Revisiting the merits of cross company data

Ekrem Kocaguneli
Lane Department of Computer Science and Electrical Engineering
West Virginia University
Morgantown, USA
ekocagun@mix.wvu.edu

Tim Menzies
Lane Department of Computer Science and Electrical Engineering
West Virginia University
Morgantown, USA
tim@menzies.us

ABSTRACT

Background: Generating a database of past projects *within* an organization requires considerable investment of 1) time, 2) money and 3) educated personnel. Hence, it is tempting to *cross* the boundaries of development type, location, language, application and hardware to use existing datasets of other organizations.
Aim: The literature is skeptical on the merits of cross-company data. Our hypothesis is that systematical investigation of project properties that are likely to define borders of *crossness*, could reveal 1) how effective they can define these borders and 2) how much *within* and *cross* data would be favored by test instances.
Method: We filtered out 8 *cross-within divisions* (21 pairs of *within-cross* subsets) out of 19 datasets and evaluated these divisions under different analogy-based estimation (ABE) methods.
Results: We have seen that *cross* and *within* data is comparable in terms of: 1) performance subject to 4 evaluation criteria and 2) percentage instances selected by ABE methods in final estimation.

Categories and Subject Descriptors

H.4 [**Software Cost Estimation**]: *k*-NN; D.2.8 [**Software Engineering**]: Cost—*within company, cross company*

1. INTRODUCTION

Accurate effort estimates of future projects are important for software organizations. With precise estimates, the organizations can better allocate their resources; thereby, increasing their competitiveness on the market. On the other hand, it is not easy to attain estimates with high accuracy values. Even in corporations that have well established measurement and estimation practices, the matter of accuracy is questionable. A well known example to that case is NASA's Check-out Launch Control System, that resulted in cancellation after doubling its initial estimate of $200M [29]. The examples in less experienced corporations are even worse [1].

The accuracy of estimates depend on two fundamental factors: 1) the model used for estimations and 2) the historical database on which the model is built. The second factor is somewhat a precondition of the first one, i.e. for a successful model, a well-maintained and precise dataset is a must. Although both factors are equally important, here our focus will be more on the dataset.

A company willing to undergo a project of a historical effort data collection *within* the organization should be able to dedicate a considerable amount of time, money and personnel to this project. Even if such an investment is ventured, the initial results may have to wait for a long time. In the case of NASA-wide software cost metrics repository, only 7 projects could have been added in a timeframe of two years. It is no surprise that a number of organizations cannot face the challenges associated with formation and maintenance of a *within-company* (from now on WC) dataset.

An alternative approach to WC dataset formation is to adapt projects from *cross* organizations. With the help of publicly available data repositories, such cross-company (from now on CC) datasets are quite easy to find. For example practitioners can find dozens of these datasets in PROMISE data repository [3].

The merits of using CC data is contradictory. A recent survey by Kitchenham et al. on the value of using CC data, shows that we are unable to make a conclusion [15]. Another study by Zimmermann et al. ends up with a similar result [33]. Turhan et al. [32] and Kocaguneli et al. [17] show that through relevancy filtering CC data can perform as well as WC data. Our work revisits the merits of *cross* data by experimenting with various ABE methods (with and without relevancy filtering) on a larger scale. We identified 21 WC-CC dataset pairs, which is orders of magnitude bigger than the amount of pairs used in previous studies [15]. Our conclusions from these experiments is that except for a very small minority of cases, CC data performance is comparable to WC.

Prior work has not deeply investigated the selection tendencies of *within* company test instances. We question how much data a test instance would select from *within* and *cross* data, when given access to both of them. The results of this investigation are: 1) test instances tend to select analogies from both *within* as well as *cross* data and 2) percentage selections from WC and CC subsets tend to be very close to one another.

1.1 Research Questions

So as to guide this research the following research questions are defined:

RQ1: What can be said about *within* and *cross* data performances?
RQ2: What is the selection tendency of *within* instances?
RQ3: What is the reason for a particular tendency of *within* instances?

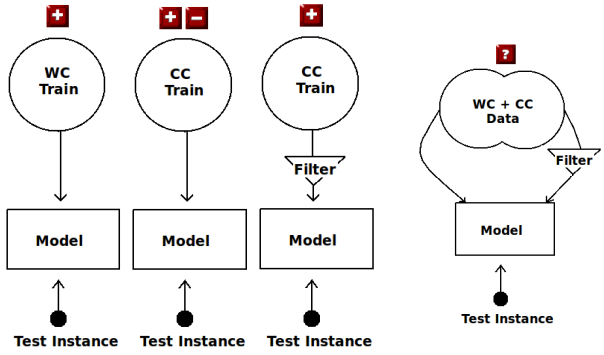
RQ4: What would be a good mixture of *wihin* and *cross* data?
RQ5: Under which conditions would *cross* data be favorable?
RQ6: Which features are likely to define *cross* and *within* boundaries?

2. MOTIVATION

There is enough empirical evidence that cross-company estimation is a serious problem to tackle [15, 17, 32, 33]. There are a still open questions to be further investigated and there is need for new questions to be raised. For example, the merits of using *within* data is known [15, 33], i.e. we know that the model of Figure 1(a) works successfully. On the other hand, merits of *cross* data in effort estimation is an open question. The research done on *cross* data so far is reported to be inconclusive [15], so we have both positive as well as negative results for the model of Figure 1(b). There is still need for a comprehensive study on this issue. In this study we investigate 21 WC-CC pairs and compare their performances.

It has been shown that performance of *cross* data can be improved via a relevancy filtering [17, 32]. So it is known that model of Figure 1(c) is a good candidate for making use of *cross* data in software effort estimation. In this work we use a variance-based relevancy filter in various scenarios.

The tendency of a test instance to select *within* or *cross* data instances has not been yet addressed. The outcomes of the model in Figure 1(d) are not known. The information about the selection tendency of a test instance can tell, if it is a good idea for an organization to combine limited *within* data with *cross* data. It can also specify in which proportions such a mixture should be. We define settings for that problem in this paper and look for answers to these problems.



(a) *Within* (b) *Cross* (c) *Filter* (d) *Within+Cross*

Figure 1: The problem types in *within* vs. *cross* data comparison and our conclusions so far. "+" and "-" signs on top of models mean positive and negative results respectively. A "?" sign means the model has not yet been investigated.

2.1 Contributions

The contributions of this research can be listed as follows:

- Analysis of effort datasets and proposition of subsets for possible WC-CC experiments.
- Investigating effort dataset features to see which features are likely to define *cross* and *within* boundaries.
- An extensive analysis on the merits of *cross* data and uniform conclusions.
- Proposing a mixture model of *within* and *cross* data.
- Investigating tendency of test instances towards *cross* and *within* data to find preferable percentage of a mixture.

3. RELATED WORK

We divide our related work into two sub-sections. §3.1 describes effort estimation in general and §3.2 provides in depth discussion of *cross* and *within* data usage in SE.

3.1 Effort Estimation

A high level taxonomy of the software effort estimation based on the adopted methodology reveals two fundamental groups: Algorithmic and non-algorithmic methods.

