How to Make Best Use of Cross-Company Data for Web Effort Estimation?

Leandro Minku*, Federica Sarro†, Emilia Mendes‡, and Filomena Ferrucci§
*School of Computer Science, University of Birmingham, UK
*Department of Computer Science, University of Leicester, UK
†CREST, Department of Computer Science, University College London, UK
‡Blekinge Institute Technology, Sweden
‡University of Oulu, Finland
§Department of Computer Science, University of Salerno, Italy
l.l.minku@cs.bham.ac.uk, f.sarro@ucl.ac.uk, emilia.mendes@bth.se, fferrucci@unisa.it

***Abstract*—[Context]: The numerous challenges that can hinder software companies from gathering their own data have motivated over the past 15 years research on the use of cross-company (CC) datasets for software effort prediction. Part of this research focused on Web effort prediction, given the large increase worldwide in the development of Web applications. Some of these studies indicate that it may be possible to achieve better performance using CC models if some strategy to make the CC data more similar to the within-company (WC) data is adopted. [Goal]: This study investigates the use of a recently proposed approach called Dycom to assess to what extent Web effort predictions obtained using CC datasets are effective in relation to the predictions obtained using WC data when explicitly mapping the CC models to the WC context. [Method]: Data on 125 Web projects from eight different companies part of the Tukutuku database were used to build prediction models. We benchmarked these models against baseline models (mean and median effort) and a WC base learner that does not benefit of the mapping. We also compared Dycom against a competitive CC approach from the literature (NN-filtering). We report a company-by-company analysis. [Results]: Dycom usually managed to achieve similar or better performance than a WC model while using only half of the WC training data. These results are also an improvement over previous studies that investigated the use of different strategies to adapt CC models to the WC data for Web effort estimation. [Conclusions]: We conclude that the use of Dycom for Web effort prediction is quite promising and in general supports previous results when applying Dycom to conventional software datasets.**

I. Introduction

Some of the early cost estimation studies (e.g., [1][13]) suggested that general-purpose models such as COCOMO [1] and SLIM [30] needed to be calibrated to specific companies before they could be used effectively. This view was also supported by Kok et al. [16], which, as per the proposals made by DeMarco [5], suggested that cost estimation models should be developed only from within-company (WC) data. However, problems may arise when relying on WC data [2], [14], e.g.: (1) the time required to accumulate enough data on past projects from a single company may be prohibitive; (2) by the time the dataset is large enough to be of use, technologies used by the company may have changed, and older projects may no longer be representative of current practices; and (3) care is necessary as data needs to be collected in a consistent manner; this means that resources need to be used so to ensure that quality control procedures during data collection are in place.

These problems motivated the use of cross-company (CC) models for effort estimation, productivity benchmarking and defect prediction. CC models are models built using CC datasets, which are datasets containing data from several companies. There are several CC software datasets available to be used for effort and defect prediction (http://openscience.us/repo/) and even organisations worldwide that use large proprietary CC datasets with tool support to provide estimation and benchmarking services. An example is the International Software Benchmarking Standards Group (ISBSG) (www.isbsg.org), which provides tools to estimate effort and benchmark productivity using their CC dataset (i.e., ISBSG dataset). They also sell the CC dataset to those companies that wish to use their own tools for estimation and benchmarking purposes.

Several studies have provided comparisons between the accuracy of CC and WC predictions. By the end of 2006, ten studies compared the prediction accuracy between CC and WC effort estimation models [11]; however only seven of these presented independent results [11]. Of these seven, three found that CC models were not significantly different to WC models and four found that CC models were significantly worse than WC models [11]. The aforementioned studies and their comparisons have been detailed in a systematic literature review [11][12] that identified and analysed studies published between 1990 and November 2006. Seven years later a second systematic literature review was carried out covering the period of 2006 to 2013 [20]. Results showed a tie, where five studies indicated that CC estimation models were not significantly worse (four showing no significant difference and one showing a difference in favour of CC models) than WC models, and five indicating that they were significantly worse.

Since the second systematic literature review [20] was published, other four primary studies [7][15][28][33] have already been published, all employing some sort of strategy to make the CC data more similar to the WC data. Three of them have used Web project data from the Tukutuku database. Kocaguneli et al. [15] employed an analogy-based technique called TEAK, while Ferrucci et al. [7] and Turhan and Mendes [33] used Nearest Neighbour filtering (NN-filtering) with step-

wise regression. TEAK provided competing CC predictions for six, out of the eight different CC models built; the NN-filtering provided competing CC predictions for seven out of the eight CC models built. However, some NN-filtering CC predictions were worse than median-based predictions. Minku and Yao [28] recently proposed a framework for learning software effort estimation models for a single company based on mapping CC models to the single company's context. They evaluated an instantiation of such framework (called Dycom) by using five different datasets. Dycom achieved always similar or better performance than WC models on those datasets. However, all those datasets were composed solely of data on conventional software projects. In this paper we investigate the effectiveness of Minku and Yao's framework [28] for Web software projects.

The domain of Web development requires specific attention since there are many differences between Web and software development projects, and the results observed using datasets of non-Web projects are not readily applicable within the context of Web development. The differences between Web and conventional software development led to the creation of a new research field called Web engineering, back in 2001. A detailed discussion on the differences between Web and software development is provided by Mendes [17]. Web development is a relatively new and rapidly growing industry, with e-commerce alone weathering the recession and growing 11% in the United States in 2009, with similar growth in 2010. Our study is geared towards enabling Web development companies to make more efficient managerial decisions worthwhile. This paper answers three research questions, as follows:

RQ1. How successful is a CC dataset at estimating effort for Web projects from a single company?

RQ2. How successful is the use of a CC dataset compared to a WC dataset for Web effort estimation?

RQ3. How does Dycom perform with respect to other techniques previously used for CC Web effort estimation?

RQ1 investigates the feasibility of CC models being applied to a test set of WC projects. To this end we compared the performance of the CC models built with Dycom for each of the considered companies with respect to two baseline approaches, namely mean and median effort. RQ2 compares the accuracy between predictions obtained using CC models and WC models in estimating the development effort for WC projects. To answer it we compared the performance of the CC models built by Dycom with respect to WC models built by using a base learner that does not benefit from the mapping. RQ3 investigates whether Dycom is a competitive approach for CC Web effort estimation given the previous results from other studies obtained by filtering the CC datasets. To address RQ3, we compare the performance of Dycom with respect to a technique successfully exploited for Web effort estimation in previous work (NN-filtering) [7][33].

RQ1 and RQ2 have also been investigated in previous studies, but in the context of this paper they were examined through the use of a CC technique (Dycom) that achieved always similar or better accuracy than WC models on conventional projects [28]. Given the promising results achieved by Dycom on conventional projects, we investigate whether Dycom can increase the benefits of CC data also for Web effort estimation.

The remainder of this paper is organized as follows. Section II describes previous related work, followed by the description of Minku and Yao's approach Dycom in Section III. The research methodology is presented in Section IV. Section V presents the results obtained using Dycom. Finally, conclusions are given in Section VI. Please note that some descriptive parts of the paper (e.g. problem description, related work, description of the database and methods) partially re-use relevant text from authors' previous publications on the topic.

