Empirical Software Engineering (2022) 27: 46
https://doi.org/10.1007/s10664-021-10103-4

An extended study on applicability and performance of homogeneous cross-project defect prediction approaches under homogeneous cross-company effort estimation situation

Sousuke Amasaki[1] · Hirohisa Aman[2] · Tomoyuki Yokogawa[1]

Accepted: 9 December 2021 / Published online: 27 January 2022

Abstract

Software effort estimation (SEE) models have been studied for decades. One of serious but typical situations for data-oriented models is the availability of datasets for training models. Cross-company software effort estimation (CCSEE) is a research topic to utilize project data outside an organization. The same problem was identified in software defect prediction research, and Cross-project defect prediction (CPDP) has been studied. CPDP and CCSEE shared the motivation and developed approaches for the same problem. A question arisen that CPDP approaches were applicable to CCSEE. This study explored this question with a survey and an empirical study focusing on homogeneous approaches. We first examined 24 homogeneous CPDP approach implementations provided in a CPDP framework and found more than half of the approaches were applicable. Next, we used the results of two past studies to check whether the applicable CPDP approaches covered strategies taken in past CPDP studies. The empirical experiment was then conducted to evaluate the estimation performance of the applicable CPDP approaches. CPDP approaches were configured with some machine learning techniques if available, and ten CCSEE dataset configurations were supplied for evaluation. The result was also compared with the results of those two past studies to find the commonalities and the differences between CPDP and CCSEE. The answers to the research questions revealed the our CPDP selection covered the strategies that CPDP approaches had taken. The empirical results supported the simple merge with no instance weighting, no feature selection, no data transformation, nor the simple voting. It was not necessarily surprising according CPDP and CCSEE studies, but had not been explored with CPDP approaches under CCSEE situation. A practical implication is: Combine cross-company data to make effort estimates.

Communicated by: Tim Menzies and Mei Nagappan

This article belongs to the Topical Collection: *Inventing the Next Generation of Software Analytics*

✉ Sousuke Amasaki
amasaki@cse.oka-pu.ac.jp

Extended author information available on the last page of the article.

Keywords Cross-company effort estimation · Cross-project defect prediction · Empirical evaluation

1 Introduction

Software effort estimation (SEE) models have been studied for decades (Wen et al. 2012; Idri et al. 2016; Sehra et al. 2017). Estimated efforts are used for many purposes such as cost estimation, budget planning, bidding, and so on. Underestimation may cause serial problems such as cost overruns, schedule delays, and quality degradation. Overestimation may also result in inappropriate project abandonment or missing bids. Researchers still have proposed new models (Mensah et al. 2018a; Phannachitta et al. 2017; Sarro et al. 2016) though many SEE models were proposed so far. SEE models were evaluated under various settings (Sigweni et al. 2016; Minku et al. 2019). In addition to effort estimation, SEE models were considered useful for exploring the best practices (Menzies et al. 2006a). Typical situations such as data missing were also treated as a topic to be studied (Jing et al. 2016).

One of the serious but typical situations for data-oriented models is the availability of datasets for training models. Advanced SEE models developed year by year would be useless if one has no or little data. Furthermore, as quoted "garbage in, garbage out," SEE models can be expected as accurate only if trained with project data similar to a target project (Briand et al. 2000). A software development organization can collect only a few project data as software projects usually need a long duration to be completed. The similarity also suffers from dataset shift (Turhan 2012).

Cross-company software effort estimation (CCSEE) is a research topic to utilize project data outside an organization (called cross-company data) to overcome the data shortage problem. Although early studies conducted this problem with the simple reuse of cross-project data (Briand et al. 2000; Mendes and Kitchenham 2004; Lokan and Mendes 2006), recent studies developed data manipulation techniques. For instance, Turhan and Mendes focused on the similarity of metrics values and proposed an instance selection approach (Turhan and Mendes 2014). Some studies regarded the problem as a transfer learning problem (Pan and Yang 2010) and proposed approaches (Kocaguneli et al. 2015; Mensah et al. 2018b; Krishna and Menzies 2019).

The shortage or absence of training data was also identified in software defect prediction research. Software defect prediction mainly aims to manage software quality efficiently. Fault-prone modules are predicted and prioritized for code review and testing. In an early phase, a project usually has a small number of fixed modules for training a software defect prediction model (Turhan et al. 2009). Cross-project defect prediction (CPDP) is a research topic to utilize project data outside a project (called cross-project data) for prediction. Researchers on CPDP also considered the problem as a transfer learning problem and have proposed new approaches. For instance, Nam et al. proposed a data transformation technique named TCA+ using a transfer component analysis (Nam et al. 2013).

CPDP and CCSEE shared the motivation and developed approaches for the same problem. Approaches developed in these research also looked similar to each other. Even the same instance selection approach using the nearest neighbor algorithm was tested in both CPDP and CCSEE (Turhan et al. 2009; Turhan and Mendes 2014). Nevertheless, most of the proposed approaches in one area have not been evaluated in another area. The difference of assumptions between them might hinder the exchange. A typical difference is what is to be predicted. A binary decision (classification) is the purpose of CPDP while a continuous number (regression) is that of CCSEE. Some difficulties were in fact found for straightforward application as shown in this paper. However, their cross-area applicability deserves to be investigated for progressing both of the research areas.

This study aims to extend our past study (Amasaki et al. 2020), which was also an extended study of another previous study (Amasaki et al. 2020) in terms of a variety of CPDP approaches under evaluation. That study had investigated the applicability and the effectiveness of CPDP in the context of CCSEE.

In this study, we still limited our study on homogeneous CPDP approaches though independent variables often differ across SEE datasets and recent CPDP studies are getting on heterogeneous CPDP approaches (Jing et al. 2015; Nam et al. 2018; Li et al. 2019). These aspects were left for future work. The rest of this paper thus used CPDP approaches to refer to homogeneous CPDP approaches.

We extended the past study to set four research questions as follows:

RQ1: Which CPDP approaches are applicable to CCSEE? CCSEE is different from CPDP in some aspects. For instance, CPDPs can give two (i.e., binary) values while CCSEEs can yield arbitrary positive real numbers. Therefore, for instance, if a CPDP makes a binary prediction based on the ratio of faulty modules in the most similar cross-project, it cannot make an effort estimate. Such differences made difficult to tell what CPDP approaches were applicable. The applicability means that a CPDP approach can process CCSEE data to produce an estimate. We took a manual inspection of some CPDP approaches as their details were the essential key to tell the applicability. This research question shed light on the difficulty and clarified CPDP approaches applicable to CCSEE.

RQ2: How much do the applicable CPDP approaches cover CPDP strategies? While various strategies had been employed in CPDP approaches, the applicable CPDP approaches were selected considering the difficulty from a limited CPDP set. Therefore, the selected set might be biased and missed important strategies. We thus focused on a coverage, which was another perspective independent of the applicability. This question gave answer whether the applicable CPDP approaches covered enough range of CPDP strategies. As we found a catalog of strategies in past studies, the applicable approaches were mapped onto the catalog.

RQ3: Which CPDP approaches can be useful for CCSEE in terms of the estimation performance? As CPDP and CCSEE were different problems, the preference of approaches would have also been different. The applicable CPDP approaches were needed to be evaluated under CCSEE situation empirically. No comprehensive survey on that cross-domain application had exist. We employed an empirical evaluation.

RQ4: Is the effectiveness in CCSEE consistent to the effectiveness in CPDP? If the preference of CPDP approaches was completely different from the preference under CCSEE, it was not reasonable to bring a good CPDP approach to CCSEE. Examining their commonality and difference helped to understand what idea was effective in further study. We thus conducted comparisons between the results in RQ3 and past CPDP study results.

Regarding RQ1, we evaluated the same CPDP approaches as our previous study (Amasaki et al. 2020), and the same ones were selected as applicable. This study newly added RQ2 to ask whether the selected applicable CPDP approaches were enough to the majority of CPDP approaches as the number of them was not large compared with enormous CPDP approaches proposed year by year. We confirmed that our selection covered most of the strategies found in past survey papers. Regarding RQ3, we refined the experimental setup adopted in the previous study to compare with each other more reasonably. The change led to a change in the conclusion we remarked in the previous study. Namely, it supported an ordinal procedure in software effort estimation. RQ4 was a newly added research question that matched our results on the CCSEE situation with past CPDP studies to examine how much a trend found under the CCSEE situation was common under the CPDP situation. This comparative analysis realized the similarity and the difference between CPDP and CCSEE.

2 Related Work

CCSEE is an endeavor to seek project data suitable for estimating a target effort when an effort estimation model suffers from the shortage of training dataset. Although the name suggests bringing project data outside a target company, it actually means project data not here, for instance, from other teams in the same company.

An early study analyzed seven past studies that used Cross-Company (CC) data straightforwardly and pointed out that their conclusions were inconsistent (Kitchenham et al. 2007). A consideration for chronological orders of project data was then noticed. The chronological splitting was thus adopted for evaluation, and it was found that CC data was inferior to Within-Company (WC) data in this usage (Mendes and Lokan 2009). An extended systematic literature survey included it but led to the same conclusion (Mendes et al. 2014). These studies used CC data as they were and left the difference among WC and CC data.

The idea of CCSEE was extended to change CC data, and several strategies of CCSEE approaches have been studied so far. A typical strategy for CCSEE was instance selection. An instance selection approach based on K-nearest neighbors was applied for effort estimation and found as comparable to WC data (Turhan and Mendes 2014). TEAK (Kocaguneli and Menzies 2011) was also an instance selection approach that variably selects an instance subset considering its variance. It was shown that TEAK transferred cross-domain data and achieved better estimation than the same domain data. In another study, they also showed that TEAK could transfer old data (considered as CC data) so that it could perform as well as new (i.e., WC) data (Kocaguneli et al. 2015). Apart from automatic CCSEE, Ferrucci and Gravino proposed an instance selection approach based on experts' opinions and found a promising result (Ferrucci and Gravino 2019). The use of experts' opinions was rarely considered in CPDP approaches probably due to the size of the dataset to be processed. Our study also excluded this type of study as out of scope.

