cases, one can see a degree of instability in the plots especially with respect to the precision and recall.

In the AUC plot, OO+SCM+LOC has the lowest median value; lower than Process metrics which has the lowest f-measure and recall medians. Despite having the highest median value, OO+LOC does not seem to be very stable in comparison with the other metric suites. Finally, the stability of SCM+LOC seems to be fine, considering the huge amount of data contributing to it with respect to the other metric sets in the plots.

3.2 Theme: Modelling Techniques

Almost all techniques are single objective. NB (43%) and LR (32%) are the most widely used learning techniques in CPDP. NN, SVM, and DTree based models achieve the highest median f-measure in CPDP ($\approx$ 0.5). Despite its popularity, naïve Bayes (NB) technique seems to have an average performance. Performance of the ensembles varies greatly when assessed by f-measure vs. AUC. Table 10 shows the frequency of learners used in CPDP studies.

Various modelling techniques have been used by the categorical studies. The majority of studies have used only base learners to build prediction models. There are few studies which use various optimisation techniques on top of the base learners. Table 10 shows the number of cases for which the base learners are used as they are, the number of cases where they are used in the context of an optimisation/ensemble technique, and the studies in which they appear (either as they are or in ensembles).

Naïve Bayes (NB) is the most commonly used modelling technique in the categorical studies (513 data rows in 12 studies). Logistic Regression (LR) is the next frequently used modelling technique (455 data rows in 9 studies). Other base learners such as Decision Tree (DTree), Support Vector Machine (SVM), and Random Forests (RF) are also used to build prediction models in a considerable number of cases. Note also that these modelling techniques are being used as the underlying learner for some of the optimisation techniques as well. For example, 160 of the predictions in the extracted data use boosting naïve Bayes, a subset of the 184 cases for which NB is used in ensembles. LR has the highest number of predictions when used in the context of ensembles (344 data rows). Study [S4] has the highest amount of contribution to the ensemble LR predictions by providing 264 of 344 data rows.

Four studies have used the boosting approach in their reported models ([S18], [S26], [S42], [S44]). Two of these studies ([S18], [S44]) use NB and boosting as a part of their proposed CPDP approaches. Zhang et al. [S42] build their models with both NB and DTree and boosting. The study by Ryu et al. [S26] considers the applicability of SVM learner with boosting in the context of CPDP.

Among the selected studies, only two ([S11], [S14]) utilise a search based method to create a defect prediction model. In [S14], a genetic programming method is proposed in which optimal mathematical expressions that could best fit the training data are sought for. Further, [S11] is the only study which has proposed a search based multi-objective approach (MODEP) to CPDP. They used NSGA-II algorithm and targeted the cost effectiveness as their goal.

Some studies ([S4], [S5], [S8], [S42], [S43]) have used other ensemble techniques in their experiments. The ensemble of weak learners is used in [S4] to avoid over-fitting. In [S8], the authors combine six different learning algorithms using two underlying learning techniques, namely LR and BN. They propose to use a meta-learner called Combined Defect Predictor (CODEP) which creates a prediction model from the prediction results of multiple other prediction models. In a similar manner, Zhang et al. [S42] investigated seven ensemble algorithms composed of various modelling techniques, six of which are used to benchmark CODEP. In [S5], the authors generate a meta learner model using the decision tree learner based on the distributional characteristics of the source and target datasets and the prediction result from the source to the target (success, failure) from another learner (Decision Table in this case). They use this tree to predict the performance of prediction models trained with a specific source set and tested on a particular target set. Ryu et al. [S43] proposed a Hybrid Instance Selection technique built on top of the Nearest-Neighbour algorithm to capture the local knowledge in data by NN and the global knowledge by NB.

3.2.1 Performance in relation to modelling techniques

Figure 4 shows the model performances in relation to the modelling techniques. As per the f-measure plots, models based on NN, SVM, and DTree seem to perform well while the models using LR and RF do not seem to achieve overall good performances. More notably, Simple NB performs as good as most of the models, yet it is not among the top three techniques with respect to the median values. The performance in terms of precision and recall also confirm this conclusion as NB achieves a medium performance for

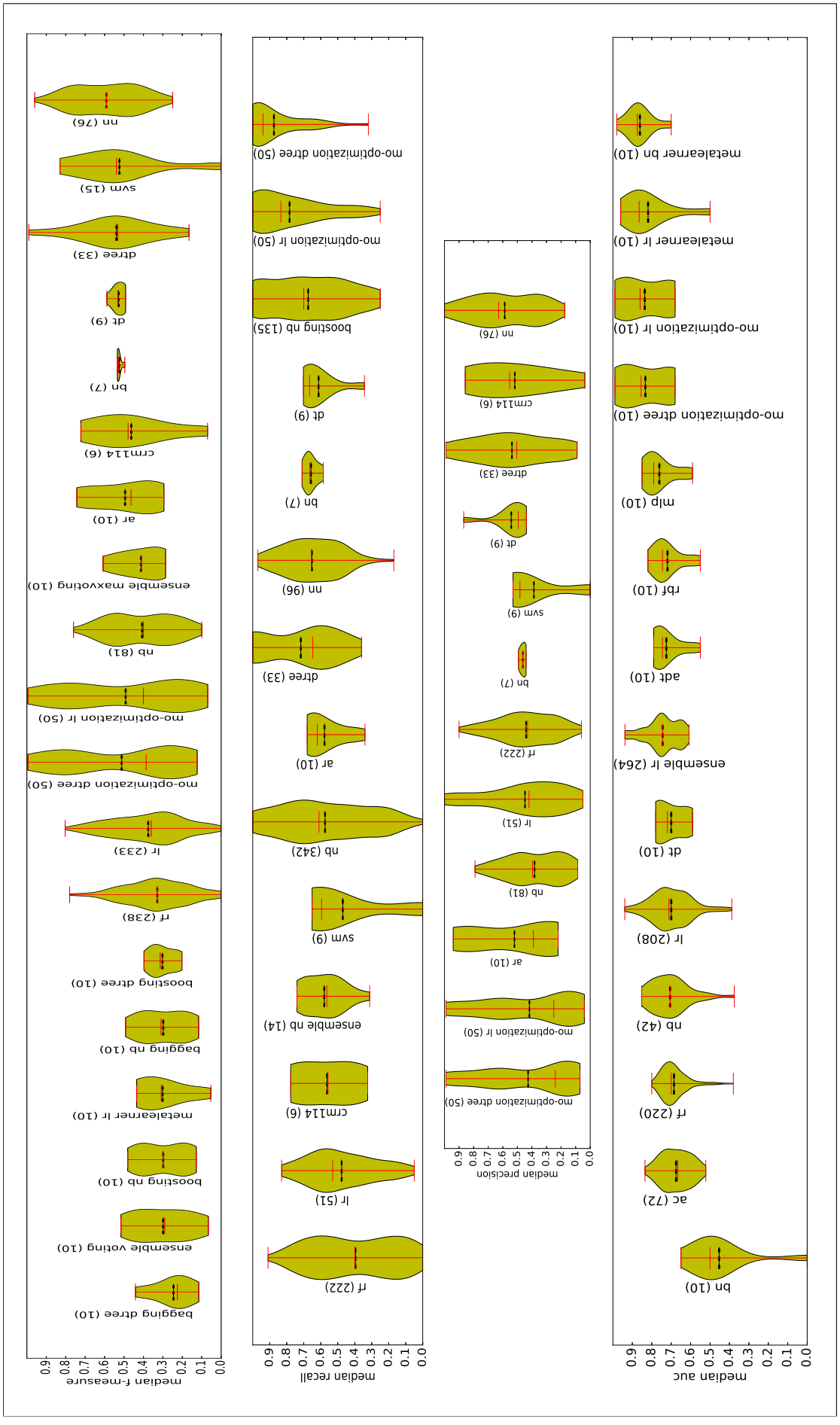


Fig. 4. Performance of the modelling techniques: f-measure, recall, precision and AUC

TABLE 10
Popularity of the modelling techniques

Learner	#As is	#Ensemble	Studies
Naïve Bayes	513	184	[S3], [S7], [S13], [S18], [S20], [S25], [S26], [S40], [S41], [S43], [S44], [S46]
Logistic Regression	455	344	[S3], [S4], [S8], [S11], [S15], [S24], [S35], [S37], [S41]
Random Forest	248		[S7], [S26], [S39], [S41], [S42]
Nearest Neighbour	102		[S7], [S27], [S41]
Association Classifier	72		[S16]
Decision Tree	43	102	[S2], [S3], [S7], [S11], [S26]
SVM	34	10	[S3], [S7], [S23], [S26], [S41]
Genetic Programming	21		[S14]
Decision Table	19		[S3], [S7], [S8]
Bayesian Net	17	10	[S3], [S8]
CRM 114	12		[S1]
Part	10		[S26]
ADT	10		[S8]
IBk	10		[S26]
RBF	10		[S8]
Max voting	n/a	10	[S42]
MLP	10		[S8]
Association Rules	10		[S11]
Average Voting	n/a	10	[S42]

both. While MO-optimisation models have very high recall values, they lack the good precision and they seem to be unstable as well. This instability in precision could be one of the reasons for their overall unstable performance. They have the highest median recall values while having the lowest precision. Surprisingly, the ensembles under perform most of the other counterparts in terms of f-measure, with the catch that the relevant data mostly comes from only one study ([S42]) for the majority of them.

The Meta-learner BN, proposed in [S8] has the highest median AUC value. Beside that, its stability seems to be much better than the other approaches in this category. The authors claim that combining classifiers can improve the prediction performance as in $\text{CODEP}_{\text{Bayes}}$, a meta-learner on top of Bayesian Networks [S8]. This approach is followed by Meta-learner LR, MO-Optimisation, and MLP. The data for these meta-learners as well as MLP are coming from one study [S8]. Later, [S42] assessed the usefulness of Meta-learner LR with respect to alternative performance measures, i.e., f-measure and Nof20 and concluded that its performance is comparable to other composite algorithms except for MAX voting. In their experiments, Max voting significantly improved the performance of other composite algorithms, one of which is Meta-learner LR ($\text{CODEP}_{\text{Logistic}}$) [S42]. These techniques are followed by RBF, ADT and Ensemble LR. Ensemble techniques seem to affect the performance of LR, one of the most common learning techniques in the studies. LR is improved by both Ensemble LR and Meta-learner LR according to the plots. The data for Ensemble LR comes from the Ensemble of weak classifiers model in [S4]. In contrast with f-measure results, ensembles tend to improve CPDP according to the AUC plots.

To have a better understanding on how different learners behave according to different performance measures, the results for PF and Balance are also assessed. Similarly, NN provides the best overall median balance value and the lowest median PF confirming its good performance with respect to multiple measures. Boosting has not improved NB dramatically in terms of the median, but the stability is improved. Of course one might notice a performance ceiling with Boosting NB compared with NB as is, considering the number of data points that contribute to them. Boosting is related to both higher recall and higher Probability of False alarm. In that sense, the performance of Boosting NB is similar to the simple NB technique as it has the worst median probability of false alarm, making it potentially less suitable overall for building prediction models when the precision of the model is targeted.

3.3 Theme: Performance Evaluation Criteria

No set of common measures is shared by all studies. Recall (75%), probability of false alarm (43%), precision (39%), f-measure (39%) and AUC (36%) are the most frequently used measures, respectively. Table 11 shows all performance measures reported in CPDP studies along with their frequencies.

No common set of performance evaluation criteria is used to gauge the performance of different CPDP models. Table 11 presents the most widely used performance measures used in the categorical studies. Last column of the table represents the studies in which the performance measure is used. To be able to make the comparisons, additional measures are calculated whenever possible. These include the cases in which average and median values are reported as well. For example in [S11], f-measure values are calculated from the reported median recall and precision values. Moreover, some of the less common performance measures such as Overall Error Rate, G-mean, H-measure, and cost related measures to mention some, are labeled as miscellaneous in the table.

Even though most of the studies use multiple performance evaluation measures, a few studies have used only one ([S4], [S24], [S37], [S46]). Three of these studies have used AUC and one study has used f-measure. A depiction of the number of studies for each performance evaluation measure is shown in Table 11. According to the table, recall is the most commonly used performance measure (21

TABLE 11
The definition and usage of different performance measures across the categorical studies

Measure (#Studies)	Description	Definition	Studies
Precision (11)	The proportion of the predicted positive cases that were correct	$\frac{TP}{TP+FP}$	[S1]–[S3], [S5], [S7], [S11], [S15], [S16], [S23], [S39], [S40]
Recall, pd (21)	The proportion of positive cases that were correctly identified	$\frac{TP}{TP+FN}$	[S1]–[S3], [S5], [S7], [S11], [S13]–[S16], [S18], [S20], [S23], [S25]–[S27], [S39]–[S41], [S43], [S44]
Accuracy (2)	Proportion of correctly classified units	$\frac{TP+TN}{TP+FP+TN+FN}$	[S1], [S16]
Probability of False Alarm, pf (12)	Proportion of non-faulty units incorrectly classified as fault-prone	$\frac{FP}{FP+TN}$	[S7], [S13], [S14], [S18], [S20], [S25]–[S27], [S40], [S41], [S43], [S44]
AUC, Area Under the Curve (10)	The area under the receiver operating characteristics curve. Independent of the cutoff value		[S4], [S8], [S11], [S16], [S25], [S26], [S37], [S39], [S40], [S46]
F-measure (11)	Harmonic mean of precision and recall	$\frac{2\times Precision\times Recall}{Precision+Recall}$	[S1], [S3], [S5], [S7], [S16], [S24], [S25], [S39]–[S42]
G-measure (3)	Harmonic mean of pd and (1-pf)	$\frac{2\times pd\times(1-pf)}{pd+(1-pf)}$	[S18], [S27], [S40]
Balance (4)	The Euclidean distance from the (pd, pf) point to (pd=1, pf=0) in the ROC curve	$1-\frac{\sqrt{(0-pf)^2+(1-pd)^2}}{\sqrt{2}}$	[S13], [S20], [S43], [S44]
MCC (3)	A compound measure considering all true and false positives and negatives	$\frac{TP\times TN-FP\times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$	[S18], [S40], [S41]
Miscellaneous	FN rate, G-mean, H-measure, Cost, Hubert Stat. Procedure	N/A	[S1], [S14], [S26], [S35], [S44]

studies) followed by the probability of false alarm, precision, f-measure, and AUC.

Cost related measures such as inspection cost ([S8], [S11]), cost effectiveness ([S42]), and expected cost of misclassification ([S14]) were used in multiple studies and their importance were justified. The inspection cost is based on LOC and is an approximation of the effort needed to analyse the classified faulty units. On the other hand, cost effectiveness deals with the number of faulty units detected when inspecting first n% of lines of code. Finally, two continuous models studies used Nagelkerke's R2, precision, and RMSE (Root Mean Square Error). The descriptions and definitions of these measures can be found in the online appendix.

3.4 Theme: CPDP Approaches

In this section, an overview of the different approaches in CPDP will be presented. Different studies have used different processing methods to build their models. In the following, these methods are presented.

3.4.1 Data Related Issues and Approaches

A summary of all observed issues with a mapping to how they are addressed is provided in Table 12. Among them, data heterogeneity is the most frequently addressed CPDP specific issue (68% of studies), while the remaining ones are not unique to CPDP, e.g., class imbalance (25%), noise in data (7%), redundant (14 %), or correlated features (14%).

The majority of the categorical studies have addressed different data related issues such as data heterogeneity, class imbalance, highly skewed data, etc. as a part of their proposed approach. Table 12 describes these data related issues and explains how they are addressed in different studies. The last column of the table presents the studies addressing each issue. If an approach in a study is specifically named, the name is mentioned along with the study. Three studies ([S1], [S7], [S46]) are absent from this table as they have not addressed any issue in any of their reported models.

Class imbalance. Usually, defects are not distributed evenly and the number of defective modules is much less compared to the non defective modules. This issue is called the class imbalance problem [13]. Data imbalance greatly impacts the performance of the classification models as well as decreasing their generalisability [S44]. Learning minority class is difficult when a classifier is trained on imbalanced data [1]. Therefore, the classifier is skewed toward the majority class resulting in a lower rate of detection. This issue is addressed by several studies using different methods such as data re-sampling(RS) ([S18], [S26], [S39], [S44]), instance re-weighting(RW) ([S23], [S25]), and selective learning strategy ([S43]).

Data heterogeneity. The similarity of source and target data distributions is believed to have a large effect on the outcome of predictions. This problem is sometimes referred to as data heterogeneity in software projects ([S11], [S35], [S37], [S41]). As expressed by Canfora et al. [S11], software projects are often heterogeneous and they have different software metric distributions. Moreover, heterogeneity is believed to be affected by certain context factors such as domain, size, and programming language to name some [S15]. Many machine learning algorithms work well under the assumption that source and target data are drawn from the same distribution [33]. This assumption typically holds for WPDP due to the nature of the data, but it might not hold for CPDP [S24] and therefore, CPDP does not necessarily work in all situations [S9], [S20]. Various

methods such as data transformation (DT), filtering (F), feature matching, and normalization (N) are proposed in studies to tackle the problem.

Highly skewed data. As presented in Table 12, few studies have addressed the skewed data problem [S4], [S13], [S20], [S43]. This issue is sometimes one of the contributors to poor model performances [45]. Menzies et al. [41] pointed out that spreading the skewness of the data evenly can cause a significant improvement in the effectiveness of the data mining models. Logarithmic transformation is used in these studies for addressing the issue.

Irrelevant and redundant features. A set of studies ([S3], [S15], [S37], [S42]) use feature selection (FS) technique to identify and remove irrelevant and redundant features. Reducing the dimensionality of the data, this process improves the effectiveness of the learning algorithm [30].

Continuous data. The values of numerical features (integer or real) or those taking on linearly ordered range of values are considered as continuous features [34]. However, some classification algorithms assume nominal feature spaces during the learning process and hence, making these algorithms inapplicable in some cases [46]. One approach toward this problem is the discretization (D) process. This approach was used by two studies [S16], [S25].

Privacy. The privacy issue arises when the confidentiality of the data is a concern for the project owners. This issue in turn causes the owners to rarely contribute to the pool of available data even though such data might contribute to further research efforts. Privacy is considered by Peters et al. [S27] in the context of CPDP. They obfuscate the data in order to hide the project details when they are shared among multiple parties.

Collinearity among metrics. Collinearity and multi-collinearity among independent variables might decrease the effectiveness of the prediction models. This issue happens when two or more independent variables are strongly correlated. Calculating Pearson and Spearman correlations are two of the most common ways of detecting such issues. Dimensionality reduction and PCA [66] (Principal Component Analysis) in particular are other approaches to tackle strongly correlated features. PCA is being used by [S31], [S33], [S34], [S36] in the primary studies. In [S39] the authors remove highly correlated metrics (Spearman $\rho > 0.8$) to deal with the problem of collinearity.

Noise in data. Data instances with excessive deviation from the acceptable range of the feature space are known as outliers [30]. Classifiers trained on noisy data are expected to be less accurate [47]. Two of the studies among the others try to overcome this problem, one of which performs outlier removal [S43] and the other does noise filtering [S27].