II. Related Work

Seven studies to date, detailed next, have used datasets of Web projects in order to investigate the abovementioned research questions [9][21][18][19][7][15][33].

S1: The first study (S1) was carried out in 2004 by Kitchenham and Mendes [9]. It investigated, using data on 53 Web projects from the Tukutuku database (40 CC and 13 from a single company), to what extent a CC effort model could be successfully employed to estimate development effort for WC Web projects. Their effort models were built using Forward Stepwise Regression (SWR) and they found that CC predictions were significantly worse than WC predictions.

S2: The second study (S2) extended S1, also in 2004, by Mendes and Kitchenham [21], who used SWR and Case-based reasoning (CBR), and also data on 67 Web projects from the Tukutuku database (53 CC and 14 from a single company). They built two CC and one WC model, and found that both SWR CC models provided predictions significantly worse than the WC predictions, whereas CBR CC data provided predictions significantly better than the WC predictions.

S3: By 2007, another 83 projects had been volunteered to the Tukutuku database (68 CC and 15 from a single company), and were used by Mendes et al. [18] to carry out a third study (S3) partially replicating S2 (only one CC model was built), and using SWR and CBR. They corroborated some of S2's findings (SWR CC model provided predictions significantly worse than WC predictions). However, S2 found CBR CC predictions to be superior to CBR WC predictions, which is the opposite of what was obtained in S3.

S4: Later, in 2008, Mendes et al. [19] conducted a fourth study (S4) that extended S3 to fully replicate S2. They used the same dataset used in S3, and their results corroborated most of those obtained in S2. The main difference between S2 and S4 was that one of S4's SWR CC models showed similar predictions to the WC model, which contradicts the findings from S2.

S5: After S4 was published, another 45 projects (i.e., 31 coming from a single company and 14 from different companies) were volunteered to the Tukutuku database. Therefore, a fifth study (S5) [7] extended S3 using the entire set of 195 projects from the Tukutuku database, and two WC datasets (31 and 18 projects respectively). In addition, they also investigated to what extent applying a filtering mechanism to CC datasets prior to building prediction models can affect the accuracy of the effort estimates they provide. Their results (without filtering) corroborated those from S3. However, the filtering mechanism significantly improved the prediction accuracy of CC models when estimating WC projects, making their prediction accuracy similar.

S6: This study used data on 125 projects from the Tukutuktu database (representing eight single companies), and a self-tuning analogy-based effort estimation method called TEAK [15] to compare CC versus WC predictions. They used seven single companies at a time as their CC dataset, and the eight as the testing set. This means that eight different models were built. Results showed that six CC models presented competitive accuracy to WC models, and two CC models presented worse accuracy than WC models.

S7: This study [33] followed the same approach used in S5, where only the WC datasets were used in the analysis. Data from eight companies with a total of 125 projects was employed, using SWR and relevancy filtering; results were reported per company as well as via a meta-analysis. The CC models in general provided poor predictions for the WC projects. However, when compared to the WC predictions, results were mixed: CC models built using regular (i.e., no filtering) regression models provided predictions significantly worse than the predictions obtained from WC models. Nevertheless, when built using the Nearest Neighbour (NN) filtering with stepwise regression, CC models presented competing accuracy to WC models in 7 out of 8 datasets.

Based on the above mentioned studies, we can see that analogy-based approaches (CBR and TEAK) sometimes provided worse, sometimes similar, and sometimes better results for CC models, when compared to WC models. The use of NN-filtering made the CC models accuracy similar to the one achieved using WC projects in most of the cases. These approaches can be seen as local approaches, as their predictions are based on the projects most similar to the specific project for which effort is being estimated. Approaches whose estimations are not based on any sort of locality (SWR) tended to perform similar or worse than WC models.

III. The Relationship Between CC and WC Contexts

Minku and Yao [28] observed that there is a relationship between the Software Effort Estimation (SEE) context of a certain company and other companies. They formalise the relationship between two companies C_A and C_B as follows:

$$f_A(\mathbf{x}) = g_{BA}(f_B(\mathbf{x})) \quad (1)$$

where C_A is the company in which we are interested; C_B is another company (or a section of this other company); f_A and f_B are the true functions providing the required effort to C_A and C_B, respectively; g_{BA} is a function that maps the effort from C_B's context to C_A's context; and $\mathbf{x} = [x_1, x_2, \cdots, x_n]$ are the input features of a software project. As an illustrative example, consider the software projects in Table I. In this case, the true effort of a project in C_A is 1.2 times the effort that would be required in C_B, i.e., $f_A(\mathbf{x}) = g_{BA}(f_B(\mathbf{x})) = 1.2 \cdot f_B(\mathbf{x})$. Even though the relationship between the effort in C_A and C_B is linear in this example, our CC learning scenario does not restrict g_{BA} to linear functions. Note also that f_A and f_B can be functions of any type. For instance, f_A (or f_B) could be composed of sub-functions representing different clusters of the company's data in order to represent the level of heterogeneity within a company itself [27], [24]. In practice, it is also likely that there will be some noise in the efforts.

TABLE I: Illustrative example of linear relationship between company C_A and company C_B. In this case, the true effort in person-hours for a given project in C_A is 1.2 times its true effort in person-hours in C_B.

ID	Functional Size	Development Type	Lang. Type	C_B's True Effort	C_A's True Effort
0	100	Enhancement	3GL	500	600
1	300	Re-development	4GL	1300	1560
2	400	New Development	4GL	2000	2400
3	500	New Development	3GL	3000	3600

Given equation 1, the learning task of creating an SEE model to a certain company C_A can involve the task of learning the relationship between C_A and other companies. Therefore, they proposed a framework for learning SEE models for a company C_A based on mapping CC models to C_A's context. The general idea of the framework is to use CC data to create one or more CC models, and to use a very limited number of WC training examples to learn a function that maps the estimations given by each CC model to estimations in the WC context. It is hoped that the task of learning the mapping functions is less difficult than the task of learning a whole WC model based solely on the WC training examples, as this would considerably reduce the amount of WC examples required for learning. This framework can be used both when WC and CC training examples arrive continuously (online) and when they comprise pre-existing sets of fixed size (offline). Please refer to Minku and Yao's work [28] for details on the framework.

A. Dycom

Minku and Yao [28] presented an online learning instantiation of the framework above, called Dynamic Cross-company Mapped Model Learning (Dycom). Dycom was validated using data on conventional software projects, and shown to achieve similar or better performance than a corresponding WC approach [28]. Dycom is described next:

CC training data: CC training examples are available beforehand. They are split into sections based on some clustering algorithm, or on their productivity, or on the size of the projects. Each section C_{Bi} is considered as a separate CC training set. For example, if there are N companies and the training examples from each company are split into S sections, then there will be $M = N \cdot S$ different CC training sets. If the company from which each CC training example comes from is not known, then the entire CC training set is split into different sections, as if $N = 1$. The reason for the splitting will be explained later in the paragraph on mapping functions. Note that we use the term CC loosely herein. For example, projects from different departments within the same company could be considered as CC projects if such departments employ largely different practices.

