CHALMERS

Unsupervised Outlier Detection in Software Engineering

Master of Science Thesis in Software Engineering

HENRIK LARSSON
ERIK LINDQVIST

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
Göteborg, Sweden June 2014

Unsupervised Outlier Detection in Software Engineering

Henrik Larsson Erik Lindqvist

2014

Unsupervised Outlier Detection in Software Engineering
HENRIK LARSSON
ERIK LINDQVIST

Examiner: Matthias Tichy
Supervisor: Richard Torkar

Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone +46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden June 2014

Abstract

The increasing complexity of software systems has lead to increased demands on the tools and methods used when developing software systems. To determine if a tool or method is more efficient or accurate than others empirical studies are used. The data used in empirical studies might be affected by outliers i.e. data points that deviates significantly from the rest of the data set. Hence, the statistical analysis might be distorted by these outliers as well.

This study investigates if outliers are present within Empirical Software Engineering (ESE) studies using unsupervised methods for detection. It also tries to assess if the statistical analyses performed in ESE studies are affected by outliers by removing them and performing a re-analysis. The subjects used in this study comes from a narrow literature review of recently published papers within Software Engineering (SE). While collecting the samples needed for this study the current state of practise regarding data availability and analysis reproducibility is investigated.

This study's results shows that outliers can be found in ESE studies and it also identifies issues regarding data availability within the same field. Finally, this study presents guidelines for how to improve the way outlier detection is presented within ESE studies as well as guidelines for publishing data.

Contents

1 Introduction

Software is increasingly important for both national and international infrastructure. Hence, it is increasingly important to produce software more cost-efficiently (Sommerville, 2006). This has led to an increased interest, the last 30 years, in Software Engineering (SE). The field of SE do not just covers the technical aspects of creating software, it also attends to the aspects of managing software projects. Empirical studies plays an important role in order to study the effects of developed methods and tools within SE. These empirical studies are considered to be an accepted discipline within SE (Ko et al., 2013).

Data collected from empirical studies are used to draw conclusions regarding the study. However, this data can contain outliers, values that deviates significantly from the rest, which may or may not impact the analysis of the study. It is therefore important to understand the impact of outliers in order to interpret the results correctly and to draw valid conclusions (Kriegel et al., 2008). Additionally, the task of identifying and removing outliers needs to be documented for the study to be more easily replicated. This is of importance since replication experiments, which confirms the findings of a study, helps building confidence in the result and procedures. Therefore, replication is considered as one of the cornerstones in a scientific community and an indicator of how mature a scientific discipline is (Brooks et al., 2008).

In this study the presence of outliers will be investigated, within the field of Empirical Software Engineering (ESE). Furthermore, the effect on the conclusions drawn in ESE studies, when potential outliers are removed, will be investigated to determine the applicability of outlier detection within ESE. Additionally, to which extent the presence of outliers is documented and outlier detection is conducted will also be investigated. Ultimately, this study aims at providing guidelines regarding if/how outlier detection should be conducted and presented in empirical studies.

This report is divided as follows: Section 2 presents background information regarding outliers and replication as well as related work within these areas. In Section 3 the methodology used for the pre-study is presented. Section 4 presents the candidate papers and algorithms from the pre-study and is followed by Section 5 where the results from the application of algorithms (presence of outliers) as well as a deeper analysis of a few selected papers is presented. The report is concluded with a discussion in Section 6, threats to validity in Section 7 and finally conclusions in Section 8.

1.1 Problem and Purpose

As empirical studies within SE grows more complex, with the maturity of the field, the quantities of data that is handled grow as well. In order to be able to replicate these studies with large data sets it is of importance that all steps are well documented, including the data post-processing such as outlier detection. Furthermore, performing data post-processing, such as outlier detection, manually on a large data set can be a tedious and error-prone task, which may

complicate replication of the study. Therefore, unsupervised and 'automatic' methods are of interest.