Effort estimates generated by algorithmic methods are the product of a model that is built on historical data. Such methods may entail the adaptation of an expert proposed model to historical data. A very well known example to that scenario is Boehm's COCOMO method [2]. However, this option requires a long time interval for *within* data collection and model adaptation to local data. An alternative is the processing of historical data by generic methods. Regression methods [25], neural nets [20], model trees [31] and instance-based models [11, 17, 25] are examples to this category.

Another proposed effort estimation strategy is the non-algorithmic methods, a.k.a. expert-based estimation. The models under this category can be defined as a human-intensive process of estimate negotiation between domain experts [8]. These negotiations continue until a concensus is reached among the experts. Possible pitfalls related with this family of methods are threefold:

1. obvious need for high-quality experts,
2. poor cabability of humans to improve their expert judgment skills [9] and
3. possible conflict of interest among domain experts [17].

3.2 Within-Company vs. Cross-Company

A baseline for successful algorithmic models is the historical effort datasets of past projects. An organization willing to employ an algorithmic model-based effort estimation in their processes may choose to benefit from one of the following:

- *within* data that is required to be collected *within* the organization
- *cross* data that was collected elsewhere and that needs to be adapted
- *combination* of *within* and *cross* data

The merits of *within* data has been shown in the literature; for accurate estimates previous work suggest *within* data that bears locality-speicific features. However, there are multiple issues associated with collecting and maintaining *within* data [15, 17, 22, 23]:

1. Long time requirement for accumulation of enough local data
2. Possibility of technology change by the time *within* data is ready
3. Sensitivity to possible human errors in data collection
4. Loss of managerial interest due to long time constraint

Unlike *within* data, the reported results regarding *cross* data are inconclusive. An extensive systematic review by Kitchenham et al. on the value *cross* data reports that only 7 out of 10 studies in the review are able to show independent evidence in their comparisons

of *within* and *cross* data [15]. Out of this 7, 4 studies favor *within* data, whereas the remaining 3 report that *cross* data performance is not significantly worse than *within*.

Another field of SE that questions the merits of *cross* data is defect prediction and the inconclusive scenario endures. Zimmermann et al. [33] study *cross* data in an as is manner and out of 622 *cross* predictions only 3.4% is reported to work. Turhan et al. reports successful applications of *cross* data: Filtering *cross* data through a nearest-neighbor (NN) based filter increases defect prediction probability [32]. This filtering approach has also inspired effort estimation domain. A variance based filtering on *cross* data has attained comparable performance to that of *within* data [17]. The problem of *cross* data has been paid little attention [33] and new questions like the mixture of *within* and *cross* data) are yet to be raised. Various aspects of this mixture approach are investigated in this paper.

4. METHODOLOGY

The details of dataset filtering and division of them into *cross-within divisions* are presented here. We also explain the selected performance measures and our experimental set-up.

4.1 Datasets

There are 2 fundamental factors that were considered for selection of the datasets used in this research:

- Public availability: For reproducibility purposes
- Cross-within divisibility: For enabling *cross* vs. *within* experimentation

A critical issue in software engineering is the ability of the proposed results to be reproducible [10, 15] and use of proprietary data is a major obstacle towards this goal. Therefore, all our datasets are publicly available through PROMISE data repository [3]. Although there are more than twenty effort datasets available in PROMISE, they are not all available for *cross-within* experimentation. The available datasets should be able to provide *cross-within division*. We define *cross-within division* as the subset(s) of effort data that are formed through division of a nominal attribute: Instances having the same value for that nominal attribute form a subset. The nominal attribute should be a plausible candidate for a *cross* company setting, i.e. the attribute should be likely to change from one company to other.

After manually inspecting more than 20 datasets and 6 are selected for *cross-within* experimentation. 6 datasets defined 8 *cross-within divisions* according to their available nominal attributes. 8 divisions include 21 subsets, i.e. 21 different WC-CC pairs. The datasets, *cross-within divisions* (subsets) and the division criteria are given in Figure 10. The selected division criteria include:

- project type: embedded, organic and semidetached (*cocomo81*),
- center: geographical development center (*nasa93*),
- language type: programming language used for development (*desharnais*),
- application type: on-line service program, production control program etc. (*finnish* and *maxwell*),
- hardware: PC, mainframe, networked etc. (*kemerer* and maxwell),
- source: whether in-house or outsourced (maxwell).

Note that each subset in Figure 10 is named with a self-explanatory abbreviation. The numbers at the end of abbreviations correspond to values of the nominal attribute used to form the subsets. If a name has multiple numbers at the end (e.g. finnishAppType2345) this means that all instances with these nominal attribute values are combined in a single subset. Also note that the terms *within/cross* and WC/CC will be used interchangeably for the rest of the text.

Dataset	Criterion	Subsets	Subsets Size
cocomo81	project type	cocomo81e	28
		cocomo81o	24
		cocomo81s	11
nasa93	development center	nasa93_center_1	12
		nasa93_center_2	37
		nasa93_center_5	39
desharnais	language type	desharnaisL1	46
		desharnaisL2	25
		desharnaisL3	10
finnish	application type	finnishAppType1	17
		finnishAppType2345	18
kemerer	hardware	kemererHardware1	7
		kemererHardware23456	8
maxwell	application type	maxwellAppType1	10
		maxwellAppType2	29
		maxwellAppType3	18
maxwell	hardware	maxwellHardware2	37
		maxwellHardware3	16
		maxwellHardware5	7
maxwell	source	maxwellSource1	8
		maxwellSource2	54

Figure 2: 6 datasets are selected from 20+ candidates. Then selected datasets are divided into subsets according to a criterion that can define a *cross-within division*. The datasets, subset sizes as well as the selection criteria are provided here.

We acknowledge that *cross-within division* is a validity concern (a detailed discussion is given in §6). However, we hasten to say that it is an acceptable experimentation method, because: a) Despite all our efforts, public effort datasets are still limited; b) For a comprehensive analysis (as this study does on 21 subsets), alternative use of available datasets should be considered.

4.2 Methods

We used 2 different ABE methodologies in this paper: a relevancy filtering-based ABE method called "Test Essential Assumption Knowledge" (TEAK) [19] and ABE0 [17, 19].

4.2.1 ABE0: A Baseline Analogy-based Estimation

ABE methods generate an estimate for a *test* project by retrieving similar past projects (a.k.a. analogies) from a database of past projects and adapting their effort values into an estimate. There are various design options associated with ABE methods such as the distance measure for nearness [24], adaptation of analogy effort values [24], row processing [4, 13], column processing [13, 21] and so on. Elsewhere [12] we show that these options can easily lead to more than 6000 ABE variants. When ABE methods proposed by Kadoda & Shepperd [11], Mendes et al. [24], and Li et al. [21] are followed, a baseline variant emerges:

- Form a database of past projects, whose rows are projects instances and whose columns are *independent* variables (that define projects) and a *dependent* variable (effort value).
- Decide how many similar projects (*analogies*) are to be used from the training set, i.e *k*-value.
- For each test instance, retrieve *k* analogies from the database.
  - For selection of *k* analogies use a similarity measure like Euclidean distance measure.