Aforementioned data issues are tackled by proposing and using various data processing methods in the selected set of categorical studies. We call these approaches column and row processing methods according to the way they consume the raw data. The studies that deal with features/metrics processing are members of the column processing group (e.g., transfer learning, feature selection). Likewise, the studies which perform data processing on instances are considered in the row processing methods group (e.g., filtering, data re-sampling). The categorisation of the studies into column and row processing groups is illustrated in Tables 13 and 15. Overviews of the most common processing methods are presented in the following.

3.4.2 Column Processing Methods

Column processing techniques addressing data related issues, and the studies they appear in, are listed in Table 13. Applying normalisation (29%) and log-filtering (25%) to data features are frequent practices. Other transformations, listed in Table 14, mostly modify data features in both training and test sets, to address data heterogeneity issue.

Data transformation. As shown in the Table 13, transformation technique is used in several studies. The main premise of Transfer learning in general and transformation in particular is to neutralise the effect of heterogeneity between source and target data [S24] by extracting and learning knowledge from a dataset and applying that knowledge during the training and prediction processes. Table 14 shows a list of the studies in this category.

Metric Compensation in [S2] adjusts target data by considering the average values of the features for both training and test datasets. The study conducted in 2013 by Nam et al. [S24] proposed another transformation approach called TCA+ which tries to make feature distributions in source and target projects similar by transforming both source and target data to the same latent feature space. The universal defect prediction model proposed by Zhang et al. [S40] offers a context-aware rank transformation to address the difference in the distribution of the data. This model is trained with a large set of diverse projects from different contexts and the variations in the distribution of metrics values are high. Another CPDP study [S41], utilises an approach called Canonical Correlation Analysis (CCA) which tries to build prediction models with datasets that have different sets of metrics. They transform both training and test datasets to a new feature space in a way that they would have an equal number of features in the resulting datasets. Cruz et al. [S35] use the information form Mylyn dataset and apply a log transformation on the data.

Feature selection. Feature selection is another column processing method used by few CPDP studies ([S3], [S15], [S37], [S42]). In this case, features are the metrics that are utilized to build defect prediction models. As denoted by He et al. [S3], performing feature selection is a sensible method to deal with large number of features. This method identifies a subset of the features which can possibly deliver better predictive performances. Feature selection can be categorized into two groups: filters and wrappers. With filters, irrelevant features are removed from the feature set before it is used by a learning technique. On the other hand, wrappers use the feedback from the

TABLE 12
Data related issues and the proposed methodologies in CPDP

Issue	How the issue is been addressed	Studies
Class imbalance	Re-sampling	DTB [S18], VCB-SVM [S26], JIT [S39], TCSBoost [S44]
	Re-Weighting	[S23], TNB [S25], [S26], [S41], [S44]
	Selective learning	HISNN [S43]
Data heterogeneity	Data Transformation	Metric compensation [S2], TCA+ [S24], Log Transformation [S35], Universal model [S40], CCA+ [S41]
	Filtering	NN-filter [S13], [S20], HISNN [S43], [S5], [S18], [S23], [S25], [S27], [S41], [S44]
	Normalization	[S4], CODEP [S8], MODEP [S11], [S14], [S23], TCA+ [S24], VCB-SVM [S26], CCA+ [S41] , TCSBoost [S44]
	Metrics matching	HDP [S37]
	Clustering	[S11], [S23]
High Skeweness	Logarithmic transformation	[S4], NN-filter [S13], [S20] , HISNN[S43],[S18], [S23], [S25], [S44]
Privacy	Multi-party data sharing	LACE 2 [S27]
Irrelevant and redundant features	Feature selection	TOP-K [S3], [S15], HDP [S37], Composites [S42]
Continuous data	Discretization	[S16], [S18]
Collinearity among metrics	Remove highly correlated metrics	JIT [S39]
	Principal Component Analysis	PCA [S31], [S33], [S34], [S36]
Noise in data	Outlier removal	HISNN [S43]
	Noise filtering	LACE 2 [S27]

TABLE 13
Column processing methods

Processing method	Studies
Data Transformation	[S2], [S24], [S35], [S40], [S41]
Feature selection	[S3], [S15], [S37], [S42]
Normalization	[S4], [S8], [S11], [S14], [S24], [S26], [S41], [S44]
Log-filter	[S4], [S13], [S18], [S20], [S25], [S43], [S44]
Discretization	[S16], [S25]

TABLE 14
Data transformation approaches

Data transformation approach	Studies
Modify target data similar to source data	[S2]
Modify both source and target data	[S24], [S35], [S40], [S41]

learning algorithms to decide which feature(s) to include in building a classification model [S3]. The authors in [S3] recommended to use filtering approach, if the target is to achieve high recall values. If both higher recall and f-measure are required, their recommended approach is TOP-K approach. Further, they suggested to use minimum feature subset if appropriate precision or high f-measure is required. Nam et al. [S37], have used various feature selection approaches such as gain ratio, chi-square, relief-F, and attribute significance evaluation. Their results confirmed the usefulness of feature selection approaches toward building better defect prediction models. Yu et al. [S15] performed score test to select the best set of features. Finally, the authors in [S42] mention the use of the feature selection technique, but its details has not been reported.

Normalisation. As shown in Table 12, normalisation is being widely used in CPDP. Two types of normalisation, namely min-max and z-score, are commonly used of which z-score normalisation is more popular.

$$Z = \frac{X - \mu}{\sigma} \tag{1}$$

Equation 1 shows the formal definition for z-score normalization method. Here, μ and σ are the mean and standard deviation of X respectively. Study [S24] uses both z-score and min-max normalizations. Min-Max is formally defined using the following equation.

$$X_{new} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{2}$$

Discretization. The number of feature values is one of the contributors to the speed and effectiveness of different machine learning algorithms [30]. As a solution, some studies have used discretization, the process in which the number of possible feature values are decreased for continuous features [30]. Both [S16] and [S25] had used Minimum Description Length (MDL)-based discretization schema proposed by Irani et al. [34]. The supervised discretization algorithm proposed by the same authors uses entropy minimization heuristic for discretizing the range of a continuous-valued features to multiple intervals.

Log-Filtering. To tackle the skewness in data, some studies ([S4], [S13], [S20], [S43]) have converted the feature values by applying the logarithmic function to the original values. The rationale behind this argument is that the distribution of the log-transformed feature values better matches the assumptions of the normal distribution [S20].

3.4.3 Row Processing Methods

Table 15 lists the row processing techniques, which targets data instances as opposed to data features, along with the studies in which they appear. Filtering (36%) is the most frequently used method to address data heterogeneity via row processing.

Filtering. Turhan et al. [S20] proposed to use filtering method for CPDP. The rationale behind filtering method is that similar instances (according to the defined similarity measure) can potentially be better predictors for a corresponding test set. Therefore, they applied the k-Nearest Neighbour (k-NN) algorithm to training dataset in order to find the similar (relevant) instances for their

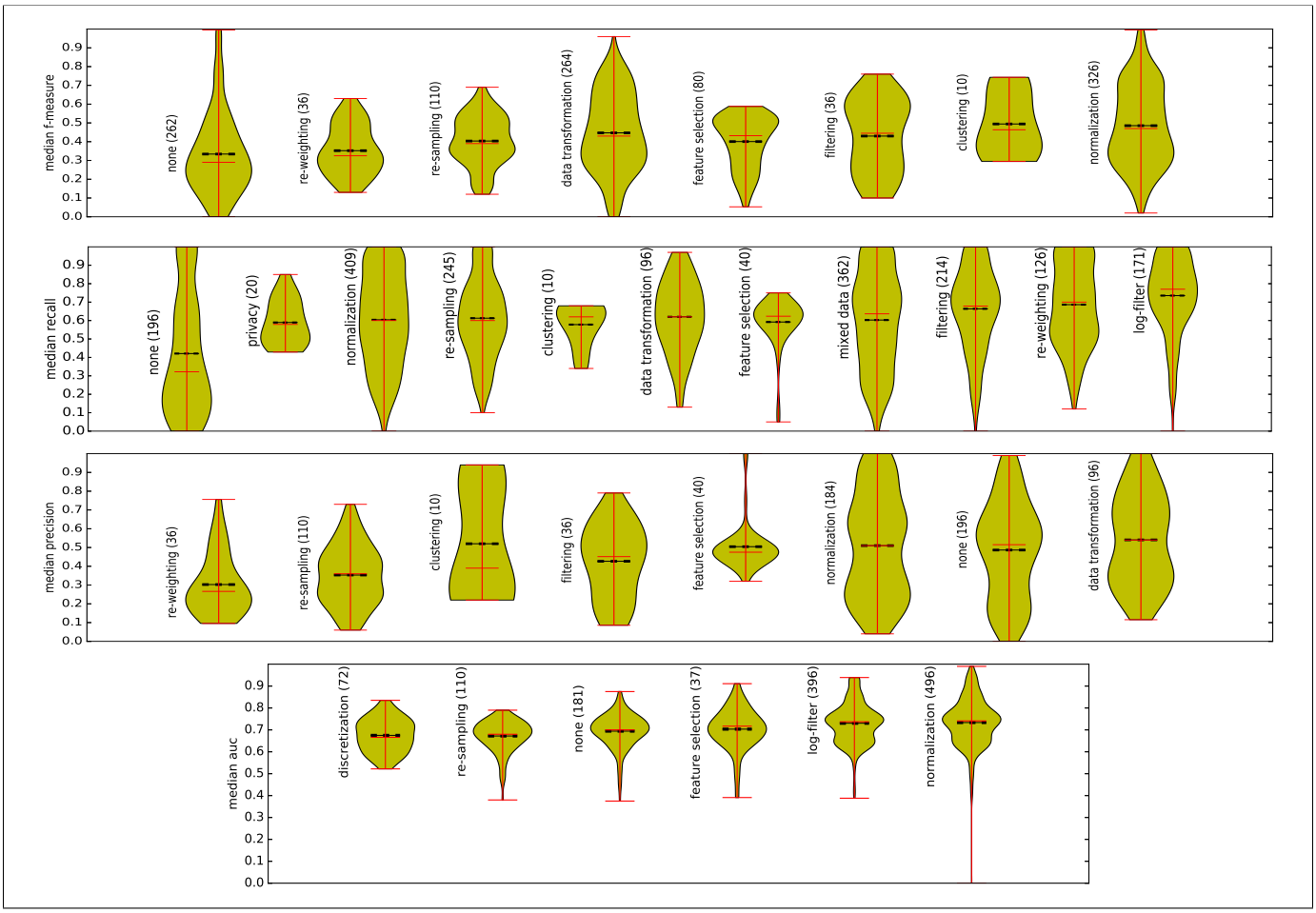


Fig. 5. Performance of the individual data approaches: f-measure, recall, precision and AUC

TABLE 15
Row processing methods

Method	Studies
Filtering	[S5], [S13], [S18], [S20], [S23], [S25], [S27], [S41], [S43], [S44]
Re-weighting	[S18], [S23], [S25], [S26], [S41], [S44]
Re-sampling	[S18], [S26], [S39], [S44]
Clustering	[S11], [S23]
Privacy	[S27]

test sets. They used Euclidean distance as the distance measure and k = 10 as the number of neighbours for k-NN. Later in 2013, they confirmed the usefulness of the filtering method in CPDP [S13]. The relevancy filtering proposed by Turhan et al. was able achieve high recall values while also increasing PF. In other words, this filter favours recall more than precision. Recently published study by Chen et al. [S18] also takes advantage of filtering method to avoid irrelevance instances in source dataset. A similar filtering method was proposed by Peters et al. [S27]. In this study, instead of using k=10 in NN-filter, "Best (K)" procedure introduced by Kocaguneli et al. [48] was utilised for determining k. HISNN [S43] also employs filtering to further reduce pf and achieves higher pd values. In HISNN, NN filtering based on a search ring of Min-Ham radius around each test data instance was performed. He et al. [S5] investigated the exhaustive dataset selection approach by creating a metalearner from the results of a large number of predictions (160K) and the distributional characteristics of training and test datasets. The goal in their approach was to filter out irrelevant datasets based on the distributional characteristics so that the most suitable datasets are selected during the training process.

Sampling/Re-sampling. Typically data sampling is referred to as the process of modifying imbalanced datasets by some mechanisms in order to achive a desired distribution of data [25]. To that end, two sampling techniques are commonly used: Over-sampling and under-sampling [26]. In over-sampling, the minority class samples are randomly selected and their copies will be added to the original dataset. Conversely, in under-sampling, the instances from the majority class are randomly discarded to achieve a balanced distribution [26]. Both over-sampling and under-sampling methods are utilised in [S26] and [S44], while [S18] uses over-sampling only and [S39] uses under-sampling only. Synthetic minority over-sampling (SMOTE) had been used in two CPDP studies [S18], [S44] as the oversampling technique. SMOTE generates new artificial minority class instances synthetically based on feature similarities to deliver more balanced class distribution [27]. Tomek links introduced in [35] is the technique of choice for

under-sampling in the study by Ryu et al. [S44].

Weighting/Re-weighting. Not all training data provide the same level of information in the prediction process. To increase the effect of particular instances on the overall model, instances are weighted according to different approaches. Weighting is also one of the methods of dealing with imbalanced data. In this case, the minority class instances are weighted in a way that a more balanced distribution of data is achieved. Further, Re-sampling has a close relationship with weighting. Two approaches are mainly considered for data weighting. If the classifier of choice supports instance weighting then weighting will be handled by the classifier. An example is the weighted k-NN classifier. Otherwise, one should re-sample the data, i.e., those instances with more weights could appear multiple times or modified instances generated form them could be added to the dataset. TNB in [S25], proposed to use gravitational weighting approach toward the data and modified NB learner to account for the assigned weights. This approach was later utilized in multiple models in the studies in either their proposed approach or in the benchmark approaches [S18], [S26], [S41], [S44]. Equally weighting the data from both classes was considered by [S23]. In this method, the data is modified in a way that the sum of the weights of the defect-prone instances is equal to the sum of the weights for non defect prone instances.

Clustering. Different clustering algorithms at different levels of granularity are used by some of the categorical studies. Herbold [S23] used EM clustering in order to create groups of characteristics vectors from the candidate source and target data for source data selection that are located in the same cluster as the target data. NN-clustering was another approach utilized in [S23]. In this case, target data similar to the source data were selected with favour to distributional characteristics. In [S11], Canfora et al. used a "local" predictor [51] based on association classification as classifier and MDS as clustering algorithm which acts as a benchmark for their MO prediction model.

While the above approaches are considered by some studies, a group of them discarded the use of any particular row processing methods. Both [S1] and [S7] investigate the applicability of different types of metrics in CPDP. Source code text in [S1] and OO metrics in [S7] were used to build the CPDP models. Similarly [S46] also targets the metrics used in CPDP, specifically focusing on design metrics which can be collected in early development stages.

3.4.4 Other CPDP Approaches

The majority of the categorical studies have mentioned specific names for their proposed approaches such as TNB[S25], DTB[S18], VCB-SVM[S26], and CODEP[S8]. However, not all of these approaches manipulate the data. Moreover, in the data oriented approaches, the models are usually comprised of one or more data processing methods listed in Tables 13 and 15.

We detected several approaches in categorical studies as described earlier. We noticed that the majority of the models were built as benchmark methods for the proposed CPDP approach(es) in the primary studies. For example, [S44] builds multiple prediction models using boosting and re-sampling as benchmarks for their proposed transfer cost sensitive boosting approach. Similarly in [S18], [S25], [S43] simple NB model with log transformation was used as benchmark for DTB, TNB, and HISNN respectively.

The majority of the studies (19 studies) present some models without any data specific approach. Finally, one study ([S27]) presents a data transformation approach that targets privacy issues in the context of CPDP. Beside the discussed approaches, three studies had reported the use of mixed data [S13], [S18], [S44]. Additionally, one study [S11] has used local methods (as a benchmark) proposed by Menzies et al. [51]. These two approaches are described briefly.

Mixed data. In the mixed data approach, CP data is combined with a portion of WP data and the defect prediction model is constructed on the new dataset [S13]. Turhan et al. [S13] evaluated the effects of mixed project data on the prediction performances. They concluded that, when limited project history data is available, defect prediction models based on the mixed data are useful. More specifically, they could perform even as good as full WP models [S13] in some cases. DTB [S18] also uses CP data mixed with 10% of available WC data. This approach trains a predictor based on boosting and mixed cross and within project data. Later, the usefulness of mixed data in the presence of the class imbalance issue was explored by Ryu et al. [S44]. During their experiments, the authors combined five, ten, and 25 percent of WP data with CP data for model construction and concluded that there is no significant performance difference between using 25 and five percent of WP data. In other words, the model with five percent of WP data performs as good as the one using 25 percent [S44].

Local Methods. Menzies et al. [51] proposed to use local models for defect prediction (and effort estimation). Before making predictions, the training and test set instances are clustered into n groups and the training instances belonging to each group are used to extract rules which minimise the defect numbers in each cluster. They concluded that the results achieved by the local (cluster) models are typically better than those from the global (cluster) models. The local models approach is considered in [S11] as a benchmark to their multi-objective approach. The data from the local models benchmark appear under clustering category and association rule learning technique as used in [S11].

3.4.5 Performance in relation to data approach

Data oriented approaches seem to improve f-measure through improvements in recall with no visible trend on precision. Higher end of the performance spectrum always includes Normalisation as a data oriented approach, so it's safe to recommend the use of Normalisation at least. In the lack of any data approach, e.g., using data as is, one may expect low recall values, though higher precision.

We assessed the impact of various data issues by investigating them in relation with the overall performances.

