The Business Case For Software Repositories

Ekrem Kocaguneli, Gregory Gay
Tim Menzies
Lane Department of Computer Science and Electrical Engineering
West Virginia University
Morgantown, WV 26505, USA
ekocagun@mix.wvu.edu,
greg@greggay.com, tim@menzies.us

Ye Yang
Institute of Software
Chinese Academy of Sciences
Beijing, People's Republic of China
ye@itechs.iscas.ac.cn

ABSTRACT

Background: Due to the inherent problems regarding within-company data such as excessive time required to collect enough data, it is tempting to use cross-company data. However, reliability of effort estimates based on cross-company data is questionable.
Aim: We aim to identify under what circumstances individual organizations can make use of cross-company data using the method of relevancy filtering, which has previously been proven beneficial in the software defect prediction domain.
Method: We test the effects of relevancy filtering on three highly elaborated effort estimation datasets and their subsets. We exploit the essential assumptions of analogy-based effort estimation and use a relevancy filtering model to improve performance variance.
Results: We have analyzed *3 datasets * 3 subsets = 9 treatments* subject to statistical tests and found out that in 7 out of 9 treatments cross-company data performed as well as within-company data.
Conclusion: Essential analogy-based effort estimation techniques aided by relevancy filtering can allow organizations to utilize large repositories of cross-company effort datasets.

Categories and Subject Descriptors

D.2.9 [**Software Engineering**]: Effort Estimation—*Management, measurement*

Keywords

Effort Estimation, estimation models, cross-company data, within-company data, software engineering

1. INTRODUCTION

Effort estimation is one of the most important activities that a software development firm will undertake. Therefore, it is essential that such estimation be accurate. Serious consequences await projects with inaccurate estimates. The project's cost could eclipse its initial budget, or worse, may have to be canceled before it has been completed. Consider the case of NASA's Check-out Launch Control System, which was cancelled when the initial estimate of $200 million was overrun by an *additional* $200M [25]. This case is not unique, despite the significant effort put into designing more accurate estimation models, it has been reported that many predictions are wrong by a factor of four or more [7, 13].

We know from advice of Barry Boehm[1] that it is *always* better to collect historical local data. Generating accurate effort estimates requires the use of such detailed and accurate historical data. Such a repository of historical records must be large enough to generate smooth, non overfitted models, but it must also be regularly edited to remove projects based on obsolete technologies and methodologies that no longer fit with the company's standards. Constructing such a database involves some inevitable amount of effort. Project metrics must be tracked and regularly updated, along with the calculated effort totals, sometimes over the course of months or years. Such a task is difficult for an established company, and almost impossible for a new firm.

In such cases, it would be desirable to use effort estimation data from some other source, perhaps even from another company. Such databases exist; the PROMISE repository [8], for instance, offers twelve effort estimation data sets for public access. However, tempting as such data can be, cross-company data comes with its own set of issues [2, 16, 18]. Accurate effort estimation functions on a theory of *locality*. New projects that follow similar practices to historical projects should require a similar amount of effort. As Chen et. al. [5] have shown, inconsistencies in data collection across multiple firms create company-specific biases in cross-company data sets. Such biases result in an unacceptable amount of variance in the effort calculations. This same problem exists in the field of defect prediction. Cross-company data sources are riddled with inconsistencies, which leads to inaccurate predictions. However, multiple authors [4, 26] have shown that it is acceptable to use cross-comapny data sources, which have been preprocessed by some sort of *relevancy filtering*, either automated or expert-guided. Such filtering removes instances that create noise in the estimation process, leaving a body of data that, in theory, follows the principle of locality.

[1]Personal communication

We want to know if such relevancy filtering could be applied in an effort estimation context. Is it feasible to *collect* data from a variety of firms, *filter* it for heavily biased measurements, and then *apply* it to testing data from an unrelated company. Our research is guided by the following research questions:

RQ1 What is the effect of selecting particular cross-company projects on estimation performance?

RQ2 Is there any evidence that cross-company data can yield accuracy values as high as within-company data?

RQ3 Do the characteristics of a particular dataset have an inf uence on the within-company versus cross-company performance?

In this paper, we demonstrate a tree-based relevancy filter that prunes the instances responsible for a large amount of variance in prediction quality. Using three cross-company datasets, each split into three firms, we demonstrate that cross-company data processed by our relevancy filter leads to more accurate predictions in over 75% of experimental cases.

2. BACKGROUND

2.1 Experts vs. Models

Software effort estimation methods can be grouped under two categories [22]: Expert judgment and model-based techniques.

Expert judgment methods are one of the most widely used estimation methods [10]. Application of expert judgment may be either explicit (following a method like Delphi [6]) or implicit (informal discussions among such experts). One problem with expert-based estimates is that they may fall victim to competing interests in the sense that a faulty estimation of a senior expert may be taken over the more accurate estimation made by a junior expert within the same company. Another problem, indicated by Jorgensen et. al., is the poor capability of humans to improve their own expert judgment [11].