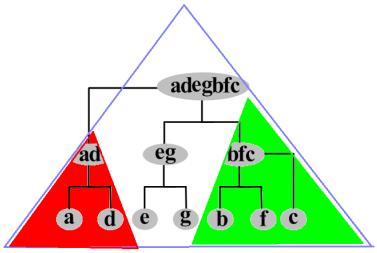


Figure 3: A sample GAC tree with regions of high variance (red) and low variance (green). GAC trees may not always be binary. For example here, leaves are odd numbered, hence node "c" is left *behind*. Such instances are pushed *forward* into the closest node in the higher level. For example, "c" is pushed forward into the "b+f" node to make "b+f+c" node.

  - Before calculating similarity, scale independent features to equalize their influence on the similarity measure.
  - Use a feature weighting scheme to reduce the effect of less informative features.
- Adapt the effort values of the k nearest analogies to come up with the effort estimate.

Following the steps of this baseline technique, we define a framework called ABE0. ABE0 uses the Euclidean distance as a similarity measure, whose formula is given in Equation 1, where w_i corresponds to feature weights applied on independent features. ABE0 framework does not favor any features over the others, therefore each feature has equal importance in ABE0, i.e. $w_i = 1$. For adaptation ABE0 takes the median of selected k projects.

$$Distance = \sqrt{\sum_{i=1}^{n} w_i (x_i - y_i)^2} \quad (1)$$

4.2.2 *TEAK: Test Essential Assumption Knowledge*

TEAK is a relevancy filtering-based ABE method that makes use of greedy-agglomerative clustering (GAC) trees. Detailed description of TEAK can be found in [19]. In summary, it is a two-pass system:

- Pass 1 removes the training instances implicated in poor decisions;
- Pass 2 selects the instances that are closest to the test instance.

In the first pass, training instances are combined into a GAC tree (called GAC1). A trivial example to GAC tree formation is provided in Figure 3. Level zero of GAC1 is formed by leaves, which are the individual project instances. These instances are greedily combined into tuples to form the nodes of upper levels. GAC1 is then traversed upwards from the root to level one (one level higher than the leaves). The variance of the effort values associated with each sub-tree (the performance variance) is then recorded and normalized to a 0-1 interval. The high variance sub-trees are then chopped-off, as these are the sub-trees that would cause an ABE method to make an estimate from a highly variable instance space. Hence, pass one prunes sub-trees with a variance greater than $\alpha\%$ of the maximum variance seen in any tree. After some experimentation, we found that $\alpha = 10$ lead to estimates with lowest errors.

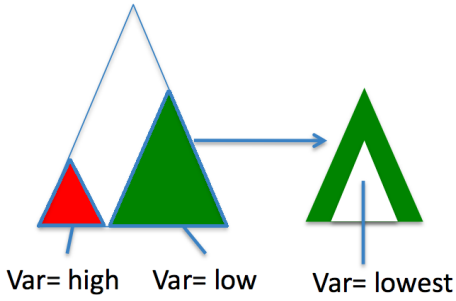


Figure 4: Execution of TEAK on 2 GAC trees, where tree on the left is GAC1 and the one on the left is GAC2 (i.e. lower variance sub-tree of GAC1). The instances in the low variance region of GAC1 (green region) are selected to form GAC2. Then test instance traverses GAC2 until no decrease in effort variance is possible. Wherever the test instance stops is selected as the subtree to be used for adaptation (white region of GAC2).

The leaves of the remaining sub-trees are the *survivors* of pass one. They are filtered to pass 2 where they are used to build a second GAC tree (GAC2). GAC2 is generated and traversed in a similar fashion to GAC1, then test instances are moved from root to leaves. Unlike GAC1, this time variance is a decision criterion for the movement of test instances: If the variance of the current tree is larger than its sub-trees, then continue to move down; otherwise, stop and select the instances in the current tree as the analogies. TEAK is a form of ABE0, so its adaptation method is the same, i.e. take the median of the analogy effort values. A simple visualization of this approach is given in in Figure 4.

4.2.3 *Why ABE methods: ABE0 and TEAK?*

The comparison of ABE methods vs. non-ABE methods (e.g. k-NN vs. neural nets) is *not* within the scope of this paper. A detailed comparison of different methods on effort estimation can be found in [25]. The reasons behind selection of ABE methods (ABE0 and TEAK) in this research are threefold: 1) they are widely investigated in software effort estimation [4, 11, 13, 17, 19, 21, 24], 2) they are particularly helpful for *cross* company examinations as they are based on distances between individual project instances and 3) analogy methods are comparable -if not better- to non-analogy methods in terms of performance.

In [19] we have compared performance of TEAK and ABE0 to non-analogy estimators (neural networks (NNet) and linear regression (LR)). An excerpt from that comparison is given in Figure 5 (for a complete analysis and for definitions of datasets please refer to Figure 7 of [19]). Note in Figure 5 that TEAK is usually the high performer in comparison to non-ABE methods.

4.3 Performance Measures

A performance measure comments on the success of an estimate, hence the predictor. Performance measures listed here have the property that there is at least one publication in effort estimation

20 × LEAVE-ONE-OUT

	TEAK	LR	NNet	*k=best*	k=1	k=16	k=2	k=4	k=8
MdMRE									
Cocomo81	▲								
Cocomo81e	▲								
Cocomo81o	▲								
Nasa93		▲							
Nasa93c2		▲							
Nasa93c5	▲								
Desharnais		▲							
Sdr	▲								
ISBSG-Banking	▲								
Count	6	3	0	0	0	0	0	0	0
Pred(25)									
Cocomo81	▲								
Cocomo81e			▲						
Cocomo81o	▲								
Nasa93		▲							
Nasa93c2		▲							
Nasa93c5	▲								
Desharnais		▲							
Sdr	▲								
ISBSG-Banking	▲								
Count	5	3	1	0	0	0	0	0	0
MAR									
Cocomo81	▲								
Cocomo81e	▲								
Cocomo81o	▲								
Nasa93		▲							
Nasa93c2		▲							
Nasa93c5	▲								
Desharnais		▲							
Sdr	▲								
ISBSG-Banking	▲								
Count	6	3	0	0	0	0	0	0	0

Figure 5: This figure displays the top performing estimation methods, measured via $(win - loss)$ and repeated for the performance measures of MdMRE, Pred(25) and MAR. The last row of each table shows the sum of times a method appeared as the top performing variant. Note that TEAK is comparable to or better than non-analogy methods.

research proposing their use. An example performance measure is the absolute residual (AR), which is the absolute difference between predicted and the actual effort values. AR formula is given in Equation 2, where $x_i, \hat{x_i}$ are the actual and predicted values respectively for test instance i. Summary of individual AR values is found by taking their mean (MAR).

$$AR_i = |x_i - \hat{x_i}| \tag{2}$$

Another performance measures is the Magnitude of Relative Error (MRE), which is a widely used method to select the best estimator from a number of competing models [6, 28]. MRE is the measure of the error ratio between the actual and the predicted effort:

$$MRE_i = \frac{\mid x_i - \hat{x_i} \mid}{x_i} = \frac{\mid AR_i \mid}{x_i} \tag{3}$$

MRE can be summarized through mean or median. The former summary defines mean MRE (MMRE) and the latter defines median MRE (MdMRE). Formulas of MMRE and MdMRE are:

$$MMRE = mean(MRE_1, MRE_2, ..., MRE_n) \tag{4}$$
$$MdMRE = median(MRE_1, MRE_2, ..., MRE_n) \tag{5}$$