A typical approach for SEE is ensemble learning. For instance, efforts of models based on different algorithms were averaged for estimation (Pospieszny et al. 2018), and a suitable regression model was selected dynamically (de Cabral and JTH 2021). These studies did not evaluate the proposed approaches under CCSEE situation. A probable reason was that a target project data was not used for selecting or transforming CC data. However, the ensemble learning strategy was also studied under CCSEE situation. Parameter tuning techniques for base learners of ensemble effort estimation had been compared with each other (Hosni et al. 2018). That study used an ensemble learning that combines models based on different algorithms. The aggregation of results was conducted with an arithmetic mean and a weighted mean based on a performance ranking.

Regarding dataset characteristics, different organizations collect different metrics for effort estimation. The studies above considered a homogeneous metrics set and were limited to organizations using the same metrics set. Although our study focused on the homogeneous CCSEE, some studies tackled heterogeneous CCSEE. A heterogeneous CCSEE approach combines canonical correlation analysis and restricted Boltzmann machine (Tong et al. 2016). The canonical correlation analysis was also used in a CPDP approach (Jing et al. 2015).

In contrast to CPDP, CCSEE often considered the chronological order of dataset instances expecting the proximity leads the similarity. An instance selection approach called the bellwether moving window was proposed (Mensah et al. 2018b). BMW was sampled from a pool of instances and considered to predict the remaining instances well. They proposed some sampling methods yielding BMWs and investigated their effectiveness.

The chronological setting also brought the idea that uses WC data and CC data together. Minku and Xiao proposed a kind of ensemble learning approach that assumed the chronological order and the availability of small WC data. Subsets of cross-company data were made so that each subset has similar projects in terms of productivity, for instance (Minku and Yao 2012, 2017). Multiple CC models were trained with the subsets and combined with one WC model using an online weighted average. The positive result implied that cross-company and even old projects could play an effective role in the effort estimation of some future projects. They also proposed Dycom that extended the method proposed in the above studies (Minku and Yao 2014; Minku and Hou 2017). Dycom adapts estimates of WC models and CC models through linear functions. Subsets of CC data can be prepared with automatic clustering (Minku et al. 2019). An active learning CCSEE approach was also proposed for using WC and CC data together (Kocaguneli et al. 2013a). It combines TEAK as an instance selection, QUICK (Kocaguneli et al. 2013b) as a feature and instance selection, and a semi-supervised learning. The chronological online learning and the mixing of CC data and WC data hardly happen under CPDP situation.

3 Applicability of CPDP Approaches to CCSEE

3.1 Characteristics of CPDP and CCSEE

Both CPDP and CCSEE are studied to improve predictions where datasets from outside are the main resources for model training. Some of their differences also stem from the differences between software defect prediction and software effort estimation. In addition to their essential difference, namely, the difference between regression and classification, a notable difference is in the data availability. Regarding software defect prediction, no independent variable nor dependent variable is almost unavailable in the early phase of a software project, although the gathering can be almost fully automated. As past projects can provide cross-project data, CPDP is thus a promising technology for defect prediction. In contrast, effort estimation, especially used in the early phase of projection as our study intends, can almost always access past project data of an organization. Furthermore, there is a widely used benchmark such as ISBSG, and the organization can assess which would be better as studied in a past study (Mendes et al. 2008). Our study aims to further improve the estimation accuracy by applying CPDP approaches.

The characteristics of datasets they use are also different as follows:

Feature Types Effort estimation datasets comprise few ratio scale variables and many nominal scale variables while defect prediction datasets hold many ratio scale variables and few nominal scale variables.

Feature Set Variation Different effort estimation datasets tend to collect different variables while defect prediction datasets have almost the same independent variables among different projects if the same unit of a module (e.g., class) is considered.

Feature-Target Relation Software size almost always has a strong relation to effort while features having a strong relation to fault-proneness are not clear.

Feature-Target Linearity The relationship between log-transformed software size and effort can often be explained by log-log regression (e.g., COCOMO model Boehm 1981) on software effort estimation while such linear relationship is not apparent in defect prediction.

These differences might affect the usefulness of CPDP approaches on CCSEE problem. Regarding feature types, CPDP approaches might become less effective because nominal scale variables are scarce and not so varied. In contrast, for feature set variation, we focused on homogeneous CPDP approaches, and it did not influence our study. For feature-target relation, as the effort-size relation always holds and works effectively, CCSEE for seeking common effective features could always find it to improve CCSEE. For feature-target linearity, CPDP approaches might treat more complex relationships than CCSEE and be less effective on CCSEE because there was no room for improving the predictability on a simpler relationship. However, ensembles have been studied for SEE (Idri et al. 2016), we left it in our study.

3.2 RQ1. What CPDP approaches are applicable to CCSEE?

3.2.1 Approach

We planned to conduct an empirical experiment to investigate the performance of CPDP approaches under CCSEE situations. A problem had been that new CPDP approaches are proposed year by year, and it is impossible to follow and check all approaches. We thus explored implementations for comprehensive performance comparisons and found CrossPare (Herbold et al. 2018).

CrossPare is a CPDP comparison framework and provides 24 CPDP approaches proposed from 2008 to 2015. It provides the most comprehensive set of CPDP approach implementations among few CPDP studies providing implementations though newer CPDP approaches were not supported. An advantage is that it enables to conduct performance comparisons under a unified framework, for instance, the same input/output formats. Although CrossPare was rarely used in other studies for performance evaluation, it distributes source code online to enable examination of implementation details of CPDP approaches by other researchers. CrossPare was written in Java and built around Weka (Frank et al. 2016) while CPDP studies often used other languages such as Python. Weka is a widely accepted library in software defect prediction and effort estimation studies. We thus considered that CrossPare was suitable for our purpose and decided to use it as a reference point for our study.

The analysis of CPDP approaches in terms of solutions they employed had seemed effective according to a CPDP survey (Hosseini et al. 2019). However, we roughly reviewed the CPDP approaches and found that the main causes related to the applicability were due to the difference between classification and regression. We thus examined the papers of those approaches and the implementations and then judged their applicability to CCSEE. Discussions regarding their typical strategies for further analysis are provided in the next section.

We developed the evaluation process below to evaluate the applicability:

1. Applicable without modification if:
   - It does *not* use the dependent variable
2. Applicable with modification if:
   - It does *not* utilize the output of inner prediction models
   - Its non-applicable components are removable
3. Otherwise, it is not applicable

The first step checks whether a CPDP approach uses the binary dependent variable to adapt cross-project data to suit a target project data characteristics. A kind of normalization, for instance, would be considered applicable for this criteria. Such CPDP approaches are often combined with arbitrary prediction models and can be configured with a regression model instead of a classification model. For the remained ones, the second step then looks into their details and checks whether its inner prediction models exist. If the outputs of the inner prediction models are not utilized for a decision, for instance, for cross-project data selection, it can also be replaced with arbitrary regression models. Therefore, it is applicable with modification. Some CPDP approaches comprise multiple components, and some components might be impediments. Such CPDP approaches would be applicable if such components can be removed.

3.2.2 Results

As a result, we identified 16 out of 24 applicable CPDP approaches. Most of them can be combined with arbitrary off-the-shelf supervised classification models for defect prediction. The classification models needed to be replaced with off-the-shelf regression models for effort estimation. Except for this inevitable change, 13 out of 16 CPDP approaches are applicable to CCSEE without any modification. Their characteristics and procedures are as follows:

Amasaki15 Amasaki et al. (2015) performs a feature selection and an instance selection, a sort of instance weighting approaches.

This approach modified an active learning method (Kocaguneli et al. 2013b) to deal with cross-project data. This approach first reduces features looking at their popularity in within-project data. Then, outlier instances are removed.

CamargoCruz09 Erika and Ochimizu (2009) performs a dataset transformation.

This approach moves medians of features of cross-project data to medians of corresponding features of a target project data.

Kawata15 Kawata et al. (2015) performs an instance selection.

This approach uses DBSCAN, a clustering algorithm, to find the nearest neighbors of instances of a target project data from cross-project data.

Koshgoftaar08 Khoshgoftaar et al. (2009) constructs an ensemble learning model. Each base learner is built with each of multiple cross-project data with the same arbitrary supervised algorithm. That is, it can work only when multiple cross-project project data are available. The prediction results of the base learners are summarized with majority voting.

The majority voting for binary classification is regarded as the averaging for regression. This approach constructs a prediction model without adjusting cross-project data to a target project data.

Ma12 Ma et al. (2012) performs an instance weighting. This approach first checks the ranges of features of cross-project data. Each feature value of an instance of the cross-project data is counted if it is within the range. Weights for the instances are based on the counts. Although Transfer Naive Bayes was adopted in the original study, arbitrary supervised classification models are supplied in CrossPare.

Menzies11 Menzies et al. (2011) performs an instance selection.

This approach first performs hierarchical clustering. An appropriate cluster at an appropriate hierarchical level is identified according to an instance of cross-project data.

CrossPare supplied WHICH algorithm (Huang et al. 2010) as one of the supervised classification models. Our experiment did not use it because the algorithm assumes a binary dependent variable.

Nam13 Nam et al. (2013) performs a dataset transformation.

This approach first normalizes cross-project data and a target project data according to heuristics. Then, it transforms the cross-project data and the target project data using transfer component analysis.

PHe15 He et al. (2015) performs a feature selection.

This approach first obtains a popular feature subset among prediction models, each of which is constructed with one of the multiple cross-project data. That is, it can work only when multiple cross-project project data are available. These results are used to make the best common feature subset among the cross-project data after removing redundant features. This approach constructs a prediction model without adjusting cross-project data to a target project data.

Turhan09 Turhan et al. (2009) performs an instance selection.

This approach selects the nearest neighbors of a target project data from cross-project data based on Euclidean distance.

Watanabe08 Watanabe et al. (2008) performs a dataset transformation.

This approach standardizes each feature of cross-project data with medians of a target project data.

YZhang15 Zhang et al. (2015b) provides 4 types of ensemble learning approaches. Three out of the four methods are applicable without any modification. (1) AVGVOTE and (2) MAXVOTE first build base learners with different supervised learning methods with the same single cross-project data. AVGVOTE uses the majority voting while MAXVOTE uses the maximum value. The majority voting for binary classification is regarded as the averaging for regression.