CC SEE models: Each of the M CC training sets is used to create a different CC model $\hat{f}_{Bi}$ ($1 \leq i \leq M$).

WC training data: Dycom considers that the WC data is entered according to the same sequence in which the projects are completed (online). In particular, it considers that one WC project arrives at each *time step*. WC projects that arrive at every p ($p > 1$) time steps contain both information on their input features and actual effort. All remaining WC

projects contain only the information on the input features, whereas their actual effort is missing. So, even though an effort estimation is required for all WC projects, only a few of them (those arriving at time stamps multiple of p) can be used as training examples. We will refer to the company for which we are interested in providing predictions as C_A.

Mapping functions: Whenever a new WC training example arrives, each model $\hat{f}_{Bi}$ is asked to perform an SEE. Each SEE is then used to create a mapping training example $(\hat{f}_{Bi}(\mathbf{x}), y)$. A mapping function $\hat{g}_{BiA}$ that receives estimations $\hat{f}_{Bi}(\mathbf{x})$ in the context of C_{Bi} as input and maps them to estimations in the context of C_A is trained with the mapping training example. Dycom considers that the relationship formalised in equation 1 can be modelled reasonably well by linear functions of the format $\hat{g}_{BiA}(\hat{f}_{Bi}(\mathbf{x})) = \hat{f}_{Bi}(\mathbf{x}) \cdot b_i$ when different sections containing relatively more similar CC training examples are considered separately. This is the reason to split CC training examples into different sections, as previously explained.

Learning a function of the format $\hat{g}_{BiA}(\hat{f}_{Bi}(\mathbf{x})) = \hat{f}_{Bi}(\mathbf{x}) \cdot b_i$ is equivalent to learning the factor b_i. This is done using equation 2:

$$b_i = \begin{cases} 1, & \text{if no mapping training example has been received yet;} \\ \dfrac{y}{\hat{f}_{Bi}(\mathbf{x})}, & \text{if } (\hat{f}_{Bi}(\mathbf{x}), y) \text{ is the first mapping training example;} \\ lr \cdot \dfrac{y}{\hat{f}_{Bi}(\mathbf{x})} + (1 - lr) \cdot b_i, & \text{otherwise.} \end{cases} \quad (2)$$

where $(\hat{f}_{Bi}(\mathbf{x}), y)$ is the mapping training example being learnt, lr $(0 < lr < 1)$ is a pre-defined smoothing factor and the factor b_i in the right side of the equation represents the latest value of b_i before receiving the current mapping training example.

While there is no mapping training example available, the mapping function performs a direct mapping $b_i = 1$. When the first mapping training example is received, b_i is set to the value $y/\hat{f}_{Bi}(\mathbf{x})$. This gives a perfect mapping for the example being learnt, as $\hat{f}_{Bi}(\mathbf{x}) \cdot b_i = \hat{f}_{Bi}(\mathbf{x}) \cdot y/\hat{f}_{Bi}(\mathbf{x}) = y$. For all other mapping training examples received, an exponential decay with smoothing factor lr is used to set b_i. This is the simple weighted average of the value that would provide a perfect mapping for the current mapping example and the previous value of b_i, which was calculated based on the previous mapping examples. A high smoothing factor lr will put more emphasis on the most recent mapping training examples and higher adaptability to changing environments, whereas a low lr will lead to a more stable mapping function. In summary, the smoothing function allows for mapping functions that provide good mappings based on previous mapping examples to be learnt, while allowing for adaptability to changes that may affect a company's required software effort.

WC SEE model: Whenever a new WC training example arrives, it is used to train a WC model $\hat{f}_{W_A}$. This model is not expected to perform very well, because it will be trained on a limited number of examples. However, its effort estimations may be helpful when used in an ensemble together with the mapped estimations, given that ensembles have been showing to improve SEE considerably [26][27].

Mapped SEE model: Both the mapped models $\hat{g}_{BiA}(\hat{f}_{Bi})$ and the WC model $\hat{f}_{W_A}$ can provide an SEE in the WC context when required. The SEE given by Dycom is the weighted average of these $M + 1$ estimations:

$$\hat{f}_A(\mathbf{x}) = \left[\sum_{i=1}^{M} w_{Bi} \cdot \hat{g}_{BiA}(\hat{f}_{Bi}(\mathbf{x})) \right] + w_{W_A} \hat{f}_{W_A}(\mathbf{x}), \quad (3)$$

where the weights w_{Bi} and w_{W_A} represent how much we trust each of the models, are positive and sum to one. So, Dycom uses an ensemble of mapped and WC SEE models.

Weights: The weights are initialised so that they have the same value for all models being used in the ensemble and are updated as as follows [26]: whenever a new WC training example is made available, the model which provided the lowest absolute error is considered to be the *winner* and the others are the *losers*. The losers have their weights multiplied by a pre-defined parameter β $(0 < \beta \leq 1)$, and then all weights are normalised in order to sum up to one.

Algorithm 1 presents Dycom's learning process. Dycom first learns the CC models (line 3). The weight associated to each CC model is initialised to $1/M$, so that each model has equal weight and all weights sum to one (line 4). The mapping functions are initialised to $b_i = 1$ (line 5). Before any WC training example is made available, the weight w_{W_A} corresponding to the WC model is initialised to zero (line 7), because this model has not received any training yet. In this way, CC models can be used to make predictions while there is no WC training example available. After that, for each new WC training example, the weights are updated (lines 10–19). If the WC training example is the first one, the weight of the WC model needs to be set (line 17). Then, the mapping training examples are created and used to update the corresponding mapping functions (lines 22 and 23). Finally, the WC model is updated with the WC training example (line 26).

IV. Methodology

A. Database

To examine the use of Dycom on Web project data, we used projects part of the Tukutuku database. This database has been created and maintained by Emilia Mendes [23] and includes information on projects developed by several companies located in different countries around the world [6]. It currently contains data on 195 projects developed by 51 companies. However, within the context of this work, we only used data from companies that provided at least 5 projects, in order to have enough WC projects for investigating the performance for each single company separately. This yielded 125 projects from 8 companies. Each company's project data was used as a WC dataset, thus leading to a total of 8 WC datasets as shown in Table II. For each WC, the remaining projects were considered as the CC data and were split into three CC sets for use with Dycom according to different ranges of productivity, as done in the original work [28]. The ranges used for the different CC sets were chosen to provide similar

Algorithm 1 Dycom

Parameters:
D_{Bi} $(1 \leq i \leq M)$: CC training sets.
β: factor for decreasing model weights

{Learn CC base models:}
for each CC training set D_{Bi} **do**
  Create CC model $\hat{b}_{Bi}$ using D_{Bi}
  $w_{Bi} = \frac{1}{M}$ {Initialise weight.}
  $b_i = 1$ {Initialise mapping function.}
end for
$w_{W_A} = 0$ {Initialise WC model weight.}
for each new WC training example $(\mathbf{x}, y)$ **do**
  {Update weights:}
  for each model $\hat{f}_{Bi}$ and $\hat{f}_{W_A}$ **do**
    Determine the model's estimation to $\mathbf{x}$.
    Calculate the absolute error AE_{Bi} (or AE_{W_A}).
  end for
  Determine loser models based on their AE.
  Multiply loser models' weights by β.
  if $(\mathbf{x}, y)$ is the first WC training example **then**
    $w_{W_A} = \frac{1}{M+1}$
  end if
  Divide each weight by the sum of all weights.
  {Update mapping functions:}
  for each model $\hat{f}_{Bi}$ **do**
    Create mapping example $(\hat{f}_{Bi}(\mathbf{x}), y)$.
    Use $(\hat{f}_{Bi}(\mathbf{x}), y)$ to update $\hat{g}_{BiA}$ based on Eq. 2.
  end for
  {Update WC model:}
  Update WC model $\hat{f}_{W_A}$ using $(\mathbf{x}, y)$.
end for

TABLE II: Average productivity and project number for each WC.