An alternative to prior performance measures is $Pred(x)$, i.e. the

win_i = 0, tie_i = 0, $loss_i$ = 0
win_j = 0, tie_j = 0, $loss_j$ = 0
if WILCOXON(P_i, P_j) says they are the same **then**
  $tie_i = tie_i + 1$;
  $tie_j = tie_j + 1$;
else
  if mean or median(P_i) < median(P_j) **then**
    $win_i = win_i + 1$
    $loss_j = loss_j + 1$
  else
    $win_j = win_j + 1$
    $loss_i = loss_i + 1$
  end if
end if

Figure 6: Pseudocode for win-tie-loss calculation between methods *i* and *j* w.r.t. performance measures P_i and P_j. Note here that only for Pred(30) the comparison is based on actual values ($Pred(30)_i$, $Pred(30)_j$) rather than mean or median values of performance measure arrays (median(P_i), median(P_j)).

percentage of estimates that fall within $x\%$ of the actual values:

$$Pred(x) = \frac{100}{N} \sum_{i=1}^{N} \begin{cases} 1 \text{ if } MRE_i \leq \frac{x}{100} \\ 0 \text{ otherwise} \end{cases} \tag{6}$$

A common value for x in $Pred(x)$ is 30 [5]. For example, $Pred(30) = 50\%$ implies that half of the estimates are within 30% of the actual values [28].

It is reported as a considerable threat to the validity of effort estimation studies, if derived performance measures are not evaluated with an appropriate statistical test [14]. In this study we use a Mann Whitney test (95%). To compare the values of MAR, MMRE, MdMRE and Pred(30), we use the following win-tie-loss procedure that incorporates Mann Whitney statistical test. We first check if two distributions i, j are statistically different according to the Mann Whitney test. In our experimental setting, i, j are arrays of performance measure results coming from two different methods. If they are not statistically different, then they are said to *tie* and we increment tie_i and tie_j. On the contrary, if they are different, we updated win_i, win_j and $loss_i, loss_j$ after a numerical comparison of performance measures. The related pseudocode is given in Figure 6. To get rid of any bias that would come from a particular experimental setting, for every experiment 20 runs are made.

4.4 Experimentation

The experimentation of this research has two different scenarios: Performance comparison and selection tendency. Performance comparison scenario compares different ABE methods (TEAK and ABE0 with *k=4,best*) when subject to *within* and *cross* company data. The selection tendency experiments question the tendency of a *within* test instance towards *within* or *cross* data. In other words, given the chance that a test instance had access to *within* and *cross* data at the same time, what percentage of every subset would be selected into *k* analogies used for estimation. The details of each scenario is presented in the following subsections.

4.4.1 Performance Comparison

For performance comparison scenario we have two settings: *Within* and *cross*. In *within* data setting, only *within* data is used as the dataset and a testing strategy of leave-one-out cross-validation (LOOCV) is employed. LOOCV works as follows: Given a *within* dataset of

T projects, 1 project at a time is selected as the test and the remaining $T-1$ projects are used for training, so eventually we have T predictions. The resulting T predictions are then used to compute 4 different performance measures of §4.3.

Cross data setting uses *within* data as the test set and the *cross* data as the training set. This setting is basically a simulation of an organization that has projects to estimate (*within* data as the test set) and uses data of past projects with recorded effort values from *cross* organization(s). In this setting LOOCV is used: Each *within* project is selected as the test instance and ABE methods derive an estimate for that instance by adapting *cross* analogies. Ultimately we end up with T predictions adapted from a *cross* dataset. Finally the performances of TEAK and ABE0 methods under *within* and *cross* data settings are compared. For that purpose we use both mere performance values as well as win-tie-loss statistics.

4.4.2 Selection Tendency

For the selection tendency scenario we select test instances according to LOOCV. For each test instance, we are left with training sets of $T-1$ *within* data and the subsets of *cross* data. After marking every *within* and *cross* instance, we combine the two datasets into a single training set and let the test instance choose analogies from the unified training set. In this setting our aim is to see what percentage of *within* and *cross* subsets would appear among selected *k* analogies. The percentage for a subset S_i is calculated in accordance with Equation 7:

$$Percentage = \frac{\#ofInstancesFromS_iInAnalogies}{Size(S_i)} \quad (7)$$

5. RESULTS

5.1 Performance Comparison

The first experimental scenario we are interested in is the performance of *within* and *cross* data. For performance comparison 4 different performance measures are employed: MAR, MMRE, MdMRE and Pred(30). The actual performance values are also evaluated subject to Mann Whitney statistical test at 95% confidence and this evaluation is summarized by win-tie-loss statistics. Performance comparison subject to estimates of TEAK, ABE0 and log+ABE0 is reported in Figure 7, Figure 8 and Figure 9 respectively.

Figure 7 shows *within* and *cross* data performance when TEAK is used as the estimation method. For each performance measure win-tie-loss statistics (abbreviated with **W**, **T**, **L** respectively) of **WC** performance when compared to **CC** over 20 runs as well as actual performance measure values are reported. For convenience, the cases where **WC** is *"dominantly"* superior to **CC** are highlighted. A *"dominant"* superior condition means winning more than half the runs, i.e. a **W** value of more than 10. Note that there are only 2 such cases, where **WC** is dominantly better than **CC**: cocomo81s and desharnaisL1. For the remaining 19 cases *within* data does not provide an advantage over *cross* data. In one particular case (kemererHardware1) the *within* data is far worse than *cross* with an **L** value of 20. These results are confirmation of previous conclusions [17, 32] in a much larger scale with 4 error measures and 21 different cases: Relevancy filtering on *cross* data improves its performance to an extent where it is no worse than *within* data. One likely question to be raised is why particular cases favor *within* or *cross* data. This question is out of our scope and is left as a future direction to this research.