(3) Bagging constructs many supervised models based on randomly sampled subsets and averages the predictions. These approaches construct a prediction model without adjusting the cross-project data to a target project data.

ZHe13 He et al. (2013) combines an instance selection technique and an ensemble learning model. The instance selection is first performed at the project data level. The instances of each cross-project data and the instances of target-project data are labeled differently. They are combined and used to train a prediction model based on a tree-based algorithm. This selection technique selects data among multiple cross-project data that are difficult to be discriminated from the target project data. That is, it can work only when multiple cross-project project data are available. The ensemble learning model is then built as follows: Each base learner is first built with the same supervised learning model with each of the selected cross-project data. The prediction results of the base learners are summarized with averaging.

Zimmermann09 Zimmermann et al. (2009) performs an instance selection at the project data level. This approach first makes similarity vectors of all pairs of multiple cross-project data. That is, it can work only when multiple cross-project project data are available. The similarity of a pair of cross-project data is first calculated with representative statistics of features such as mean and standard deviation. Then, each similarity vector is also combined with the predictive performance of a prediction model trained with one cross-project data of the pair on another paired cross-project data. A decision tree model is trained with the similarity vectors and used to decide the best cross-project data.

We also found 3 out of 16 CPDP approaches and a variation of YZhang15 are applicable with a modification. Their characteristics and procedures are as follows:

Herbold13 Herbold (2013) performs an instance selection at the project data level. That is, it can work only when multiple cross-project project data are available. The instance selection selects some of the multiple cross-project data so that their representative statistics of features such as mean and standard deviation are similar to those of a target project data. This approach then assigns weights for mitigating class imbalance. As a regression problem does not suffer from the class imbalance, the weighting procedure was removed for adapting for CCSEE.

Panichella14 Panichella et al. (2014) constructs an ensemble learning model. This approach constructs a prediction model without adjusting cross-project data to a target project data. Each base learner is built with different supervised learning methods with the same single cross-project data. A meta learner is trained with the prediction results of the base learners. The original study adopted logistic regression as the meta learner. We replaced logistic regression with linear regression, which is based on the same linear model as the logistic regression to give an estimate instead of a binary value.

Uchigaki12 Uchigaki et al. (2012) constructs an ensemble learning model. Each base learner is a logistic regression model that uses each feature in a single cross-project data as an independent variable. The prediction results from the base learners are summarized with weighted mean where the weights are based on Matthews correlation coefficients, a prediction performance measure for binary classification. This approach thus constructs a prediction model without adjusting cross-project data to a target project data. We replaced the logistic regression with the linear regression. The Matthews correlation coefficient was also replaced with the Pearson correlation coefficient to treat continuous numbers.

YZhang15 Zhang et al. (2015b) includes a boosting model with AdaBoostM1 algorithm. The AdaBoostM1 algorithm in CrossPare can support classification only. We replaced it with Additive Regression, which is a boosting algorithm supporting regression.

We excluded 8 out of 24 CPDP approaches from our experiment because they could not be adapted for CCSEE. The details are as follows:

Canfora13 Canfora et al. (2013) constructs a prediction model. It was originally proposed to construct a multi-objective prediction model that compromises two performance measures. Logistic regression models with randomly generated coefficients are constructed and refined through a genetic algorithm.

The refinement requires multiple criteria that are dependent on performance measures for binary classification.

Liu10 Liu et al. (2010) constructs a prediction model. Classification models with randomly combined features are generated and refined through genetic programming.

The refinement requires type I and II errors that assume binary classification.

Nam15 Nam and Kim (2015) performs a feature selection and an instance selection.

This approach utilizes a heuristic that a module with high metric values tends to contain a bug. Scores based on the number of features with high values are used for filtering instances and features of cross-project data. It then assigns one of the binary values (i.e., buggy or non-buggy) to the remained instances according to the scores. This reassignment cannot be modified for effort estimation that does not produce a binary value.

Peters12 Peters and Menzies (2012) performs a data transformation.

This approach first randomizes instances of cross-project data using another instance selected with the nearest unlike neighbor. The nearest unlike neighbor takes an instance and finds another instance with the closest distance but the opposite binary dependent value to the instance. It is nonsense to find a nearest unlike neighbor for instances with effort values.

Peters13 Peters et al. (2013) extends Peter12 with instance selection at the project level.

The instance selection method called CLIFF counts the number of buggy/non-buggy instances of cross-project data. This counting procedure cannot be modified for effort estimation because its dependent variable is not binary.

Peters15 Peters et al. (2015) extends Peters13 that was not applicable.

Ryu14 Ryu et al. (2014) constructs an ensemble learning model based on a boosting algorithm.

The instances of cross-project data take weights according to the similarity to a target project data. The boosting algorithm assigns weights to the instances according to the prediction performance of SVM. The weights are used when the boosting algorithm re-samples instances of the cross-project data. The resampling algorithm assumes a binary dependent variable and could not be modified for effort estimation.

Ryu15 Ryu et al. (2015) performs an instance selection.

This approach removes instances of cross-project data based on Mahalanobis distance and Hamming distance. The prediction uses the buggy/non-buggy value of the nearest neighbor instance of a target project data instance.

3.2.3 Answer

We found 16 out of 24 CPDP approaches were applicable to effort estimation. The other approaches were not applicable because a binary dependent variable is assumed. Table 1 tabulates characteristics of the applicable CPDP approaches. We analyzed the results in the table and found some trends as follows:

More than half of the applicable CPDP approaches accept a single cross-project data: If there are multiple data, these approaches combine them into a single big data. These approaches are favorable for CCSEE because it is difficult to prepare many homogeneous cross-company data. The others assume multiple cross-project project data are available. PHe15, Herbold 13, and Zimmermann09 combine the data and use it as a single big data after feature selection or instance weighting. Only ZHe13 and Koshfoftaar08 used them separately for training multiple prediction models.

More than half of the applicable CPDP approaches use a target project data to adjust cross-project data: Feature selection, instance weighting, and data transformation assume the similarity of data characteristics contribute to the predictive performance. The ensemble-based applicable CPDP approaches except for ZHe13 do not refer to a target project data. Ensemble-based approaches look ineffective for CCSEE because a local calibration is expected when an effort estimation model uses cross-company data (Menzies et al. 2017).

Table 1 Characteristics of CPDP approaches applicable to effort estimation

		Strategy			
Approach	required data	feature selection	instance weighting	data transformation	ensemble learning
Amasaki15		x	x		
Kawata15			x		
Ma12			x		
Menzies11			x		
Turhan09			x		
CamargoCruz09	1			x	
Nam13				x	
Watanabe08				x	
Panichella14					x
Uchigaki12					x
YZhang15					x
PHe15		x			
Herbold13			x		
Zimmermann09	>1		x		
ZHe13			x		x
Koshgoftaar08					x

A few of the applicable CPDP approaches combine more than one technique: Amasaki15 and ZHe13 only combine two strategies. The others dedicate to devising one strategy.

> **Answer to RQ1:** Sixteen out of twenty-four CPDP approaches of CrossPare are applicable to CCSEE. More than half of them assume a single cross-project data. More than half use a target project data to adjust cross-project data based on similarity. A few of them combine more than one strategy.

3.3 RQ2. How much do the applicable CPDP approaches cover CPDP strategies?

3.3.1 Approach

Our selection had been limited to the implementations of one framework that includes approaches proposed before 2015. As it was reasonable to examine implementations of some approaches for finding applicable ones, the selected approaches might not cover major strategies adopted in other approaches, especially those proposed after 2015. A thorough examination was still impossible as well as RQ1, and we decided to utilize past study results to examine how much our selected approaches covered strategies of CPDP approaches.

Our examination procedure is based on a survey paper (Hosseini et al. 2019) as follows:

1. Specify strategy-related solutions and CPDP approaches classified into the solutions
2. Map our strategy classification and the specified solutions
3. Check the coverage of the applicable CPDP approaches in terms of matched solutions

As our focus was on the effectiveness of CPDP approaches under CCSEE situations, we also performed an additional examination procedure based on a past study (Kamei et al. 2016) that empirically examined the effectiveness of CPDP strategies. A similar mapping approach as the above was conducted.

3.3.2 Results

Hosseini et al. summarized data-related issues into 8 types and linked CPDP studies to these types (Hosseini et al. 2019). We reviewed and found 2 out of 8 issues were directly related to CPDP approaches. We then mapped the applicable CPDP approaches onto the issues. As the issues were linked to approaches examined in that past study, we simply specify the links. Table 2 shows the two data-related issue types, their solutions, and the cited CPDP approaches in CrossPare. We analyzed the results in the table and got some findings as follows:

Some solutions were popular while others were not: As shown in Table 2, the clustering had only one approach while the filtering had 6 approaches. Note that some approaches comprise multiple solutions and are placed in multiple solutions.

All solutions were matched with our strategy classification in terms of homogeneous CPDP: Although the strategies were not identical to our strategy classification, most of them were matching. There were 6 solutions in total, and 5 out of them were linked to at least one of the applicable CPDP approaches shown in Table 1. The metrics matching was developed

Table 2 Data related issues (Hosseini et al. 2019) and the cited CPDP approaches in CrossPare

Issue	Solution	CPDP approaches
Irrelevant and redundant features	feature selection	PHe15, YZhang15
Data heterogeneity	data transformation	Watanabe08, Herbold13
	filtering	Turhan09, Ryu15, Herbold13, Ma12, Peters15, Ryu14
	normalization	Panichella14, Herbold13, Nam13, Ryu14, Ryu15
	metrics matching	–
	clustering	Herbold13

for heterogeneous CPDP. In fact, CrossPare did not include heterogeneous CPDP, and we did not treat heterogeneous CCSEE (Tong et al. 2016; Hosni et al. 2018).

Kamei et al. examined the effectiveness of four strategies on just-in-time (JIT) CPDP (Kamei et al. 2016). All strategies assume multiple cross-project data. Our selection holds four possible approaches, namely, Herbold13, Zimmermann09, ZHe13, and Koshgoftaar08. We reviewed their relationships to the applicable CPDP approaches as follows:

Similarity-based selection used the similarity between projects for selecting the best one. Although there was no exact approach, the domain agnostic (i.e., metrics-based) similarity selection corresponds to Zimmermann09 in our collection. The domain aware (i.e., context factor-based) approach was absent.