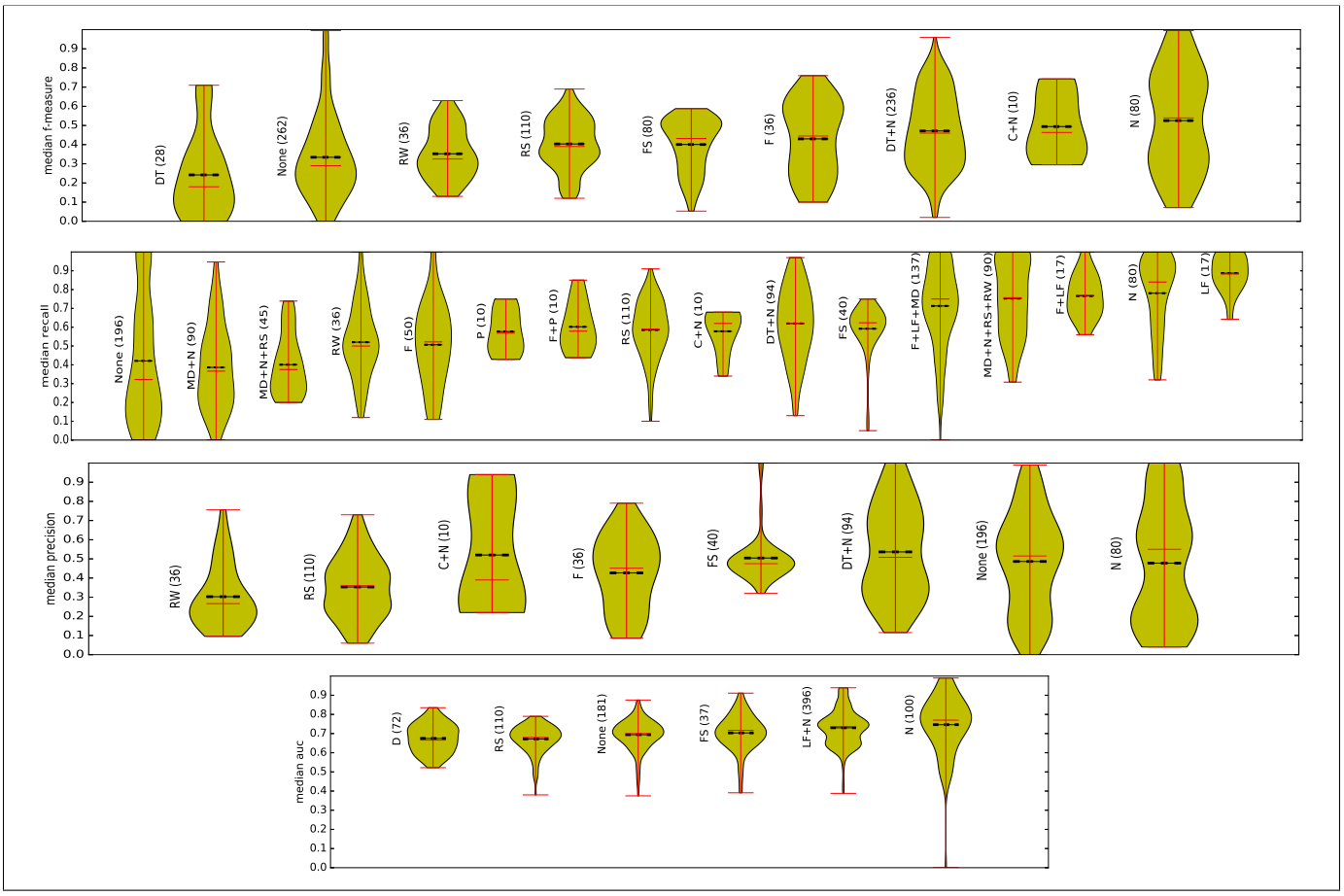


Fig. 6. Performance in relation to the combinations of the data approaches: f-measure, recall, precision and AUC

The results are illustrated in the violin plots in Figures 5 and 6. We investigated the performance in relation to individual approaches as well as their combinations. In the first case (see Figure 5), these individual approaches that are either used independently or as a part of a combination are considered. For the second part, the combinations of the approaches as they are proposed in the studies are used (see Figure 6).

According to the plots in Figure 5, Normalization (N) has the highest median f-measure while its stability is not very good. One might note the high number of values contributing to the plot for Normalization, potentially decreasing its stability. Normalization is a very common technique among the studies and hence the lack of stability was expected to some extent. Clustering is next in line with respect to f-measure with the lowest number of data points (10). Filtering has a similar median value to the other two while being more stable than Normalization. Re-sampling and Re-weighting have the lowest median values among the data approaches. The lowest median value belongs to none, the case where no data approach is considered.

The recall plots reveal that most of the data approaches are recall oriented. Using no data approach has the lowest recall value while having the second best median precision. Moreover, clustering seems to favour a balance between precision and recall as in both cases, the performance is medium to low but its f-measure performance is better than most other approaches.

Considering the AUC values, normalization and log-filter have the highest median performances. A medium performance is achieved when no data approach is used and according to the plots, this performance is better than re-sampling and discretization.

The overall performances suggest that the use of data approaches can potentially lead to better predictions in most cases. To further investigate the relation of data manipulation in CPDP performance, we checked the different combinations of them as used in different studies. Figure 6 illustrates the performance of different combinations used in the studies.

In case of the f-measure plots, N (Normalization) has the highest median performance. Other approaches are C+N (Clustering+Normalization) followed by DT+N (Data Transformation+Normalization) and F (Filtering). These three combinations with respect to the median f-measure values, all use Normalization in their settings which has also the second best median recall performance among the different data approach combinations.

We observed that the data approaches tend to favor recall more than precision. Not surprisingly, the lowest recall performances happen when no data approach is considered in this case as well. LF (Log-Filter) is used in three cases among the top five recall plots. The combinations involving MD (Mixed data) are present in both good and poor perfor-

mances. In fact the two lowest data approach combinations include MD in their settings.

The stability varies more in case of precision compared with that of recall. Except for FS (Feature Selection), all of the plots in the top half, including no data approach at all, nearly cover the whole range. The low stability of Normalisation with regard to f-measure is probably affected by its bad stability in being precise. Meanwhile, Filtering (F) and C+N seem to be more stable than the top three models while having a similar median. The recall based nature of data approaches is observed again in this set of plots as the plots for no data approach have the second best median precision and the worst median recall values.

Finally, in the case of AUCs, N and LF+N receive the top places in the plot. It is important to note that while N provides the best median value, its stability is not very promising. It is also worth noting that using no data approach is not the worst case with respect to AUC.

3.5 Theme: Overall Paper Approach

We categorize the studies based on their main focus for further analysis. Some of them optimize the learning techniques while some other focus on manipulating the data. Metrics are also another target for such models. The study approaches are categorized as follows:

- **Learner**: The focus of this set of studies is on optimising the learning approach and they usually present sophisticated learning methods toward CPDP. Different optimisation/ensemble techniques such Bagging, Boosting, Voting, Meta-Learning, and other ensembles are utilised in this category. Examples of this kind are CODEP [S8], various ensembles in [S42], Multi-objective Optimisation [S11], and ensemble of weak classifiers [S4] among the others.
- **Data**: This set of studies propose to use rigorous methods toward modification or selection of data instances. Dimensionality reduction in TCA+ [S25], mixed data in [S13], exhaustive dataset selection in [S5], and the Universal model in [S39] are some of the examples in this category.
- **Metric**: Studies in this group focus on the metric sets used in the model building. These studies argue that certain kinds or sets of metrics provide more information and, consequently, better performance can be achieved by removing redundant features [S3], [S37], or in some cases proposing new metric suites [S1].

Some studies use a combination of this three while the overall focus of some studies is not clear. For example [S44] proposes TCSBoost, a cost sensitive boosting approach that utilizes mixed data in its settings. Therefore, the focus of TCSBoost model is selected as Learner+Data. Examples of studies with no clear focus are [S7], [S40], [S46] to mention some. Additionally, multiple models in the categorical studies are presented (mostly as benchmarks for the proposed methods) with no clear focus (e.g simple NB).

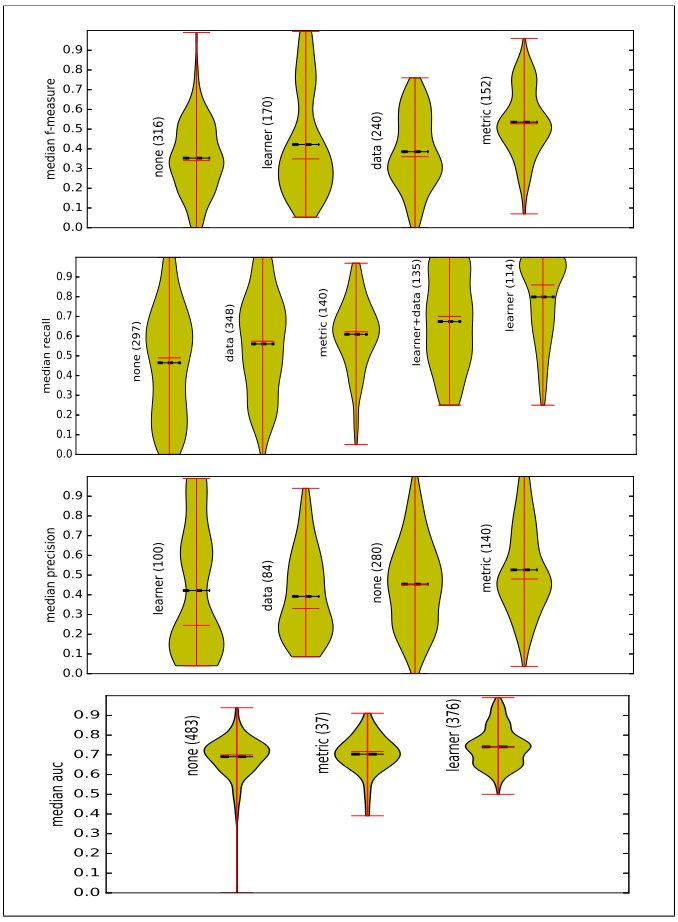


Fig. 7. Performance in relation to the main paper approaches

3.5.1 *Performance in relation to overall paper approach*

Targeting learner, data and metric can potentially lead to better predictions (in terms of f-measure, recall, AUC) and more stable models (with regard to recall). However, the most visible positive trend seems to be the case where targeting learners tend to improve recall.

These information are collected from categorical studies and their performances are presented in violin plots. Figure 7 represents the results of this categorization in the extracted data. In general, it seems that having a CPDP specific research focus on either category (learning, data, and metric) can potentially lead to better predictions than using black box learners with (row-wise or column-wise) unprocessed data. Especially in the case of recall, this approaches can provide more stable models with significant boosts in performances. The models focused on metrics have higher median f-measure and precision values. Finally, median AUC values show a similar trend, revealing that focusing on different aspects of predictions for CPDP can lead to better results, except for precision.

TABLE 16
Datasets used in Studies

Study	Open-Source	Closed-Source/Student/Proprietary
[S1]	JUR(Ant, Eclipse, Jedit, Lucene, Poi, Velocity, Xalan, Xerces)	
[S2]	Sakura, Jedit	
[S3]	JUR(Ant, Camel, Ivy, Jedit, Lucene, Poi, Synapse, Velocity, Xalan, Xerces)	
[S4]		NASA-INDUSTRIAL(CM1, JM1, KC1, KC3, MC1, MC2, MW1, PC1, PC2, PC3,PC4,PC5)
[S5]	JUR(Ant, Camel, Ivy, Jedit, Lucene, Poi, Synapse, Velocity, Xalan, Xerces)	
[S6]		Project A, Project B, Project C
[S7]	Jedit	NASA-INDUSTRIAL(KC1)
[S8]	JUR(Ant, Camel, Ivy, Jedit, Log4j, Lucene, Poi, Tomcat, Xalan)	JUR-PROPRIETARY(Prop-6)
[S9]	Firefox, Eclipse, Derby, Tomcat	MICROSOFT(Internet Explorer, DirectX, IIS, Clustering, Printing, File System, Kernel, SQL Server 2005)
[S10]		TurkCel, NASA-INDUSTRIAL(PC2, PC3, PC4, PC1, KC1, KC2, CM1, KC3, MW1, MC2)
[S11]	JUR(Ant, Camel, Ivy, Jedit, Log4j, Lucene, Poi, Tomcat, Xalan)	JUR-PROPRIETARY(Prop-6)
[S12]		PRJ1-PRJ5
[S13]	JUR(Ant, Camel, Forest, Ivy, Jedit, Log4j Lucene, Pbeans, Poi, Synapse, Tomcat, Velocity, Xalan, Xerces)	JUR-STUDENT(arc, berek, redaktor, nieruchomosci, pdftranslator, serapion, skarbonka, sklebagd, termoproject, workflow, wspomaganiepi), JUR-PROPRIETARY(Prop-1,Prop-2, Prop-3, Prop-4, Prop-5, Prop-6), NASA-INDUSTRIAL(PC1, KC1, KC2, CM1, KC3,MW1, MC2), SOFTLAB-INDUSTRIAL(AR3, AR4, AR5)
[S14]		NASA-INDUSTRIAL(KC1, KC2, KC3, CM1, MW1, PC1, JM1)
[S15]		SOFTLAB-INDUSTRIAL(AR1, AR3, AR4, AR5, AR6)
[S16]		NASA-INDUSTRIAL(PC1, KC1, PC3, CM1, KC3, MW1, MC2, JM1, KC4, MC1, PC2, PC4)
[S17]	JUR(Ant, Camel, Ivy, Jedit, Lucene, Poi, Synapse, Velocity, Xalan, Xerces)	
[S18]	JUR(Ant, Camel, Jedit, Log4j, Lucene, Poi, Synapse, Systemdata, Tomcat, Xalan, Xerces)	JUR-STUDENT(arc, elearn, redaktor), JUR-PROPRIETARY(Prop-6)
[S19]	Jruby, ArgoUML, Eclipse	
[S20]		NASA-INDUSTRIAL(PC1, KC1, KC2, CM1, KC3, MW1, MC2), SOFTLAB-INDUSTRIAL(AR3, AR4, AR5)
[S21]	Axis2, CXF, Camel, Cayenne, Derby, Lucene, OpenEJB, Wicket, XecresJ	
[S22]	JUR(*)	JUR-PROPRIETARY(*), JUR-STUDENT(*)
[S23]	JUR(Ant, Camel, Ivy, Jedit, Log4j, Lucene, Poi, Synapse, Tomcat, Velocity, Xalan, Xerces)	JUR-STUDENT(arc, redaktor)
[S24]	RELINK(Apache HTTP Server, Safe, Zxing), AEEEM(Equinox, JDT Core, Lucene, Mylyn, PDE UI)	
[S25]		NASA-INDUSTRIAL(PC1, KC1, KC2, KC3, CM1, MW1, MC2), SOFTLAB-INDUSTRIAL(AR3, AR4, AR5)
[S26]		NASA-INDUSTRIAL(PC1, KC1, KC2, KC3, CM1, MW1, MC2), SOFTLAB-INDUSTRIAL(AR3, AR4, AR5)
[S27]	JUR(Ant, Camel, Ivy, Jedit, Lucene, Poi, Synapse, Velocity, Xalan, Xerces)	JUR-PROPRIETARY(Prop-1, Prop-2, Prop-3, Prop-4, Prop-5, Prop-6)
[S28]	JUR(*)	JUR-PROPRIETARY(*), JUR-STUDENT(*)
[S29]	JUR(Ant, Camel, Xalan), RELINK(Apache HTTP Server, Safe, Zhing), AEEEM(Equinox, JDT Core, Lucene, PDE UI, Mylyn)	
[S30]	JUR(Ant, Camel, Ivy, Jedit, Lucene, Poi, Synapse, Velocity, Xalan, Xerces)	
[S31]		MICROSOFT(Windows XP SP1, Windows Server 2003)
[S32]		PROPRIETARY(Sale System, CD Selection System)
[S33]	Xpose, Jwriter	
[S34]		MICROSOFT(Internet Explorer, IIS, Process Messaging, DirectX, NetMeeting)
[S35]	Mylyn, GMF	STUDENT(ECS, BNS, CRS, FACS, ELCS)
[S36]		NASA-INDUSRIAL(CM1, KC1, KC2, KC3, MC2, MW1, PC1)
[S37]	AEEEM(Equinox, JDT Core, Lucene, Mylyn, PDE UI), RELINK(Apache HTTP Server, Safe, Zxing), JUR(Ant, Camel, Poi, Tomcat, Velocity, Xalan , Xerces)	JUR-STUDENT(arc, redaktor, skarbonka), NASA-INDUSTRIAL(CM1, MW1, PC1, PC3, PC4), SOFTLAB-INDUSTRIAL(AR1, AR3, AR4, AR5, AR6)
[S38]	JUR(Ant, Camel, Forest, Jedit, Synapse)	JUR-PROPRIETARY(Prop-1, Prop-2, Prop-3, Prop-4, Prop-5, Prop-6)
[S39]	BugZilla, Columba, Gimp, JDT, Maven-2, Mozilla, Ruby on Rails, Perl, Eclipse Platform, PostgreSQL, Rhino	
[S40]	AEEEM(JDT Core, Equinox, Lucene, Mylyn, PDE UI), Mockus(1385 open source projects from Google Code and SourceForge)	
[S41]	RELINK(Apache HTTP Server, Safe, Zxing), AEEEM(Equinox, JDT Core, Lucene, Mylyn, PDE UI)	NASA-INDUSTRIAL(CM1, MW1, PC1), SOFTLAB-INDUSTRIAL(AR3, AR4, AR5)
[S42]	JUR(Ant, Camel, Ivy, Jedit, Lucene, Poi, Log4j, Tomcat, Xalan)	JUR-PROPRIETARY(Prop-6)
[S43]		NASA-INDUSTRIAL(CM1, KC1, KC2, KC3, MC2, MW1, PC1)
[S44]	JUR(Ant, Camel, Jedit, Log4j, Lucene, Poi, Synapse, Systemdata, Tomcat, Xalan, Xerces)	JUR-STUDENT(arc, elearn, redaktor), JUR-PROPRIETARY(Prop-6)
[S45]		NASA-INDUSTRIAL(KC2, KC3, CM1, MW1, MC2, PC1)
[S46]		NASA-INDUSTRIAL(PC4, MC1, PC2, KC3, MW1, MC2, PC3)

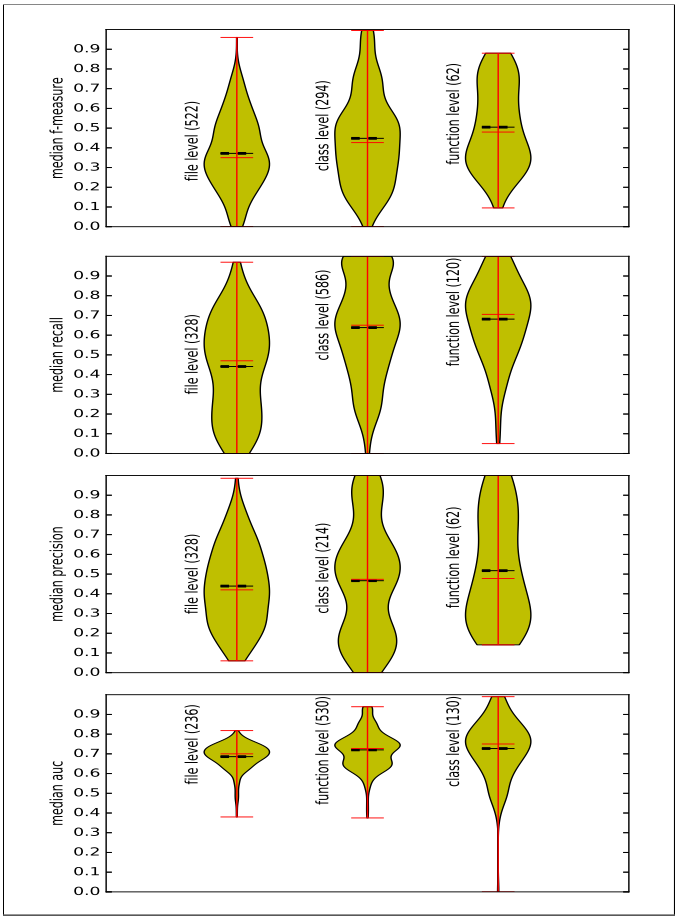


Fig. 8. Performance in relation to the levels of granularity

3.6 Theme: Datasets and Levels of Granularity

Datasets form open source projects are dominant in CPDP studies. The metrics and levels of granularity vary among the datasets such as class, function, and file levels. NASA (54%), Jureczko (65%), and Softlab (25%) datasets are the most widely used datasets in CPDP studies. Some of these datasets contain bug numbers, which are usually converted to classes for the purpose of binary classification.