Model-based techniques are the methods generated by using algorithmic and parametric approaches or by induced prediction systems. The former approach is the adaptation of an expert-proposed model to local data. A well known example to such an approach is Boehm's COCOMO method [7]. The latter approach is useful in the case where local data does not conform to the specifications of the expert's method. A few examples of induced prediction systems are linear regression, neural nets, model trees and analogies [20, 23]. All of these systems are built on inherent assumptions. In the case where data violates such assumptions, patches are applied. An example of a patch is taking the logarithm of exponential distributions before linear regression [7, 14]. However, choosing the appropriate patch again requires qualified experts.

Different organizations may choose to use an expert judgment, a model-based approach, or some combination of the two in different settings. However, the goal of any estimation model is a common one - to attain high estimation accuracy. However, estimattion accuracy does not depend only on the choice of estimation model. Another critical factor is the posession of detailed and accurate historical data that has been carefully brought up to date.

2.2 Within-Company vs. Cross-Company Data

When a model-based technique is chosen for estimation, historical data from past projects is a neccessity. Organizations may choose to collect and maintain information concerning their own past projects in a dataset - this is refered to as a within-company dataset. Previous studies have suggested using such company-specific past data for accurate estimates [13, 15]. However, collection of within-company data comes with a cost. As experts on the field, Kitchenham and Mendes have reported three problems likely to occur when an organization wishes to rely on company-specific data [2, 16, 18]:

1 The time required to accumulate enough local data may be prohibitive.

2 By the time that the local dataset is large enough, the technologies employed by the company may have changed and old projects may have become obselete.

3 For collecting local data in a consistent manner, care is necessary.

On top of the before mentioned reasons, based on our personal experience, we can say that in the result-oriented environment of today's corporations, the time that must be spent collecting data (and the inevitable postponment of tangible results) is likely to decrease the enthusiasm of managers for estimation practices.

The problems regarding local data collection have motivated researchers to search for alternative solutions like the use of cross-company datasets (that is, datasets containing data from several different companies) [2, 18]. Although software effort estimation is a relatively young research field, the collective effort of many contributors have enabled the building of considerable repositories of software effort datasets. For example, in the PROMISE data repository alone[2], there are currently twelve effort prediction datasets that are available for public access [8]. However, the use of cross-company datasets is not a silver-bullet solution and comes with its own set of problems [2, 16, 18]:

1) It is even more difficult to ensure the consistency of data collection accross multiple organizations.

2) Process and practices differ accross companies, leading to differing trends in the data.

3) In the absense of a proper sampling strategy, it is difficult to make sure that randomly-selected cross-company projects truly form a random sample of the defined population.

[2]http://promisedata.org

Knowing the pros and cons of the within and cross-company approaches, we can take a look at their reflections on practice. In their extensive systematic review on the issue, Kitchenham and Mendes have made contraversial claims [3,16] concerning the results, as well as methodologies, of many studies adressing the use of cross-company data in software effort estimation. Among the ten studies under consideration in their 2006 systematic review, only seven of them showed independent evidence in their comparisons of accuracies between cross-company and within-company prediction models. Within those seven valid studies, three studies found that cross-company model performance was not significantly worse than within-company performance. The remaining four found that within-company models significantly outperformed cross-company models [3]. Therefore, the findings on cross-company vs. within-company usage are currently inconclusive.

3. MOTIVATION

Cross-company data use for prediction purposes is not limited to software effort estimation. The subject has also been addressed in the software defect prediction domain. Turhan et. al. questioned the applicability of cross-company data in defect prediction and found that cross-company data can be successfully used [26]. Cross-company data, taken *"as is"*, is only useful under relatively limited conditions [26]. However, cross-company data filtered for irrelevancies through the use of a nearest-neighbor technique has yielded surprisingly high accuracy results [26].

Zimmermann et. al. have addressed the same question at the individual project level and have come up with similar conclusions [4]. They have defined the cross-prediction models as a serious challenge, and they have also reported that using the cross-company data *as is* does not lead to useful predictions [4]. In order to increase these accuracy values, Zimmermann has proposed an approach that selects projects to be used for cross-company predictors [4]. This approach can be regarded as an expert-guided relevancy filtering. Taking these findings in defect prediction domain as an impetus, we would like to explore the applicability and performance of applying a *relevancy filtering* technique to cross-company data in the effort estimation domain.

4. APPROACH

4.1 ABE0

Analogy-based estimation (ABE), in the simplest terms, generates an estimate for a test project by gathering evidence from the effort values of similar projects in some training set. By analyzing the previous research of experts like Shepperd et. al. [24], Mendes et. al. [19] and Li et. al. [17] on the field of analogy-based estimation, we can come up with a baseline technique:

- Form a table whose rows are completed past projects (this is a training set).
- The columns of this set are composed of *independent* variables (the features that define projects) and a *dependent* variable (the recorded effort value).
- Decide on how many similar projects (*analogies*) to use from the training set when examining a new test instance , i.e. k-values.
- For each test instance, select those k analogies out of the training set.
  - While selecting analogies, use a similarity measure (such as the Euclidean distance of features).
  - Before calculating similarity, apply a scaling measure on independent features to equalize their influence on this similarity measure.
  - Use a feature weighting scheme to reduce the influence of less informative features.
- Use the effort values of the k nearest analogies to calculate an effort estimate.