Similarity merge comprises two variations. The one was called the data merging that combines all project data. It was absent in our selection. The another selects multiple data using the similarity-based selection and combines them. The exact approach was absent in our selection, but Herbold13 is based on the same idea.

Universal prediction Zhang et al. (2014, 2016) was referred to as a single model built from the entire set of projects. In that sense, all the applicable CPDP approaches correspond to the same idea though any of those approaches were not similar to the specific clustering and ranking procedure.

Ensemble of models comprises two variations. These variations used voting of prediction models trained with different cross-project data. The one uses simple voting while the other assigns weights based on the similarity between projects. Koshgoftaar08 corresponds to simple voting. As ZHe13 selects some cross-project data before ensembles prediction models, it can be considered as a kind of weighted voting technique.

3.3.3 Answer

Our selection covered all solutions of two data-related issues in the past survey (Hosseini et al. 2019) except for heterogeneous CPDP, which was out of our scope. The simple merge was missed in our collection in comparison to the past study (Kamei et al. 2016). In fact, it is an ordinal approach in SEE including CCSEE. Not a few development teams use a dataset of their department including other teams' project data. Some benchmark datasets such as ISBSG dataset, which is composed of different projects of different domains, are also utilized for estimation. Also, the simple merge was implemented as “ALL” in our previous study (Amasaki et al. 2020) as a baseline to be surpassed. We thus decided to regard it as one CPDP approach in our collection.

Answer to RQ2: The applicable CPDP approaches covered most of the solutions summarized in the survey paper (Hosseini 2019). Heterogeneous models were excluded. The applicable CPDP approaches covered most of the strategies examined in the past study (Kamei et al. 2016) after adding a simple merge.

4 Experimental Setup

As we found 17 CPDP approaches (i.e., 16 from Table 1 and a simple merge) applicable to CCSEE, an experimental setup of our empirical performance evaluation was described in this section for answering RQ3 and RQ4 in the next section.

4.1 CPDP Approach Configurations

The seventeen CPDP approaches were designed for binary classification and needed to be adapted for regression. As CrossPare was implemented around Weka (Frank et al. 2016), we considered using supervised regression models of Weka with those CPDP approaches.

Table 3 shows supervised classification models combined in the selected CPDP approaches. `DecisionTable`, `MultilayerPerceptron`, `RandomForest`, and `RBFNetwork` were applicable as regression models.

Contrastingly, we found no supervised regression model based on Bayes theory and applicable for a regression problem. Therefore, configurations with `NaiveBayes` except

Table 3 Supervised models used in CPDP approaches and their treatments

	Treatment		
Models	As is	Remove	Replace with
`DecisionTable`	x		
`MultilayerPerceptron`	x		
`RandomForest`	x		
`RBFNetwork`	x		
`NaiveBayes`		x	
`BayesNet`			`SMOreg`
`Logistic`			`LinearRegression`
`SMO`			`SMOreg`
`J48`			`M5P`, `REPTree`
`ADTree`			`AdditiveRegression` with `M5P` and `REPTree`

for Koshgoftaar08 were removed from our experiment. Koshgoftaar08 used it as a base learner and needed a replacement. We used MultilayerPerceptron for keeping the variety of base learners pool.

BayesNet was used as a base learner of an ensemble learning in Panichella14 and voting variations of YZhang15. We thus replaced it with SMOreg with RBFKernel for complementing a variety of base learners. Panichella14 also used BayesNet as a meta learner. We removed this configuration because the small number of training data for the meta learner would be too small to train SMOreg, the support vector regression.

For Logistic and SMO, regression versions, namely, LinearRegression and SMOreg, were found. We thus used them as replacements to handle regression problems.

J48 was a tree model and was often used in software defect prediction studies. There were two tree models applicable to regression problems in Weka: M5P and REPTree. M5 was a variant of C4.5 (Zhang et al. 2015a) used in J48 and performed well (Bennin et al. 2016). REPTree was often used in effort estimation studies and showed good performance (Bennin et al. 2016). We thus decided to use them as a replacement for J48. That is, configurations using J48 doubled.

ADTree was a boosting of tree models. No appropriate model was found in Weka. Instead, we found that AdditiveRegression in Weka could make a boosting model with an arbitrary supervised model. It was thus combined with tree models, namely, M5P and REPTree.

Hyper-parameter optimization was often omitted in CrossPare if supervised models were used as base learners of ensemble learning approaches. We kept the implementation and followed it. For easy comparisons, some models were not tuned even when used as a single supervised model in CPDP approaches. RandomForest and REPTree were insensitive to hyper-parameters (Minku and Yao 2013; Tantithamthavorn et al. 2017) and used default parameters. M5P was supplied to AdditiveRegression as a base learner. To see the effect of the ensemble learning approach, and M5P was not tuned. LinearRegression brings a ridge parameter, which would yield a ridge regression beyond a simple regression, and was not tuned. MultilayerPerceptron and RBFNetwork were tuned according to a study (Minku and Yao 2013). The complexity parameter of SMOreg was tuned among 1 and 100. The shrinkage parameter of AdditiveRegression was tuned among 0.1 and 1.0.

4.2 Effort Estimation Datasets

We used the same eight datasets as the comparative study (Kocaguneli and Menzies 2011) in our study. In that comparative study, the authors said that many publications assumed the locality of data could be discovered using a single feature. Also, they considered that data divided into subsets according to locality could be used for within or cross effort modeling. The authors then explored datasets on PROMISE repository to determine the criteria to divide them and reached to use the eight datasets. We followed this idea to prepare within and cross data though each dataset does not necessarily come from one organization.

Table 4 shows their names, criteria for subsets, and sizes. These datasets were freely accessible from PROMISE repository and commonly used for effort estimation studies. Groups of subsets for evaluating transfer learning in effort estimation were prepared by dividing these datasets. We adopted the same criteria for division as the comparative study, namely, project type, center, language type, application type, hardware type, source, and year. Note that all subsets comprise the same variables as we focused on homogeneous

Table 4 Datasets and subsets used for experiment

Dataset	Criterion	Subset	Size
cocomo81	project type	cocomo81e	28
Boehm (1981)		cocomo81o	24
		cocomo81s	11
cocomo81	year	coc6075	30
		coc76test	43
nasa93	development	nasa93_center_1	12
Menzies et al. (2006b)	center	nasa93_center_2	37
		nasa93_center_3	39
nasa93	year	nasa7079	39
		nasa80test	54
desharnais	language type	desharnaisL1	46
Desharnais (1989)		desharnaisL2	25
		desharnaisL3	10
finnish	application type	finnishAppType1	17
Kitchenham and Kansala (1993)		finnishAppType2345	18
kemerer	hardware	kemererHardware1	7
Kemerer (1987)		kemererHardware23456	8
maxwell	application type	maxwellAppType1	10
Maxwell (2002)		maxwellAppType2	29
		maxwellAppType3	18
maxwell	hardware	maxwellHardware2	37
		maxwellHardware3	16
		maxwellHardware5	7
maxwell	source	maxwellSource1	8
		maxwellSource2	54

CCSEE approaches. The suffixes of year-based subsets of cocomo81 and nasa93 represented years from when projects in a subset developed. For instance, coc6075 meant projects developed from 1960 to 1975. coc76test had projects developed from 1976 and used as a testing subset in another study (Kocaguneli et al. 2015). In total, ten dataset configurations were supplied to our experiment.

The size of datasets and the number of subsets were small compared to those of defect prediction datasets. Thus, cocomo76test and nasa80test were also used for training for estimating coc6075 and nasa7079, respectively though it is unrealistic to estimate the efforts of past projects with future projects.

Effort and size-related metrics were log-transformed before prediction. The performance of CPDP approaches was evaluated with their raw metric values. It is a common technique in effort estimation for improving the estimation accuracy and had been often used with these datasets in past studies (Kitchenham and Mendes 2009) and confirmed as effective. From a theoretical point of view, as described in Section 3.1, the log-transformation makes that primary relationship linear. Furthermore, estimates in a raw metric (e.g., man-months) are always positive as the inverse of log-scale values never produces a negative value. These properties were more suitable for effort estimation.

4.3 Performance Measure

SEE models are compared with each other using performance measures. MMRE and PRED(25) had been the most popular basis for this purpose. They are based on the magnitude of relative error (MRE), the absolute difference between actual and predicted efforts divided by the actual value. However, using MRE-based measures had been controversial (Port and Korte 2008) for its bias and called for using other measures (Foss et al. 2003).

Standardized accuracy (SA) (Shepperd and MacDonell 2012) aims to overcome the bias and make comparisons across datasets possible. It got popular in recent studies. SA was defined on an unbiased measure MAE (mean absolute error) as follows:

$$\mathrm{SA} = 1 - \frac{\mathrm{MAE}_{P_i}}{\mathrm{MAE}_{P_0}}$$

MAE_{P_i} represents a mean absolute error of the prediction system P_i. MAE_{P_0} represents a mean absolute error of a prediction system of random guessing. MAE_{P_0} was calculated following Langdon et al. (2016). SA takes more than 0 if an SEE model is better than random guessing. Note that SA often resulted in negative in our study because of cross-company effort estimation.

5 Performance of Applicable CPDP Approaches

5.1 RQ3. Which CPDP approaches can be useful for CCSEE in terms of the estimation performance?

5.1.1 Approach

We conducted an empirical experiment to evaluate the performance of applicable CPDP approaches under a homogeneous CCSEE situation. Figure 1 shows the procedure of the performance evaluation experiment.