The purpose of this study is to investigate to what extent outlier detection is conducted and documented within the field of SE. Furthermore, outlier detection using unsupervised and 'automatic' methods will be conducted on data sets from a selected collection of papers in order to investigate the presence of outliers. Additionally, a verification will be carried out, on a subset of papers containing outliers, in order to verify that their conclusions hold in the absence of outliers. Finally, guidelines will be proposed for how to present the data analysis performed with regards to outlier detection and removal. These guidelines are meant to support researchers in presenting their studies within SE so that the studies can be more easily replicated.

1.2 Hypotheses and Research Questions

Outliers can cause a large impact on the mean value of a data set. Hence, our first hypothesis is related to the occurrence of outliers within ESE studies.

> $\mathbf{H_{0}}_{\text{Outliers}}$ Data from software engineering studies do not contain outliers.
> $\mathbf{H_{1}}_{\text{Outliers}}$ Data from software engineering studies contain outliers.

If $\mathbf{H_{0}}_{\text{Outliers}}$ can be rejected it is of interest to investigate if removing the outliers could lead to different conclusions than those drawn in the original study.

> $\mathbf{H_{0}}_{\text{Concl}}$ Removing outliers from software engineering studies does not lead to different conclusions than the conclusions drawn in the original study.
> $\mathbf{H_{1}}_{\text{Concl}}$ Removing outliers from software engineering studies leads to different conclusions than the conclusions drawn in the original study.

The documentation of the presence outliers and outlier detection in SE studies are of importance for the study's ability to be replicated. If data points are excluded for the sake of being outliers it is especially important to explain why, so others can reproduce this data post-processing step.

> **RQ1:** To what extent is the presence of outliers and outlier detection documented in software engineering studies?

Having access to the original data of a study is of great help when replicating the study. Therefore, Research Question 2 aims at describing the current state of practice when it comes to data availability in SE studies.

> **RQ2:** To what extent is data available in research papers and in which form is the data made available?

The connection between the two hypotheses and the focus of the two research questions are visualized in Figure 1.

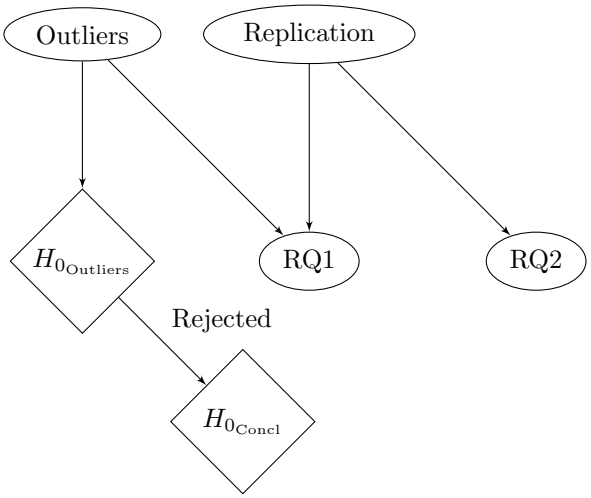


Figure 1: The relationship between hypotheses and research questions.

2 Related Work and Theoretical Background

In this section two areas of importance for this study, outliers and replication will be presented. In Section 2.1 outliers will be defined, look at issues regarding outliers and finally present related work regarding this area. Section 2.2 aims at presenting replication as a concept and its importance as well as some related work regarding replication within SE.

2.1 Outliers

Outliers are data points that differ significantly from other data points in a data set. A commonly used definition for outliers is "an outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs" (Osborne and Overbay, 2004). It is also stated by Osborne and Overbay (2004) that the effect of outliers can impair the statistical analysis. Therefore, determining if the collected data contains any data values that should be regarded as outliers is of importance.