We can refer to this baseline framework as ABE0. ABE0 uses the Euclidean distance as a similarity measure, whose formula is given in Equation 1.

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

We can see the weighting approach used for project features in Equation 1. In Equation 1, w_i corresponds to feature weights applied to independent features. For our case we do not favor any features over the others, therefore ABE0 uses a uniform weighting, i.e. $w_i = 1$.

The adaptation strategy for the effort estimate is not necessarily a complex process. ABE0 simply returns the median of the effort values of the k nearest analogies. The reason why ABE0 uses median instead of mean in our research is due to the fact that the number of instances that are selected for making the estimate after filtering are quite few in number (2 to 10 instances). In that case, use of mean may let extreme effort values have a very strong influence on the estimation. However, we want our estimates to represent the majority of selected instances and not greatly affected by extreme values, which may or may not be noise. Therefore, we use median instead of mean.

4.2 Relevancy Filtering

Using cross-company data in an *as is* fashion has been shown to have a negative effect on accuracy values [4]. However, the use of a relevancy filtering method has been shown to increase the accuracy of cross-company models in software defect prediction domain [4,26]. Although different researchers have used different relevancy filters (Turhan et. al. use a nearest neighbor filter, whereas the filtering employed by Zimmermann et. al is closer to guided expert judgment), it is obvious that filtering for relevant data is helpful when attempting to increase accuracy in cross-company models in the defect prediction domain.

We have described an analogy-based estimation baseline (ABE0) in Section 4.1. For our relevancy filtering, we will use a form of ABE0 in which the selection of analogies will be based on the calculated performance variance.

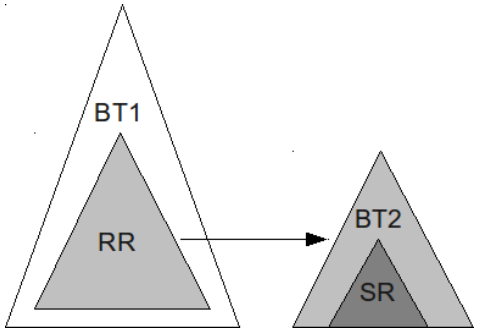

Figure 1: A representation of relevancy filtering. RR region contains instances selected by **R**andom **R**elevancy filtering in BT1 tree and SR region has the instances of **S**elective **R**elevancy filtering for a particular test instance in BT2.

In our relevancy filtering approach, the training projects are used to generate a binary tree. The leaves of this binary tree (level zero of the tree) are formed by the individual training projects, which are then greedily clustered in tuples to form the parent levels. This binary tree, which we will call BT1, is then traversed upwards from the root to height level one (one higher level than the leaves). The variance of the effort values in each sub-tree (the performance variance) is then recorded and normalized to a 0-1 interval. A random relevancy filtering is applied on these normalized variances as follows:

- Generate a random number from a normal distribution.
- Prune nodes with a normalized performance variance above the random value.

The remaining training instances in the pruined tree are used to build a second binary tree (called BT2). This second tree is then subject to a second relevancy filtering. BT2 is generated and traversed in the same fashion as BT1. This time, while traversing the tree, instead of storing the variances of sub-trees, we use the variance as a decision criterion. If the variance of the current tree is larger than its sub-trees, then continue to move down the subtree; otherwise, stop moving and select the instances in the current tree as the relevant instances and adapt them for estimation. Since the described model is a version of ABE0, the adaptation used for the selected relevant instances of BT2 is the same as ABE0, taking the median. A simple visualization of this relevancy filtering approach is given in in Figure 1.

This relevancy filtering method is very similar to the NN-filtering employed by Turhan et.al. [26], except that, for each test instance, the number of analogies to be selected (k-value) is determined online dependent on the characteristics of the current train set. Furthermore, unlike the models proposed in previous cross-company research, relevancy filtering on the basis of performance variance makes no preassumptions of the dataset and allows the test instances to adapt to the characteristics of the training set in order to prune irrelevancies.

5. METHODOLOGY

Previous studies on cross-company vs. within-company effort estimation were criticized for their lack of proper statistical analysis or due to the limited datasets used. Therefore, while conducting our research, we have paid particular attention to meeting the requirements of Kitchenham et al [16]. The details regarding dataset selection criteria as well as our statistical analysis methods will be provided in Sections 5.1 and 5.2 respectively.

The reproducibility of results is another critical issue in software engineering [12] and a valid criticism of much of the previous work in this field [3, 16]. As we want our results to be reproduced in both similar and different settings, we have chosen only publicly available datasets and we have provided as much details as possible regarding our experimental settings.

5.1 Data

In our research, we have used subsets of three commonly-used datasets in software effort estimation research: Nasa93, the original Cocomo81 [7], and Desharnais [9].

We will denote the subsets of Nasa93 as Nasa93c1, Nasa93c2 and Nasa93c5. Nasa93c1, Nasa93c2 and Nasa93c5 contain projects from different NASA development center around the United States (denoted as development centers 1, 2 and 5 in the complete dataset). In a similar fashion, subsets of Cocomo81 will be denoted as Coc81o, Coc81e and Coc81s and refer to:

- Coc81o contains "organic projects" that come from small teams with high experience working with less than rigid requirements.
- Coc81e contains "embedded projects" that are developed within tight constraints (hardware, software, operational etc.).
- The "semidetached projects" denoted by Coc81s are at an intermediate stage between the organic and embedded modes.