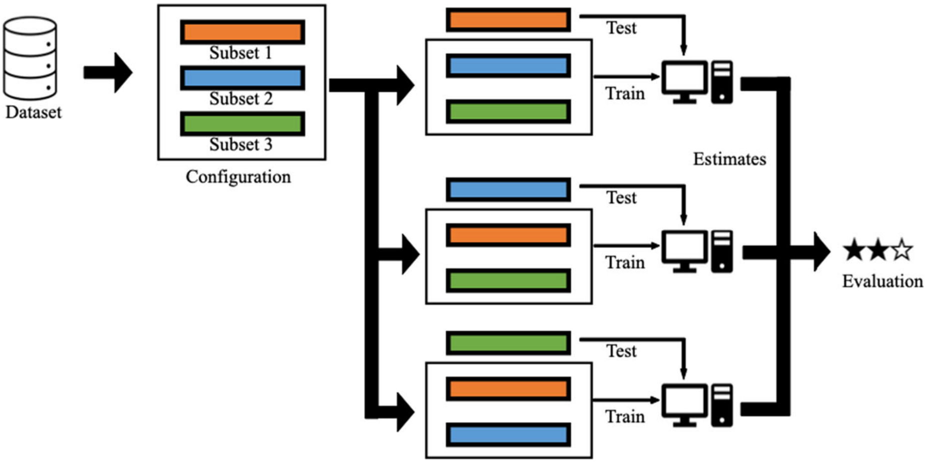

Fig. 1 Procedure of Performance Evaluation Experiment

A dataset was first divided into some subsets according to a configuration shown in Table 4. Each subset played a target project data (i.e., test dataset) to be estimated in turn. The other subsets of the configuration played cross-project data for the target project data. Each CPDP approach configuration (i.e., a combination of approach and supervised learning method) was trained on the cross-project data for effort estimation. The target project data was passed to the trained model to obtain estimates for projects in that data. The estimation results were used for performance evaluation. For one data configuration, the number of obtained estimation results was as same as the multiplication of the number of CPDP approaches and the number of subsets of the data configuration.

Table 4 shows that half of the configurations divided a dataset into 2 subsets though Fig. 1 shows that a configuration divides a dataset into 3 subsets. The difference hindered a straightforward empirical experiment. Table 5 shows the relationship between configuration types and CPDP approach types. Possible combinations are checked. The configurations with 2 subsets could not be used for evaluating CPDP approaches that assume multiple CC data. Because one of the subsets was used for testing as shown in Fig. 1 while only one subset remained.

The comparison of all CPDP approaches was comprehensive and desired. That comparison could be performed only on the configurations with ¿2 subsets as shown in the bottom row of Table 5. The comparison using on all configurations was also desirable in terms of result generality. That comparison could include only CPDP approaches that assume single CC data as shown in the left column of Table 5. For the configurations with ¿2 subsets, they simply combined training subsets. We thought that either of them was not enough and decided to conduct comparisons on both settings.

Half of the dataset configurations had three subsets and could be used for evaluating Herbold13, Koshgoftaar08, PHe15, ZHe13, and Zimmermann09. Therefore, comparisons among all CPDP approaches were conducted with the five dataset configurations. All ten dataset configurations could be used for comparisons among the CPDP approaches that assume single CC data.

The CPDP approach configurations were ranked using Scott-Knott ESD test (Tantithamthavorn et al. 2017). It first produced statistically distinct groups of CPDP configurations regarding their predictive performance results. Some groups with negligible differences in Cohen's *d* effect size were then merged. That is, the performance between any pair of the final groups was not only statistically significant but also non-negligible. They were ordered according to the performance. Note the best configurations of each CPDP approach were used for evaluation. A CPDP approach configuration would be regarded as useful if it were in a higher-ranked group than the best simple merge configuration, introduced in response to RQ2. Cross-validation results were also added as an alternative to within-company SEE to see how a CPDP approach configuration was practically useful. The performance of cross-validation is usually expected to surpass that of CPDPs.

Table 5 Relationship between configuration types and approach types

Configuration	CPDP types	
# of subsets	single	multiple
2	✓	
¿2	✓	✓

5.1.2 Results

We first examined the comparison results based on the experimental setting in Section 4.

Comparison of All CPDP approach configurations is shown in Fig. 2. Y-axis represents the means of SA values, and X-axis represents configuration names. The configuration names comprise a CPDP approach and an adopted supervised learning method. The ranking is based on the results of the experiment on the five dataset configurations having three subsets. Configurations grouped by the same color are considered to have a negligible difference and thus have the same performance. The groups on the left side are higher in performance.

For the configuration names, SR, RN, RF, LR, and A5 specify `SMOreg`, `RBFNetwork`, `RandomForest`, `LinearRegression`, and `AdditiveRegression` with `M5P`, respectively. CV and COMBINE represent the cross-validation and the simple merge, respectively. YZhang15-BAG-M5 specifies the bagging of predictors learned by M5P. YZhang15-AVGVOTE specifies the average voting of predictors learned by different supervised models. Note that the boosting of YZhan15 was absent as it is identical to COMBINE with `AdditiveRegression`, a boosting algorithm. Also, Uchigaki12 was not shown due to its considerably worse performance.

The top group contains CV, COMBINE, and 7 CPDP approach configurations. Most of them used M5 for estimation. Although the y-axis indicated CV was the top approach and COMBINE followed it, the performance of all configurations was statistically equivalent. A CPDP approach configuration would have been useful if it had been significantly better than COMBINE. In fact, no CPDP approach configuration was placed ahead of COMBINE, and the simple merge was considered as the best choice.

The top-ranked approaches used a merged single cross-company data for model construction. Five out of the seven CPDP approaches used instance weighting strategies as shown in Table 1. The rest, namely, YZhang15-BAG and YZhang15-AVGVOTE, used the ensemble model strategy. No data transformation, no feature selection, nor no instance selection at the project level was found in the top group. These CPDP strategies looked harmful because their performance was significantly lower than the simple merge.

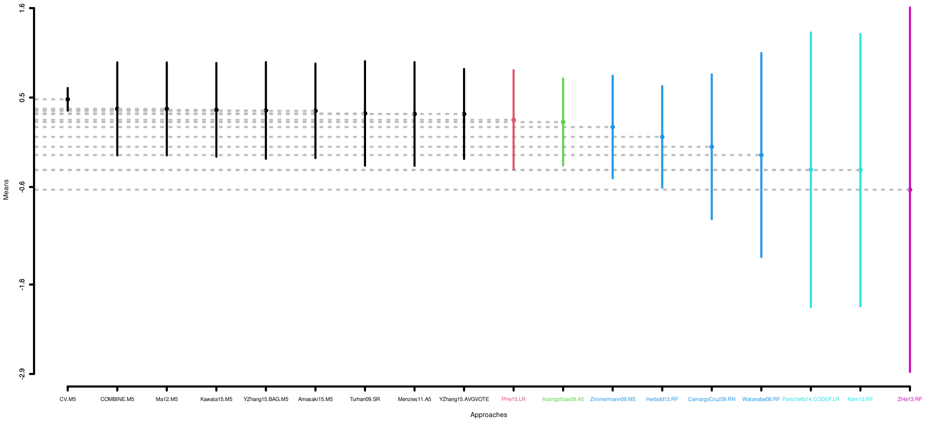

Fig. 2 Ranking of All Applicable CPDP approaches

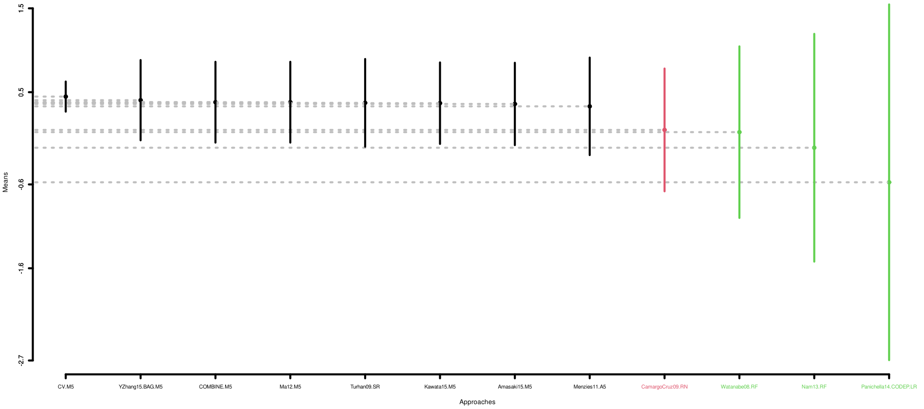

Fig. 3 Ranking of CPDP approaches for Single Cross-Project Data

Comparison of single CC data CPDP approaches is shown in Fig. 3. The ranking is based on the results on the ten dataset configurations. The top group contains CV, COMBINE, and 6 CPDP approach configurations. Most of them used M5 for estimation. Note that YZhang15-AVGVOTE was diminished because it was significantly worse than YZhang15-BAG. The same 6 configurations as Fig. 3 remained in the top group. That is, the difference in the dataset configurations was not critical for evaluation.

Regarding the effect of dataset configuration, the estimation performance depends on what subsets were supplied as a target project data. That is, the difference in the predictability among the subsets might influence the results. We thus explored this aspect through the removal of subsets.

Predictability of data subsets was first glimpsed using COMBINE (i.e., the simple merge) configurations. Figure 4 shows the distributions of SAs among the test subsets. SA takes less than 0 if an effort estimation model is worse than a random guess. It was obvious that the performance of `cocomo81o`, `desharnaisL3`, `kemererHardware1`, and `maxwellAppType1` was nearly the random guess. That is, they were dissimilar to the other subsets. We thus conducted replicated comparisons without these subsets for conclusion robustness.