There are several approaches for classifying outlier detection methods, three of them are mentioned by Hodge and Austin (2004) as unsupervised, supervised and semi-supervised detection techniques. Unsupervised detection determines if a data point is an outlier with no prior knowledge of the data set and flags the data points most separated from the normal data as outliers. Supervised detection use data that is pre-labeled as normal or not normal in order to determine if the other data points are outliers or not. Finally, semi-supervised detection uses a small set of training data to detect outliers (Hodge and Austin, 2004).

Rousseeuw and Van Zomeren (1990) state that in data sets with more than two dimensions it can be difficult to detect outliers because visual inspection is not as reliable. Therefore, it is important to use other methods than visual inspection to detect outliers in multi-dimensional data. However, some outlier detection techniques require information about the distribution of the data set. Determining this distribution for data sets with more than two dimensions can be challenging. Hence, if one is not certain of the distribution, choosing a method for outlier detection that does not make assumptions about the distribution is prefered (whether having a large, multidimensional data set, or not).

Seo and Bae (2013) studied the effect of outliers in software effort estimation. This was done by investigating the effect outliers have on the estimation accuracy of commonly used software estimation methods. The methods are evaluated on industrial data collected from publicly available repositories such as PROMISE and ISBSG. A Wilcoxon ranked sum test was used to see if removing outliers made a significant difference. Seo and Bae (2013) reported that there was a positive effect in estimation accuracy when removing outliers but not enough to say that it is significantly better. The study by Seo and Bae (2013) differs from this study by being focused on effort estimation and using publicly available data sets only. Furthermore, this study relies on data retrieved from recently published papers within SE and takes a more automated

approach to outlier detection. The focus in this study is on describing the state of practice when it comes to data availability and how research data is made available (**RQ2**).

Yuan and Bentler (2001) evaluates how outliers distorts the results in covariance structure analysis and the quantitative effect of outliers on statistical tests. Covariance tests are used to determine how different variables are affected of each other. Since covariance tests are used within SE, see e.g. Card et al. (1987), it may be of importance for the SE community to understand what impact outliers have on the end result. Yuan and Bentler (2001) reports that effects of a few outliers can discredit the value of using a model. They also report that outliers does not need to be very extreme to break down the covariance analysis.

2.2 Replication

As empirical studies have become more common within SE the importance of being able to replicate studies increases. Such replications are important since they help increase the body of knowledge around SE, which in turn leads to an increased maturity of the field. A replication of a study also comes with benefits for the original study in terms of increased confidence for the conducted experiment and the reported findings (e.g. tightened confidence intervals). Furthermore, the success of a conducted replication is not mainly depending on how well the replicated result conforms to the original but on the contribution to the body of knowledge (Basili et al., 1999; Shull et al., 2008; Brooks et al., 2008).

There are in general two forms of replication, internal and external. Internal replication is carried out by the original researcher or team while external replication is carried out by someone else than the original author (Brooks et al., 2008). Furthermore, the degree to which a replication is carried out can be divided into exact and conceptual replication. An exact replication is a replication which follows the original procedure as closely as possible, whereas a conceptual replication is a replication where the same hypothesis is validated through a different procedure (Shull et al., 2008). However, Juristo and Vegas (2011) states that exact replication within SE is close to non-existent due to the difficulties in recreating the exact conditions from the original experiment. Furthermore, Juristo and Vegas (2011) also propose that promoting non-exact experiments could encourage more researchers to perform replication experiments.

Sjøberg et al. (2005) found in their survey that only 18% of the surveyed papers from SE were replications. This was a surprising finding considering that replication is seen as important in science (Lindsay and Ehrenberg, 1993). The reason for this lack of replication studies is interesting and Lindsay and Ehrenberg (1993) propose that it might be due to that replicated experiments do not reward the researcher as much as an original experiment.