In CPDP, a diverse range of datasets are used for building the models from different sources. Open source datasets are the main contributors in this area, making the replications for such experiments much more easier. Moreover, the maturity of the NASA datasets is not considered in our study as many studies in this area are validating the proposed models using them. Utilized datasets, their metrics and their level of granularity is collected from the studies and are subjected to our analysis. The utilised datasets by the studies are summarized in Table 16. The table distinguishes among different kinds of datasets, i.e., open source, industrial, student, and proprietary. The used datasets are categorised based on their type and suit they belong to in the table. The following is also a short summary of the datasets from the studies which passed the quality assessment phase:

- **NASA:** NASA MDP datasets contain function/method level software metrics for 13 NASA Software projects. Of these 13 datasets, one is written in Java (KC3) and the rest use C/C++. The defect rate in these datasets ranges from 0.5% (in PC2) to 32.3% (in MC2). Additionally, the metrics in these datasets are not exactly the same and some of them contain additional metrics. For the datasets in this group, eight (CM1, KC3, MC2, MW1, PC1, PC2, PC3, PC4) contain 40, two (MC1, PC5) contain 39 and the other three (JM1, KC1, KC2) have 21 software metrics. One should note the two available sources for these datasets, namely PROMISE and MDP. More importantly, the data in these sources are not the same and the datasets are different in terms of number of features as well as the number of instances [49]. The feature information provided in this study are based on the datasets available from PROMISE repository. See [49] for the detailed information regarding (the quality of) these datasets.
- **Jureczko (JUR):** These suite contains 48 releases from 15 open source projects (Ant, Camel, Ckjm, Forrest, Ivy, JEdit, Log4j, Lucene, PBeans, POI, Synapse, Tomcat, Velocity, Xalan, Xerces). Another 27 releases of 6 proprietary projects belong to this suit. All of these six projects are developed by the same company. Finally, 17 academic projects, with one release each, also belong to the same category. These projects are developed by groups of students within a period of one year. Each of these datasets contain 20 software metrics including OO, SCM, and LOC. The metrics and defect information are extracted using an extension of CKJM [2] and BugInfo [3] tools.
- **AEEEM:** AEEEM suite is collected by D'Ambros [37] and is comprised of metric and bug data from five open source projects (JDT:Eclipse JDT core, PDE:Eclipse PDE UI, EQ:Equinox Framework, Myl:Mylyn, Luc:Apache Lucence). Each dataset contains 61 software metrics including OO, previous defects, and change metrics. To access these suite visit: http://bug.inf.usi.ch/
- **ReLink:** Three datasets are present in this group: Apache HTTP Server, OpenIntents Safe, and ZXing. These datasets contain 26 static code metrics and the defect labels are manually verified after the automatic detection. The suite is collected by Wu et al. [38] and is publicly available: http://www.cse.ust.hk/{$\sim$} scc/ReLink.htm.
- **SoftLab:** This collection contains five datasets (ar1, ar3, ar4, ar5, ar6) donated by Softlab. These datasets contain 29 static code metrics as well as manually verified defect labels for different systems. These datasets share 17 of their metrics with NASA datasets and usually appear together in the studies [S20], [S24], [S26], [S37].
- **Mockus:** A subset of this suite is used in one study [S40] for training their proposed universal model. Originally, the suite contains the defect data for about 235K projects hosted on SourceForge and GoogleCode [39]. Of this huge set of datasets, 1385 were selected to train the Universal model as many of these datasets were either trivial or did not contain adequate data to be considered for defect prediction.

2. http://gromit.iiar.pwr.wroc.pl/p_inf/ckjm/
3. http://kenai.com/projects/buginfo

- **Other suits:** Study [S1] uses the source code text tokens as independent variables and uses the dependent variable information from Jureczko suite and Zimmerman et al. [44]. Study [S2] extract 63 software metrics from Sakura and JEdit editors using Understand tool for C and Java [4]. They use eight OO+LOC metrics for their predictions. Study [S7] uses an extension of KC1 dataset for which object oriented metrics are computed. Originally, the independent variables for this dataset were at the function level. Study [S39] extracts and uses the process metrics for multiple projects (Bugzilla (BUG), Columba (COL), Gimp (GIP), Eclipse JDT (JDT), Maven-2 (MAV), Mozilla (MOZ), Perl (PER), Eclipse, Platform (PLA), PostgreSQL (POS), Ruby on Rails (RUB), Rhino (RHI)).

Beside the datasets themselves, the level in which the predictions are performed are captured for all studies. Three levels of granularity were detected during the data extraction phase. Function level predictions are performed in the case of most predictions on NASA and SOFTLAB datasets. The information on most of these datasets are at the lowest unit of code in their systems, i.e., functions. This level is finer than the level of the Jureczko datasets, in which the class level metrics are extracted. In this case, each instance in a particular dataset represents a Java class. Additionally, some of the studies [S13], [S37] perform predictions at multiple levels of granularity such as function/class/file levels.

The datasets are of great importance when conducting empirical research. In case of defect prediction, the generalizability of the models is impacted greatly by the selection of the datasets. One of the sources for the huge variance in the stability of different learning techniques, data approaches, etc. is probably the use of different sets of datasets that are utilized for building prediction models in different studies. Hence, various aspects of the datasets such as metrics used, their distributional characteristics, and their level of granularity should be considered when making claims for the usefulness of different benchmarks.

3.6.1 Performance in relation to levels of granularity

Performance of different models with respect to the degree of details in granularity is illustrated in Figure 8. We observe a consistent behaviour with respect to different levels of granularity in terms of f-measure, precision, and recall. In all three, function and file level predictions have the highest and the lowest median values respectively. With respect to AUC, the class level achieves the top median value while the file level predictions still have the lowest. Except for AUC, class level predictions have less stability as they cover a higher range. With these in mind, one can argue that function level predictions could potentially lead to better performance while being more effective in practice.

4 CPDP vs. WPDP

To measure the effect and power of the proposed approaches, we collected both within and cross project data for all the models that pass through our quality checklist. Studies [S1]–[S3], [S7], [S11], [S13], [S15], [S16], [S20], [S24], [S27], [S37], [S39], [S43], [S46] have compared the performance of their proposed approach to that of WPDP (with one or more of f-measure, recall, precision and AUC). Our comparisons however are not limited to the proposed approaches in the studies as we have combined the data from all of the reported models with enough information. We need to point out that when we draw conclusions from the results, we have to consider that the data for these models always include those of proposed approaches, i.e, the best CPDP models and approaches, while this is not the case for the WPDP models. Usually, the simplest WPDP models are considered in the studies when they act as benchmarks for proposed CPDP approaches and no rigorous and sophisticated optimisation or data manipulation methods are applied. This is also one more reason for including all of the benchmark cross project approaches used for the sake of comparisons in the studies.

Meta-analysis was employed to assess and compare the performance of CPDP and WPDP as there is no clear picture of how they perform against each other. Finally, we have to consider the fact that no set of evaluation measures and none of the types of reporting (median, average, original) are shared among all of the studies that contribute to this analysis. This is why multiple plots matching different criteria are presented.

4.1 Meta-analysis

In its essence, a meta-analysis incorporates statistical approaches to combine multiple studies in an effort to increase power over individual studies while improving the estimates of the effect sizes and resolve uncertainties when different studies disagree [50], [69], [76]. Therefore, it can be interpreted as a statistical overview of the results from multiple studies. Extraordinarily, very significant increases in power are less likely in general and one could reasonably expect that new methods will lead to modest improvements in the majority of experiments. This, however, is not to undermine the importance of these improvements. On the contrary, these small steps can be extremely important and provide significant benefits if applied under suitable circumstances. Perhaps the greatest advantage of using a meta-analysis is its ability in generalising the results to a larger population and proposing bigger pictures of the state for the research target. Individual studies are often too small for drawing reliable general conclusions. Combining the results of multiple studies using meta-analysis technique provides higher numbers of participants, less random error, and narrower confidence intervals [75]–[77]. Consequently, these factors could provide better evidence for verifying the validity of a true effect and investigating its statistical significance [78].

Having said these, meta-analysis have their downsides as well. Such a case is the occasional failure of this method to detect the source of bias. For example, meta-analysis cannot correct the poor design and bias in the original studies [70]. This has led to the use of different strategies to include/exclude certain studies based on sets of defined quality measures. However, using such filters could lead to another probable level of bias, i.e., selection bias (subjectivity) into the method [71], [72].

4. http://www.scitools.com/index.php

Nonetheless, the advantages are far more valuable than these few downsides if the analyses are carried out and interpreted carefully. For details on common criticisms of meta-analysis, and how they are addressed, we refer the reader to Borenstein's book [96]. In software engineering, Kitchenham et al. endorses the use of meta-analysis [95]. Examples of meta-analysis in software engineering include investigations of test-driven development [92] and pair-programming [94].

The first step in meta-analysis is to calculate the outcome of interest and summary statistics for each individual study. In the second stage, the calculated statistics are combined to give an overall summary estimate. The weighted mean difference technique is considered in this study as it is one of the most popular techniques for meta-analysis with continuous variables and the one that best fits our available data [73], [74], [76]. In this approach, each study receives a weight. The greater the weight awarded to a study, the more it influences the overall estimate. Different weighting models exist for meta-analysis, most notably, the inverse variance method. The inverse variance method essentially assigns higher weight to larger studies and less weight to smaller studies. Alternatives, based on other factors such as trial quality exist but such methods are rarely used and not recommended [78].

4.2 Fixed and Random Effect Models

Two popular methods of performing meta-analysis are fixed and random effect models [76]. The fixed effect model assumes that the subjects share a common effect [76], [77]. In other words, it considers only one source of variability, i.e., the within study error. As a result, the contribution of each study is proportional to the amount of information observed in that study. Consequently, this implies that the differences among studies are solely due to sampling error, i.e., by increasing the sample size it is likely that the effects converge to one true effect [76], [77]. The assumption of a common effect shared by the studies does not usually hold. Therefore, the random effect model which considers a second source of variability, i.e., the between study error, is usually reported.

The random effect model is built on top of the results of the fixed effect model. The description of the random effect model and its procedures will be discussed after presenting the required calculations for the fixed effect model. For more detailed analysis and discussions, see [75]–[77].

Fixed effect model: To estimate the individual effect for a specific study one needs to assess the difference between the experimental and control groups. One popular method of doing such comparisons is through the effect size calculations. The following equation is used to calculate Cohen's d:

$$cd = \frac{X_e - X_c}{std_{pooled}} \tag{3}$$

Here, X_c and X_e are the means of the control and experimental groups, respectively. std_{pooled} represents the pooled standard deviation which can be calculated as follows:

$$std_{pooled} = \sqrt{\frac{(n_e - 1) * (s_e)^2 + (n_c - 1) * (s_c)^2}{n_e + n_c - 2}} \tag{4}$$

Where s_c, n_c, s_e and n_e are the standard deviation of control group, number of subjects in the control group, standard deviation of experimental group, and number of subjects in the experimental group, respectively. Multiplying Cohen's d by Hedges' correction factor (Eq. 5) will result in Hedges' d (Eq. 6):

$$J(m) = 1 - \frac{3}{4m - 1} \tag{5}$$

$$d = J(n_c + n_e - 2) \times cd \tag{6}$$

Having the effect size calculated, the confidence interval can easily be estimated using the following equation:

$$d - Z_{\frac{\alpha}{2}}\sqrt{v} \leq I \leq d + Z_{\frac{\alpha}{2}}\sqrt{v} \tag{7}$$

In this equation d is the effect size for the individual study and $Z_{\frac{\alpha}{2}} = 1.96$ for $\alpha = 0.05$. Further, v represents the estimated variance which can be calculated as follows [50]:

$$v = \frac{n_e + n_c}{n_e \times n_c} + \frac{d^2}{2(n_e + n_c)} \tag{8}$$

The following equation, calculates the overall effect:

$$d_* = \frac{\sum_i w_i \times d_i}{\sum_i w_i} \tag{9}$$

where w_i is the weight assigned to study i and is equal to the inverse of its observed variance:

$$w_i = \frac{1}{v_i} \tag{10}$$

The variance for the overall effect, therefore, can be calculated as follows:

$$v_* = \frac{1}{\sum_i w_i} \tag{11}$$

Finally, the confidence interval for the overall effect is:

$$d_* - Z_{\frac{\alpha}{2}}\sqrt{v_*} \leq I \leq d_* + Z_{\frac{\alpha}{2}}\sqrt{v_*} \tag{12}$$

Random effect model: The fixed effect model relies on the assumption that the true effect is shared among the studies. This assumption, however, might not always be plausible. Even though the studies for meta-analysis are usually similar when considered, there is generally no reason to assume that they share a common effect. Therefore, rather than one true effect, we assume the existence of a distribution of true effects. The random effect model assumes that the differences among individual effect sizes are due to sampling error as well as other variables and factors that have not been accounted for. The combined effect consequently, cannot be represented as one, but instead is represented by the mean of the population of true effects [76].

The calculations for the random effect model, involve changing the weighting mechanism used by the fixed effect

model. Specifically, Q which represents the total variance, and df, which represents the expected variance if all studies have the same true effect need to be calculated first. The difference, $Q - df$, is called the excess variance which after transformation into the same scale as the within study variance is denoted by τ^2 (tau-squared). The Q statistic is computed as follows:

$$Q = \sum_i w_i \times (d_i - d_*) \tag{13}$$

where $w_i = 1/v_i$, the inverse variance, is the weight for individual effects in the fixed effect model. When the only source of variance is the within study error, then the expected value of Q would be the degrees of freedom (df) where

$$df = \#studies - 1 \tag{14}$$

Consequently, the between studies variance, τ^2, can be calculated according to

$$\tau^2 = \begin{cases} \frac{Q-df}{C} & if\ Q > df \\ 0 & if\ Q \leq df \end{cases} \tag{15}$$

Where

$$C = \sum_i w_i - \frac{\sum_i {w_i}^2}{\sum_i w_i} \tag{16}$$

is the scaling factor for transformation into the same scale as the within study error. Using τ^2, the weights and the overall effect can be calculated as follows:

$$w'_i = \frac{1}{v_i + \tau^2} \tag{17}$$

$$d'_* = \frac{\sum_i w'_i \times d_i}{\sum_i w'_i} \tag{18}$$

Similar to the fixed effect model, the overall variance and the 95% confidence interval could be computed as

$$v'_* = \frac{1}{\sum_i w'_i} \tag{19}$$

$$d'_* - Z_{\frac{\alpha}{2}} \sqrt{v'_*} \leq I \leq d'_* + Z_{\frac{\alpha}{2}} \sqrt{v'_*} \tag{20}$$

Finally, the one and two tailed $p - value$ for random effect model can be computed using z'_* value and the standard normal cumulative distribution function, $\phi(z)$ by the following (similar for fixed effect using, d_* and v_*):

$$z'_* = \frac{d'_*}{\sqrt{v'_*}} \tag{21}$$

$$p - value = 1 - \phi(z'_*) \tag{22}$$

$$p - value = 2 \times (1 - \phi(|z'_*|)) \tag{23}$$

Using the random effect model, it is expected to observe changes in the fixed effect model by modifying the weights to be more balanced. The location of the combined effect could potentially change due to the weight updates. Finally, the confidence interval for the combined effect is expected to increase, which in turn might lead to a non-significant overall effect (intersection with the y-axis or $p - value \geq \alpha$).

4.3 Meta-analysis Results

Very briefly, WPDP outperforms CPDP. CPDP challenges WPDP only with respect to recall, but that is at the cost of precision. This trend is hidden when checking performance with the compound f-measure. NASA and Jureczko datasets are likely to be involved in the performance shift toward CPDP. Both fixed and random effect meta-analysis models per study, learner and dataset agree on the results for most cases (though not in 100% agreement).

In practice, with well-defined questions, both fixed and random effect models lead to very similar results and it is recommended to perform and report both as a measure of robustness for the choice of statistical model [78]. Therefore, both fixed and random effect meta-analysis were performed for studies, datasets, and learners, i.e., the difference between CPDP and WPDP was assessed among the studies and further in relation to the datasets and learners. Furthermore, multiple plots are prepared based on the measures of performance (f-measure, precision, recall, AUC). The results of our analysis are represented in Forest Plots. Each contributing case in the forest plots appears as a row in the plots (a study, a dataset or a learner depending on the analysis category). The filled black box in each row represents the weight of that particular instance (study, dataset, learner) for the random effect model. The centre of the box is where the mean is located. The fixed effect weights for each instance are represented by unfilled rectangles. When the boxes for the fixed effect weights are not visible, they are covered by the boxes from the random effect model. This can happen when the random effect weights (for smaller instances) become larger or when the weights from the two models are very close. The latter occurs when the between study variance is very small or when the two models match, i.e., $\tau^2 = 0$. The line on the sides of each box are the 95% confidence interval associated with each instance. The thin dashed line shows the line of no effect for the random effect model, i.e., the average overall random model effect size. Similarly, the thin red dotted line represents the line of no effect for the fixed effect model. The diamonds, their placements, and their sizes show the state of the overall effect (filled black diamond for random effect and the unfilled diamond below it for the fixed effect model). An intersection between the y-axis and the confidence interval for an instance shows a non-significant difference. This is also the case for the overall effect and if a diamond intersects with the y-axis, the overall performance difference for that model won't be statistically significant. The $p - values$ for significance and the 95% CI for both models are also represented through summary statistics in