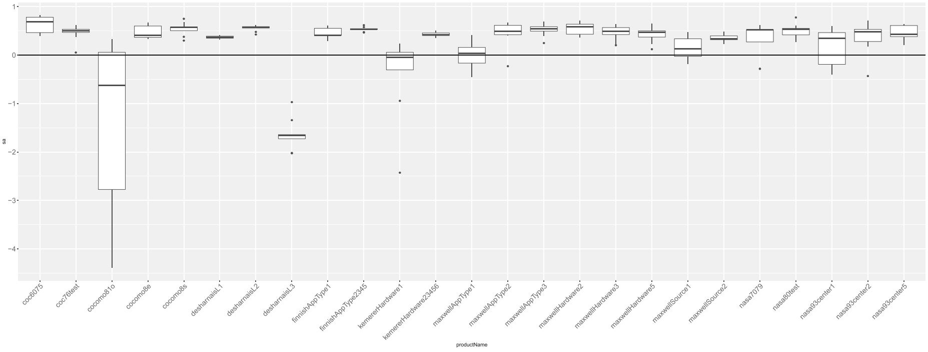

Fig. 4 The estimation performance distributions of test subsets with COMBINE configurations

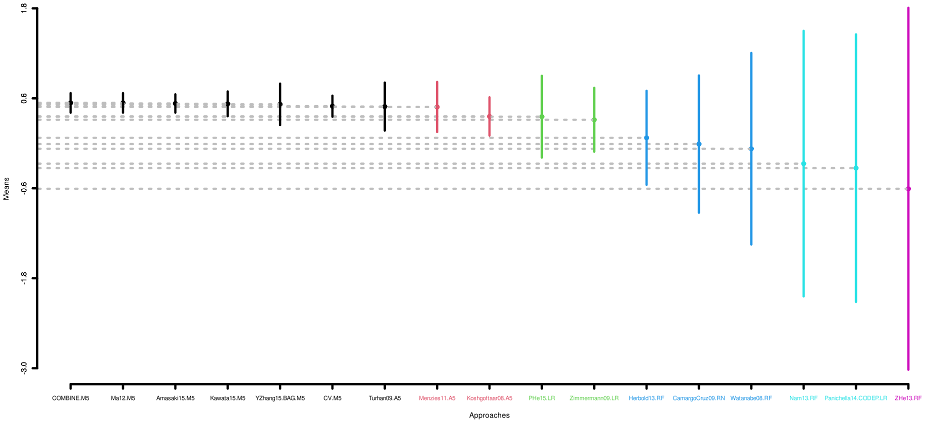

Fig. 5 Ranking of All Applicable CPDP approaches (selected subsets)

Comparison of All CPDP approaches without outlier subsets is shown in Fig. 5. The top group contains CV, COMBINE, and 5 CPDP approach configurations. In comparison to Fig. 2, YZhang15-AVGVOTE was diminished because it was significantly worse than YZhang15-BAG. Menzies11 was moved to the 2nd group. Turhan09 with A5 (Additive Regression with M5P) got better than Turhan09 with M5P. As a result, the top group only contains instance weighting strategy approaches and an ensemble strategy. Furthermore, they did not surpass the simple merge significantly.

Comparison of single CC data CPDP approaches without outlier subsets is shown in Fig. 6.

The top group contains COMBINE and 6 CPDP approach configurations. In comparison to Fig. 2, the same CPDP approaches were ranked in the top group though M5 was often replaced with A5. These CPDP approach configurations were significantly better than the best cross-validation in the 2nd group. The simple merge was still in the top group, and no reason could bring other CPDP approaches for significant performance improvement.

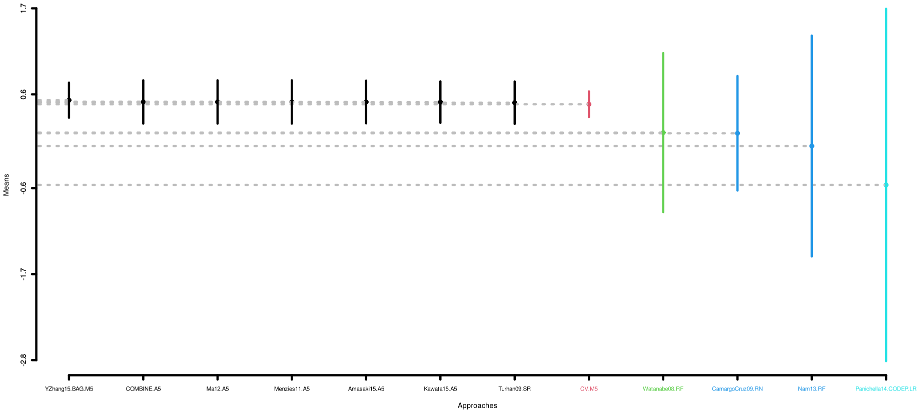

Fig. 6 Ranking of CPDP approaches for Single Cross-Project Data (selected subsets)

5.1.3 Answer

We conducted four types of empirical evaluation of the predictive performance among the applicable CPDP approaches. Figures 2 – 6 brought some observations as follows:

Approaches using (combined) single CC data was better than approaches using multiple CC data: The top groups found in the two evaluation using five dataset configurations, shown in Figs. 2 and 5, contained at least 5 CPDP approaches other than CV and COMBINE. They assume single CC data. No approach assuming multiple CC data was not found in the top group. That is, no clear preference for approaches assuming multiple CC data was not observed.

Outlier elimination affected the members of top-ranked approaches to some extent: The difference between Figs. 2 and 5 was the absence of outliers. When the outliers were inside, seven CPDP approaches other than CV and COMBINE were in the top group. When they were excluded, five CPDP approaches were remained except for CV and COMBINE.

For the comparisons among the approaches assuming single CC data, the top-ranked CPDP approaches were not affected by the outliers. That is, the elimination worked to some extent.

The difference of the dataset configurations affected the members of top-ranked approaches: Regarding the approaches assuming single CC data, the top-ranked approaches in Figs. 2 and 3 were different. They had common 6 CPDP approaches in their top groups. The cause of the difference was the number of dataset configurations used for the evaluations. When the outliers were excluded, the top-ranked approaches in Figs. 5 and 6 were also different. They had common 5 CPDP approaches in their top groups. To be conservative, we could say that the common 5 approaches among the evaluations would be a better answer for comparisons. Also we could support the first observation; the approaches using single CC data were better.

The top-ranked approaches were based on instance weighting and ensemble learning The five common approaches were either instance weighting or ensemble learning in Table 1. The other strategies were worse than them at least on the dataset configurations we employed.

The simple merge was the best choice COMBINE, namely, the simple merge, is an ordinal approach in SEE as described in Section 3.3.3. It was considered as a criterion for evaluating the usefulness of CPDP approaches. We observed that the simple merge placed the highest rank among CPDP approaches and was competitive with CV. Also, it is better than random guessing as SA was greater than 0.0. In summary, we recommend using the ordinal approach. There was nothing better than it.

> **Answer to RQ3:** The simple merge was found useful through conservative analyses. The others were practically ineffective or harmful.

5.2 RQ4. Is the effectiveness in CCSEE consistent to the effectiveness in CPDP?

5.2.1 Approach

The two papers referred to in the previous section (Hosseini et al. 2019; Kamei et al. 2016) analyzed the performance of CPDP approaches they examined. We thus investigated the consistency between our results and the results in those papers.

Hosseini et al. put some claims regarding the performance (Hosseini et al. 2019): The simple merge was not considered the best choice. It was surpassed by other approaches. The ensemble learning was at the medium place of a prediction model ranking. A meta learner (Panichella14-CODE-LR in our study) was better than other ensemble learning approaches and base learners in AUC.

Kamei et al. showed that filtering out cross-project data based on the similarity did not contribute to the improvement (Kamei et al. 2016). The simple merge and the simple voting were enough. They worked as well as WPDP. Note that similar results were found in other following studies (Catolino et al. 2019; Tabassum et al. 2020).

5.2.2 Results

Our results supported the simple merge as it always placed in the top groups as well as CV. We found no evidence that the simple voting worked better than the simple merge. The simple voting of base learners trained with the same supervised learning algorithm (i.e., Koshgotaar08) did not appear in the top groups. The simple voting of base learners trained with different supervised algorithms (i.e., YZhang15-AVGVOTE) was sometimes ranked in the top groups. Introducing a meta learner did not improve the performance as Panichella14-CODE-LR never appeared in the top groups.

We also found that a typical ensemble learning, the bagging (i.e., YZhang15-BAG-M5), always ranked in the top groups. The boosting by Additive Regression (i.e., XXX-A5) was often found in the top groups. These approaches accept a single training data and partially supported the simple merge compared to the simple voting.

Furthermore, filtering out cross-project data at the project level such as Herbold13 never appeared in the top groups. Instance weighting based approaches at the module level, such as Kawata15, always appeared in the top groups. However, this result did not support the instance weighting because they did not improve the simple merge without filtering.

5.2.3 Answer

The results showed that our study results did not fully fit the past studies. The summary was as follows:

Answer to RQ4: Our results partially fit Kamei et al. but not Hosseini et al. The simple merge sufficed and performed as well as cross-validation. The simple voting was weakly supported.

5.3 Discussion

The answers to the RQs were based on limited experiment settings and unable to generalize with experiment results only. We thus provided some possible explanations to help supporting the observations.

For RQ1, the simplicity is a possible explanation for the distribution bias of strategies of applicable approaches towards single cross-project data. Those approaches are favorable because they are applicable even when a single cross-project data is available. The exclusion of approaches using the dependent variable also had some effects on the bias.

For RQ2, our study relied on CrossPare. The good coverage thus seems due to the wide-range of implementations in CrossPare.

For RQ3, a primary possible cause was the small size of subsets in configurations in comparison to that of CPDP. The small number of the subsets up to 3 subsets might also influence on that. Section 5.1.3 showed the use of multiple CC data was not effective. The subset selection approaches from multiple CC data were based on prediction models or representative statistics. Small subsets could provide small information for that purpose and might make such approaches imprecise. The lower ranks of data transformation approaches using single CC data, shown in Fig. 6, can also be explained by the same cause. Median values of small samples, for example, would not represent a precise position to be standardized. Finally, the preference of simple merge against instance weighting in CPDP was reported by (Bin et al. 2017). We think the same result was also observed in the top-ranked approach group.

For RQ4, the differences in comparison to the past studies can be attributed the differences described for RQ3. CPDP and CCSEE treat datasets of different nature and yielded the differences.

6 Implications

The results for RQ1 specified the same CPDP approaches as our previous study (Amasaki et al. 2020). The classification regarding the applicability led to the knowledge that most of the CPDP approaches were applicable. It also revealed that the key to cross-application was a dependency on the target variable type, namely, a binary variable. We clarified a lightweight applicability assessment process in Section 3.1. It can help researchers to seek other applicable CPDP approaches for further cross-domain evaluation. We also showed that CPDP approaches might be adapted to CCSEE. It was an ad hoc process, and a need for developing an adaptation procedure was identified to increase applicable CPDP approaches.

The answer to RQ2 assured that we explored all strategies identified in past CPDP studies. That is, there was no strategy specific to CPDP. All strategies can be examined in CCSEE. Table 1 also suggests what strategies were not explored well. Furthermore, the table revealed that most of the approaches used a single strategy. These new findings are valuable for researchers to classify what strategies they are taking and combining to seek effective CCSEE approaches.