The previously referenced papers regarding replication within SE are focusing mostly around replicating the experiment and do not regard data analysis to a great extent. Though, within the field of bioinformatics, where large data sets are common, more work has been carried out in order to promote a more reproducible way of presenting the analysis for a study Tan et al. (2010); Gentleman (2004); Preeyanon and Brown (2014). The main purpose of these proposals is to make the analysis clear to the reader and reproducing a study's report is

merely executing an accompanying script in the report's repository. This allows the reader to easily reproduce artifacts such as plots and tables, and the reader can even perform changes to the analysis and study the result of them *in vivo*. If it is assumed that the amount of data used by empirical studies within SE is increasing, the need for standardized tools for handling data, such as those promoted for bioinformatics, increases as well.

Although it might sound simple in theory to ensure that a study is reproducible, the previously mentioned low outcome of replications could be an indicator that it is harder in practice.

In this study 'reproducibility' will be referred to as the ability to replicate a study's 'result' based on the information stated in the original study. The 'result' can refer to intermediate results as well as the final result of a study. In this study the focus is on reproducibility of the analysis, known as "re-analysis" (Gómez et al., 2010) i.e. verifying the results of the analysis based on data from the original study.

3 Methodology

By using a method for conducting a study the steps taken can be communicated more clearly. Furthermore, by using known methods for a study the credibility of it increases since it will be easier for the reviewer to understand the steps conducted. Using known methods also has the benefit of not needing to define each step but only motivate why this method is suitable. Lastly, by using a method for a study, compared to doing it *ad hoc*, it is more likely that the study is reproducible since, as stated before, the understandability is likely to be higher.

There are multiple methods to choose from when conducting research within SE. Two common ways of conducting research are experiments and case studies. Experiments can be conducted both within a controlled environment such as a laboratory setting or a real life setting. Common for both settings are that some of the variables can be controlled and changed in order to observe effects and compare outcomes of the experiment (Wohlin et al., 2012). Case studies are well suited for investigating a single entity or comparing two different methods for conducting a task within a certain context. They are also suitable for industrial evaluation of SE methods. However, case studies have the disadvantage of being difficult to generalize as a representative for the population (Runeson and Höst, 2009). This study can be seen as a case study in a broader context. In this case the context is SE and the phenomenon under investigation is outlier detection.

The method used in this study consisted of a *pre-study*, the *application of algorithms* and a *replication of analysis* for selected papers.

3.1 Pre-Study

The pre-study consisted of two different parts, the search for papers and the search for algorithms. The result from both of these tasks were then used for the application of algorithms where the algorithms were applied on the papers in order to be able to answer the hypotheses. A description of how the search was conducted for papers is presented in Section 3.1.1 and for the search for algorithms in Section 3.1.2.

3.1.1 Search Process for Papers

The process of searching and reviewing papers was carried out in a systematic way albeit not strictly following the guidelines for conducting systematic literature reviews in SE. Selection and quality assessment criteria were defined and used to determine which papers to include for the study. The process was meant to provide a sample of published papers to give an idea of the current state of the research field. Therefore, an exhaustive review over all papers published within the field was not carried out. Due to this limitation, this review cannot be considered a "systematic literature review" as defined in (Kitchenham and Charters, 2007). The methodology used for the review is further elaborated in the following paragraphs.

Papers used in this study were selected from the research field SE. To refine the scoop only papers from recognizable sources within SE were considered.