WC Data	Avg Productivity	# of Projects	% Projects
C1	2.03	14	11.2
C2	4.61	20	16
C3	0.87	15	12
C4	2.49	6	4.8
C5	1.42	13	10.4
C6	0.67	8	6.4
C7	0.90	31	24.8
C8	1.20	18	14.4
Total	-	125	100

size partitions and are shown in Table III. Herein, productivity was computed as the ratio between the AdjustedSize and Actual Effort, where the AdjustedSize measure includes only those size measures that together have a significant relationship with effort and has been obtained by using a size-based effort estimation model built with Manual StepWise Regression, as proposed by Kitchenham and Mendes [10].

Each project in the Tukutuku database is characterized by a total of 19 independent variables and a dependent variable (i.e., the total effort in person hours). Similar to other studies (e.g., [7][33]), we excluded categorical variables to increase the degrees of freedom for analysis. A short description of the independent variables used herein is given in Table IV, and summary statistics are reported in Table V. A more comprehensive description of the Tukutuku's features can be

TABLE III: Productivity ranges for CC data.

CC Data	Productivity Band	# Projects
CC-C1	Low: (0.13,0.65]	37
	Medium: (0.65,1.36]	37
	High: (1.36,21.60]	37
CC-C2	Low: (0.17,0.73]	35
	Medium: (0.73,1.34]	35
	High: (1.34,5.78]	35
CC-C3	Low: (0.15,0.69]	36
	Medium: (0.69,1.31]	36
	High: (1.31,16.78]	38
CC-C4	Low: (0.11,0.65]	39
	Medium: (0.65,1.27]	39
	High: (1.27,21.59]	41
CC-C5	Low: (0.16,0.71]	37
	Medium: (0.71,1.25]	37
	High: (1.25,10.69]	38
CC-C6	Low: (0.10,0.71]	39
	Medium: (0.74,1.33]	39
	High: (1.33,17.50]	39
CC-C7	Low: (0.15,0.66]	31
	Medium: (0.66,1.35]	31
	High: (1.36,8.15]	32
CC-C8	Low: (0.14,0.67]	35
	Medium: (0.67,1.31]	35
	High: (1.30,15.81]	37

TABLE IV: Variables used for the Tukutuku database.

Variable	Description
nlang	Number of different development languages used.
DevTeam	Size of a project's development team.
TeamExp	Avg team experience with the development language(s) used.
TotWP	Total number of Web pages (new and reused).
NewWP	Total number of new Web pages.
TotImg	Total number of images (new and reused).
NewImg	Total number of new images created.
Fots	Number of features reused without any adaptation.
HFotsA	Number of reused high-effort features/functions adapted.
Hnew	Number of new high-effort features/functions.
totHigh	Total number of high-effort features/functions
FotsA	Number of reused low-effort features adapted.
New	Number of new low-effort features/functions
totNHigh	Total number of low-effort features/functions
TotEff	Actual total effort used to develop the Web application.

found in [23]. The companies used in this study are small Web development companies whose projects were volunteered to the Tukutuku database within the period between 2003 and 2006. The descriptive statistics suggest that datasets are fairly homogeneous. Statistics for each company can be found at http://cc.oulu.fi/~bturhan/ccsc/ and were omitted due to space constraints. Box plots of effort, for each of the companies, are shown in figure 1; they suggest that, apart from company C3, effort variance for each single company is not very large.

Minku and Yao [28] investigated Dycom in an online scenario based on the chronological order of the WC projects. However, since the Tukutuku database does not include chronological information, Dycom has been used herein in an offline scenario, as explained in Section IV-B.

B. Evaluation Procedure

RQ1 investigates the feasibility of CC models being applied to a test set of WC projects. To this end we compared the performance of the CC models built with Dycom for each

TABLE V: Descriptive statistics for the variables used in our study.

Variable	Mean	Median	Std.Dev.	Min	Max
nLang	3.89	4	1.45	1	8
DevTeam	2.58	2	2.38	1	23
TeamExp	3.83	4	2.03	1	10
TotWP	69.48	26	185.69	1	2000
NewWP	49.55	10	179.14	0	1980
TotImg	98.58	40	218.37	0	1820
NewImg	38.27	1	125.47	0	1000
Fots	3.19	1	6.24	0	63
HFotsA	11.96	0	59.85	0	611
Hnew	2.08	0	4.7	0	27
TotHigh	14.04	1	59.63	0	611
FotsA	2.24	0	4.53	0	38
New	4.24	1	9.65	0	99
TotNHigh	6.48	4	13.22	0	137
TotEff (dependent var.)	468.11	88	938.51	1.1	5000

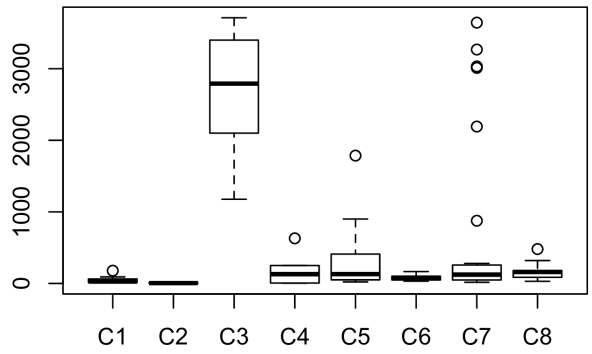

Fig. 1: Effort required to develop software projects in each company.

of the considered companies with respect to CC baseline approaches, namely the mean effort and median effort. These baselines always make predictions equal to the mean and median of a given training set, respectively. In order to answer RQ1, the mean and median models used the same training set as Dycom.

In order to evaluate Dycom, the CC baseline approaches, and CC random guess in offline mode, we repeated the following steps 30 times (1000 times for random guess):

1) Shuffle the WC data.
2) Select every p WC example as a training example, as well as all CC examples, and use them to train the approach.
3) Use all other WC examples to test the approach. This will lead to $|WCdata| - |WCtrainingexamples|$ effort estimations made by the approach.
4) Calculate the approach's performance based on these effort estimations using the chosen performance measure (e.g., Mean Absolute Error).