The answer to RQ3 demonstrated that the simple merge was the best. It was different from our previous study (Amasaki et al. 2020). This new finding was valuable for practitioners to confirm their ordinal procedure was the best among the applicable CPDP approaches. They did not need to explore the complicated approaches originally developed for CPDP. The result was a challenge for researchers though those approaches were not developed for CCSEE.

The answer to RQ4 examined that that challenge was common between CPDP and CCSEE. The results did not fully fit the past studies. Our findings were partially supported by an empirical study (Kamei et al. 2016). A survey (Hosseini et al. 2019) did not support our results. The empirical study was limited to some specific approaches as well as our study. The survey explored a wider range of studies and provided the observations. This gap noticed that further empirical evaluations are needed to conclude the usefulness of CPDP approaches on CCSEE. It also drives further replication studies of us with other CPDP approaches.

7 Threats to Validity

Regarding possible biases in our study design, we decided the study design especially for empirical performance evaluation for their recognition in SE community. The evaluation criterion was decided as SA because it was known as an unbiased measure while MRE-based measures were known as biased. Since SA was proposed, it has been used widely in other studies. Scott-Knott ESD was also popular in SDP studies. It was also used in different types of studies such as technical debt determination (Yan et al. 2019) and document categorization (Prana et al. 2019). Regarding the implementation, this study relied on CrossPare framework for evaluation. CrossPare was built around Weka. The tool choice might cause the difference in the results in comparison to, for instance, the results using scikit-learn for Python. However, no library was perfect, and it has often been used by many researchers with a mention of a threat to validity. From these aspects, our choices were considered as a commonly recognized combination. At least, we could avoid an unusual choice.

For comprehensibility, although the coverage of CPDP strategies was confirmed as enough, the progress in the strategies was not explored comprehensively. Therefore, some unknown CPDP approaches might be significantly effective under CCSEE situation. Unknown CPDP approach configurations might also be found as we explored limited combinations of the approaches and the supervised regression models. Our result was not inconsistent with the CPDP paper that investigated the effectiveness of CPDP strategies. Therefore, we think the effectiveness of the simple merge was reliable to some extent.

The number of datasets was small, and different configurations used the same dataset repeatedly. This treatment might cause a bias when we lead a conclusion based on the number of configurations supporting an approach. This threat would be mitigated if not a trivial number of datasets were available. As we followed a previous study to use those dataset configurations, we covered not a small number of datasets if not perfect.

8 Conclusions

This paper examined 24 CPDP approaches regarding its applicability to CCSEE. As a result, we found 16 out of 24 CPDP approaches were applicable. The coverage of those approaches over CPDP studies was examined through reviews based on two CPDP papers. Although it missed heterogeneous CPDP, most of the strategies were included in our selection.

An empirical evaluation of the applicable CPDP approaches was also conducted to observe whether they could improve the effort estimation performance. As a result, we found that the sample merge was the best choice. The others were found ineffective or harmful. This conclusion was as same as our previous study (Amasaki et al. 2020). We further investigated whether this finding was consistent with CPDP studies. The investigation partially supported that the simple merge was considered a promising CPDP approach and might be more effective than the others.

A practical implication from this study is very simple: Combine CC data and use it as if it was WC data. The result also supported a past study (Mendes et al. 2014): there was no strong evidence to support the clear advantage of WC data.

Section 2 shows the limitation of our study. The chronological order was not a problem in CPDP and was not treated seriously in our study. Although it was not applicable to a cross-domain situation we considered, a practical evaluation of CPDP approaches under CCSEE situation will help to evaluate the applicable CPDP approaches. As described, this study only examined a limited set of CPDP approaches. Future work includes updating CPDP set and examining the effectiveness periodically. CrossPare helps this plan with its unified framework to add more CPDP approaches. In addition, we will carry out comparisons to existing CCSEE approaches such as (Kocaguneli and Menzies 2011) and additional performance evaluations on ISBSG dataset, which is often referenced as a benchmark at effort estimation.

Acknowledgments This work was partially supported by JSPS KAKENHI under Grant No. 18K11246, No. 21K11831, and No. 21K11833.

References

Amasaki S, Kawata K, Yokogawa T (2015) Improving Cross-Project defect prediction methods with data simplification. In: Proceedings of SEAA '15. IEEE, pp 96–103

Amasaki S, Aman H, Yokogawa T (2020) An exploratory study on applicability of cross project defect prediction approaches to cross-company effort estimation. In: Proceedings of PROMISE, Association for Computing Machinery, pp 71–80

Bennin KE, Toda K, Kamei Y, Keung J, Monden A, Ubayashi N (2016) Empirical evaluation of cross-release effort-aware defect prediction models. In: Proceedings of International Conference on Software Quality, Reliability and Security, pp 214–221

Bin Y, Zhou K, Lu H, Zhou Y, Xu B (2017) Training data selection for cross-project defection prediction: Which approach is better? In: Proceedings of International Symposium on Empirical Software Engineering and Measurement, ACM, pp 354–363

Boehm B (1981) Software engineering economics. Prentice-Hall

Briand LC, Langley T, Wieczorek I (2000) A replicated assessment and comparison of common software cost modeling techniques. In: Proceedings of ICSE. IEEE, pp 377–386

Canfora G, De Lucia A, Di Penta M, Oliveto R, Panichella A, Panichella S (2013) Multi-objective Cross-Project Defect Prediction. In: Proceedings of ICST '13. IEEE, pp 252–261

Catolino G, Di Nucci D, Ferrucci F (2019) Cross-project just-in-time bug prediction for mobile apps: An empirical assessment. In: 2019 IEEE/ACM 6Th international conference on mobile software engineering and systems (MOBILESoft), pp 99–110. https://doi.org/10.1109/MOBILESoft.2019.00023

de Cabral A, JTH OliveiraAL (2021) Ensemble effort estimation using dynamic selection. J Syst Softw 175:110904

Desharnais JM (1989) Analyse statistique de la producivité des projets de développment en informatique à partir de la technique des points de fonction. Master's thesis, Université du Québec à Montréal

Erika CCA, Ochimizu K (2009) Towards logistic regression models for predicting fault-prone code across software projects. In: Proceedings of ESEM '09. IEEE, pp 460–463

Ferrucci F, Gravino C (2019) Can expert opinion improve effort predictions when exploiting cross-company datasets? - a case study in a small/medium company. In: Proceedings of Product-Focused Software Process Improvement. Springer, pp 280–295

Foss T, Stensrud E, Kitchenham B, Myrtveit I (2003) A simulation study of the model evaluation criterion MMRE. IEEE Trans Softw Eng 29(11):985–995

Frank E, Hall MA, Witten IH (2016) The weka workbench. online appendix for "data mining: Practical machine learning tools and techniques" https://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf

He P, Li B, Liu X, Chen J, Ma Y (2015) An empirical study on software defect prediction with a simplified metric set. Inf Softw Technol 59:170–190

He Z, Peters F, Menzies T, Yang Y (2013) Learning from Open-Source projects: an empirical study on defect prediction. In: Proceedings of ESEM '13. IEEE, pp 45–54

Herbold S (2013) Training data selection for cross-project defect prediction. In: Proceedings of PROMISE '13. ACM, New York, pp 6:1–6:10

Herbold S, Trautsch A, Grabowski J (2018) A comparative study to benchmark Cross-Project defect prediction approaches. IEEE Trans Softw Eng 44(9):811–833

Hosni M, Idri A, Abran A, Nassif AB (2018) On the value of parameter tuning in heterogeneous ensembles effort estimation. Soft Comput 22(18):5977–6010

Hosseini S, Turhan B, Gunarathna D (2019) A systematic literature review and Meta-Analysis on cross project defect prediction. IEEE Trans Softw Eng 45(2):111–147

Huang L, Port D, Wang L, Xie T, Menzies T (2010) Text mining in supporting software systems risk assurance. In: Proceedings of International Conference on Automated Software Engineering, ASE '10. ACM, pp 163–166

Idri A, Hosni M, Abran A (2016) Systematic literature review of ensemble effort estimation. J Syst Softw 118:151–175

Jing X, Wu F, Dong X, Qi F, Xu B (2015) Heterogeneous cross-company defect prediction by unified metric representation and CCA-based transfer learning. In: Proceedings of FSE '15. ACM, pp 496–507

Jing X, Qi F, Wu F, Xu B (2016) Missing data imputation based on low-rank recovery and semi-supervised regression for software effort estimation. In: Proceedings of ICSE. IEEE, pp 607–618

Kamei Y, Fukushima T, McIntosh S, Yamashita K, Ubayashi N, Hassan AE (2016) Studying just-in-time defect prediction using cross-project models. Empir Softw Eng 21(6):2072–2106. https://doi.org/10.1007/s10664-015-9400-x

Kawata K, Amasaki S, Yokogawa T (2015) Improving relevancy filter methods for cross-project defect prediction. In: Proceedings of ACIT-CSI '15. Springer, pp 2–7

Kemerer CF (1987) An empirical validation of software cost estimation models. Commun ACM 30(5):416–429

Khoshgoftaar TM, Rebours P, Seliya N (2009) Software quality analysis by combining multiple projects and learners. Softw Qual J 17(1):25–49

Kitchenham B, Kansala K (1993) Inter-item correlations among function points. In: Proceedings of METRICS '93, pp 11–14

Kitchenham B, Mendes E (2009) Why comparative effort prediction studies may be invalid. In: Proceedings of PROMISE, p 5

Kitchenham BA, Mendes E, Travassos GH (2007) Cross versus Within-Company cost estimation studies: a systematic review. IEEE Trans Softw Eng 33(5):316–329

Kocaguneli E, Menzies T (2011) How to Find Relevant Data for Effort Estimation? In: Proceedings of ESEM '11. IEEE pp 255–264

Kocaguneli E, Cukic B, Menzies T, Lu H (2013a) Building a second opinion: learning cross-company data. In: Proceedings of ESEM '13. ACM, pp 1–10

Kocaguneli E, Menzies T, Keung J, Cok D, Madachy R (2013b) Active learning and effort estimation: Finding the essential content of software effort estimation data. IEEE Trans Softw Eng 39(8):1040–1053