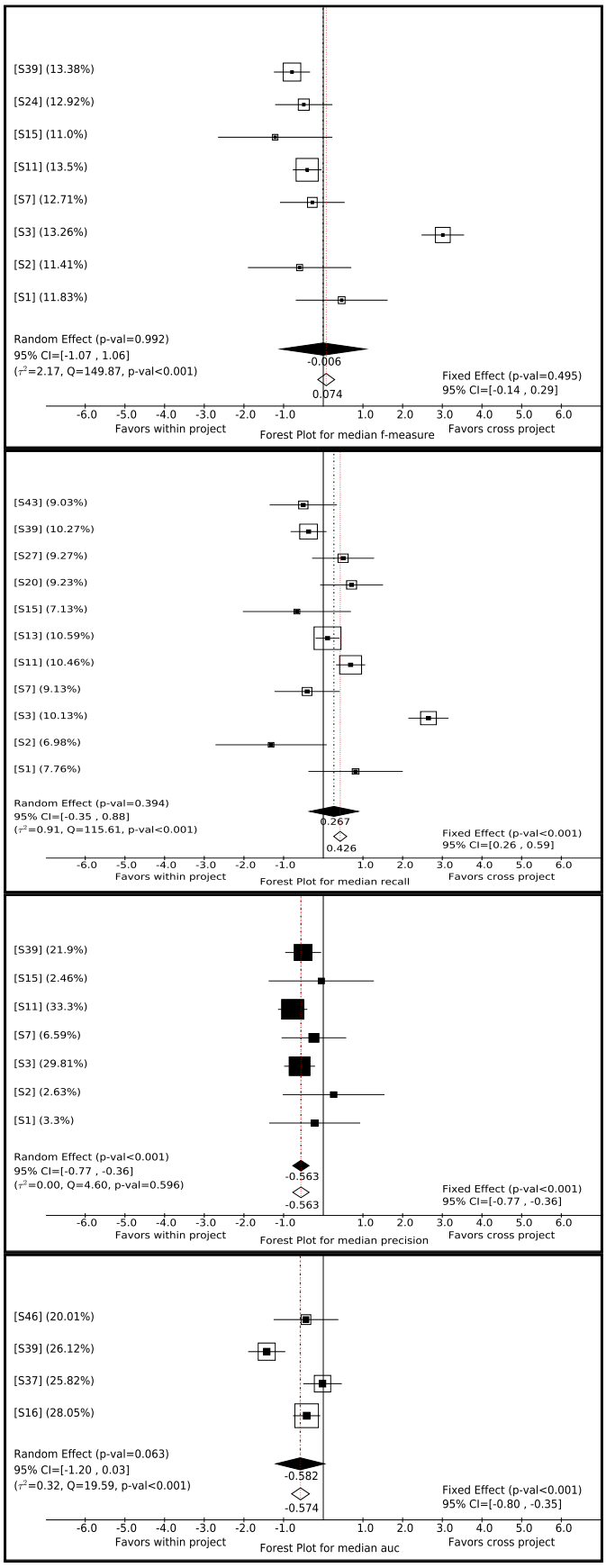


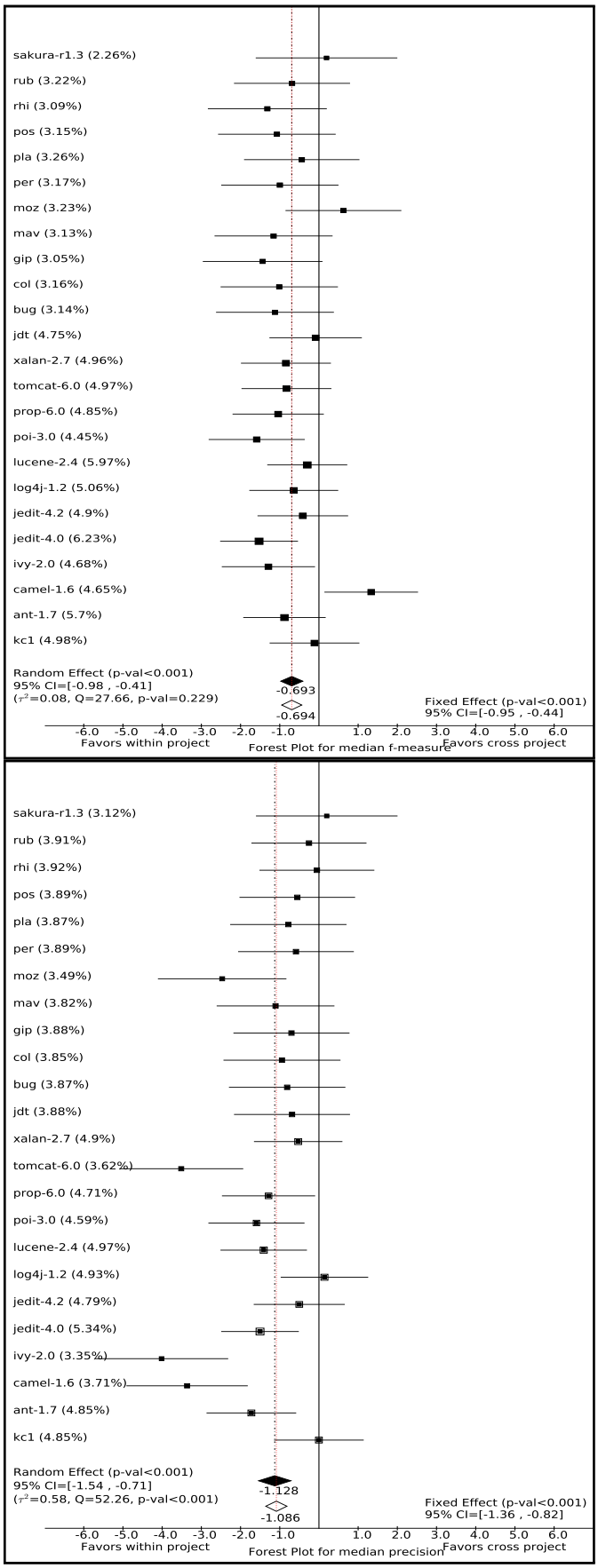
Fig. 9. Performance comparison between CPDP and WPDP across the studies: f-measure, recall, precision and AUC

the left (random effect) and right (fixed effect) sides of the plots. The placement of the boxes (for each instance) and the diamond (for each model) in the left side is for performances in favor of WPDP while the right side is for CPDP. The list of the studies, datasets, or learners and their proportion of the total weight (only for the random effect model) are shown in the left side of each plot.

Figure 9 presents the plots for f-measure, recall, and precision evaluation measures per study. With f-measure, three studies ([S3], [S11], [S39]) present the evidence for a significant difference between WPDP and CPDP while the data from other studies do not show such differences. These three studies have the highest contribution to the fixed effect model, demonstrated by their effect box sizes. The overall effect is very close to zero (-0.006) for random effect and very small for the fixed effect (0.074), non of which are neither significant nor large. The majority of the studies favour WPDP over CPDP. The significant difference observed in [S3] is the main reason for the big shift toward CPDP.

The fixed and random effect models do not agree on the significance of the overall effect for recall. With random effect, CPDP outperforms WPDP, but similar to the median f-measure case, the performance difference is not statistically significant ($p = value = 0.394$). The fixed effect model, however, shows a significant difference toward CPDP($p - value < 0.001$). The between study variance, denoted by τ^2, causes the expansion of the CI for the random effect model for both f-measure and recall (with study [S3] as a potential reason). The good performance of CPDP toward recall has been pointed out with results from multiple studies [S11], [S20], [S26], [S44] as well as our analyses in previous sections. A better WPDP performance is observed when precision is considered. The fixed and random effect models match one another in this case, demonstrated by $Q = 0$, $\tau^2 = 0$ and $p - value = 0.595$, rejecting the heterogeneity. The precision performance except in [S2] are toward WPDP. Finally, WPDP wins against CPDP when AUC is considered but the diamond representing the overall effect has an intersection with the y-axis in random effect model, making the overall effect non-significant despite the medium overall effect size The fixed effect model, however, achieves a significant difference with AUC having $p - value < 0.001$.

Forest plots for the comparison of CPDP and WPDP considering the datasets are presented in Figures 10 and 11. In terms of f-measure, beside the overall effect sizes, the majority of the datasets favor WPDP over CPDP. Only three datasets (Camel 1.6, MOZ, Sakura r1.3) are in favor of CPDP. Recall with a much higher number of observations is significantly better with CPDP, however, the effect sizes are small for both random and fixed effect models. Jureczko datasets make up the majority of the datasets in favour of CPDP with recall. The performance difference with precision is also more vivid in this case. WPDP is significantly more precise than CPDP and the achieved large effect size confirms this finding as well. With AUC, most of the datasets have a better prediction performance with WPDP. NASA datasets are the only datasets in favor of CPDP in this case. In particular, three NASA datasets (cm1, kc3,pc2) have better AUC with CPDP in comparison with those gained from WPDP. In general, NASA and Jureczko datasets have highly



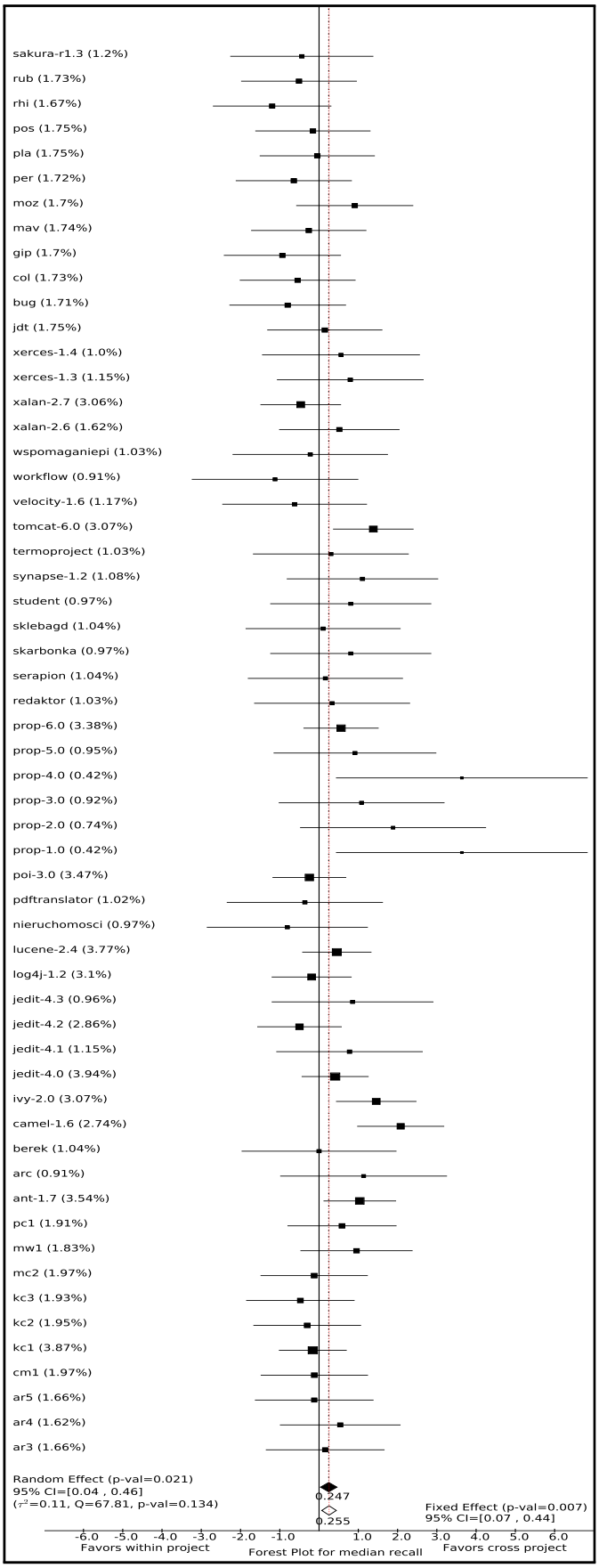


Fig. 10. Performance comparison between CPDP and WPDP across different datasets: f-measure, recall and precision

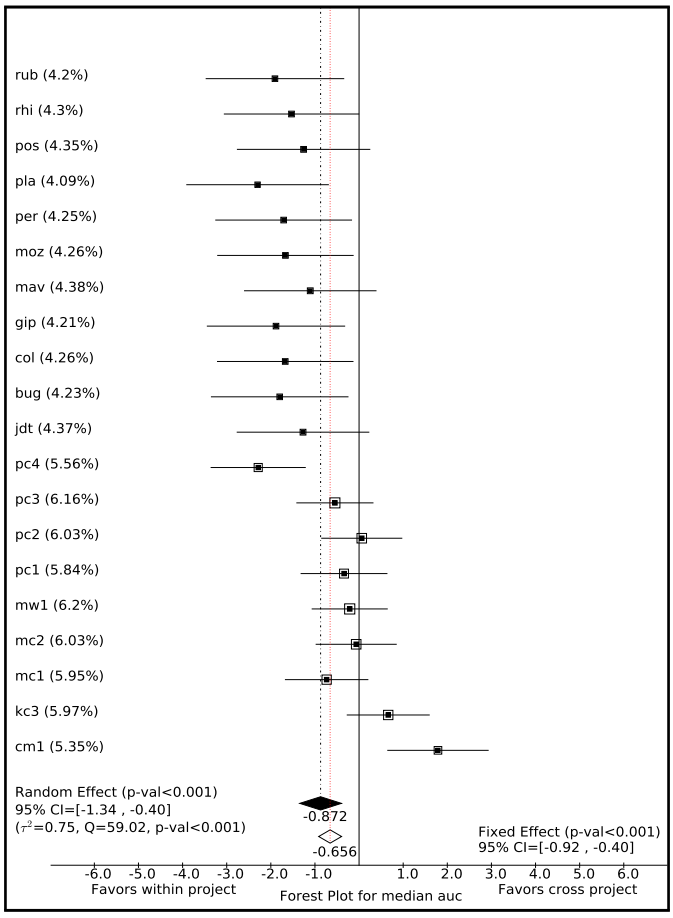


Fig. 11. Performance comparison between CPDP and WPDP across different datasets: AUC

contributed to the performance shift toward CPDP. Further, the share for some of these particular datasets is higher than the rest of the datasets as they have been used and validated in more studies. Fixed and random effect models are similar in the case of datasets and they agree with each other beside the differences in effect sizes and confidence intervals. The overall effect sizes for precision and AUC are higher than those of fixed effect model, but the confidence intervals are wider as well. The effect size for recall has become smaller in response to the more balanced weightings of the random effect model and the confidence interval, similar to the other measures is larger as well.

As another probable source of influence on the performance difference between CPDP and WPDP, we have evaluated the performances with respect to the learning techniques. The forest plots in Figure 12 present the results in this regard, extracted from the categorical studies. The overall random model effect with f-measure, a medium effect size, is toward CPDP but it is not significant. The overall effect for the fixed effect model, however, is tiny (-0.039) and not significant. The shift in location of the overall effect is clearly caused by the high weights assigned to the LR and RF learners, while the high performance differences observed with different learners cause the wide confidence interval for the overall random model effect. CPDP achieves a significant medium to large effect size for outperforming WPDP with respect to recall, as can be

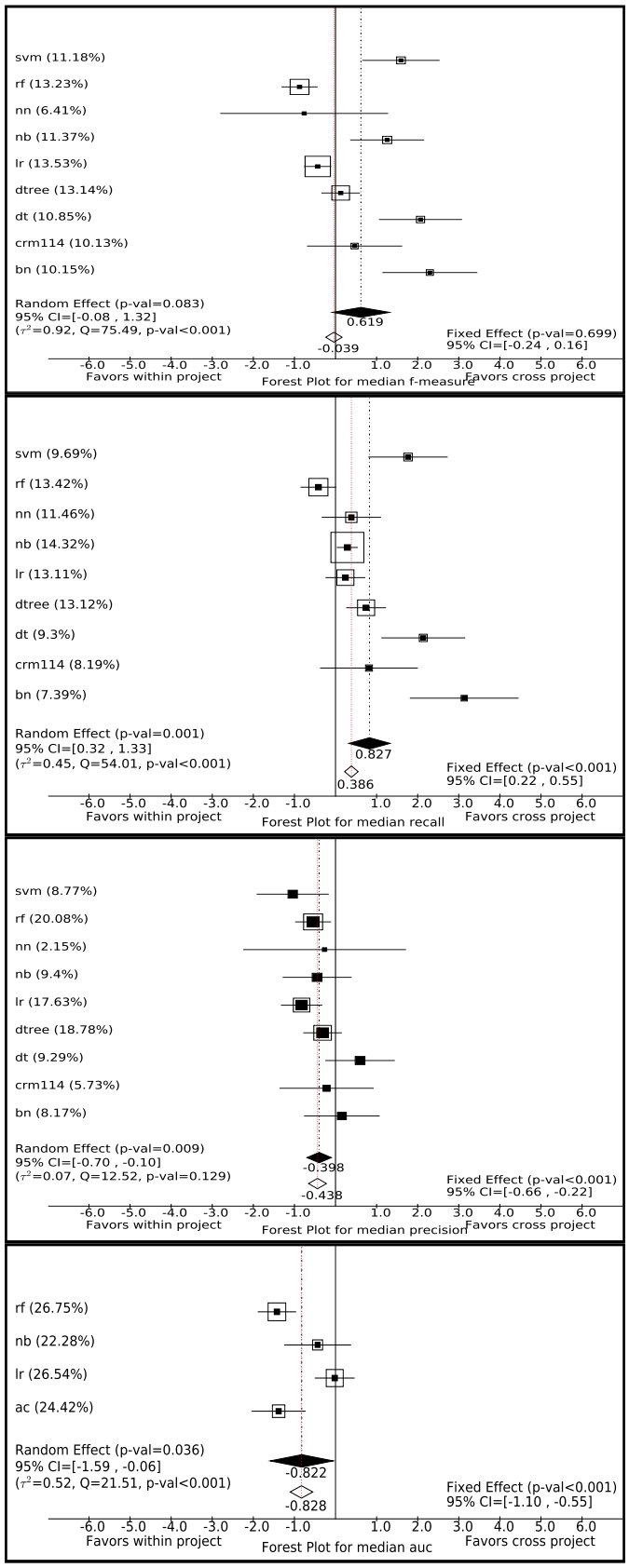


Fig. 12. Performance comparison between CPDP and WPDP with different learning techniques: f-measure, recall, precision and AUC

expected considering the observation that the majority of the learners have a favourable CPDP recall performance. In

a case of disagreement and with similar situation to that of f-measure, the very high (relative) weight assigned to a single learner, NB in this case, has caused a large shift in the location of the overall effect for the fixed effect model, causing it to be tiny and insignificant (0.084). With precision and AUC, WPDP is significantly better than CPDP while in both cases, wider confidence intervals can be observed.

With respect to the f-measure of particular learning techniques: SVM, DT, BN, CRM114, and DTree are in favor of CPDP, while RF and LR prefer WPDP. NN and NB are slightly better with WPDP but the difference is less significant for them. A very similar pattern can be observed for recall as well with the difference that LR is now marginally pro recall. With respect to precision, as expected based on the previous discussions, a significant difference in favor of WPDP can be observed. Only two techniques, i.e, BN and DT, are in favour of CPDP, neither of which are significant individually. Conversely, three of the learners achieve substantial effect sizes individually in favor of WPDP. Finally, with AUC, only one learning technique (LR) is in favor of CPDP and the rest are in favor of WPDP, resulting in a medium to high overall effect size toward WPDP.

5 Discussion

By synthesizing both qualitative and quantitative data, this section provides answers to our defined research questions. We follow a similar approach used by Hall et al. [1] when discussing model performance in relation to certain factors (e.g., independent variables, modeling techniques, etc.). First, the performance within individual studies is discussed. This helps to figure out the major predictive performance impact within each particular study. Then, the model performance across studies is discussed to get a bigger picture of how well a model performs across studies. A similar approach is considered for comparing WPDP and CPDP performances.

5.1 Independent Variables in CPDP (RQ1)

The independent variables used in the categorical studies can be mainly categorized as traditional metrics (size and complexity metrics), process metrics (code delta, code churn), and Object-Oriented metrics. In addition to the above three categories, one study uses text tokens extracted from source code [S1].

Analysis of the model performance across the categorical studies suggests that the combinations involving LOC, OO, and SCM improve the predictive performance of their respective models. Further, the analysis reveals that the process metrics have a poor performance in comparison with the previous categories. However, Kamei et al. [S39] asserted that the defect prediction models built with process metrics are more useful in comparison with the traditional defect prediction models as they are done at a finer level of granularity and the responsible developers for inspections can be determined more rapidly. Their results showed that WPDP models built with code changes outperform CPDP counterparts in terms of AUC, precision, recall, and f-measure.

CPDP models built with only OO metrics tends to perform better than the other CPDP models built with only one set of metrics (e.g., only process metrics) in terms of f-measure and precision. Source code text from [S1] also has a comparable performance, but the evidence for it is not as strong as of OO.

We observed that the models built upon the combination of different independent variables perform the best. Watanabe et al. [S2] and Canfora et al. [S11] used OO and LOC. The combination of OO, SCM and LOC is utilised by several studies ([S3], [S5], [S23], [S40]). Further, Jing et al. [S41] used the combination of OO, SCM, and process metrics along with LOC. Moreover, OO+LOC, which has the top spot with regard to recall, has the lowest median precision.