Lastly, the Desharnais dataset will be split into three different subsets based on the three different programming languages used for development projects. We will denote the subsets of Desharnais as DesL1, DesL2 and DesL3 (languages 1, 2 and 3 respectively). The subsets of each dataset are chosen because they form self-consistent and reliable data groupings. Since each of these subsets have certain common criteria (the development center, development mode, or development language), each subset will be treated as a seperate within-company dataset. The details such as feature and instance sizes as well as content of datasets are given in Figure 2. All of the datasets used in this research are available in PROMISE data repository [8].

Kitchenham and Mendes attribute a particular importance to dataset size in their review [16] and state that larger within-company datasets lead to more reliable comparisons between within-company and cross-company models. In order to evaluate the goodness of within-company datasets,

Dataset	Features	Projects	Content	Historical Effort Data Units
Cocomo81	17	63	NASA projects	months
Coc81o	17	24	Cocomo81 organic projects	months
Coc81e	17	28	Cocomo81 embedded projects	months
Coc81s	17	11	Cocomo81 semidetached projects	months
Nasa93	17	93	NASA projects	months
Nasa93c1	17	12	Nasa93 projects from center 1	months
Nasa93c2	17	37	Nasa93 projects from center 2	months
Nasa93c5	17	39	Nasa93 projects from center 5	months
Desharnais	12	81	Canadian software projects	hours
DesL1	11	46	Desharnais projects developed with language 1	hours
DesL2	11	25	Desharnais projects developed with language 2	hours
DesL3	11	10	Desharnais projects developed with language 3	hours
		Total: 469		

Figure 2: The 469 projects used in this study come from *3 datasets + 9 subsets = 12* data sets. Indentation in column one denotes that indented dataset is a subset of another dataset.

they propose a quality scoring of four values: poor (less than ten projects), fair (between ten to twenty projects), good (between twenty to fourty projects) and excellent (more than forty projects) [16]. According to quality criteria indicated by Kitchenham et. al. [16], the nine within-company datasets used in this study are grouped as follows: one of excellent quality, five of good quality, and three of fair quality.

5.2 Experiments

For each of the three main datasets (Nasa93, Cocomo81 and Desharnais) in our research, we have conducted within-company and cross-company experiments. The division of main datasets into its subsets is structured such that the subsets have a self consistent structure according to a dataset-specific characteristic. Therefore, each subset can be considered similar to a within-company dataset that contains projects sharing the particular characteristics of a single development firm.

To understand the within-company and cross-company data formation, assume that a dataset X with its three subsets X_1, X_2 and X_3 is under consideration. For within-company experiments, the relevancy filtering described in Section 4.2 is applied on each one of X_1, X_2 and X_3 seperately and the median of the filtered project instances in the training set is stored as the effort estimate for the test instance. For the separation of training and testing sets, the leave-one-out method is used. Leave-one-out selects one instance out of a dataset of n instances as the test instance and uses the remaining $n-1$ instances as the training set.

For the cross-company experiments, one of X_1, X_2 or X_3 is chosen as the test set and the combination of the remaining two forms the cross-company testing dataset. This time, the relevancy filtering is applied on the cross-company dataset, and the estimations for projects in the test set are stored. An illustration showing how the cross-company experiments are performed is given in Figure 3

Each of the within-company and cross-company experiments are repeated twenty times ir order to remove any bias that would otherwise be brought by a particular test and training set combination.

5.3 Performance Criteria

$win_i = 0$, $tie_i = 0$, $loss_i = 0$
$win_j = 0$, $tie_j = 0$, $loss_j = 0$
if MANN-WHITNEY($MRE's_i$, $MRE's_i$) says they are the same **then**
  $tie_i = tie_i + 1$;
else
  if median($MRE's_i$) < median($MRE's_j$) **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 4: Pseudocode for Win-Tie-Loss Calculation Between Method i and j

In order to compare the performance of within-company and cross-company datasets subject to our relevancy filtering approach, we have used two measures: the magnitude of relative error (MRE) and win-tie-loss values generated by a statistical rank-sum test. MRE is utilized by the authors because it is the most commonly used performance criterion for software effort estimation [1]. Furthermore, as we can see from Formula 2, MRE gives a per-instance based estimation performance evaluation.

$$MRE = \frac{|actual_i - predicted_i|}{actual_i} \quad (2)$$

However, MRE is subject to many pitfalls. When MRE is used as a stand-alone evaluation criterion (i.e. not combined with appropriate statistical tests), it may lead to biased or even false conclusions. To prevent us from falling into MRE-related pitfalls, we use another performance criterion called a win-tie-loss calculation. A win-tie-loss calculation states the fact that comparison between two methods i and j makes sense only if they are statistically significant. If they are statistically the same, that could indicate that they are observations coming from the same distribution, therefore they are noted as a tie and their *tie* values (tie_i and tie_j) are incremented. On the other hand, if there is a statistical difference between two methods, then the method with a lower median MRE score, say i, is identified as a "winner" and the one with the lower MRE , say j, is identified as a "loser." The