	MAR					MMRE					MdMRE					Pred(30)				
Dataset	**W**	**T**	**L**	**WC**	**CC**	**W**	**T**	**L**	**WC**	**CC**	**W**	**T**	**L**	**WC**	**CC**	**W**	**T**	**L**	**WC**	**CC**
cocomo81e	0	20	0	1.0E+3	1.1E+3	0	16	4	2.4	0.9	4	16	0	0.7	0.9	4	16	0	0.1	0.1
cocomo81o	0	20	0	8.2E+2	8.1E+2	2	18	0	0.8	2.7	2	18	0	0.8	0.9	2	18	0	0.1	0.2
cocomo81s	18	2	0	3.6E+1	1.8E+2	15	5	0	1.0	8.6	15	5	0	0.5	1.7	13	5	2	0.2	0.1
nasa93_center_1	0	20	0	1.4E+2	1.3E+2	0	20	0	1.2	2.0	0	20	0	0.8	0.8	0	20	0	0.6	0.5
nasa93_center_2	4	16	0	1.8E+2	2.1E+2	2	18	0	1.3	2.8	2	18	0	0.7	0.8	2	18	0	0.2	0.2
nasa93_center_5	0	20	0	6.9E+2	8.9E+2	0	12	8	0.9	0.7	8	12	0	0.6	0.8	8	11	1	0.2	0.2
desharnaisL1	11	9	0	9.9E+2	2.0E+3	9	11	0	0.6	2.4	9	11	0	0.4	1.7	9	11	0	0.4	0.3
desharnaisL2	0	20	0	2.8E+3	2.8E+3	0	20	0	0.5	0.6	0	20	0	0.5	0.5	0	20	0	0.2	0.3
desharnaisL3	0	20	0	2.8E+3	3.2E+3	2	18	0	0.5	0.5	2	18	0	0.4	0.5	2	18	0	0.2	0.2
finnishAppType1	0	20	0	3.2E+3	3.8E+3	0	20	0	1.1	1.0	0	20	0	0.5	0.6	0	20	0	0.3	0.2
finnishAppType2345	0	20	0	7.1E+3	5.4E+3	0	17	3	2.2	0.9	0	17	3	0.8	0.7	0	17	3	0.1	0.2
kemererHardware1	0	0	20	1.4E+2	5.5E+1	0	0	20	1.3	0.3	0	0	20	1.1	0.3	0	0	20	0.4	0.5
kemererHardware23456	0	20	0	2.0E+2	2.0E+2	0	20	0	0.7	0.7	0	20	0	0.6	0.5	0	20	0	0.1	0.1
maxwellApp1type1	6	14	0	1.4E+3	3.2E+3	1	19	0	0.8	1.9	1	19	0	0.4	0.7	0	19	1	0.3	0.3
maxwellApp1type2	0	18	2	6.6E+3	5.4E+3	0	19	1	1.2	0.9	0	19	1	0.5	0.4	0	19	1	0.3	0.3
maxwellApp1type3	0	20	0	5.6E+3	6.6E+3	1	19	0	1.0	1.0	1	19	0	0.5	0.6	1	19	0	0.2	0.2
maxwellHardware2	0	20	0	5.6E+3	5.3E+3	0	20	0	0.8	1.0	0	20	0	0.5	0.5	0	20	0	0.2	0.2
maxwellHardware3	0	20	0	5.3E+3	5.9E+3	0	20	0	0.9	0.7	0	20	0	0.4	0.5	0	20	0	0.3	0.4
maxwellHardware5	0	20	0	3.6E+3	3.6E+3	0	20	0	3.7	2.8	0	20	0	0.7	0.8	0	20	0	0.1	0.1
maxwellSource1	6	14	0	1.5E+3	3.3E+3	1	19	0	0.3	0.4	1	19	0	0.1	0.4	1	19	0	0.8	0.8
maxwellSource2	0	20	0	6.0E+3	6.0E+3	0	20	0	1.2	1.9	0	20	0	0.6	0.7	0	20	0	0.2	0.2

Figure 7: Results of TEAK: Comparison of performance between *within* and *cross* data w.r.t. 4 different performance measures (MAR, MMRE, MdMRE, Pred(30)) as well as *W, T, L* statistics. Highlighted rows are the cases, where *within* data is *"dominantly"* better than *cross*, i.e. wins more than half the time. Under the columns of *WC* and *CC* the actual performance values associated with *within* and *cross* company datasets are provided respectively.

Another intriguing question is what happens in this large-scale comparison, when we remove the relevancy filtering. The performance comparison of *within* and *cross* data subject to estimates of ABE0 with *k={4, best}* is given in Figure 8. The cells of Figure 8 can have 3 values: *"+", "-" and "o"*.. The *"+"* sign tells that *within* data performance is *"dominantly"* better than that of *cross* data, i.e. *within* won more than 10 out of 20 runs, whereas a *"-"* sign

tells that *within* lost more than 10 runs. If none of these conditions occur, i.e. *within* and *cross* performances tie, then a "o" sign is assigned to the cell. The cases where *within* data is *"dominantly"* better than *cross* are highlighted.

Dataset	MAR		MMRE		MdMRE		Pred(30)	
	k=4	*k*=best	*k*=4	*k*=best	*k*=4	*k*=best	*k*=4	*k*=best
cocomo81e	o	o	o	o	o	o	o	o
cocomo81o	o	o	+	o	-	o	-	o
cocomo81s	o	o	o	o	o	o	o	o
nasa93_center_1	o	-	o	-	o	-	o	-
nasa93_center_2	o	o	o	o	o	o	o	o
nasa93_center_5	o	o	+	o	-	o	-	o
desharnaisL1	-	-	o	o	o	o	o	o
desharnaisL2	-	o	+	o	-	o	-	o
desharnaisL3	o	o	o	o	o	o	o	o
finnishAppType1	o	o	o	o	o	o	o	o
finnishAppType2345	o	o	o	o	o	o	o	o
kemererHardware1	o	o	o	o	o	o	o	o
kemererHardware23456	o	o	o	o	o	o	o	o
maxwellAppType1	o	o	o	o	o	o	o	o
maxwellAppType2	o	o	o	o	o	o	o	o
maxwellAppType3	o	o	o	o	o	o	o	o
maxwellHardware2	o	o	o	o	o	o	o	o
maxwellHardware3	o	o	o	o	o	o	o	o
maxwellHardware5	o	o	o	o	o	o	o	o
maxwellSource1	o	o	o	o	o	o	o	o
maxwellSource2	o	o	o	o	o	o	o	o

Figure 8: ABE0 performance comparison between *within* and *cross* data w.r.t. 4 different performance measures (MAR, MMRE, MdMRE, Pred(30)) with different *k* values. Each cell in this table can have three values: *"+", "-" and "o"*.. A *"+"* sign indicates that *within* performance is *"dominantly"* better than *cross*, i.e. it won more 10 of the 20 runs, whereas a *"-"* sign tells that *cross* lost more than 10 runs. If none of these conditions occur, i.e. *within* and *cross* performances tie, then a "o" sign is assigned to the cell. For convenience *"+"* signs are highlighted.

Notice in Figure 8 that out of *21 Subsets* $\times$ *4 error measures* $\times$ *2 ABE0 methods = 168 cases*, there are only 3 cases where *within* performance is dominantly better. For the majority of the remaining cases, *within* and *cross* performance is comparable. Surprisingly for a minority of the remaining cases (see "-" signs), *cross* data performance is dominantly better than that of *within*. These results are important in the sense that unlike previous work that report inconclusive results on the merits of cross data [16,33]; we are able to see enough uniformity in a large scale experiment with 168 cases. The uniformity concludes that *cross* data is as high performing as *within* data. However, we should remember that such uniformity may come from *cross-within divisions*, which assume *cross* subsets of a division are all collected through the same methodology.

The pre-processors applied on the data before an estimation model can have a significant effect on the performance [25]. In [12] we investigate the stability of rankings among 90 different methods that include both analogy and non-analogy methods. We have seen that applying a *log* transformation on the data as a pre-processing step to ABE0 (log+ABE0), can improve the ranking of ABE0 by orders of magnitude. The results of log+ABE0 is summarized in Figure 9. The notation of Figure 9 is the same as that of Figure 8. See that the general picture of Figure 8 repeats in Figure 9: There are only a few cases that favor *within* data and in the majority of the cases *within* and *cross* data have comparable performances.