The performance measures used in the comparisons were Mean Absolute Error (MAE), Standardized Accuracy (SA) and Mean Absolute Error in the Logarithmic Scale (MAEL). MAE has been recommended for the evaluation of software effort estimators because it is unbiased towards over or underestimations. SA is based on MAE and represents the ratio of how much better an approach is than random guess [31]. Random guess is defined as uniformly randomly sampling the true effort over all the projects of a given training set. Different from MAE, MAEL is a measure independent of project size. These performance measures are defined as follows, where T is the number of examples used for evaluating the performance, y_i is the actual effort for the example i, and $\hat{y}_i$ is the estimated effort for example i:

- $MAE = \frac{1}{T}\sum_{i=1}^{T}|\hat{y}_i - y_i|$;
- $MAEL = \frac{1}{T}\sum_{i=1}^{T}|e_i|$, where $e_i = \ln y_i - \ln \hat{y}_i$;
- $SA = \left(1 - \frac{MAE_{P_i}}{MAE_{rguess}}\right) \cdot 100$, where MAE_{P_i} is the MAE of the approach P_i being evaluated and MAE_{rguess} is the MAE of a large number (e.g., 1000 runs) of random guess.

MAE and MAEL are measures to be minimised, i.e., the lower their values, the better the estimation method. SA is a measure to be maximised.

RQ2 compares the accuracy between predictions obtained using Dycom and WC models in estimating the effort for WC projects, i.e., models trained solely on WC data.

In order to evaluate the WC models, we repeated the following steps 30 times:

1) Shuffle the WC data using the same order as for Dycom.
2) Forbid every p example to ever be used as a test example.
3) For each WC example that can be used for testing, select it as a test example, use all remaining WC examples for training, and get the WC model's estimation to the test example.
4) There will now be $|WCdata| - |WCtrainingexamples|$ effort estimations.
5) Calculate the performance based on these effort estimations using the chosen performance measures.

The above RQs questions have also been the ones investigated in previous studies, but within the context of this paper they were examined using Dycom and in an iterative fashion: first, we replicated the use of Dycom in exactly the same way proposed by Minku and Yao [28], without any dataset-specific fine tuning of parameters; second, we re-applied Dycom after fine tuning the parameter p to the Tukutuku database's context, as detailed in section IV-C.

RQ3 compares the accuracy between predictions obtained using Dycom and NN-filtering [33] in estimating the effort for WC projects. We chose NN-filtering to answer R3 because it is the only CC approach that managed to achieve similar performance to WC Web effort estimation models for seven out of eight Tukutuku single companies. Therefore, it can be considered as the most competitive CC Web effort estimation approach in the literature. The training and testing datasets used with NN-filtering were exactly the same as the ones used by Dycom, i.e., the results were evaluated by using the same procedure employed to answer RQ1.

The performance measures were compared based on Wilcoxon Sign Rank tests. The comparisons were based on Holm-Bonferroni corrections considering eight comparisons (corresponding to the eight Tukutuku single companies) at the overall level of significance of 0.05.

C. Experimental Setup

Dycom can be used with any base learner. We employed Regression Trees (RTs) as they were the base learners used in the original study [28]. Minku and Yao [28] chose to use RTs since they are local approaches in which estimations are based on the projects that are most similar to the project being predicted. This can help dealing with the heterogeneity within each dataset [27]. In order to provide a fair comparison, the models used for WC learning and NN-filtering were also RTs. Given the base learner used in the experiments, Dycom will be referred to as Dycom-RT, the WC model will be referred to as WC-RT, and NN-filtering as NN-filtering-RT in our analysis. As Minku and Yao [28], we used the RT implementation *REPTree* provided by WEKA [8], where splits are created so as to minimise the variance of the targets of the training examples in the nodes. The study of Dycom with other base learners is left as future work.

Dycom's parameters were set to the values used in the original work [28] as follows: Dycom's parameter β was set to the default value of 0.5, which has been previously used in other studies for similar weight update mechanisms [26]. Dycom's parameter lr was set to 0.1, a value chosen after some initial experimentation with 0.05 and 0.1 in [28]. The parameters used with each RT were the ones more likely to obtain good results in previous work [27]: minimum total weight of 1 for the instances in a leaf, and minimum proportion of the variance on all the data that need to be present at a node in order for splitting to be performed 0.0001. The parameter p, which controls the amount of WC training examples that can be used by Dycom, was first set to $p = 10$, meaning that Dycom uses only 10% of the WC training examples used by the WC model. This is the same as the value used in Minku and Yao's work [28]. In addition, we also investigated the use of Dycom-RT with $p = 2$. In this case Dycom uses half of the WC training examples used by the WC-RT. The reason for investigating Dycom with $p = 2$ is that the Tukutuku WC datasets are very small in comparison to the WC datasets used in the original Dycom study. Using $p = 10$ resulted in Dycom employing too few WC training projects to train the mapping function (in some cases even only 1 WC training project), which led to very poor results. Due to space restrictions, only the results obtained with $p = 2$ will be reported herein.

As for the NN-filtering setup, similarly to previous work where NN-filtering was applied to Web effort estimation using Tukutuku data [7][33], we used 10 nearest neighbours and the Euclidean distance as similarity measures.

D. Threats to Validity

Internal validity regards to establishing that a certain observable event was responsible for a change in behaviour. It is related to the question "Is there something other than the treatment that could cause the difference in behaviour?" [29]. When using machine learning approaches, it is important that the approaches being compared use fair parameter choices in comparison to each other [3][4][25][32]. In this paper, both the RTs used as WC learners and within Dycom and NN-filtering employed the same parameters, which were the ones more likely to obtain good results in the literature [27]. Dycom contains two extra parameters which were set to the same value for all companies used in our analysis, i.e., they were not fine tuned for each company separately. Therefore, the results presented herein do not depend on the user fine tuning Dycom for each individual company. Future work will investigate whether Dycom's results could be improved further by fine tuning parameters for each company.

Construct validity regards to accurately naming our measures and manipulations [29]. The size measures and effort drivers used in the Tukutuku database, and therefore in our study, have been obtained from the results of a survey investigation and have also been confirmed by an established Web company and a second survey [22]. Consequently, it is our belief that the variables identified are measures that are meaningful to Web companies and are constructed from information their customers can provide at a very early stage in the project development. As for data quality, it was found that at least for 93.8% of Web projects in the Tukutuku database effort values were based on recorded data [22]. With respect to performance measures, we used MAE, SA, and MAEL in order to evaluate Dycom. These measures are unbiased towards under or over-estimations. We believe these form a good set of measures for evaluating performance in SEE, given that MAE considers project size, MAEL is independent of project size and SA allows for easier interpretability. Should the reader be interested in other performance measures, we provide results for Dycom-RT, WC-RT and NN-filtering-RT using several other performance measures in a separate report available at http://www0.cs.ucl.ac.uk/staff/F.Sarro/esem15measures.pdf. These are logarithmic standard deviation, correlation, root mean squared error, mean magnitude of the relative error and percentage of predictions within 25% of the actual values. Wilcoxon statistical tests with Holm-Bonferroni corrections were used to check the statistical significance of the differences in overall performance. In this paper, we report the median performances over all 30 runs. The average performances were also computed as in Minku and Yao's work [28] and are available in the report abovementioned. The use of averages and medians to compare the approaches led to the same conclusions in terms of which approach is better.