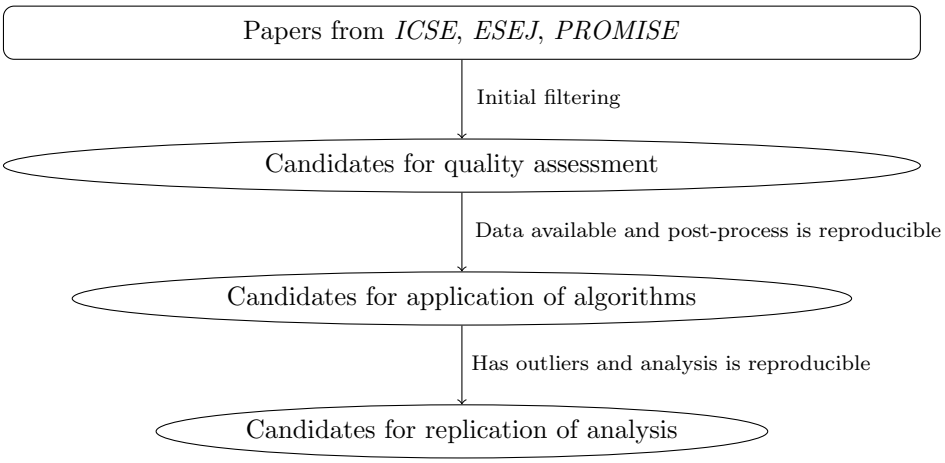


Figure 2: The figure describes how the paper filtering process was implemented. The filtering done in "Data available and post-process reproducible" refers to the *pre-study* while the filtering carried out in "Has outliers and analysis is reproducible" refers to the *replication of analysis*.

More specifically the sources were the *Empirical Software Engineering Journal (ESEJ)*, *International Conference on Software Engineering (ICSE)* and *International Conference on Predictive Models in Software Engineering (PROMISE)*. ESEJ was chosen since it is a journal featuring articles on empirical research within SE. Proceedings from ICSE was chosen since the conference is considered to be among the leading conferences within SE. The PROMISE conference proceedings include empirical research and associated to this is an online repository[1] containing paper data. Therefore, PROMISE was deemed as suitable source for elicitation. Furthermore, only papers from 2013–2014 were considered, in order to get a current sample of the field, i.e. an exhaustive search was not conducted. As a final filter only papers regarding empirical studies and numerical data were chosen. The papers that passed the mentioned criteria were then considered for this study.

After the previously mentioned initial filtering a selection based on quality attributes was carried out. A selection process containing two steps was created to assess data availability and data post-processing reproducibility.

In the first step a paper's data availability was assessed. If the paper's data was published in the paper, or accessible online, or provided directly by the authors it passed the first step. The purpose of this step was to filter out papers where the raw data set is not accessible.

In the second step the possibility of being able to replicate exactly any processing done on the raw data set was assessed. If a paper's post-processing was deemed reproducible, or if none was conducted, the paper was kept and outlier detection algorithms were applied on its data set. Being able to reproduce the post-processing is important if further analysis of the paper's data set is to be possible. The filtering process is visualized in Figure 2.

The data needed to answer **RQ1** ("To what extent is the presence of out-

[1] `https://code.google.com/p/promisedata/`

liers and outlier detection documented in software engineering studies?") was gathered simultaneously as the paper search was conducted. In order to answer the research question the extent to which the presence of outliers and outlier detection is documented in the studies selected, from the previous filtering and selection, was investigated.

To answer **RQ2** ("To what extent is data available in research papers and in which form is the data made available?") it was investigated how the data used in the reviewed papers are made available to the public. This was carried out by trying to access the raw data from the papers selected previously. As the first step, a check was made to see if the data was available in the paper. If no data was found in the paper a search was made for references to webpages or online repositories in the paper to see if the data was made available online. The third, and last, step was contacting the lead author via email and asking for the data. The email sent was a short email acknowledging the author's paper and asking for access to the study's data without elaborating the intentions further. However, if an author asked about the intentions a description of this study was given. By sending a sparse email the willingness of the author to give access to the study's data was tested. If a reply did not come within four weeks the data was regarded as not being available.

All papers assessed during the pre-study can be found in Appendix B together with the assessment of the different attributes.