5.2 Selection Tendency

The second experimental scenario of this research is the selection tendency. In this setting LOOCV is used to select out single test instances one by one from a *within* dataset of size T. Remaining $T-1$ *within* instances are combined with the *cross* subsets.

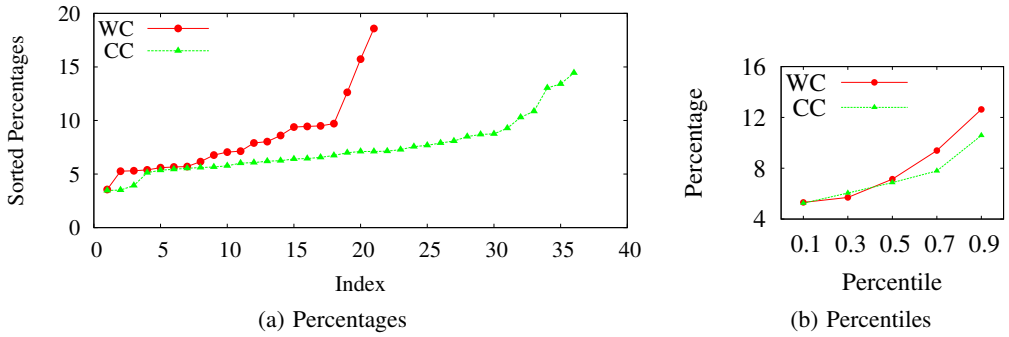
Dataset	MAR		MMRE		MdMRE		Pred(30)	
	k=4	*k*=best	*k*=4	*k*=best	*k*=4	*k*=best	*k*=4	*k*=best
cocomo81e	o	o	o	-	o	-	o	-
cocomo81o	o	-	+	-	-	-	-	-
cocomo81s	o	o	o	o	o	o	o	o
nasa93_center_1	-	-	+	-	-	-	-	-
nasa93_center_2	o	o	o	o	o	o	o	o
nasa93_center_5	o	o	o	-	o	-	o	-
desharnaisL1	-	-	o	o	o	o	o	o
desharnaisL2	o	o	+	o	-	o	-	o
desharnaisL3	o	o	o	o	o	o	o	o
finnishAppType1	o	o	o	o	o	o	o	o
finnishAppType2345	o	o	o	o	o	o	o	o
kemererHardware1	o	o	o	o	o	o	o	o
kemererHardware23456	o	o	o	o	o	o	o	o
maxwellAppType1	o	o	o	o	o	o	o	o
maxwellAppType2	o	o	o	o	o	o	o	o
maxwellAppType3	o	o	o	o	o	o	o	o
maxwellHardware2	o	o	o	o	o	o	o	o
maxwellHardware3	o	o	o	o	o	o	o	o
maxwellHardware5	o	o	o	o	o	o	o	o
maxwellSource1	o	o	o	o	o	o	o	o
maxwellSource2	o	o	o	o	o	o	o	o

Figure 9: *log*+ABE0 performance comparison between *within* and *cross* data w.r.t. 4 different performance measures (MAR, MMRE, MdMRE, Pred(30)) with different *k* values. The notation used here is the same as Figure 8. The highlighted cells are the cases, where *within* data is dominantly better than *cross* data.

Test Set	Zone	From S1	From S2	From S3
cocomo81e (28)	3.7	1.0 (3.6%)	1.1 (4.8%)	1.6 (14.4%)
cocomo81o (24)	4.3	1.8 (6.6%)	1.3 (5.6%)	1.1 (10.4%)
cocomo81s (11)	4.1	1.4 (5.1%)	1.7 (7.0%)	1.0 (9.4%)
nasa93_center_1 (12)	5.6	1.0 (8.1%)	2.9 (7.9%)	1.7 (4.3%)
nasa93_center_2 (37)	10.0	1.6 (13.0%)	4.6 (12.4%)	3.8 (9.8%)
nasa93_center_5 (39)	5.1	0.8 (6.7%)	2.2 (6.0%)	2.1 (5.4%)
desharnaisL1 (46)	5.0	2.5 (5.5%)	1.7 (7.0%)	0.8 (7.9%)
desharnaisL2 (25)	4.8	2.6 (5.6%)	1.5 (6.1%)	0.7 (6.7%)
desharnaisL3 (10)	3.5	1.9 (4.1%)	1.3 (5.0%)	0.4 (4.0%)
finnishAppType1 (17)	3.1	1.6 (9.1%)	1.6 (8.8%)	
finnishAppType2345 (18)	3.0	1.4 (8.2%)	1.6 (8.8%)	
kemererHardware1 (7)	1.5	0.6 (8.8%)	0.9 (10.7%)	
kemererHardware23456 (8)	1.4	0.5 (7.3%)	0.8 (10.6%)	
maxwellAppType1 (10)	3.5	0.7 (7.1%)	1.7 (5.9%)	1.0 (5.8%)
maxwellAppType2 (29)	3.2	0.4 (3.7%)	1.8 (6.2%)	1.0 (5.5%)
maxwellAppType3 (18)	2.5	0.6 (6.3%)	0.9 (3.2%)	1.0 (5.6%)
maxwellHardware2 (37)	2.9	1.7 (4.6%)	0.8 (4.9%)	0.4 (6.0%)
maxwellHardware3 (16)	3.9	2.5 (6.8%)	1.1 (6.8%)	0.3 (4.3%)
maxwellHardware5 (7)	3.4	2.3 (6.2%)	0.8 (5.0%)	0.3 (4.5%)
maxwellSource1 (8)	3.0	0.1 (1.6%)	2.8 (5.2%)	
maxwellSource2 (54)	3.2	0.4 (4.6%)	2.8 (5.3%)	

Figure 10: The amount of instances selected from within and cross company datasets. In parenthesis the percentage of selected instances out of the actual within company dataset is given. The diagonal entries that are highlighted with gray are the within company selection amounts and percentages.

Prior to combination, every training instance is marked with the source that it belongs to. Then the test instance is allowed to choose *k* analogies from a training set of *within* and *cross* data. After processing test instances via TEAK, ABE0 and log+ABE0; we can see the percentage selection of analogies from each one of the *within* and *cross* subsets. Figure 10 shows the size of prediction ***"Zone"*** used by TEAK for estimation as well as the percentage selection of instances from *within* and *cross* subsets into this prediction zone. Each *cross-within division* is represented with a row of 2 or 3 subsets; columns named *"From* S_i*"* where $i \in \{1, 2, 3\}$ represent the subsets of the rows. The highlighted diagonal entries of each cell show the amount of instances (in percentage and in number) selected from *within* subset. The off-diagonal values are the amount of instances selected from *cross* datasets. See in Figure 10 that *within* test instances do not necessarily select all the analogies from *within* subsets. On the contrary, the percentage of instances selected from *within* and *cross* datasets are very close to one another.