External validity regards to generalizing the results to a wider context [29]. The Tukutuku database comprises data on projects volunteered by individual companies, and therefore it does not represent a random sample of projects from a defined population. This means that we cannot conclude that the results of this study apply to other companies different from the ones that volunteered the data used herein. However, we believe that Web companies that develop projects with similar characteristics to those used in this paper may be able to apply our results to their Web projects. We have also used RTs as the base learners in this study, as these were the base learners used in the original work [28]. This choice has been made because it guarantees that any different results obtained in this paper in comparison to the original work are not due to the use of a different base learner from the original work. Even though RTs have been shown to perform well on conventional

software data as well as Web data in the past, our future work will further investigate if the results obtained herein generalise to other base learners that have also been successfully used with Web data (e.g., stepwise regression).

V. Results

A. RQ1

Table VI shows the comparisons between Dycom-RT and the mean and median effort baseline approaches when using $p = 2$. Under this setting, except for C8 in terms of MAE against the median, Dycom-RT achieved similar or better MAE and MAEL than both mean and median in all cases. These results are confirmed by Wilcoxon sign-rank tests (two-sided) with Holm-Bonferroni corrections at the overall level of significance of 0.05. In particular, Dycom-RT performed significantly better than the median effort in four cases in terms of MAE (C2–4, C7), significantly better in seven cases in terms of MAEL (C1–5,C7,C8) and significantly worse in only one case in terms of MAE (C8). These results show some improvement over previous research where, for the same set of 125 Web projects, NN-filtering CC stepwise regression models presented significantly worse accuracy than median-based predictions in five cases (C2, C4, C6–8) [33].

It is interesting to note that the characteristics of the data sets can influence the performance of the approaches, and that different performance measures capture such effects differently. For instance, C3 is the single company with the highest median effort (see Figure 1). This was associated to higher MAE for all approaches shown in Table VI. However, the higher efforts were not associated to higher MAEL because this measure is independent of project size, which is typically positively correlated with effort.

RQ1 asked how successful a CC dataset is at estimating effort for projects from a single company. The above results show that, when using Dycom-RT to convert CC data to the WC context, CC datasets can be successful in estimating effort for projects from a single company. The measure SA suggests that Dycom-RT performed from around 10 to 97 percent better than a CC random guess approach when using $p = 2$.

B. RQ2

Table VII shows Dycom-RT's results in comparison to WC-RT. According to Wilcoxon sign-rank (two-sided) statistical tests with Holm-Bonferroni corrections at the overall level of significance of 0.05, Dycom-RT obtained similar or better performance than WC-RT in most cases. Note that, in the case of Dycom, similar performance already represents a good advantage over WC models, because Dycom requires a lower number of WC training examples, saving the cost of collecting the required effort for WC projects. Obtaining similar or better performance thus indicates a strong advantage of Dycom-RT over WC-RT.

The datasets for which Dycom-RT obtained the best behaviour were C2, C6 and C7. The performance measure SA suggests that Dycom-RT performed up to around 97 percent better than a WC-based random guess approach, even though for C8 it performed worse. These results present some improvement in terms of CC models' accuracy, compared to WC models, with respect to previous studies that used different strategies to adapt the CC models to the WC context. Kocaguneli et al.'s results [15] were not based on statistical significance; however, their win-tie-loss results showed CC models built using TEAK to provide competing accuracy (not superior) for 6 out of 8 models. Turhan and Mendes' results [33] were more competitive, showing that CC models built using NN-filtering with stepwise regression presented competing accuracy to WC models for 7 out of 8 models. Still, NN-filtering never achieved better performance than WC models on these datasets.

RQ2 asked how successful the use of a CC dataset is compared to a WC dataset for Web effort estimation. The analysis presented above shows that, when using Dycom-RT, CC data can help to improve performance for Web effort estimation. Overall, our experiments show that Dycom-RT not only managed to use only half of the WC training examples, but also was usually able to provide similar or better average performance than WC-RT.

We believe that the implications that these results have for research and practice are that CC datasets may be a competitive choice for Web companies that have just a few available WC projects, when CC models are built using a strategy where they are explicitly mapped to the WC context. Such findings hold for companies that develop applications similar to those described herein, and for the eight companies who have volunteered data for the Tukutuku database.

C. RQ3

Table VIII shows the results of the comparison between Dycom-RT and NN-Filtering-RT. We can observe that Dycom-RT obtained significantly better MAE than NN-Filtering-RT in 3 cases (i.e., C2, C4, C7) and worse in only one case (C8); in the remaining cases (C1, C3, C5, C6) no significant statistical differences were found. MAE allows larger projects to have larger influence on the performance than MAEL. In terms of MAEL, Dycom-RT obtained significantly better performance in 2 cases (C2 and C7) and worse in one case (C8).

RQ3 focused on the performance of Dycom with respect to other techniques previously used for CC Web effort estimation where some sort of 'filtering' was applied to the CC data during model building. Our results show that Dycom-RT tends to obtain similar or better performance than a competitive approach from the literature (NN-Filtering-RT) specially when considering larger projects, given the difference in MAE and MAEL results. Therefore, it can be worth to separate and map CC projects into the WC context for achieving good performance in Web effort estimation.

VI. Conclusions

In this study, we have used data from eight different single company datasets in the Tukutuku database to empirically compare the accuracy between estimates obtained using cross-company (CC) and within-company (WC) models. Dycom CC models using Regression Trees (RTs) as the base learners were evaluated and benchmarked against baseline models (mean and median effort), a WC base learner (WC-RT) and a technique previously used for CC Web effort estimation (NN-filtering).

TABLE VI: RQ1. Comparison between Dycom-RT vs. CC baseline approaches (mean and median effort).