3.1.2 Search Process for Algorithms

The search for suitable candidate algorithms was carried out with the aim of finding algorithms that can be used in an automated environment. That is, algorithms that can be used in, for example, a script that takes a data set as input. In order to conduct this search three criteria were defined that an algorithm must fulfill in order to be considered for this study:

- The algorithm has to be unsupervised. The reason for this being that unsupervised algorithms requires no training data and with this requirement the effort and information needed to use the algorithm should be lower, compared to a supervised algorithm.
- The algorithm does not make any assumptions about the distribution of the data set, also known as being robust. This criterion was important due to the difficulties of determining the distribution using an automated approach, and due to the fact that SE research many times contain data sets of various (small) sizes leading to the conclusion that a non-parametric approach is more applicable, generally speaking.
- The algorithm must not require any input parameters, except for the data set destined to analyze, as algorithms not requiring parameters have the benefit of being suitable for use in an automated environment.

3.2 Application of Algorithms

The goal of this step was to run the candidate algorithms on the data sets from the pre-study. In order to achieve this an automated process was created, referred to as the pipeline. This pipeline takes a data set as input and applies

a suitable outlier detection algorithm on the set depending on if the data set is one-dimensional or multi-dimensional.

For each one-dimensional data set that was processed, the following information was produced:

- The original data set with the identified outliers labelled.
- Descriptive statistics for original and modified data sets:
  - Mean
  - Median
  - Standard deviation
  - Number of outliers
- Density and QQ-plots[2] for the original data set and the modified data set.
- Plot with outliers marked and the change of mean if they are removed.
- Output of a Shapiro-Wilks normality test for both the original and modified data set.
- Output of a Welch's t-test. Significance testing for difference between the original and modified data set.
- Output of a Mann-Whitney U test. Significance testing for difference between the original and modified data set.

In order to make this study reproducible all results are automatically generated and can be recreated at any time by downloading the study's resources[3] and executing the analysis script as described in the accompanying documentation. This script then inputs all data sets into the pipeline and the results can then be viewed in the result folder. Software requirements for using the pipeline can be found in Appendix A.

3.2.1 Analyzing the Results

The analysis of the results consisted of two parts. In the first part, the presence of outliers was determined in order to see if $\mathbf{H_{0_{\text{Outliers}}}}$ could be rejected. While in the second part it was investigated if the original data set without outliers differed significantly from the original data set. This comparison was conducted in order to get a general overview of the impact of outlier removal on the data set.

The criterion for rejecting $\mathbf{H_{0_{\text{Outliers}}}}$ was that at least one outlier was found in any data set investigated after the application of algorithms was performed. If $\mathbf{H_{0_{\text{Outliers}}}}$ could be rejected $\mathbf{H_{0_{\text{Concl}}}}$ was investigated.

In order to pick a suitable significance test for one-dimensional data a normality test of both the original and modified data set was carried out. For this purpose the Shapiro-Wilk test was chosen as it has been shown capable of performing normality tests (Razali and Wah, 2011). To complement the

[2] QQ-plots are probability plots that compare two distributions. In this case, the studied data set's distribution is compared with that of a normal distribution.

[3] `https://github.com/linqcan/odser2014`

Shapiro-Wilk test, QQ-plots and density plots were used to allow visual inspection of the distribution. If any of the two data sets were reported as having a non-normal distribution a non-parametric significance test, Mann-Whitney U, was used. Contrary, if both data sets were normally distributed a parametric significance test, Welch's t-test, was used to decide if they were significantly different. If nothing else is mentioned, a significance level of 95% was used for all significance testing.

The output from using a multi-dimensional algorithm was unknown beforehand. Therefore, the design of the analysis for multi-dimensional algorithms was postponed until the pre-study was conducted. The result from the pre-study and the changes to the analysis can be found in Section 4.3 and Section 4.3.5.

3.3 Replication of Analysis

In order to determine if $\mathbf{H_0}_{\text{Concl}}$ can be rejected the original analysis of the data set was investigated for its reproducibility. If the data set had outliers and the analysis was deemed reproducible further analysis was conducted. This analysis was a replication of the original analysis but with the outliers removed from the data set. $\mathbf{H_0}_{\text{Concl}}$ was rejected if at least one case existed where the conclusion in the original study did not hold when the modified data set was used.