Finally, Zhang et al. [S40] showed that context factors such as programming languages, the presence of issue tracking systems, the total number of commits, and the total number of developers can further increase the predictive performance of the models using only code and process metrics.

5.2 Modeling Techniques in CPDP (RQ2)

Naïve Bayes (NB) and Logistic Regression (LR) are the most commonly used modeling techniques in CPDP. Most of the studies have constructed multiple models with different modeling techniques and their comparative performance are reported. When considering individual studies, it is not possible to see a clear picture of which modeling technique performs the best in CPDP context. Singh et al. [S7] concluded that J48 (a variant of C4.5 implemented in Java) performs better than the other benchmark classifiers (NB, SVM, RF, NN, and DT). Further, they revealed that NB classifier favors CPDP over WPDP in terms of precision. He et al. [S5] also reported that the best prediction results are provided by J48. They later observed that simple classifiers like NB could perform well in CPDP context with respect to the overall performance [S3]. Additionally they asserted that NB and BN are relatively stable in presence of different metric sets in source and target data sets. Liu et al. [S14] reported that GP-based models perform better than the non-GP models. They presented these results by comparing GP-based models against 17 non-GP models. Chen et al. [S18] observed that DTB, which is based on NB and boosting, improves CPDP performance by reducing negative samples in CP data. VCB-SVM, a model based on SVM and boosting, can better classify defect-prone code units by considering the class imbalance issue [S26]. Similarly, better results were observed by Ryu et al. [S44] with TCSBoost which uses NB with boosting and utilizes mixed WP and CP data.

Detailed analysis of the categorical studies suggest that the CPDP performance may associate with modeling technique used. As mentioned earlier, NN, SVM, and DTree achieve the highest median f-measure performances and they seem to be performing relatively well in CPDP context. Moreover, base learners perform better with respect to f-measure compared with the ensembles. Conversely, ensembles seem to improve the performance of base learners in terms of recall and AUC. Notably, standalone BN has the lowest AUC while metalearner BN provides the highest. Ensembles seem to improve LR as well in terms of AUC. Despite its complicated nature, RF classifier has a low median

performance with respect to multiple measures. Further, the performance of NB is medium in terms of f-measures and recall and medium to low with AUC and precision. LR has a medium to low performance and is comparable or inferior to NB while being more complicated.

5.3 Performance Evaluation in CPDP (RQ3)

Analysing performance evaluation criteria of 28 categorical studies disclosed that the majority studies had computed various compound performance measures using the confusion matrix to evaluate the performance of their prediction models. Few studies had employed only AUC to measure the model performance ([S4], [S37], [S46]).

Almost all the studies construct single objective defect prediction models. Single-objective defect prediction models detect a set or a ranked list of likely fault-prone code units [S11]. Multi-objective Defect Prediction, on the other hand, targets multiple goals at once. MODEP [S11] uses a multi-objective approach to target both cost and effectiveness. MODEP tries to achieve a compromise between cost and effectiveness. Therefore, software engineers can select predictors that achieve their desired level of compromise.

The majority of the studies had given some sort of justification for selecting their particular set of measures. In some studies, popularity is one of the reasons in that regard ([S5], [S7], [S25], [S39], [S41], [S46]). Theoretical justifications are another approach that some of the studies have taken into consideration. Arguments such as being threshold sensitive and biased against the class imbalance problem fit into this category as well. A number of the studies have taken these issues into account for not selecting precision and accuracy [S13], [S20], [S25], [S27]. For example, [S20] had used recall, probability of false alarms and balance and dismissed the use of accuracy, mentioning that such measures are poor indicators of performance considering the balance between the defect prone and defect free classes, i.e., the class imbalance [67]. AUC, on the other hand, is threshold insensitive while not being influenced by the class imbalance issue. This has justified the use of AUC in some of the studies either individually [S4], [S37], [S46] or with other measures [S26]. The justifications for using particular performance measures were collected when possible for the primary studies. A summary of these justifications are presented in our online appendix.

While the selected reporting measures by the studies could provide some means of comparisons, the lack of common measures make such tasks difficult or even impossible. Hall et al. [1] presented alternatives to be used for reporting. They particularly suggested to report AUC and confusion matrices and presented references for useful means of reporting [1], [67], [68] (for confusion matrices). Further, they recommended Average Relative Error (ARE) for continuous studies [1]. The concern of using particular performance evaluation measures in different contexts has been raised by different studies [1], [67], a summary of which is presented by Hall et al. [1]. For example, Menzies et al. in [67] did not use precision and accuracy arguing that they are poor performance indicators when the instances of the target class are rare.

5.4 CPDP Approaches (RQ4)

A variety of approaches are proposed in CPDP, addressing different data and learner related issues. Of those issues, class imbalance and data heterogeneity are the most commonly investigated by the proposed models.

Approaches such as DTB [S18], VCB-SVM [S26], TCSBoost [S44], and JIT [S39] had used data re-sampling to deal with the class imbalance issue.

NN-filter [S20] tries to come up with similar data to those of the test set, resulting in a more suitable training dataset. Transforming the data to make the data distributions more similar is another approach considered by the categorical studies. Approaches such as metric compensation [S2], TCA+ [S24], Universal model [S40], CCA+ [S41] belong to this category. LACE2 [S27] also fits into the data transformation category but its transformation is more focused on the privacy issues rather than the performance.

Some approaches have used multiple methods in their settings to achieve their desired goals. Turhan et al. [S13], DTB [S18] and TCSBoost [S44] utilised mixed data to create CPDP models.

Only one study [S23] in the included set uses clustering in their proposed approach. Two methods are utilized to find similar data in this case, namely NN and EM clustering.

Feature selection is also used by multiple studies [S3], [S15], [S37], [S42]. Heterogeneous Defect Prediction (HDP) approach proposed by Nam et al. [S37] uses feature selection and feature matching in its defect prediction process.

Basic data processing methods such as data normalization and logarithmic transformation had been used widely in CPDP as they are believed to have positive effect on the performance of CPDP.

Benchmark methods are also an interesting point to stress out. Usually the proposed approaches are compared with other well-studied methods in the literature. As such, He et al. [S18] compared DTB with other famous approaches in the CPDP such as TNB, NN-filtering, and mixed data approach. They stated that, DTB performs significantly better than all the other counterparts statistically in terms of g-measure. Similarly, in [S25], the predictive performance of TNB is compared with NN-filtering approach, leading to the conclusion that TNB outperforms filtering in terms of AUC and f-measure.

Jing et al. [S41] had compared their proposed approach (CCA+) with various state-of-the-art approaches such as TCA+, NN-filtering, and TNB. Their experimental results indicated that CCA+ is superior to the benchmark CCDP methods in terms of three widely used measures namely recall, pf, and f-measure.

Even though some studies have provided comparisons with few CPDP approaches, there was no clear consensus on which CPDP approach performs the best. Thorough analysis of the model performance across studies suggest that the performance of the model links to the data and overall paper approaches used. More specifically, the general trend seems to be the recall based nature of the data approaches. This good recall based performance has probably contributed more to the overall increase in f-measure performance when data approaches are utilised. This can also be argued as the data approaches do not seem to have a profound effect on

precision and using no data approach yields better precision performances compared to the other counterparts.

5.5 CPDP vs. WPDP (RQ5)

We performed three sets of analysis for benchmarking the performance of CPDP models against that of WPDP as no general conclusion could be made by just considering individual studies. Some studies support WPDP performance (e.g [S15], [S39]) while another group make claims in favor of CPDP (e.g [S1], [S3], [S5], [S11], [S23], [S40], [S41]). These claims vary in terms of specific performance measures (e.g., recall, AUC) or modeling technique (e.g., NB, J48) to mention some.

He et al. [S5] asserted that source data from the same project does not always lead to better prediction results and CP data could potentially provide better predictive performance results when the most suitable source datasets could be determined. Herbold [S23] observed that the CPDP performs better with recall, but it lacks the precision of WPDP models. Similarly, Mizuno and Hirata [S1] revealed that WPDP with source code text tokens as independent variables achieves better precision compared with that of CPDP models, but underperforms CPDP in terms of recall.

Zhang et al. [S40] observed that there are clear differences in the performance between the universal model and WPDP models with all performance measures except AUC. Specifically, the universal model yielded lower precision, but achieved higher recall values. Further, they asserted that the universal model is as effective as WPDP models in terms of AUC. CCA+ [S41] not only considers the heterogeneity in the data but also obtains comparable prediction results to WPDP. The results of the study conducted by He et al. [S3] indicated that WPDP models capture higher precision while CPDP models achieve better recall or f-measure. Canfora et al. [S11] revealed that CP predictions are worse than WP predictions in terms of precision and recall. However, MODEP introduced by them achieved a better cost-effectiveness compared with single-objective predictors trained with WP data.

We performed two sets of meta-analysis, i.e., fixed and random effect models to investigate the relationships and their directions between CPDP and WPDP. For each effect model, we performed three sets of analysis targeting different aspects of the studies. Our first set of meta-analysis grouped by primary studies revealed that CPDP could achieve comparable performance to WPDP in terms of f-measure. Even though the majority of the studies show a positive effect for WPDP, the overall result is neither toward CPDP nor WPDP and the difference is not significant. With recall, CPDP achieves a better performance resulting in a non-significant medium effect size with random effect model and a small to medium effect size with fixed effect model. Conversely, WPDP wins with precision both in terms of number of studies and the overall effect. The random and fixed effect models match in this case as the test for heterogeneity is rejected ($p - value = 0.596$). The achieved effects for precision are medium and significant in favor of WPDP. Finally, despite having similar medium effect sizes, the fixed and random effect models do not agree on significance for AUC. The fixed effect model, clearly impacted by [S39], shows significance ($p - value < 0.001$) while the balanced weights and consequently, the wider overall confidence interval for the random effect model demonstrate insignificance ($p = value = 0.063$).

To reach a higher degree of confidence we conducted further investigations on the factors that are likely to impact the peformance differences. This time, we categorized the collected data with respect to the datasets that are used. The results in this case show better WPDP performance in terms of precision and AUC and f-measure. With recall, CPDP performs better than WPDP and the difference is significant. Moreover, we observed that the main contributors to the good results of CPDP are Jureczko datasets which tend to improve recall.

The next set of analysis revealed a similar pattern with precision and AUC, but the performance difference with f-measure were not significant. We further observed that some of the learning techniques might be more suitable for CPDP in relation to different performance metrics. At the same time, the fixed effect model results demonstrated different outcomes with recall and the reason for the difference, i.e., the higher assigned weights to the most widely used learners, was detected as the reason behind the difference.

While the fixed and random effect models did not agree on all cases, in summary, we observed the recall based nature of CPDP with multiple sets of analysis. The conclusions of different studies about the precision and AUC of WPDP models were confirmed as well. It is important to remember that the best results of CPDP are included in these comparisons while the results for WPDP are provided as a part of the benchmark methods for the proposed CPDP approaches. These WPDP models are usually built in simpler manners, without much sophisticated approaches and one could argue that, in practice, WPDP could outperform CPDP with possibly even larger difference margins.

6 Guidelines and Recommendations for Future Research

The vast history of defect prediction has lead to proposal of many theories, approaches, and models. Hall et al. [1] performed a comprehensive systematic review on the subject. Many of the proposed ideas have been adapted and applied by CPDP branch despite its recent history. We summarized the state of CPDP in this study. Below is a list of the suggestions for future research.

6.1 On data quality and SZZ

As pointed out by Hall et al. [1], researchers should seriously consider data quality for their research. As seen with our list of primary studies, the majority of the datasets are quite old and their quality and usefulness (e.g., NASA dataset) are under question in two ways. The first issue, raised by multiple studies [1], [40], [49] in this area, is the quality of the datasets. This is stressed upon to such a degree by Hall et al. [1] that they have failed almost every study that experiment with NASA datasets in their quality assessment stage (114 failed studies of which 58 use NASA datasets. These 58 studies belong to a set of 62 papers in total, which use NASA datasets meaning that only 4 passed the

quality assessment stage). Another issue with these datasets is the alignment of their practices with today's development practices, which make the data potentially irrelevant and, if not selected/handled/treated carefully, even harmful. On the quality issue, we strongly feel the need of performing additional studies so as [60] to evaluate the validity of the data in the first place.

In addition, the true origin of the bugs and the number of bugs in each unit of interest in both code and change metric datasets have potential for further research. The majority of the datasets for CPDP and defect prediction in general are generated through automated heuristic approaches such as SZZ [61]. However, SZZ and its extensions [61]–[64], for example, can not handle an entire line of bug fixing, i.e., when bugs are fixed only by addition of new code (or sometimes moving the code in the source files) [64]. This is not to be interpreted as a recommendation to discard all SZZ generated datasets, but rather a call for more research effort for compiling better quality defect datasets.

Hence, the quality and representativeness of the datasets and the process of mapping bugs to code units needs better solutions and serious attention for the validity of future research. For reviewers of CPDP papers, analysis of a *conveniently* random subset of datasets should raise doubts.

6.2 On performance measures

Regarding the predictive performance, it is important to report multiple performance measures to fully grasp the nature and capabilities of the proposed models [1], especially with regard to compound measures such as f-measure. Being a compound measure, f-measure can be misleading in differentiating the differences across precision and recall. Reporting base measures instead of compounds would help computing additional measures in the future. Reporting performance based on confusion matrices and AUC is also a highly encouraged practise [1]. AUC is fundamentally different from f-measure, recall and precision such that it is threshold independent. Hall et al. [1] not only provided suggestions for what to use, but they also warned the researchers about the conclusion invalidity in case of using inappropriate performance evaluation measures. Therefore, the choice of evaluation measure should be thought out carefully.

We observed that in most cases CPDP studies report higher recall at the expense of lower precision (and this insight is lost with f-measure). Ideally, defect prediction models are decision support systems for the developer or QA personnel, raising flags for potentially defective cases. Herzig and Nagappan argue that practitioners prefer precision over recall in their workflow [93]. For example, practitioners will lose faith in a system that makes too many false alarms and cause them extra work, whereas they will trust a system with high precision even if it cannot detect all cases.

Therefore, we recommend that future CPDP research is benchmarked more on precision than recall, while reporting AUC or base measures such as true/false positives/negatives.

6.3 On the use of statistical tests and effect sizes

Performing appropriate statistical tests and computing and reporting relevant effect sizes would help in demonstrating the validity of the conclusions. In our list of 46 primary studies, 10 present effect sizes ([S3], [S13], [S18], [S22], [S26], [S28], [S29], [S40], [S43], [S44]) three of which are published by the same authors ([S26], [S43], [S44]). Before using a statistical test and/or an effect size calculation method however, one should investigate the possibility of using them based on their assumptions. Using inappropriate tests could lead to significance when there are none. Having said these, it is important to consider the power of the utilised tests when drawing conclusions. The less powerful non-parametric tests could show lack of significance due to their weaker power in comparison with parametric tests, however, parametric tests require more assumptions, some of which are violated in many experiments.

Different types of tests have been proposed and employed in software engineering, some of which are described in [56]. While discussing the subject, one must notice the possible disadvantages of using particular tests despite their occasional usefulness. For example, while the Friedman-Nemenyi test, used and encouraged in [7], [56] provides useful insights, it can sometimes be confusing. Specifically, the results of such tests should be interpreted with caution as the ranking procedure does not differentiate between a good performing approach that has a slightly lower performance among the benchmarks on one hand, and an absolute worst performing approach, not even close to the other benchmarks in terms of performance, on the other hand. Hence, the decision between a good and a bad approach becomes more difficult with such tests.

Therefore, for the sake of simplicity and the big picture, the use of Scott-Knott test [102], which groups the methods into distinct classification ranks is encouraged. This test does not suffer from the overlapping groups issue, which is present in several other post hoc tests, one of which is Nemenyi's test.

Further, effect sizes like Cliff's d [103] (which is a measure of how often values in one distribution are larger than the values in a second distribution) are preferable to parametric counterparts, since they do not require any assumptions about the shape or spread of the distributions. Finally, the non-parametric bootstrapping technique [104] (estimation by measuring properties when sampling from an approximating distribution) can be useful in the context of (cross project) defect prediction.

While we are not the first to make this recommendation, we see that this issue is not considered as seriously as it should. Therefore, we encourage further research, once more, to choose statistical tests appropriate for the study design rather than employing them based on their popularity, to check and report on the underlying assumptions of the statistical tests, and to report effect sizes.

6.4 On the lack of search based and multi-objective methods for CPDP

Very few studies have focused on search based approaches in CPDP. We observed that manipulating data can potentially improve the performance as described earlier in the

data approaches section. These approaches open new perspectives to search based methodologies for CPDP. Specifically, we observed only one study [S11] with a multi-objective approach. However, we rarely encounter single-objective tasks in reality, something that is seldom targeted by CPDP. As such, targeting multiple goals with potential focus on data and learning technique optimisation would probably be of great value and a step toward practical applications for research in CPDP.

There is a research gap on search-based and multi-objective methods for CPDP. We recommend further research to exploit and fill this gap

6.5 On the use of metric sets as features

We observe that feature *extraction* is seldom used in CPDP studies. Adapting and applying state of the art techniques in feature extraction and applying the knowledge learnt from the body of work on lower-dimensional manifolds, spectral learning, and random projections may prove to be more useful in building predictors [57]–[59]. Only a few studies [52], [S3], [S15], [S37], [S40] (very limited in some) have considered the feature and metric manipulation by means of feature *selection* and obviously more research in the area is required. Moreover, the feature manipulation approaches can be used in multi-objective methods as one of the data manipulation approaches as discussed earlier.

Most studies utilise software metrics without pre-processing, e.g., feature selection/extraction, even though the complex CPDP algorithms prevent reasoning based on these metrics. There is a research gap on applying state of the art feature extraction techniques and we recommend further research to exploit and fill this gap

6.6 On tuning hyper-parameters of learners

We observed that almost all studies use the default options set by the learning environments for setting the hyper-parameters of learning techniques. A recent study [53] asserted that the majority of the most commonly used classifiers require the setting of at least one parameter. Similar observations are also presented in [1], [54], [55], raising concerns regarding the usage of default parameter values, which, if tuned properly, often lead to better performance.

We recommend that hyper-parameter tuning for existing and proposed approaches in CPDP research should not be considered optional, but rather necessary.

6.7 On the lack of CPDP studies on commercial/closed source systems

There are very few examples that demonstrate the applicability of CPDP approaches in industrial settings. This lack of demonstration of practical applications lead to concerns regarding the actual value of CPDP. Applying the techniques proposed in the literature to modern day software systems especially on the proprietary and closed source software could lead to further progresses in the field. The majority of the studies have focused on open source data, which, despite their great value, might not be representative enough for the software industry as a whole.

While most of our recommendations target the rigour of further studies, this one specifically calls for the need for (industrial) relevance in CPDP studies that are performed in real settings.