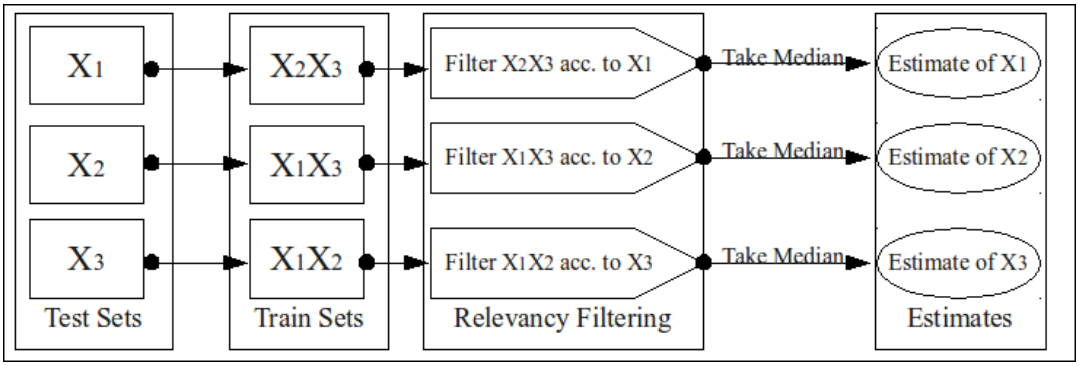

Figure 3: In cross-company experiments for subsets X_1, X_2 and X_3 we have 3 treatments. At each treatment, one subset is selected as test set and the remaining two become the cross-company data, i.e. train set. After filtering, median of the remaining instances from the train set become our estimate.

related values win_i and $loss_j$ are incremented by one. As we repeat each treatment twenty times, win-tie-loss calculation provides us with a good perspective on the success of each method across different datasets. The pseudocode for a win-tie-loss calculation is given in Figure 4. For the comparison of methods in win-tie-loss calculation, a non-parametric statistical test (the Mann-Whitney rank-sum test) is used at a significance level of 95%.

6. RESULTS

In our experiment, we analyzed *3 datasets * 3 subsets = 9 treatments*. For each treatment, we applied relevancy filtering on the training data, assessing on the basis of performance variance. We evaluated cross-company and within-company performances of each particular dataset, subject to statistical tests.

6.1 Preliminary Results

Based on the earlier results (as well as the related literature reviews [2–4,16,18,26]), we were quite aware of the fact that the usage of cross-company versus within-company data for effort estimation purposes is a contraversial issue. Therefore, before going into the details of our results, we took the time to observe the behaviour of cross-company projects from different datasets when subject to a relevancy filter.

In Figure 5, we can clearly see the effect of relevancy filtering for the Nasa93 dataset. The instances of the cross-company dataset that come from different centers are not favored over one another. On the contrary, some instances were commonly selected from every center that formed a cross-company treatment. In the second column of Figure 5, we can see how many instances were selected on average from each center in each cross-company treatment. While we could note that the relevancy filter chooses certain projects from every center, we were unable to observe any definitive patterns.

Note that in Figure 5, each row has one center whose number of selected instances is zero. This is due to the fact that one particular center is always kept as the seperate test set. Hence, no instances from that center can be selected for that row.

Cross-Companies	**Selected**		
	Nasa93c1	Nasa93c2	Nasa93c5
Nasa93c2 and Nasa93c5	0	2.35	1.17
Nasa93c1 and Nasa93c5	1.41	0	1.79
Nasa93c1 and Nasa93c2	2.36	1.08	0

Figure 5: After relevancy filtering on cross-company datasets, the distribution of *selected* instances do not come from a particular center of Nasa93.

Figure 6 shows the average number of projects selected from each subset in the cross-company datasets of Cocomo81. The same relevancy filtering effect that appeared in Nasa93 can also be observed on Cocomo81. As we can see in Figure 6, relevancy filtering selects instances from all of the subsets forming a particular cross-company treatment. For the last treatment (where Coc81o and Coc81e are the cross-company firms) in Figure 6, we see that the effect of relevancy filtering for selecting instances from each subset is quite limited. The average number of instances chosen from Coc81o is 2.75 whereas the same number for Coc81e is 0.25.

Cross-Companies	**Selected**		
	Coc81o	Coc81e	Coc81s
Coc81e and Coc81s	0	1.98	1.29
Coc81o and Coc81s	3.57	0	1.36
Coc81o and Coc81e	2.75	0.25	0

Figure 6: Effect of relevancy filtering on Cocomo81 cross-companies is similar to those of Nasa93. Instances are selected from all subsets.

Figure 7 shows the same effect for Desharnais that we observed for Nasa93 and Cocomo81 in Figures 5 and 6. That is, the selection of instances is never limited to a single subset of the cross-company treatment. However, if we look at row two (DesL1 and DesL3) of Figure 7, we can see that this effect is again extremely limited - an average of 2.05 instances are chosen from DesL1, while only 0.1 instances are chosen from DesL2. This same situation was encountered for one of the Cocomo81 treatments. Therefore, we would suggest that there is a hidden influence of certain data features on the selection of instances from cross-company datasets. The

task of discovering such features and defining their influences is a potential basis for future work.