Test Set	Mean vs. Dycom	MAE	SA	MAEL	Median vs. Dycom	MAE	SA	MAEL
C1	Mean-P2	542.9989	-747.3199	3.0267	Median-P2	57.9686	9.5432	1.2305
	Dycom-RT-P2	36.6201	42.8563	0.65105	Dycom-RT-P2	36.6201	42.8563	0.65105
	P-value	2.00E-06		2.00E-06	P-value	5.72E-01		1.00E-05
C2	Mean-P2	590.8896	-588.0089	4.6718	Median-P2	89.5760	-4.2988	2.8610
	Dycom-RT-P2	4.2004	95.1092	0.553	Dycom-RT-P2	4.2004	95.1092	0.553
	P-value	2.00E-06		2.00E-06	P-value	2.00E-06		2.00E-06
C3	Mean-P2	2170.0519	13.5880	1.7619	Median-P2	2523.5714	-0.4892	3.4836
	Dycom-RT-P2	734.1386	70.7664	0.23605	Dycom-RT-P2	734.1386	70.7664	0.23605
	P-value	4.10E-05		2.00E-06	P-value	2.80E-05		2.00E-06
C4	Mean-P2	392.4489	-71.6249	2.1131	Median-P2	185.2500	18.9869	2.0480
	Dycom-RT-P2	110.96	51.4752	0.7837	Dycom-RT-P2	110.96	51.4752	0.7837
	P-value	2.00E-06		2.00E-06	P-value	4.00E-06		2.00E-06
C5	Mean-P2	465.0405	-19.2921	1.6425	Median-P2	325.4167	16.5242	1.2256
	Dycom-RT-P2	321.19815	17.6063	0.9441	Dycom-RT-P2	321.19815	17.6063	0.9441
	P-value	1.25E-01		4.00E-06	P-value	5.30E-01		4.90E-04
C6	Mean-P2	490.7463	-545.7188	2.0440	Median-P2	29.6250	61.0197	0.4109
	Dycom-RT-P2	36.052	52.5632	0.41305	Dycom-RT-P2	36.052	52.5632	0.41305
	P-value	2.00E-06		2.00E-06	P-value	3.82E-01		7.97E-01
C7	Mean-P2	802.1830	-13.2760	1.8029	Median-P2	605.6667	14.4740	1.2335
	Dycom-RT-P2	23.234	96.7191	0.1244	Dycom-RT-P2	23.234	96.7191	0.1244
	P-value	2.00E-06		2.00E-06	P-value	2.00E-06		2.00E-06
C8	Mean-P2	421.2622	-194.8180	1.5218	Median-P2	94.6667	33.7481	0.7119
	Dycom-RT-P2	128.99745	9.7218	0.5614	Dycom-RT-P2	128.99745	9.7218	0.5614
	P-value	4.00E-06		2.00E-06	P-value	2.11E-03		3.85E-03

MAE, SA, and MAEL are the median performances over 30 runs. Cells in lime (light grey) represent better values for Dycom-RT, whereas cells in orange (dark grey) represent better values for the baseline approach. P-values of Wilcoxon sign rank tests to compare Dycom-RT against the baseline approaches for each company are also shown. When in cells highlighted in lime or orange, these p-values indicate statistically significant difference using Holm-Bonferroni corrections at the overall level of significance of 0.05 considering the eight companies. Statistical tests were not performed for SA because this measure is an interpretable equivalent to MAE. Therefore, none of the SA cells have been coloured.

TABLE VII: RQ2: Comparison between Dycom-RT vs. WC-RT.

	Approach	MAE	SA	MAEL
C1	WC-RT	22.8779	43.8107	0.7362
	Dycom-RT	36.6201	10.0591	0.6511
	P-value	4.99E-03		4.41E-01
C2	WC-RT	5.2373	26.1423	0.6399
	Dycom-RT	4.2004	40.7643	0.5530
	P-value	1.71E-03		2.77E-03
C3	WC-RT	627.7143	28.5761	0.2615
	Dycom-RT	734.1386	16.4667	0.2361
	P-value	8.59E-02		7.04E-01
C4	WC-RT	64.7500	73.4631	0.4000
	Dycom-RT	110.9600	54.5246	0.7837
	P-value	3.32E-04		1.70E-06
C5	WC-RT	374.0833	6.8672	0.9879
	Dycom-RT	321.1982	20.0337	0.9441
	P-value	7.34E-01		6.00E-01
C6	WC-RT	44.7500	-7.1856	0.5736
	Dycom-RT	36.0520	13.6479	0.4131
	P-value	5.71E-02		7.73E-03
C7	WC-RT	223.7953	68.5268	0.3517
	Dycom-RT	23.2340	96.7325	0.1244
	P-value	2.40E-06		4.20E-04
C8	WC-RT	76.6667	27.7487	0.4242
	Dycom-RT	128.9975	-21.5683	0.5614
	P-value	2.16E-05		4.53E-04

TABLE VIII: R3: Comparison between Dycom-RT vs. NN-filtering.

	Approach	MAE	SA	MAEL
C1	NN-Filtering-RT	22.0922	45.7405	0.9009
	Dycom-RT	36.6201	10.0591	0.6511
	P-value	4.99E-03		4.41E-01
C2	NN-Filtering-RT	15.8203	-123.1032	1.0056
	Dycom-RT	4.2004	40.7643	0.5530
	P-value	1.71E-03		2.77E-03
C3	NN-Filtering-RT	670.8572	23.6671	0.2864
	Dycom-RT	734.1386	16.4667	0.2361
	P-value	8.59E-02		7.04E-01
C4	NN-Filtering-RT	125.8413	48.4257	0.7564
	Dycom-RT	110.9600	54.5246	0.7837
	P-value	3.32E-04		1.70E-06
C5	NN-Filtering-RT	400.0417	0.4046	1.1105
	Dycom-RT	321.1982	20.0337	0.9441
	P-value	7.34E-01		6.00E-01
C6	NN-Filtering-RT	35.8375	14.1617	0.5393
	Dycom-RT	36.0520	13.6479	0.4131
	P-value	5.71E-02		7.73E-03
C7	NN-Filtering-RT	226.3800	68.1633	0.4112
	Dycom-RT	23.2340	96.7325	0.1244
	P-value	2.40E-06		4.20E-04
C8	NN-Filtering-RT	73.0556	31.1518	0.4309
	Dycom-RT	128.9975	-21.5683	0.5614
	P-value	2.16E-05		4.53E-04

MAE, SA, and MAEL are the median performances over 30 runs. Cells in lime (light grey) represent better values for Dycom-RT, whereas cells in orange (dark grey) represent better values for WC-RT or NN-Filtering-RT. P-values based on the Wilcoxon sign rank tests (used to compare Dycom-RT against WC-RT and NN-Filtering-RT for each company) are also shown; when in cells highlighted in lime or orange, these p-values indicate a statistically significant difference using Holm-Bonferroni corrections at the overall level of significance of 0.05 considering the eight companies. Statistical tests were not performed for SA because this measure is an interpretable equivalent to MAE. Therefore, none of the SA cells have been coloured.

With regard to RQ1, our results show that, when using Dycom-RT to map CC data to the WC context, CC datasets can be successful in estimating effort for projects from a single company. Dycom-RT always performed significantly better than the mean effort in terms of MAE and MAEL except for one company in terms of MAE, where it performed similarly. Dycom-RT always performed similarly or significantly better than the median effort in terms of MAE and MAEL, except for one company in terms of MAE, where it performed worse These results represent an improvement over previous studies employing other strategies to adapt CC models to the WC data [15][33] for Web effort estimation. With regard to RQ2, Dycom-RT provided similar or significantly superior performance to WC-RT in most cases. These results are also an improvement over the strategies proposed in previous studies [15][33]. As for RQ3, we found that Dycom-RT performs similarly or significantly better than NN-filtering-RT for all companies except one. Overall the results presented herein are quite promising, and in general support previous results when applying Dycom-RT to conventional software datasets.

Our work has implications to both practice and future research. Regarding implications to practice, Dycom presents, in our view, the following benefits:

1) Dycom in combination with CC datasets has the

potential to provide competitive accuracy for Web effort estimation in industry by mapping estimations from the CC to the WC context. This would require Dycom to be embedded into a simple interface for use by companies, which is part of our future work.