6.8 On unfair comparisons of CPDP vs. WPDP

One of the key promises of CPDP is its competitiveness with WPDP, at least as a stop-gap measure. We observe that most WPDP counterparts, which are benchmarked with CPDP methods, usually employ the simplest form of learners. On the other hand, the algorithmic complexity of CPDP methods seems to be increasing with every new proposal. These sophisticated methods are generally not applied to the within project data as benchmarks. This is justifiable in the sense that some of the approaches are specifically designed for CPDP and can not be applied to WPDP. However, this should not prevent/limit the researchers to compare their proposed models to the state of the art in WPDP. Such comparisons would show the real value of CPDP and its current state. Further, it is of great value if the studies use up-to-date and state of the art benchmarks in CPDP for validation and demonstration of their proposed approaches.

Whenever a new technique is proposed for CPDP, we recommend applying that new technique in a WPDP setting as well, for fair benchmarking. If this is not possible, a state-of-the-art WPDP counterpart, e.g., Random Forests, with hyper-parameter tuning should be used. Further, a comparison with a CPDP technique must be presented.

6.9 On the lack of replication packages

There are very few replication packages available for CPDP. The replication packages not only allow researchers to validate the results achieved by individual studies, but also could lead to better and more powerful studies/approaches as follow ups. The implementations may contain small tweaks and details which sometimes are impossible to extract from the proposed algorithms presented in the papers. Documenting and presenting the code alongside the presented algorithms as replication packages could resolve confusions in this regard. The practice of sharing your data and experiments has been recommended in the past [87], [88] and such recommendations has lead to repositories such as PROMISE Repository [89] and SeaCraft [90] among others. In an ideal scenario, best way for technology transfer is to provide tools that can work along with and integrate into programming IDEs to provide insight for the developer/tester.

Further, the research on datasets, specifically compiling new versions of the existing datasets, is encouraged as one of the areas of interest for CPDP. As discussed earlier, the defect datasets and their reliability can change over time, as new defects affecting earlier versions of the software systems are discovered. Different levels of granularity, other types of metrics, other methods of blaming (code and data flow), and datasets from different companies with different practices are highly required.

Publish your scripts and data! We recommend reusable scripts, and whenever possible, the data to be shared whenever a new CPDP approach is proposed. This will help further studies to easily include your technique as

benchmarks, improving the impact of your research. This is also useful for the training of junior researchers.

6.10 On the lack of regression studies in CPDP

Very few studies have considered continuous models. In our list of 30 primary studies which passed the quality assessment phase, only two perform regression analysis to predict for the number of defects in software units. We are not sure as to why research chose to focus on binary classification. Indeed, most defect datasets contain defect counts, but these are converted to binary defect labels in CPDP studies. With regression studies, broader decisions on allocating QA resources can be taken.

There is a research gap on regression based methods for CPDP. We recommend further research to exploit and fill this gap

7 Threats to Validity

It is important to clarify the possible threats that influence the outcomes of our research. The following are the validity threats identified during this SLR.

7.1 Publication Bias

Publication bias denotes the issue of publishing more positive results over negative results [36]. Literature reviews on claims to support or reject a hypothesis will be biased if the original literature is suffering from the publication bias. While some preferences in publishing are useful (e.g., not publishing suspected flawed studies) a tendency toward some outcomes rather than few others leads to biased and possibly incorrect conclusions. Studies with significant results might not always be better than the studies with a null result with respect to quality of design. However, statistically significant results have a much higher chance of getting published.

7.2 Search Terms

Finding all relevant primary studies is always a challenge in any SLR. To address this issue, a detailed search strategy was prepared and presented in our study. Search string was constructed with different terms identified by checking titles and keywords from the relevant papers already known to the authors. Alternative spellings and synonyms for search terms were then added by consulting an expert in the area. Search string was applied to the full text of the papers. Moreover, the applicability of the search string was piloted and the identified studies were compared with the list of studies that were already known and the search string then was altered accordingly. In addition to the automated search in six electronic databases, additional search strategies, i.e., snowballing, was carried out to find other relevant studies that might have been excluded. These procedures provided a high confidence that the majority of the key studies were identified.

7.3 Study Selection Bias

The study selection process was carried out in two phases. In the first round, studies were excluded based on the title and abstract independently by two researchers. The pilot study of the selection process was conducted to place a foundation for better understanding the inclusion/exclusion criteria. Potential disagreements were resolved during the pilot study and inclusion/exclusion criteria were refined. Inter-rater reliability was evaluated to mitigate the threat emerged from the researchers' personal subjective judgment. Agreement between the two researchers was "substantial" for selecting relevant papers from the full set of papers. The selection process was repeated until a full agreement was achieved. When the researchers could not make a decision about a particular study, a third researcher was consulted. In the second phase, studies were excluded based on the full text. Due to this well-established study selection process, it is unlikely that any relevant studies were missed.

7.4 Quality Assessment and Data Extraction

Two researchers independently investigated the quality of each study. Quality assessment criteria were piloted and modified based on the results from the pilot study. Inputs from an expert were taken in the cases where the researchers could not come to an agreement on a particular study. Aforementioned actions mitigated the risk of missing any relevant study. For data extraction, the studies were divided between two researchers; each researcher extracted the data from the relevant studies and the extracted data were rechecked by the other researcher. Issues in data extraction were discussed after the pilot data extraction and the researchers were able to complete the data extraction process following the refinement of the criteria. Extracted data were then inspected by automated scripts to check the correctness of the extracted values across the paper content, improving the validity of our analysis.

7.5 Violin Plots

The number of data points included in our violin plots to synthesise the primary studies varies and is limited to what is reported in primary studies. The number of data rows were also varied in the plots that are used to compare performance in relation to various factors. For example, during investigation of model performance in relation to modelling techniques, RF plot was drawn using 238 data rows while DT was created using only 9 data rows for f-measure. This issue may potentially skew the results and affect our conclusions. The medians of the data groups are selected as the basis of our analysis. This is because no formal statistical tests could be applied to our data when they are grouped together based on different (individual) themes. This point raises another potential threat about the interacting factors involved in the models. Similar to [1], we argue that limiting the analysis to single model factors, despite their usefulness is simplistic. Hall et al. [1] argue that the performance of the models could be impacted by other sources , i.e., the combination of the involved factors in the models are more important than any one of them

alone [1]. Additionally, the good behaviour of a single factor in the context of other factors and their influence on it is not investigated. Having said these, we tried to consider methodological issues that are believed to have impact on the performance [1] by performing additional comparisons in two levels (aggregated and combinations) into the data approaches which are heavily targeted by the studies.

7.6 Meta-analysis and Forest Plots

Similar to the case for the violin plots, the data contributing to the forest plots are not just from the proposed approaches, but the benchmarks also contribute to the data. As such, the data and its size from different cases may vary highly. We have to note that the data for CPDP contains the best of its domain, while the best WP data is not usually presented when the comparisons are made. To have a robust conclusion, both fixed and random effect models were included in our meta-analysis instead of reporting only one model. Despite the disagreements in some cases, the results of fixed and random effect models confirm one another in the majority of cases, which is expected with well-defined targets. The random effect model considers both within and between study errors while the fixed effect model only deals with the within study error. Inclusion of both models is a step toward demonstrating the validity of our conclusions. However, it is worth noting that these are parametric models and their assumptions might not hold for highly skewed data.

7.7 Data Quality

The quality of data for defect prediction is always a threat to the validity of the conclusions [1], [40], [41], [43]. In our study, we did not consider the low quality nature of NASA datasets. We also skipped the maturity test for them as a large portion of the studies have used them in their experiments and their effectiveness has been investigated extensively. Hall et al. [1] excluded the models built on NASA datasets as no maturity information is available for them.

8 Conclusion

The main objective of this SLR was to summarize and synthesize the existing CPDP studies in order to identify what kind of independent variables, modeling techniques, performance evaluation criteria, and approaches are used in building CPDP models. Moreover, this study aimed to explore the predictive performance of cross project defect prediction models compared with that of within project models.

A systematic literature review accompanied by meta-analysis was conducted to fulfil the study objective and answer the defined research questions. After a comprehensive analysis by following a systematic series of steps and assessing the quality of the studies, 30 studies were identified, of which 28 were about categorical models. Beside the guidelines and recommendations for future research presented earlier, the main findings obtained from this SLR are summarized below according to the defined research questions.

- The majority of the CPDP models are constructed using combinations of independent variables. The models that are trained with these combinations seem to perform better than individual metric sets. OO, SCM, and source code text metrics could have acceptable performance in CP context, while process metrics show comparatively low performance.
- NB and LR are the most widely used learning techniques in CPDP. This is true for their use as is and in the context of ensembles too. NB, which is one of the most widely used techniques in CPDP, seems to have an average performance among other modeling techniques. NN, SVM, and DTree are the modeling techniques with the highest median f-measure performance values in CPDP. Ensembles show a different behaviour with f-measure and AUC, where they perform below average for the former and best for the latter.
- Recall, precision, f-measure, pf, and AUC are the most frequently used performance metrics in CPDP.
- The majority of the CPDP approaches address one or more data related issues using various row and column processing methods. Data approaches can increase f-measure and recall performance measures, but they do not seem to have a positive effect on precision.
- Even though in some cases CPDP and WPDP have a comparable performance, WPDP still seems to be better with respect to f-measure, precision, and AUC. If there does not exist enough WP data, however, then CPDP could be a reasonable replacement. Moreover, CPDP models are mostly recall based while having low performances toward precision. Datasets and learners used for the verification of CPDP techniques may be biased, especially NASA and Jureczko datasets.

To conclude, cross project defect prediction model performance is influenced by the way it is built. Specifically, the predictive performance of the model is associated with the independent variables used, modelling techniques on which CPDP models were built and the CPDP approaches followed when building the models, as well as the benchmark datasets used for verification. Cross project defect prediction still remains as a challenge due to its recall based yet low precision performance, but it can potentially achieve comparative predictive performance to within project models when the factors influencing the performance are optimized. In this respect, we hope that this SLR will provide a reference point for conducting future research and the recommendations provided will lead to higher quality research in this area.

Acknowledgments

The authors would like to thank Mr. Olli-Pekka Pakanen from M3S Oulu, who contributed to the initial phase of this literature review by evaluating the primary studies, and the anonymous reviewers, whose invaluable feedback resulted in significant improvements over the initial version of the manuscript. Data underlying this article can be accessed on Zenodo at https://doi.org/10.5281/zenodo.833011

Primary Studies

[S1] O. Mizuno and Y. Hirata, "A cross-project evaluation of text-based fault-prone module prediction," in *Empirical Software Engineering in Practice (IWESEP), 2014 6th International Workshop on*. IEEE, 2014, pp. 43–48.

[S2] S. Watanabe, H. Kaiya, and K. Kaijiri, "Adapting a fault prediction model to allow inter languagereuse," in *Proceedings of the 4th international workshop on Predictor models in software engineering*. ACM, 2008, pp. 19–24.

[S3] P. He, B. Li, X. Liu, J. Chen, and Y. Ma, "An empirical study on software defect prediction with a simplified metric set," *Information and Software Technology*, vol. 59, pp. 170–190, 2015.

[S4] S. Uchigaki, S. Uchida, K. Toda, and A. Monden, "An ensemble approach of simple regression models to cross-project fault prediction," in *Software Engineering, Artificial Intelligence, Networking and Parallel & Distributed Computing (SNPD), 2012 13th ACIS International Conference on*. IEEE, 2012, pp. 476–481.

[S5] Z. He, F. Shu, Y. Yang, M. Li, and Q. Wang, "An investigation on the feasibility of cross-project defect prediction," *Automated Software Engineering*, vol. 19, no. 2, pp. 167–199, 2012.

[S6] J. Wang and Q. Wang, "Analyzing and predicting software integration bugs using network analysis on requirements dependency network," *Requirements Engineering*, pp. 1–24, 2014.

[S7] P. Singh, S. Verma, and O. Vyas, "Cross company and within company fault prediction using object oriented metrics," *International Journal of Computer Applications*, vol. 74, no. 8, 2013.

[S8] A. Panichella, R. Oliveto, and A. De Lucia, "Cross-project defect prediction models: L'union fait la force," in *Software Maintenance, Reengineering and Reverse Engineering (CSMR-WCRE), 2014 Software Evolution Week-IEEE Conference on*. IEEE, 2014, pp. 164–173.

[S9] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and B. Murphy, "Cross-project defect prediction: a large scale experiment on data vs. domain vs. process," in *Proceedings of the the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*. ACM, 2009, pp. 91–100.

[S10] B. Turhan, G. Kocak, and A. Bener, "Data mining source code for locating software bugs: A case study in telecommunication industry," *Expert Systems with Applications*, vol. 36, no. 6, pp. 9986–9990, 2009.

[S11] G. Canfora, A. D. Lucia, M. D. Penta, R. Oliveto, A. Panichella, and S. Panichella, "Defect prediction as a multiobjective optimization problem," *Software Testing, Verification and Reliability*, vol. 25, no. 4, pp. 426–459, 2015.

[S12] A. Pravin and S. Srinivasan, "Detecting software bugs in source code using data mining approach," *National Journal on Advances in Computing and Management*, vol. 3, no. 2, 2012.

[S13] 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*, vol. 55, no. 6, pp. 1101–1118, 2013.

[S14] Y. Liu, T. M. Khoshgoftaar, and N. Seliya, "Evolutionary optimization of software quality modeling with multiple repositories," *IEEE Transactions on Software Engineering*, vol. 36, no. 6, pp. 852–864, 2010.

[S15] L. Yu and A. Mishra, "Experience in predicting fault-prone software modules using complexity metrics," *Quality Technology & Quantitative Management*, vol. 9, no. 4, pp. 421–434, 2012.

[S16] B. Ma, H. Zhang, G. Chen, Y. Zhao, and B. Baesens, "Investigating associative classification for software fault prediction: An experimental perspective," *International Journal of Software Engineering and Knowledge Engineering*, vol. 24, no. 01, pp. 61–90, 2014.

[S17] Z. He, F. Peters, T. Menzies, and Y. Yang, "Learning from open-source projects: An empirical study on defect prediction," in *2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*. IEEE, 2013, pp. 45–54.

[S18] L. Chen, B. Fang, Z. Shang, and Y. Tang, "Negative samples reduction in cross-company software defects prediction," *Information and Software Technology*, vol. 62, pp. 67–77, 2015.

[S19] R. Premraj and K. Herzig, "Network versus code metrics to predict defects: A replication study," in *2011 International Symposium on Empirical Software Engineering and Measurement*. IEEE, 2011, pp. 215–224.

[S20] 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.

[S21] 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*. ACM, 2012, p. 61.

[S22] M. Jureczko and L. Madeyski, "Towards identifying software project clusters with regard to defect prediction," in *Proceedings of the 6th International Conference on Predictive Models in Software Engineering*. ACM, 2010, p. 9.

[S23] S. Herbold, "Training data selection for cross-project defect prediction," in *Proceedings of the 9th International Conference on Predictive Models in Software Engineering*. ACM, 2013, p. 6.

[S24] J. Nam, S. J. Pan, and S. Kim, "Transfer defect learning," in *Proceedings of the 2013 International Conference on Software Engineering*. IEEE Press, 2013, pp. 382–391.

[S25] Y. Ma, G. Luo, X. Zeng, and A. Chen, "Transfer learning for cross-company software defect prediction," *Information and Software Technology*, vol. 54, no. 3, pp. 248–256, 2012.

[S26] D. Ryu, O. Choi, and J. Baik, "Value-cognitive boosting with a support vector machine for cross-project defect prediction," *Empirical Software Engineering*, vol. 21, no. 1, pp. 43–71, 2016.

[S27] 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-Volume 1. IEEE Press, 2015, pp. 801–811.

[S28] M. Jureczko and L. Madeyski, "Cross–project defect prediction with respect to code ownership model: An empirical study," *e-Informatica Software Engineering Journal*, vol. 9, no. 1, 2015.

[S29] P. He, B. Li, and Y. Ma, "Towards cross-project defect prediction with imbalanced feature sets," *arXiv preprint arXiv:1411.4228*, 2014.

[S30] P. He, B. Li, D. Zhang, and Y. Ma, "Simplification of training data for cross-project defect prediction," *arXiv preprint arXiv:1405.0773*, 2014.

[S31] N. Nagappan, T. Ball, and B. Murphy, "Using historical in-process and product metrics for early estimation of software failures," in *2006 17th International Symposium on Software Reliability Engineering*. IEEE, 2006, pp. 62–74.

[S32] M. Thongmak and P. Muenchaisri, "Predicting faulty classes using design metrics with discriminant analysis." in *Software Engineering Research and Practice*, 2003, pp. 621–627.

[S33] L. C. Briand, W. L. Melo, and J. Wust, "Assessing the applicability of fault-proneness models across object-oriented software projects," *IEEE transactions on Software Engineering*, vol. 28, no. 7, pp. 706–720, 2002.

[S34] N. Nagappan, T. Ball, and A. Zeller, "Mining metrics to predict component failures," in *Proceedings of the 28th international conference on Software engineering*. ACM, 2006, pp. 452–461.

[S35] A. E. Camargo Cruz and K. Ochimizu, "Towards logistic regression models for predicting fault-prone code across software projects," in *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*. IEEE Computer Society, 2009, pp. 460–463.

[S36] A. Nelson, T. Menzies, and G. Gay, "Sharing experiments using open-source software," *Software: Practice and Experience*, vol. 41, no. 3, pp. 283–305, 2011.

[S37] J. Nam and S. Kim, "Heterogeneous defect prediction," in *Proceedings of the 2015 10th joint meeting on foundations of software engineering*. ACM, 2015, pp. 508–519.

[S38] M. Chen and Y. Ma, "An empirical study on predicting defect numbers," in *Proc. of SEKE*, 2015, pp. 397–402.

[S39] 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*, pp. 1–35, 2015.

[S40] F. Zhang, A. Mockus, I. Keivanloo, and Y. Zou, "Towards building a universal defect prediction model with rank transformed predictors," *Empirical Software Engineering*, pp. 1–39, 2015.

[S41] X. Jing, F. Wu, X. Dong, F. Qi, and B. Xu, "Heterogeneous cross-company defect prediction by unified metric representation and cca-based transfer learning," in *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*. ACM, 2015, pp. 496–507.

[S42] Y. Zhang, D. Lo, X. Xia, and J. Sun, "An empirical study of classifier combination for cross-project defect prediction," in *Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual*, vol. 2. IEEE, 2015, pp. 264–269.

[S43] D. Ryu, J.-I. Jang, and J. Baik, "A hybrid instance selection using nearest-neighbor for cross-project defect prediction," *Journal of Computer Science and Technology*, vol. 30, no. 5, pp. 969–980, 2015.

[S44] D. Ryu, J. I. Jang, and J. Baik, "A transfer cost-sensitive boosting approach for cross-project defect prediction," *Software Quality Journal*, pp. 1–38, 2015.

[S45] Ç. Çatal, "The use of cross-company fault data for the software fault prediction problem," *Turkish Journal of Electrical Engineering & Computer Sciences*, pp. 3714–3723, 2016.

[S46] P. Singh and S. Verma, "Cross project software fault prediction at design phase," *World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering*, vol. 9, no. 3, pp. 800–805, 2015.

References

[1] 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*, vol. 38, no. 6, pp. 1276–1304, 2012.

[2] B. Kitchenham, R. Pretorius, D. Budgen, O. P. Brereton, M. Turner, M. Niazi, and S. Linkman, "Systematic literature reviews in software engineering–a tertiary study," *Information and Software Technology*, vol. 52, no. 8, pp. 792–805, 2010.