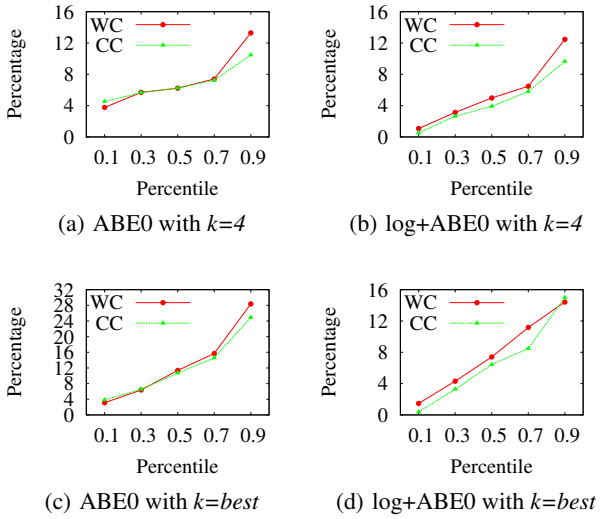
(a) Percentages (b) Percentiles

Figure 11: Percentages and percentiles of instances selected by TEAK from WC and CC datasets. The CC percentages are very similar to shifted version of WC percentages, the shift-effect is due to different number of subsets. The percentile graph removes the shift-effect and we see that *within* test instances select very close percentages of *within* and *cross* company instances.

To better see the percentages of *within* and *cross* subsets, we sorted and plotted these percentage values in Figure 11. Figure 11(a) shows the sorted percentage values, where the WC data percentages are shown with circles, whereas the CC data percentages are represented by rectangles. See in Figure 11(a) how *cross* percentage values are shifted versions of *within* percentages. The shift-effect comes from the fact that there are more *cross* subsets than *within* subsets. The percentiles from 10^{th} to 90^{th} with increments of 20 are given in Figure 11(b). When we plot the percentiles, the shift-effect due to subset number disappears and we are able to observe the surprising fact that WC and CC percentages at indicated percentile values are very close. In other words, subject to a variance based relevancy filtering ABE model, a *within* test instance selects equal percentages from *within* and *cross* datasets.

The percentage ordering plots of ABE0 and log+ABE0 are very similar to Figure 11(a). Therefore, we did not include these figures here due to space constraints. Instead, we give the percentile plots in Figure 12. See in Figure 12 that percentage values of WC and CC are very close to one another for all percentile values. This means that subject to ABE methods with or without relevancy filtering, test instances select close amounts of WC and CC data.

We can now ask what percentage should be used when an organization has certain amount of *within* data but also needs the use of a *cross* dataset. The median values of the percentiles (i.e. 50^{th} percentile) in Figure 12 lie somewhere between 5% to 12%. This number can be an indicator for a mixture amount. However, we do not have enough evidence to claim such a value yet. Obviously further research is required to propose such a mixture amount and on the relative performances of different mixtures.



(a) ABE0 with *k=4* (b) log+ABE0 with *k=4*

(c) ABE0 with *k=best* (d) log+ABE0 with *k=best*

Figure 12: Percentiles of instances selected by ABE0 and log+ABE0 with *k={4, best}* from WC and CC datasets.

6. THREATS TO VALIDITY

External validity questions whether the results can be generalized outside the specifications of a study [26]. For the purpose of external validity, we use of 21 WC-CC dataset pairs. Among 10 studies investigated by Kitchenham et al. in [15], 9 of them used single WC-CC dataset pairs, and 1 study used 6 pairs. In terms of external validity, this report has higher validity than a standard *within* vs. *cross* data comparison effort estimation study.

Another consideration for external validity is the employed methods. Due to the nature of our research questions, particularly regarding selection tendency, we utilized ABE methods. There are thousands of possible ABE variants and there is no way that this study covers them all. There is obviously need for future research that repeats these experimentations with different ABE variants. However, experiments reported here include base variants (ABE0 and log+ABE0) as well as filtering based variants (TEAK) run on 21 WC-CC pairs. Therefore, the extent of the experimentation in this research offers enough support for the claims that 1) *cross* data performs no worse than *within* data and 2) a *within* test instance tends to create an equal mixture of *within* and *cross* projects.

Construct validity (i.e. face validity) asks if we are measuring what we actually intended to measure [27]. Previous studies have concerned themselves with the construct validity of different performance measures for effort estimation (e.g. [30]). So as not to bias our conclusions due to a limited number of performance measures, we used 4 different performance measures aided with win-tie-loss statistics. Another threat to construct validity is the formation of our *cross-within divisions*. The proposed division criteria for *within* and *cross* data may as well define different corporations in real world. Also the use of available public data promotes the reproducibility of the conclusions of a research. However, we acknowl-

edge that using *within* and *cross* data coming from completely different organizations would be a better option. On the other hand, the issue with using such proprietary data is the fact that they are difficult to be shared with research community and therefore proposed results cannot be reproduced, validated or refuted.

In terms of *internal validity* of our results, there is one dimension of experimental conditions not explored. We are making use of LOOCV, whose a possible alternative would be N-Way cross-validation. In N-Way cross-validation, data is randomly divided into B bins and each bin is tested on a model learned from the combination of other bins (typical values for B are 3 or 10). From a theoretical point of view, not controlling the stability of our results across different testing strategies is a threat to validity, as different testing strategies entails different bias and variance conditions [7]. Elsewhere [18], we show that there is very little difference in the bias and variance values generated for LOOCV and N-way cross-validation. Since two testing strategies have similar bias-variance characteristics for effort datasets, we opted for LOOCV due to the fact that LOOCV is a deterministic procedure that can be exactly repeated by any other researcher with access to a particular data set. N-way cross-validation on the other hand requires a random number generator and a stratification heuristic (to maintain same class distribution in each bin). Without access to exact same random number generator and stratification heuristic, it would be difficult for a researcher "A" to reproduce results of researcher "B".

7. CONCLUSION

At the end of an extensive experimentation, we can answer the research questions that were defined to guide this research.

RQ1: What can be said about *within* and *cross* data performances? The results of TEAK experiments confirm the findings of prior work [17, 32] on a larger scale: After relevancy filtering *cross* data performance is comparable to that of *within* data. The results of ABE0 experiments are quite interesting. For the selected effort datasets, there are very limited cases, where *within* data outperforms *cross* data. For the majority of the cases, *within* and *cross* data performances are comparable. The possible explanation for this outcome may be hidden in the fact that subsets of a particular effort dataset share similar methodologies and similar concerns in the collection process. However, we do not yet have enough evidence to strongly support that hypothesis.

RQ2: What is the selection tendency of *within* instances? A surprising result of this research is the fact that under different ABE models (TEAK, ABE0, log+ABE0) test instances selected analogies from *within* as well as *cross* data. This may be interpreted as a hint that organizations with very limited *within* data can incorporate *cross* data into their database.

RQ3: What is the reason for a particular tendency of *within* instances? In our experiments we have identified possible features in effort datasets for *cross-within divisions*. Since test instances select analogies from all the subsets, the *cross* subsets defined by these features are obviously good candidates to provide connections between *cross* company data.

RQ4: What would be a good mixture of *wihin* and *cross* data? The percentage distributions observed in §5 show that the percentage amount of instances selected from *within* and *cross* datasets are very close to one another. So a good mixture seems to be equal percentages from different datasets, i.e. if an organization uses $x\%$ from their *within* dataset, then the amount of instances selected from a *cross* dataset could be close to $x\%$. Furthermore, the percentile figures show that median percentage value is between 5% to 12%. This can be an indicator interval for the mixture of *within* and *cross* subsets.