Cross-Companies	Selected		
	DesL1	DesL2	DesL3
DesL2 and DesL3	0	2.27	0.65
DesL1 and DesL3	2.05	0	0.1
DesL1 and DesL2	3.17	1.59	0

Figure 7: Similar trend of relevancy filtering continues for Desharnais dataset as well. Selected instances come from all datasets of cross-company datasets.

As a result of our preliminary research, we have seen that relevancy filtering, in the majority of cases, does not favor any single subset of a cross-company dataset. On the contrary, the instances selected as relevant (i.e. the instances in the SR tree of Figure 1) come from all of the "companies" that compose a single cross-company dataset. A summary of our preliminary results is given in Figure 8. We can observe from this figure that, for each cross-company dataset, there are instances from every subset being selected by the relevancy filter. From this, it can be deduced that - although the definition of within-company data is based on single dimensions such as geographical proximity of development teams (centers 1, 2 and 5 of Nasa93), mode of development (organic, embedded and semi-detached modes of Cocomo81) or particular languages used to develop projects (languages 1, 2 and 3 of Desharnais) - it is actually the individual independent variables that have an influence on the selection of projects as relevant to a particular test instance.

6.2 Performance Results

These initial results motivated us to go further with our studies and discover how the observed filtering effect reflected on the performance of cross-company datasets.

In Figure 9, we list the win-tie-loss values for the subsets of Nasa93 subject to three treatments. Because we compare cross-company and within-company performance in twenty randomized assessments for each treatment, the sum of *win*, *tie* and *loss* values at each row is twenty. This table shows us that, in all three treatments, the *tie* values are quite high. The *tie* values between within-company and cross-company datasets based on Nasa93 are 19, 17 and 15. This indicates that, for at least 75% of the tests, there is no statistical difference between filtered cross-company and within-company results. Even for the treatment with the lowest *tie* value of 15, within-company has a *win* value of 3 and cross-company has a *win* value of 2 Therefore, for Nasa93, the performance of cross-company data (filtered for relevancy) is indistinguishable from the performance of within-company data.

Figure 10 shows the win-tie-loss values for the subsets of Cocomo81. In two out of the three treatments the *tie* values are 19, which tells us that for these treatments, within-company and cross-company performance are almost identical. However, the last treatment shows a preference for within-company data on thirteen of the twenty tests. If we take a look at Figure 6 from our preliminary results, we see that the Coc81s subset showed a selection bias towards a certain domain. That is, it selected for an average of 2.75 instances from Coc81o and only 0.25 from Coc81e.

Dataset	Method	Win	Tie	Loss
Nasa93c1	WC-Data	1	19	0
Nasa93c2 and Nasa93c5	CC-Data	0	19	1
Nasa93c2	WC-Data	3	17	0
Nasa93c1 and Nasa93c5	CC-Data	0	17	3
Nasa93c5	WC-Data	3	15	2
Nasa93c1 and Nasa93c2	CC-Data	2	15	3

Figure 9: MRE win-tie-loss values for Nasa93 from 20 *randomized assessments*. In all treatments *tie* values are quite high. For Nasa93, the performance of cross-company data (CC-Data) is mostly same as within-company data (WC-Data).

Dataset	Method	Win	Tie	Loss
Coc81o	WC-Data	0	20	0
Coc81e and Coc81s	CC-Data	0	20	0
Coc81e	WC-Data	1	19	0
Coc81o and Coc81s	CC-Data	0	19	1
Coc81s	WC-Data	13	7	0
Coc81o and Coc81e	CC-Data	0	7	13

Figure 10: MRE win-tie-loss values for Cocomo81 from 20 *randomized assessments*. In 2 treatments CC-Data is same as WC-Data. However, in the case of Coc81s WC-Data outperforms CC-Data.

The win-tie-loss values for subsets of Desharnais are given in Figure 11. The derived results for the Desharnais subsets are similar to those of the Cocomo81 treatments Two out of the three treatments show identical *tie* values of 19, which again suggests that the performance of filtered cross-company datasets statistically identical to within-company datasets. However, in one of the treatments, within-company outperforms cross-company on sixteen of the twenty trials.

Dataset	Method	Win	Tie	Loss
DesL1	WC-Data	16	4	0
DesL2 and DesL3	CC-Data	0	4	16
DesL2	WC-Data	1	19	0
DesL1 and DesL3	CC-Data	0	19	1
DesL3	WC-Data	1	19	0
DesL1 and DesL2	CC-Data	0	19	1

Figure 11: MRE win-tie-loss values for Desharnais from 20 *randomized assessments*. In the case of DesL1 WC-Data is much better than CC-Data. For other treatments, WC and CC-Data are statistically the same.