2) Dycom reduces the amount of WC data that needs to be collected by a single company in order to obtain Web effort estimates.
3) Dycom can also be used to provide a better understanding of the relationship between the effort required to develop projects within a company and the effort required by other companies. This can be used by software engineers when they are planning strategies to improve their company's productivity. An example of that has been given by Minku and Yao [28]. As future work, this can also be investigated for Web projects.
4) By providing a better understanding of the relationship between the effort of different companies, Dycom can also be used by a single company to check whether improvements in productivity are being obtained in comparison to other companies.

Some of the implications of this work for future research are the following: Regression Trees (RTs) were the sole base learners employed in our study. However, other base learners should also be investigated and compared against RTs' results, which is part of our future work. Dycom has now been assessed on both conventional as well as Web-based project data, all with promising results. However, there are also more recent releases of the ISBSG database (R13), and other large datasets (e.g., Finnish dataset) that can be used to assess Dycom further, and which are also part of our future work. In addition, we have separated the CC data into different subsets according to productivity in order to generate the CC models. A comparison of this strategy against the use of clustering algorithms to separate the CC data would be a valuable contribution.

REFERENCES

[1] B. Boehm. *Software Engineering Economics*. Prentice-Hall, Englewood Cliffs, NJ, 1981.

[2] L. Briand, T. Langley, and I. Wieczorek. A replicated assessment of common software cost estimation techniques. In *ICSE*, pages 377–386, 2000.

[3] A. Corazza, S. Di Martino, F. Ferrucci, C. Gravino, F. Sarro, and E. Mendes. How effective is tabu search to configure support vector regression for effort estimation? In *PROMISE*, pages 4:1–4:10, 2010.

[4] A. Corazza, S. D. Martino, F. Ferrucci, C. Gravino, F. Sarro, and E. Mendes. Using tabu search to configure support vector regression for effort estimation. *Empirical Software Engineering*, 18(3):506–546, 2013.

[5] T. DeMarco. *Controlling Software Projects: Management, Measurement, and Estimates*. Prentice Hall PTR, 1986.

[6] F. Ferrucci, C. Gravino, R. Oliveto, F. Sarro, and E. Mendes. Investigating tabu search for web effort estimation. In *Euromicro Conference on Software Engineering and Advanced Applications*, pages 350–357, 2010.

[7] F. Ferrucci, E. Mendes, and F. Sarro. Web effort estimation: The value of cross-company data set compared to single-company data set. In *PROMISE*, pages 29–38, 2012.

[8] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. *SIGKDD Explorations*, 11(1):10–18, 2009.

[9] B. Kitchenham and E. Mendes. A comparison of cross-company and single-company effort estimation models for web applications. In *Empirical Assessment in Software Engineering*, pages 47–55, 2004.

[10] B. Kitchenham and E. Mendes. Software productivity measurement using multiple size measures. *IEEE TSE*, 30(12):1023–1035, 2004.

[11] B. Kitchenham, E. Mendes, and G. H. Travassos. A systematic review of cross- vs. within-company cost estimation studies. In *EASE*, pages 10–12, 2006.

[12] B. Kitchenham, E. Mendes, and G. H. Travassos. Cross versus within-company cost estimation studies: A systematic review. *IEEE TSE*, 33(5):316–329, May 2007.

[13] B. Kitchenham and N. Taylor. Software cost models. *ICL Technical Journal*, pages 73–102, 1984.

[14] B. A. Kitchenham and E. Mendes. A comparison of cross-company and within-company effort estimation models for web applications. In *International Symposium on Software Metrics*, pages 348–357, 2004.

[15] E. Kocaguneli, T. Menzies, and E. Mendes. Transfer learning in effort estimation. *Empirical Software Engineering*, pages 1–31, 2014.

[16] P. Kok, B. Kitchenham, and J. Kirawkowski. The mermaid approach to software cost estimation. In *ESPRIT*, pages 296–314. Springer Netherlands, 1990.

[17] E. Mendes. *Practitioner's Knowledge Representation*. Springer-Verlag, 2014, DOI: 10.1007/978-3-642-54157-5_2.

[18] E. Mendes, S. Di Martino, F. Ferrucci, and C. Gravino. Effort estimation: How valuable is it for a web company to use a cross-company data set, compared to using its own single-company data set? In *International Conference on World Wide Web*, pages 963–972, 2007.

[19] E. Mendes, S. Di Martino, F. Ferrucci, and C. Gravino. Cross-company vs. single-company web effort models using the tukutuku database: An extended study. *JSS*, 81(5):673–690, May 2008.

[20] E. Mendes, M. Kalinowski, D. Martins, F. Ferrucci, and F. Sarro. Cross- vs. within-company cost estimation studies revisited: an extended systematic review. In *EASE*, pages 12:1–12:10, 2014.

[21] E. Mendes and B. Kitchenham. Further comparison of cross-company and within-company effort estimation models for web applications. In *International Symposium on Software Metrics*, pages 348–357, 2004.

[22] E. Mendes, N. Mosley, and S. Counsell. Comparison of cross-company and single-company effort estimation models for web applications. In *International Conference on Empirical Software Engineering*, pages 1–22, 2003.

[23] E. Mendes, N. Mosley, and S. Counsell. Investigating web size metrics for early web cost estimation. *JSS*, 77(2):157–172, 2005.

[24] T. Menzies, A. Butcher, D. Cok, A. Marcus, L. Layman, F. Shull, B. Turhan, and T. Zimmerman. Local vs. global lessons for defect prediction and effort estimation. *IEEE TSE*, 39(6):822–834, 2013.

[25] T. Menzies and M. Shepperd. Special issue on repeatable results in software engineering prediction. *Empirical Software Engineering*, 17:1–17, 2012.

[26] L. Minku and X. Yao. Can cross-company data improve performance in software effort estimation? In *PROMISE*, pages 69–78, 2012.

[27] L. Minku and X. Yao. Ensembles and locality: Insight on improving software effort estimation. *IST*, 55(8):1512–1528, 2013.

[28] L. L. Minku and X. Yao. How to make best use of cross-company data in software effort estimation? In *ICSE*, pages 446–456, 2014.

[29] M. L. Mitchell and J. M. Jolley. *Research Design Explained*. Cengage Learning, USA, 7th edition, 2010.

[30] L. H. Putnam. A general empirical solution to the macro software sizing and estimating problem. *IEEE TSE*, 4(4):345–361, 1978.

[31] M. Shepperd and S. McDonell. Evaluating prediction systems in software project estimation. *IST*, 54(8):820–827, 2012.

[32] L. Song, L. Minku, and X. Yao. The impact of parameter tuning on software effort estimation using learning machines. In *PROMISE*, pages 9:1–9:10, 2013.

[33] B. Turhan and E. Mendes. A comparison of cross- versus single-company effort prediction models for web projects. In *Euromicro Conference on Software Engineering and Advanced Applications*, pages 285–292, 2014.