RQ5: Under which conditions would *cross* data be favorable? Our results show that under 2 conditions *cross* data would be beneficial for an organization: 1) When a relevancy filter is used and 2) when *cross* data has a common feature (as those listed in this research) to the *within* company.

RQ6: Which features are likely to define *cross* and *within* boundaries? Our aim with this research question was to find features that would define boundaries to separate *within* and *cross* data from one another so that test instances would select mostly from its related *within* dataset. However, none of the features we have identified resulted in such a scenario. On the contrary, they defined links between *cross* datasets so that test instances selected from all subsets.

8. FUTURE DIRECTIONS

Some of the most likely future directions to this research are:

- Reproduction of this work on proprietary data.
- Investigating why particular subsets (cocomo81s, desharnaisL1) favor *within* data, whereas the rest favors both *within* and *cross*.
- Using different ABE or non-ABE methods under similar settings.
- Experimenting if limited *within* data can be supplemented with *cross* data.
- Using different features on different datasets to see if they can define a border between *within* and *cross* data.

9. REFERENCES

[1] B. Boehm, C. Abts, and S. Chulani. Software development cost estimation approaches âĂŞ A survey. *Annals of Software Engineering*, 10:177–205, 2000.

[2] B. W. Boehm. *Software Engineering Economics*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1981.

[3] G. Boetticher, T. Menzies, and T. Ostrand. PROMISE.

[4] C.-l. Chang. Finding Prototypes for Nearest Classifiers. *IEEE Transactions on Computer*, C(11), 1974.

[5] S. Chulani, B. Boehm, and B. Steece. Bayesian Analysis of Empirical Software Engineering Cost Models. *IEEE Trans. Softw. Eng.*, 25(4):573–583, 1999.

[6] T. Foss, E. Stensrud, B. Kitchenham, and I. Myrtveit. A simulation study of the model evaluation criterion mmre. *IEEE Transactions on Software Engineering*, 29(11):985–995, Nov. 2003.

[7] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer, 2 edition, 2008.

[8] M. Jorgensen. A review of studies on expert estimation of software development effort. *Journal of Systems and Software*, 70(1-2):37–60, Feb. 2004.

[9] M. Jorgensen and T. Gruschke. The impact of lessons-learned sessions on effort estimation and uncertainty assessments. *Software Engineering, IEEE Transactions on*, 35(3):368–383, 2009.

[10] N. Juristo and S. Vegas. Using Differences among Replications of Software Engineering Experiments to Gain Knowledge. In *International Symposium on Empirical*

Software Engineering and Measurement, pages 356–366, 2009.

[11] G. Kadoda, M. Cartwright, and M. Shepperd. On configuring a case-based reasoning software project prediction system. In *UK CBR Workshop, Cambridge, UK*, pages 1–10. Citeseer, 2000.

[12] J. Keung, E. Kocaguneli, and T. Menzies. A Ranking Stability Indicator for Selecting the Best Effort Estimator in Software Cost Estimation. *Submitted to Automated Software Engineering*, 2011.

[13] J. W. Keung, B. Kitchenham, and D. R. Jeffery. Analogy-X: Providing Statistical Inference to Analogy-Based Software Cost Estimation. *IEEE Trans. Softw. Eng.*, 34(4):471–484, 2008.

[14] B. Kitchenham and E. Mendes. Why comparative effort prediction studies may be invalid. In *PROMISE '09: Proceedings of the 5th International Conference on Predictor Models in Software Engineering*, pages 1–5, New York, NY, USA, 2009. ACM.

[15] B. Kitchenham, E. Mendes, and G. H. Travassos. Cross versus Within-Company Cost Estimation Studies: A Systematic Review. *IEEE Trans. Softw. Eng.*, 33(5):316–329, 2007.

[16] B. Kitchenham, E. Mendes, and G. H. Travassos. Cross versus Within-Company Cost Estimation Studies: A Systematic Review. *IEEE Trans. Softw. Eng.*, 33(5):316–329, 2007.

[17] E. Kocaguneli, G. Gay, T. Menzies, Y. Yang, and J. Keung. When to use data from other projects for effort estimation. In *Proceedings of the IEEE/ACM international conference on Automated software engineering*, pages 321–324. ACM, 2010.

[18] E. Kocaguneli and T. Menzies. The Effects of Test Set Selection on Effort Estimation (in preperation), 2011.

[19] E. Kocaguneli, T. Menzies, A. Bener, and J. Keung. Exploiting the Essential Assumptions of Analogy-based Effort Estimation. *To Appear in IEEE Trans. Softw. Eng*, 2011.

[20] Y. Kultur, B. Turhan, and A. B. Bener. ENNA: software effort estimation using ensemble of neural networks with associative memory. In *SIGSOFT '08/FSE-16: Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering*, pages 330–338, New York, NY, USA, 2008. ACM.

[21] Y. LI, M. XIE, and T. GOH. A study of project selection and feature weighting for analogy based software cost estimation. *Journal of Systems and Software*, 82(2):241–252, Feb. 2009.

[22] E. Mendes and B. Kitchenham. Further comparison of cross-company and within-company effort estimation models for web applications. *10th International Symposium on Software Metrics, 2004. Proceedings.*, pages 348–357, 2004.

[23] E. Mendes, C. Lokan, R. Harrison, and C. Triggs. A Replicated Comparison of Cross-Company and Within-Company Effort Estimation Models Using the ISBSG Database. *11th IEEE International Software Metrics Symposium (METRICS'05)*, pages 36–36, 2005.

[24] E. Mendes, I. Watson, C. Triggs, N. Mosley, and S. Counsell. A comparative study of cost estimation models for web hypermedia applications. *Empirical Software Engineering*, 8(2):163–196, 2003.

[25] T. Menzies, Z. Chen, J. Hihn, and K. Lum. Selecting Best Practices for Effort Estimation. *IEEE Transactions on Software Engineering*, 32(11):883–895, 2006.

[26] D. Milic and C. Wohlin. Distribution Patterns of Effort Estimations. In *Euromicro*, 2004.

[27] C. Robson. *Real world research: a resource for social scientists and practitioner-researchers*. Blackwell Publisher Ltd, 2002.

[28] M. Shepperd and C. Schofield. Estimating Software Project Effort Using Analogies. *IEEE Trans. Softw. Eng.*, 23(11):736–743, 1997.

[29] Spareref.com. NASA to Shut Down Checkout & Launch Control System.

[30] E. Stensrud, T. Foss, B. Kitchenham, and I. Myrtveit. An empirical validation of the relationship between the magnitude of relative error and project size. *Proceedings Eighth IEEE Symposium on Software Metrics*, pages 3–12, 2002.

[31] B. Stewart. Predicting project delivery rates using the NaiveâĂŞBayes classifier. *Journal of Software Maintenance and Evolution: Research and Practice*, 14(3):161–179, May 2002.

[32] B. Turhan, T. Menzies, A. Bener, and J. Di Stefano. On the relative value of cross-company and within-company data for defect prediction. *Empirical Software Engineering*, 14(5):540–578, 2009.

[33] T. Zimmermann, N. Nagappan, H. Gall, E. Giger, and B. Murphy. Cross-project defect prediction. *Proceedings of the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering on European software engineering conference and foundations of software engineering symposium - E*, page 91, 2009.