The summary of our results in terms of win-tie-loss values are given in Figure 12. When we look at the win-tie-loss values, the first noticable fact is the high number of *tie* values between cross-company and within-company models for each treatment. In seven out of the nine treatments (which is more than 75% of all treatments), the *tie* values are equal to or greater than fifteen. Another interpretation of the high *tie* values is that, in more than 75% of cases, there is no statistical difference between filtered

Cross-Company Dataset	Test Set	Instance Size in SR Tree		
		From Nasa93c1	From Nasa93c1	From Nasa93c1
Nasa93c2 and Nasa93c5	Nasa93c1	0	2.35	1.17
Nasa93c1 and Nasa93c5	Nasa93c2	1.41	0	1.79
Nasa93c1 and Nasa93c2	Nasa93c5	2.36	1.08	0
		From Coc81o	From Coc81e	From Coc81s
Coc81e and Coc81s	Coc81o	0	1.98	1.29
Coc81o and Coc81s	Coc81e	3.57	0	1.36
Coc81o and Coc81e	Coc81s	2.75	0.25	0
		From DesL1	From DesL2	From DesL3
DesL2 and DesL3	DesL1	0	2.27	0.65
DesL1 and DesL3	DesL2	2.05	0	0.1
DesL1 and DesL2	DesL3	3.17	1.59	0

Figure 8: The instance sizes in SR tree are mean of remaining instances after filtering in 20 runs. Cross-company datasets are combination of two within-company datasets tested on another within-company dataset. We see that relevancy filtering selects instances from both centers forming a cross-company dataset.

cross-company and within-company datasets. That is, cross-company datasets have performed just as well as within-company datasets. There are only two treatments, DesL1 and Coc81s, where within-company performance was significantly better than cross-company performance. A possible explanation for those two scenarios may be hidden in the dataset size or in the quality of the within-company datasets, but the currently-available information makes it is difficult to suggest any conclusive reason for the situation.

6.3 Answers to Research Questions

With these results, we can now address the questions that we proposed to guide this research.

RQ1) What is the effect of selecting particular cross-company projects on the estimation performance?

We have seen that, in more than 75% of tests, cross-company datasets filtered for relevancy have performed as well as within-company datasets. Therefore, we can state that selecting particular cross-company projects, guided by a relevancy filter, has a positive effect on the estimation performance of analogy-based estimation.

RQ2) Is there evidence that cross-company data can yield accuracy values equal to that of within-company data?

For seven out of the nine treatments used in our experiments, filtered cross-company data has performed equally to within-company data. Our experiments have yielded statistical evidence that cross-company data can be used to obtain high accuracy values. However, for cross-company data to attain such values, we cannot use the data *"as is"*. Rather, we must use a relevancy filtering method to remove the instances that cause a high performance variance.

RQ3) Do the characteristics of particular datasets have an influence on the within-company versus cross-company performance?

In two treatments, we have observed that within-company datasets significanlty outperformed cross-company datasets. Therefore, we cannot say that the raw characteristics of particular datasets have no influence on effort estimation performance. However, the exact reason behind this is not yet obvious. In our case, the two treatments that yielded higher within-company performance were datasets with 46 and 11 projects respectively. Based on dataset size alone, it is difficult to observe any connection between number of projects and estimation performance. One potential reason could be the data quality of each set. However, we do not yet have any evidence to back such claims.

7. THREATS TO VALIDITY

We will address the threats to the validity of our results under two categories - the internal and external validity.

Internal validity asks to what degree the cause-effect relationship between dependent and independent variables holds [21]. The ideal case for satisfying internal validity would be to learn a theory from a past situation and then to apply the theory to a new setting. However, general software effort studies make use of commonly-explored past datasets such as the ones used in this research (Nasa93, Cocomo81, and Desharnais). This issue of internal validity threatend all effort studies using only past data. We can mitigate this problem by simulating the behaviour of a learned theory in new settings. In our study, we make use of the leave-one-out method for all treatments to address such internal validity issues. Leave-one-out selection enables us to separate the training and test sets completely in each experiment.

External validity refers to the ability to generalize our results. A potential threat to the external validity of our research is that we use subsets of three existing datasets for our within-company data. This approach may seem non-traditional when compared to previous cross-company studies. However, each dataset used in our study is actually composed of projects coming from different vendors with various different properties. That is, these datasets actually do represent cross-company data. Therefore, when splitting these datasets into subsets, we chose dimensions that would both validate the subsets as seperate companies while still producing high quality datasets (according to definition set forth by Kitchenham et. al. [16]). For Nasa93, this dimension was a physical seperation (development centers). For Cocomo81, it was the organization method of teams (also

Data set	Train Set	Test Set	Method	Win	Tie	Loss
Nasa93	Nasa93c1	Leave-one-out test instance	Within-Company	1	19	0
	Nasa93c2 and Nasa93c5	Nasa93c1	Cross-Company	0	19	1
	Nasa93c2	Leave-one-out test instance	Within-Company	3	17	0
	Nasa93c1 and Nasa93c5	Nasa93c2	Cross-Company	0	17	3
	Nasa93c5	Leave-one-out test instance	Within-Company	3	15	2
	Nasa93c1 and Nasa93c2	Nasa93c5	Cross-Company	2	15	3
Cocomo81	Coc81o	Leave-one-out test instance	Within-Company	0	20	0
	Coc81e and Coc81s	Coc81o	Cross-Company	0	20	0
	Coc81e	Leave-one-out test instance	Within-Company	1	19	0
	Coc81o and Coc81s	Coc81e	Cross-Company	0	19	1
	Coc81s	Leave-one-out test instance	Within-Company	13	7	0
	Coc81o and Coc81e	Coc81s	Cross-Company	0	7	13
Desharnais	DesL1	Leave-one-out test instance	Within-Company	16	4	0
	DesL2 and DesL3	DesL1	Cross-Company	0	4	16
	DesL2	Leave-one-out test instance	Within-Company	1	19	0
	DesL1 and DesL3	DesL2	Cross-Company	0	19	1
	DesL3	Leave-one-out test instance	Within-Company	1	19	0
	DesL1 and DesL2	DesL3	Cross-Company	0	19	1