[3] C. Catal and B. Diri, "A systematic review of software fault prediction studies," *Expert systems with applications*, vol. 36, no. 4, pp. 7346–7354, 2009.

[4] D. Radjenović, M. Heričko, R. Torkar, and A. Živkovič, "Software fault prediction metrics: A systematic literature review," *Information and Software Technology*, vol. 55, no. 8, pp. 1397–1418, 2013.

[5] N. E. Fenton and M. Neil, "A critique of software defect prediction models," *IEEE Transactions on software engineering*, vol. 25, no. 5, pp. 675–689, 1999.

[6] K. Herzig and N. Nagappan, "Empirically detecting false test alarms using association rules," in *Proceedings of the 37th International Conference on Software Engineering-Volume 2*. IEEE Press, 2015, pp. 39–48.

[7] 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.

[8] M. Jureczko and D. Spinellis, "Using object-oriented design metrics to predict software defects," *Models and Methods of System Dependability. Oficyna Wydawnicza Politechniki Wrocławskiej*, pp. 69–81, 2010.

[9] D. Radjenović, M. Heričko, R. Torkar, and A. Živkovič, "Software fault prediction metrics: A systematic literature review," *Information and Software Technology*, vol. 55, no. 8, pp. 1397–1418, 2013.

[10] C. Catal and B. Diri, "Investigating the effect of dataset size, metrics sets, and feature selection techniques on

software fault prediction problem," *Information Sciences*, vol. 179, no. 8, pp. 1040–1058, 2009.

[11] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering*. Springer Science & Business Media, 2012.

[12] T. Menzies, J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors," *IEEE transactions on software engineering*, vol. 33, no. 1, pp. 2–13, 2007.

[13] B. Turhan, "On the dataset shift problem in software engineering prediction models," *Empirical Software Engineering*, vol. 17, no. 1-2, pp. 62–74, 2012.

[14] V. R. Basili, F. Shull, and F. Lanubile, "Building knowledge through families of experiments," *IEEE Transactions on Software Engineering*, vol. 25, no. 4, pp. 456–473, 1999.

[15] G. Boetticher, T. Menzies, and T. Ostrand, "Promise repository of empirical software engineering data," *West Virginia University, Department of Computer Science*, 2007.

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

[17] N. Ohlsson and H. Alberg, "Predicting fault-prone software modules in telephone switches," *IEEE Transactions on Software Engineering*, vol. 22, no. 12, pp. 886–894, 1996.

[18] K. El Emam, W. Melo, and J. C. Machado, "The prediction of faulty classes using object-oriented design metrics," *Journal of Systems and Software*, vol. 56, no. 1, pp. 63–75, 2001.

[19] R. Subramanyam and M. S. Krishnan, "Empirical analysis of ck metrics for object-oriented design complexity: Implications for software defects," *IEEE Transactions on software engineering*, vol. 29, no. 4, pp. 297–310, 2003.

[20] T. Gyimothy, R. Ferenc, and I. Siket, "Empirical validation of object-oriented metrics on open source software for fault prediction," *IEEE Transactions on Software engineering*, vol. 31, no. 10, pp. 897–910, 2005.

[21] N. Nagappan and T. Ball, "Static analysis tools as early indicators of pre-release defect density," in *Proceedings of the 27th international conference on Software engineering*. ACM, 2005, pp. 580–586.

[22] N. Nagappan, T. Ball, and A. Zeller, "Mining metrics to predict component failures," in *Proceedings of the 28th international conference on Software engineering*. ACM, 2006, pp. 452–461.

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

[24] R. Moser, W. Pedrycz, and G. Succi, "A comparative analysis of the efficiency of change metrics and static code attributes for defect prediction," in *2008 ACM/IEEE 30th International Conference on Software Engineering*. IEEE, 2008, pp. 181–190.

[25] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[26] B. W. Yap, K. A. Rani, H. A. A. Rahman, S. Fong, Z. Khairudin, and N. N. Abdullah, "An application of oversampling, undersampling, bagging and boosting in handling imbalanced datasets," in *Proceedings of the First International Conference on Advanced Data and Information Engineering (DaEng-2013)*. Springer, 2014, pp. 13–22.

[27] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority oversampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[28] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*. ACM, 2014, p. 38.

[29] J. Cohen, "A coefficient of agreement for nominal scales," in *Educational and Psychological Measurement*, 1960, pp. 37–46.

[30] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, "Data preprocessing for supervised leaning," *International Journal of Computer Science*, vol. 1, no. 2, pp. 111–117, 2006.

[31] J. L. Hintze and R. D. Nelson, "Violin plots: a box plot-density trace synergism," *The American Statistician*, vol. 52, no. 2, pp. 181–184, 1998.

[32] P. Brereton, B. A. Kitchenham, D. Budgen, M. Turner, and M. Khalil, "Lessons from applying the systematic literature review process within the software engineering domain," *Journal of systems and software*, vol. 80, no. 4, pp. 571–583, 2007.

[33] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[34] U. M. Fayyad and K. B. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning." in *IJCAI*, R. Bajcsy, Ed. Morgan Kaufmann, 1993, pp. 1022–1029.

[35] I. Tomek, "Two modifications of cnn," *IEEE Trans. Systems, Man and Cybernetics*, vol. 6, pp. 769–772, 1976.

[36] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," 2007.

[37] M. D'Ambros, M. Lanza, and R. Robbes, "An extensive comparison of bug prediction approaches," in *2010 7th IEEE Working Conference on Mining Software Repositories (MSR 2010)*. IEEE, 2010, pp. 31–41.

[38] 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*. ACM, 2011, pp. 15–25.

[39] A. Mockus, "Amassing and indexing a large sample of version control systems: Towards the census of public source code history." in *MSR*, vol. 9, 2009, pp. 11–20.

[40] G. Liebchen and M. Shepperd, "Data sets and data quality in software engineering: Eight years on," in *Proceedings of the The 12th International Conference on Predictive Models and Data Analytics in Software Engineering*, ser. PROMISE 2016. New York, NY, USA: ACM, 2016, pp. 7:1–7:4. [Online]. Available:

http://doi.acm.org/10.1145/2972958.2972967

[41] S. Hosseini, B. Turhan, and M. Mäntylä, "Search based training data selection for cross project defect prediction," in *Proceedings of the The 12th International Conference on Predictive Models and Data Analytics in Software Engineering*, ser. PROMISE 2016. New York, NY, USA: ACM, 2016, pp. 3:1–3:10. [Online]. Available: http://doi.acm.org/10.1145/2972958.2972964

[42] D. Bowes, T. Hall, and D. Gray, "Dconfusion: a technique to allow cross study performance evaluation of fault prediction studies," *Automated Software Engineering*, vol. 21, no. 2, pp. 287–313, 2014.

[43] S. Hosseini, B. Turhan, and M. Mäntylä, "A benchmark study on the effectiveness of search-based data selection and feature selection for cross project defect prediction," *Information and Software Technology*, Jun 2017. [Online]. Available: http://dx.doi.org/10.1016/j.infsof.2017.06.004

[44] T. Zimmermann, R. Premraj, and A. Zeller, "Predicting defects for eclipse," in *Predictor Models in Software Engineering, 2007. PROMISE'07: ICSE Workshops 2007. International Workshop on*. IEEE, 2007, pp. 9–9.

[45] T. Menzies, J. Greenwald, and A. Frank, "Data mining static code attributes to learn defect predictors," *IEEE transactions on software engineering*, vol. 33, no. 1, pp. 2–13, 2007.

[46] J. Dougherty, R. Kohavi, M. Sahami *et al.*, "Supervised and unsupervised discretization of continuous features," in *Machine learning: proceedings of the twelfth international conference*, vol. 12, 1995, pp. 194–202.

[47] C.-M. Teng, "Correcting noisy data." in *ICML*. Citeseer, 1999, pp. 239–248.

[48] E. Kocaguneli, T. Menzies, A. Bener, and J. W. Keung, "Exploiting the essential assumptions of analogy-based effort estimation," *IEEE Transactions on Software Engineering*, vol. 38, no. 2, pp. 425–438, 2012.

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

[50] L. Hedges and I. Olkin, *Statistical Methods for Meta-analysis*. Academic Press, 1985. [Online]. Available: https://books.google.ca/books?id=brNpAAAAMAAJ

[51] T. Menzies, A. Butcher, D. Cok, A. Marcus, L. Layman, F. Shull, B. Turhan, and T. Zimmermann, "Local versus global lessons for defect prediction and effort estimation," *IEEE Transactions on software engineering*, vol. 39, no. 6, pp. 822–834, 2013.

[52] R. Malhotra and A. J. Bansal, "Fault prediction considering threshold effects of object-oriented metrics," *Expert Systems*, vol. 32, no. 2, pp. 203–219, 2015.

[53] C. Tantithamthavorn, S. McIntosh, A. E. Hassan, and K. Matsumoto, "Automated parameter optimization of classification techniques for defect prediction models," in *Proceedings of the 38th International Conference on Software Engineering*. ACM, 2016, pp. 321–332.

[54] T. Menzies and M. Shepperd, "Special issue on repeatable results in software engineering prediction," 2012.

[55] B. Ghotra, S. McIntosh, and A. E. Hassan, "Revisiting the impact of classification techniques on the performance of defect prediction models," in *Proceedings of the 37th International Conference on Software Engineering-Volume 1*. IEEE Press, 2015, pp. 789–800.

[56] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine learning research*, vol. 7, no. Jan, pp. 1–30, 2006.

[57] S. Wang, T. Liu, and L. Tan, "Automatically learning semantic features for defect prediction," in *Proceedings of the 38th International Conference on Software Engineering*. ACM, 2016, pp. 297–308.

[58] K. Kamvar, S. Sepandar, K. Klein, D. Dan, M. Manning, and C. Christopher, "Spectral learning," in *International Joint Conference of Artificial Intelligence*. Stanford InfoLab, 2003.

[59] Y. Gao and C. Yang, "Software defect prediction based on manifold learning in subspace selection," in *Proceedings of the 2016 International Conference on Intelligent Information Processing*. ACM, 2016, p. 17.

[60] D. A. da Costa, S. McIntosh, W. Shang, U. Kulesza, R. Coelho, and A. Hassan, "A framework for evaluating the results of the szz approach for identifying bug-introducing changes," *IEEE Transactions on Software Engineering*, 2016.

[61] J. Śliwerski, T. Zimmermann, and A. Zeller, "When do changes induce fixes?" in *ACM sigsoft software engineering notes*, vol. 30, no. 4. ACM, 2005, pp. 1–5.

[62] T. Zimmermann, S. Kim, A. Zeller, and E. J. Whitehead Jr, "Mining version archives for co-changed lines," in *Proceedings of the 2006 international workshop on Mining software repositories*. ACM, 2006, pp. 72–75.

[63] C. Williams and J. Spacco, "Szz revisited: verifying when changes induce fixes," in *Proceedings of the 2008 workshop on Defects in large software systems*. ACM, 2008, pp. 32–36.

[64] S. Davies, M. Roper, and M. Wood, "Comparing text-based and dependence-based approaches for determining the origins of bugs," *Journal of Software: Evolution and Process*, vol. 26, no. 1, pp. 107–139, 2014.

[65] M. DAmbros, M. Lanza, and R. Robbes, "Evaluating defect prediction approaches: a benchmark and an extensive comparison," *Empirical Software Engineering*, vol. 17, no. 4-5, pp. 531–577, 2012.

[66] J. E. Jackson, *A user's guide to principal components*. John Wiley & Sons, 2005, vol. 587.

[67] T. Menzies, B. Turhan, A. Bener, G. Gay, B. Cukic, and Y. Jiang, "Implications of ceiling effects in defect predictors," in *Proceedings of the 4th international workshop on Predictor models in software engineering*. ACM, 2008, pp. 47–54.

[68] N. J. Pizzi, A. R. Summers, and W. Pedrycz, "Software quality prediction using median-adjusted class labels," in *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on*, vol. 3. IEEE, 2002, pp. 2405–2409.

[69] R. DerSimonian and N. Laird, "Meta-analysis in clinical trials," *Controlled clinical trials*, vol. 7, no. 3, pp. 177–188, 1986.

[70] R. E. Slavin, "Best-evidence synthesis: An alternative to meta-analytic and traditional reviews," *Educational*

researcher, vol. 15, no. 9, pp. 5–11, 1986.

[71] J. E. Hunter, F. L. Schmidt, and G. B. Jackson, *Meta-analysis: Cumulating research findings across studies*. Sage Publications, Inc, 1982, vol. 4.

[72] G. V. Glass, B. MacGaw, and M. L. Smith, *Meta-analysis in social research*. Sage Beverly Hills, CA., 1984.

[73] R. Coe, "It's the effect size, stupid: What effect size is and why it is important," in *presented at the Annual Conference of the British Educational Research Association*, 2002.

[74] J. Cohen, "Statistical power analysis for the behavioural sciences (rev. ed.)," *New York: Academic*, 1977.

[75] A. P. Field, "Meta-analysis of correlation coefficients: a monte carlo comparison of fixed-and random-effects methods." *Psychological methods*, vol. 6, no. 2, p. 161, 2001.

[76] M. Borenstein, L. V. Hedges, J. Higgins, and H. R. Rothstein, "A basic introduction to fixed-effect and random-effects models for meta-analysis," *Research synthesis methods*, vol. 1, no. 2, pp. 97–111, 2010.

[77] R. D. Riley, J. P. Higgins, and J. J. Deeks, "Interpretation of random effects meta-analyses," *Bmj*, vol. 342, p. d549, 2011.

[78] C. for reviews and dissemination (CRD), *Systematic reviews: CRD's guidance for undertaking reviews in health care*. Centre for Reviews and Dissemination, 2009.

[79] B. Kitchenham, E. Mendes, and G. H. Travassos, "A systematic review of cross-vs. within-company cost estimation studies," in *Proceedings of the 10th international conference on Evaluation and Assessment in Software Engineering*. British Computer Society, 2006, pp. 81–90.

[80] B. A. Kitchenham, E. Mendes, and G. H. Travassos, "Cross versus within-company cost estimation studies: A systematic review," *IEEE Transactions on Software Engineering*, vol. 33, no. 5, 2007.

[81] B. W. Boehm, "Understanding and controlling software costs," *Journal of Parametrics*, vol. 8, no. 1, pp. 32–68, 1988.

[82] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and B. Murphy, "Cross-project defect prediction: a large scale experiment on data vs. domain vs. process," in *Proceedings of the the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*. ACM, 2009, pp. 91–100.

[83] B. Turhan, G. Kocak, and A. Bener, "Data mining source code for locating software bugs: A case study in telecommunication industry," *Expert Systems with Applications*, vol. 36, no. 6, pp. 9986–9990, 2009.

[84] 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.

[85] B. Turhan, A. Tosun, and A. Bener, "Empirical evaluation of mixed-project defect prediction models," in *Software Engineering and Advanced Applications (SEAA), 2011 37th EUROMICRO Conference on*. IEEE, 2011, pp. 396–403.

[86] G. Canfora, A. De Lucia, M. Di Penta, R. Oliveto, A. Panichella, and S. Panichella, "Multi-objective cross-project defect prediction," in *Software Testing, Verification and Validation (ICST), 2013 IEEE Sixth International Conference on*. IEEE, 2013, pp. 252–261.

[87] G. Boetticher, T. Menzies, and T. Ostrand, "Promise repository of empirical software engineering data," *West Virginia University, Department of Computer Science*, 2007.

[88] T. Menzies, E. Kocaguneli, B. Turhan, L. Minku, and F. Peters, *Sharing data and models in software engineering*. Morgan Kaufmann, 2014.

[89] "The promise repository of empirical software engineering data," 2015.

[90] "The seacraft repository of empirical software engineering data," 2017.

[91] A. T. Misirli, B. Caglayan, A. Bener, and B. Turhan, "A retrospective study of software analytics projects: In-depth interviews with practitioners," *IEEE Software*, vol. 30, no. 5, pp. 54–61, 2013. [Online]. Available: https://doi.org/10.1109/MS.2013.93

[92] Y. Rafique and V. B. Misic, "The Effects of Test-Driven Development on External Quality and Productivity: A Meta-Analysis," *Software Engineering, IEEE Transactions on*, vol. 39, no. 6, pp. 835–856, Jun. 2013.

[93] K. Herzig and N. Nagappan, "Empirically detecting false test alarms using association rules," in *Proceedings of the 37th International Conference on Software Engineering - Volume 2*, ser. ICSE '15. Piscataway, NJ, USA: IEEE Press, 2015, pp. 39–48. [Online]. Available: http://dl.acm.org/citation.cfm?id=2819009.2819018

[94] J. E. Hannay, T. Dyb, E. Arisholm, and D. I. Sjberg, "The effectiveness of pair programming: A meta-analysis," *Information and Software Technology*, vol. 51, no. 7, pp. 1110 – 1122, 2009, special Section: Software Engineering for Secure Systems. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950584909000123

[95] B. A. Kitchenham, D. Budgen, and P. Brereton, *Evidence-Based Software Engineering and Systematic Reviews*. Chapman & Hall/CRC, 2015.

[96] M. Borenstein, L. V. Hedges, J. P. T. Higgins, and H. R. Rothstein, *Criticisms of Meta-Analysis*. John Wiley & Sons, Ltd, 2009, pp. 377–387. [Online]. Available: http://dx.doi.org/10.1002/9780470743386.ch43

[97] M. Lanza, A. Mocci, and L. Ponzanelli, "The tragedy of defect prediction, prince of empirical software engineering research," *IEEE Software*, vol. 33, no. 6, pp. 102–105, 2016. [Online]. Available: https://doi.org/10.1109/MS.2016.156

[98] V. Nair, T. Menzies, N. Siegmund, and S. Apel, "Using bad learners to find good configurations," *CoRR*, vol. abs/1702.05701, 2017. [Online]. Available: http://arxiv.org/abs/1702.05701

[99] M. Rees-Jones, M. Martin, and T. Menzies, "Better predictors for issue lifetime," *CoRR*, vol. abs/1702.07735, 2017. [Online]. Available: http://arxiv.org/abs/1702.07735

[100] S. Hosseini, D. Gunarathna, and B. Turhan, "Cpdp slr replication package," Jul. 2017. [Online]. Available: https://doi.org/10.5281/zenodo.833011

[101] "Scitools source code analysis and metrics, understand for java, available at." [Online]. Available: http://www.scitools.com

[102] E. G. Jelihovschi, J. C. Faria, and I. B. Allaman, "Scottknott: a package for performing the scott-knott clustering algorithm in r," *TEMA (São Carlos)*, vol. 15, no. 1, pp. 3–17, 2014.

[103] G. Macbeth, E. Razumiejczyk, and R. D. Ledesma, "Cliff's delta calculator: A non-parametric effect size program for two groups of observations," *Universitas Psychologica*, vol. 10, no. 2, pp. 545–555, 2011.

[104] N. Mittas and L. Angelis, "Comparing cost prediction models by resampling techniques," *Journal of Systems and Software*, vol. 81, no. 5, pp. 616–632, 2008.

[105] N. Nagappan and T. Ball, "Static analysis tools as early indicators of pre-release defect density," in *Proceedings of the 27th international conference on Software engineering*. ACM, 2005, pp. 580–586.