Kocaguneli E, Menzies T, Mendes E (2015) Transfer learning in effort estimation. Empir Softw Eng 20(3):813–843

Krishna R, Menzies T (2019) Bellwethers: A baseline method for transfer learning. IEEE Trans Softw Eng 45(11):1081–1105

Langdon WB, Dolado J, Sarro F, Harman M (2016) Exact mean absolute error of baseline predictor, MARP0. Inf Softw Technol 73:16–18

Li Z, Jing XY, Zhu X, Zhang H, Xu B, Ying S (2019) On the multiple sources and privacy preservation issues for heterogeneous defect prediction. IEEE Trans Softw Eng 45(4):391–411

Liu Y, Khoshgoftaar TM, Seliya N (2010) Evolutionary optimization of software quality modeling with multiple repositories. IEEE Trans Softw Eng 36(6):852–864

Lokan C, Mendes E (2006) Cross-company: Single-company effort models using the ISBSG database A further replicated study. In: Proceedings of ISESE '06, vol 2006, pp 75–84E

Ma Y, Luo G, Zeng X, Chen A (2012) Transfer learning for cross-company software defect prediction. Inf Softw Technol 54(3):248–256

Maxwell KD (2002) Applied statistics for software managers. Prentice Hall

Mendes E, Kitchenham B (2004) Further comparison of cross-company and within-company effort estimation models for Web applications. In: Proceedings of METRIC '04, pp 348–357

Mendes E, Lokan C (2009) Investigating the use of chronological splitting to compare software cross-company and single-company effort predictions: a replicated study. In: Proceedings of EASE. ACM, pp 11–20

Mendes E, Di Martino S, Ferrucci F, Gravino C (2008) Cross-company vs. single-company web effort models using the Tukutuku database: An extended study. J Syst Softw 81(5):673–690

Mendes E, Kalinowski M, Martins D, Ferrucci F, Sarro F (2014) Cross-vs. Within-company cost estimation studies revisited: An extended systematic review. In: Proceedings of EASE. ACM

Mensah S, Keung J, Bosu MF, Bennin KE (2018a) Duplex output software effort estimation model with self-guided interpretation. Inf Softw Technol 94:1–13

Mensah S, Keung J, MacDonell SG, Bosu MF, Bennin KE (2018b) Investigating the significance of the bellwether effect to improve software effort prediction: Further empirical study. IEEE Trans Reliab 67(3):1176–1198

Menzies T, Chen Z, Hihn J, Lum K (2006a) Selecting best practices for effort estimation. IEEE Trans Softw Eng 32(11):883–895

Menzies T, Chen Z, Hihn J, Lum K (2006b) Selecting best practices for effort estimation. IEEE Trans Softw Eng 32(11):883–895

Menzies T, Butcher A, Marcus A, Zimmermann T, Cok D (2011) Local versus global models for effort estimation and defect prediction. In: Proceedings of ASE '11. IEEE, pp 343–351

Menzies T, Yang Y, Mathew G, Boehm B, Hihn J (2017) Negative results for software effort estimation. Empir Softw Eng 22(5):2658–2683

Minku LL, Hou S (2017) Clustering dycom. In: Proceedings of PROMISE '17. ACM, pp 12–21

Minku LL, Yao X (2012) Can Cross-company Data Improve Performance in Software Effort Estimation? In: Proceedings of PROMISE '12. ACM, pp 69–78

Minku LL, Yao X (2013) Ensembles and locality: Insight on improving software effort estimation. Inf Softw Technol 55(8):1512–1528

Minku LL, Yao X (2014) How to make best use of cross-company data in software effort estimation? In: Proceedings of ICSE. ACM, pp 446–456

Minku LL, Yao X (2017) Which models of the past are relevant to the present? a software effort estimation approach to exploiting useful past models. Autom Softw Eng 24(3):499–542

Minku LL, Bowes D, Shihab E, Turhan B (2019) A novel online supervised hyperparameter tuning procedure applied to cross-company software effort estimation. Empir Softw Eng 24:3153–3204

Nam J, Kim S (2015) CLAMI Defect prediction on unlabeled datasets (t). In: Proceedings of ASE '15. IEEE, pp 452–463

Nam J, Pan SJ, Kim S (2013) Transfer defect learning. In: Proceedings of ICSE '13. IEEE, pp 382–391

Nam J, Fu W, Kim S, Menzies T, Tan L (2018) Heterogeneous defect prediction. IEEE Trans Softw Eng 44(9):874–896

Pan SJ, Yang Q (2010) A survey on transfer learning. IEEE Trans Knowl Data Eng 22(10):1345–1359

Panichella A, Oliveto R, De Lucia A (2014) Cross-project defect prediction models: L'Union fait la force. In: Proceedings of CSMR-WCRE '14. IEEE, pp 164–173

Peters F, Menzies T (2012) Privacy and utility for defect prediction: experiments with MORPH. In: Proceedings of ICSE '12. IEEE, pp 189–199

Peters F, Menzies T, Gong L, Zhang H (2013) Balancing privacy and utility in Cross-Company defect prediction. IEEE Trans Softw Eng 39(8):1054–1068

Peters F, Menzies T, Layman L (2015) LACE2: Better Privacy-Preserving data sharing for cross project defect prediction. In: Proceedings of ICSE '15. IEEE, pp 801–811

Phannachitta P, Keung J, Monden A, Matsumoto K (2017) A stability assessment of solution adaptation techniques for analogy-based software effort estimation. Empir Softw Eng 22(1):474–504

Port D, Korte M (2008) Comparative studies of the model evaluation criterions MMRE and PRED in software cost estimation research. In: Proceedings of International Symposium on Empirical Software Engineering and Measurement. ACM, pp 51–60

Pospieszny P, Czarnacka-Chrobot B, Kobylinski A (2018) An effective approach for software project effort and duration estimation with machine learning algorithms. J Syst Softw 137:184–196

Prana GAA, Treude C, Thung F, Atapattu T, Lo D (2019) Categorizing the Content of GitHub README Files. Empir Softw Eng 24:1296–1327

Ryu D, Choi O, Baik J (2014) Value-cognitive boosting with a support vector machine for cross-project defect prediction. Empir Softw Eng 21(1):1–29

Ryu D, Jang JI, Baik J (2015) A hybrid instance selection using nearest-neighbor for cross-project defect prediction. J Comput Sci Technol 30(5):969–980

Sarro F, Petrozziello A, Harman M (2016) Multi-objective software effort estimation. In: Proceedings of ICSE. ACM, pp 619–630

Sehra SK, Brar YS, Kaur N, Sehra SS (2017) Research patterns and trends in software effort estimation. Inf Softw Technol 91:1–21

Shepperd MJ, MacDonell S (2012) Evaluating prediction systems in software project estimation. Inf Softw Technol 54(8):820–827

Sigweni B, Shepperd M, Turchi T (2016) Realistic assessment of software effort estimation models. In: Proceedings of EASE '16. ACM

Tabassum S, Minku LL, Feng D, Cabral GG, Song L (2020) An investigation of cross-project learning in online just-in-time software defect prediction. In: ACM/IEEE 42Nd international conference on software engineering, New York, pp 554–565. https://doi.org/10.1145/3377811.3380403

Tantithamthavorn C, McIntosh S, Hassan AE, Matsumoto K (2017) An empirical comparison of model validation techniques for defect prediction models. IEEE Trans Softw Eng 43(1):1–18

Tong S, He Q, Chen Y, Yang Y, Shen B (2016) Heterogeneous Cross-Company effort estimation through transfer learning. In: Proceedings of APSEC '16. IEEE, pp 169–176

Turhan B (2012) On the dataset shift problem in software engineering prediction models. Empir Softw Eng 17(1-2):62–74

Turhan B, Mendes E (2014) A comparison of Cross-Versus Single-Company effort prediction models for web projects. In: Proceedings of SEAA '14. IEEE, pp 285–292

Turhan B, Menzies T, Bener AB, Di Stefano J (2009) On the relative value of cross-company and within-company data for defect prediction. Empir Softw Eng 14(5):540–578

Uchigaki S, Uchida S, Toda K, Monden A (2012) An ensemble approach of simple regression models to Cross-Project fault prediction. In: Proceedings of SNPD '12. IEEE, pp 476–481

Watanabe S, Kaiya H, Kaijiri K (2008) Adapting a fault prediction model to allow inter languagereuse. In: Proceedings of PROMISE '08. ACM, New York, pp 19–24

Wen J, Li S, Lin Z, Hu Y, Huang C (2012) Systematic literature review of machine learning based software development effort estimation models. Inf Softw Technol 54(1):41–59

Yan M, Xia X, Shihab E, Lo D, Yin J, Yang X (2019) Automating Change-level Self-admitted Technical Debt Determination. IEEE Trans Softw Eng 45:1221–1229

Zhang F, Mockus A, Keivanloo I, Zou Y (2014) Towards building a universal defect prediction model. In: Proceedings of MSR, pp 182–191

Zhang F, Mockus A, Keivanloo I, Zou Y (2016) Towards building a universal defect prediction model with rank transformed predictors. Empir Softw Eng 21(5):2107–2145

Zhang W, Yang Y, Wang Q (2015a) Using bayesian regression and em algorithm with missing handling for software effort prediction. Inf Softw Technol 58:58–70

Zhang Y, Lo D, Xia X, Sun J (2015b) An empirical study of classifier combination for Cross-Project defect prediction. In: Proceedings of COMPSAC '15. IEEE, pp 264–269

Zimmermann T, Nagappan N, Gall H, Giger E, Murphy B (2009) Cross-project defect prediction: a large scale experiment on data vs. domain vs. process. In: Proceedings of ESEC/FSE '09. ACM, pp 91–100

Affiliations

Sousuke Amasaki[1] · **Hirohisa Aman**[2] · **Tomoyuki Yokogawa**[1]

Hirohisa Aman
aman@ehime-u.ac.jp

Tomoyuki Yokogawa
t-yokoga@cse.oka-pu.ac.jp

[1] Okayama Prefectural University, Soja, Japan

[2] Center for Information Technology, Ehime University, Matsuyama, Japan