Figure 12: MRE win-loss-tie results from 20 *randomized assessments*. Comparisons are between within-company and cross-company performance of each subset of every dataset.

called the development mode). For Desharnais, the splitting criteria was the language used to develop a project.

8. CONCLUSIONS AND FUTURE WORK

In our research, we have explroed the use of cross-company data in the software effort estimation domain. We have seen that, in the past, using cross-company data in an *"as is"* manner has lead to poor estimation accuracy. However, research in the defect prediction field has hinted that subjecting cross-company data to a process that filters for relevancy could improve the estimates made. To test this theory, we used a filtering method that builds binary trees and prunes these trees using the performance variance along the branches. For our estimation model, we have used a simple version of analogy-based estimation.

In a series of experiments, we discovered statistical evidence to support our theories. In more than 75% of cases, we have observed that the filtered cross-company data performance was statistically identical to that of within-company data. Based on such evidence, we claim that the positive effect of relevancy filtering previously reported in the defect prediction field [4, 26] also apply to software effort estimation.

Our research has shown that filtered cross-company treatments can attain estimation accuracies just as high as those of within-company data. At the same time, equal results are not the same as better. Boehm's advice holds - if possible, a firm should use their own locally-collected data. However, if a company is new or has eschewed data collection in the past, such data may not be available. Based on our experimental conclusions, we would suggest that any organization that wishes to utilize effort estimation, but lacks the resources or experience, could make use of publically-available cross-company data as long as they apply some sort of relevancy filtering method to tune the data to local practices. Filtered cross-company models can be used to provide accurate estimates until a firm has collected enough historical records to switch to a localized data source.

Going forward, we would like to learn exactly why an instance is deemed "relevant" or "irrelevant" by the filter for a particular treatment. In other words, we would like to know exactly which features are most influential when assigning relevancy. The ability to identify these exact dimensions would make the selection of appropriate projects easier for any institution that utilizes cross-company data and would lead to more accurate filtering techniques.

9. ACKNOWLEDGEMENTS

This work was conducted under contract with the Institute of Software, Chinese Academy of Sciences.

10. REFERENCES

[1] *A Survey on Software Estimation in the Norwegian Industry*, Washington, DC, USA, 2004. IEEE Computer Society.

[2] *A Replicated Comparison of Cross-Company and Within-Company Effort Estimation Models Using the ISBSG Database*, Washington, DC, USA, 2005. IEEE Computer Society.

[3] *A systematic review of cross- vs. within-company cost estimation studies*, 2006.

[4] *Cross-project defect prediction: a large scale experiment on data vs. domain vs. process*, New York, NY, USA, 2009. ACM.

[5] *"Reducing the Local Bias in Calibrating the General COCOMO"*, 2009.

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

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

[8] G. Boetticher, T. Menzies, and T. Ostrand. PROMISE repository of empirical software engineering data, 2007.

[9] J. Desharnais. Analyse statistique de la productivitie des projets informatique a partie de la technique des point des fonction. Master's thesis, Univ. of Montreal, 1989.

[10] M. Jø rgensen. A review of studies on expert estimation of software development effort. *Journal of Systems and Software*, 70:37–60, February 2004.

[11] 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, May-June 2009.

[12] 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.

[13] C. Kemerer. An empirical validation of software cost estimation models. *Communications of the ACM*, 30(5):416–429, May 1987.

[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 and N. Taylor. Software cost models. *ICL Technical J.*, pages 73–102, 1984.

[16] B. A. 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] 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*, 82:241–252, 2009.

[18] E. Mendes and B. Kitchenham. Further comparison of cross-company and within-company effort estimation models for web applications. pages 348–357, 2004.

[19] E. Mendes, I. D. 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.

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

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

[22] M. Shepperd. Software project economics: a roadmap. In *FOSE '07: 2007 Future of Software Engineering*, pages 304–315, 2007.

[23] M. Shepperd and G. Kadoda. Comparing software prediction models using simulation. *IEEE Transactions on Software Engineering*, pages 1014–1022, 2001.

[24] M. Shepperd, C. Schofield, and B. Kitchenham. Effort estimation using analogy. In *ICSE '96: Proceedings of the 18th international conference on Software engineering*, pages 170–178, Washington, DC, USA, 1996. IEEE Computer Society.

[25] Spareref.com. Nasa to shut down checkout & launch control system, August 26, 2002. http://www.spaceref.com/news/viewnews.html?id=475.

[26] 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*, 14:540 – 578, 2009.