.

Table 1: Descriptive statistics of the paper search pre-study

Papers gathered	43
Papers missing contact information	2
Data requests made	30
Replies stating data is confidential	3
Replies stating 'other reason' to data not available	2
Replies with data	5
Papers with data available (online, paper, from author)	16
Papers with data online	4
Papers with data in the paper	7
Papers disregarded	2
Papers documenting outliers	16
Papers describing their outlier detection	4

4 Results from Pre-Study

The goal of the pre-study is to provide papers and algorithms that could be used for the application of algorithms in Section 5.

In this section descriptive statistics regarding the paper search will be presented before the results from the same search is presented. Furthermore, results from the algorithm search will be presented in the proceeding section.

4.1 Descriptive Statistics for the Paper Search

In Table 1 descriptive statistics for the paper search is presented. From this table one can note that 39% of the papers had data available after a request was sent to the authors. As additional information regarding data availability, the source with the highest rate of papers with data available was PROMISE (60%) followed by ESEJ (35%) and ICSE (33%). Furthermore, it is stated in the table that two replies were given as 'other reasons', these reasons were that the data's size was too large for it to be handed over and that the data was not easily available to the author. Additionally, two papers were disregarded due to their unsuitability to outlier detection. This unsuitability is further elaborated in Section 6.3.

Regarding **RQ1**, PROMISE (80%) had the largest percentage of papers documenting outliers and was followed by ESEJ (53%) and ICSE(14%). The descriptive statistics in Table 1 will be used to answer **RQ1** and **RQ2** in Section 6.1.

4.2 Candidates from the Paper Search

In this section, the papers that were chosen based on the criteria described in Section 3.1.1 are presented. That is, papers whose data was available and whose pre-process was deemed possible to replicate were selected as candidates. All papers reviewed in the paper search are presented in Appendix B with their

corresponding assessment. Information regarding the sample sizes for all data sets part of this study can be found in Appendix D.1.

4.2.1 An empirical study on the developers' perception of software coupling

The study by Bavota et al. (2013) looks into how coupling is perceived by developers by letting two groups of developers study coupling in three software systems. Developers assessed the level of coupling using a 1–5 Likert scale. In total there are 48 data sets available divided over two groups of developers, three applications and eight coupling measures. For the purpose of this study the data sets for one application (jEdit) and one group (external developers) was investigated which sums up to eight data sets. The jEdit data is described as confirming the findings of JHotDraw and ArgoUML by Bavota et al. (2013) and therefore jEdit was chosen. The group "original developers" was disregarded in this study with the same motivation as in the original study, the sample size is too low. The data from this study was made available online by the authors.

4.2.2 Are test cases needed? Replicated comparison between exploratory and test-case-based software testing

Itkonen and Mäntylä (2013) replicates a previous study that compared two manual testing techniques, Exploratory Testing (ET) and Test-Case-based Testing (TCT). This new study compares the two testing techniques using 51 students, who conducted testing on the jEdit text editor. The results from this new study reports that there were no significant difference in effectiveness when detecting defects between ET and TCT. This result is inline with the original study. For the purpose of this study all four data sets in the paper were chosen. The data sets were related to the defect count and testing effort for both of the evaluated techniques. Furthermore, Itkonen and Mäntylä (2013) conducts outlier detection with the help of box-plots. The outliers identified, subjects that are performing exceptionally well or poorly, are still included in their statistical analysis. The data for this study was available online via a reference in the paper.

4.2.3 Mining SQL injection and cross site scripting vulnerabilities using hybrid program analysis

Shar et al. (2013) are studying how SQL injections and Cross Site Scripting (XSS) can be predicted from code analysis. Two different classifiers, Logistic Regression (LR) and multi-layer perceptron, were tested on 10 projects and three attributes were assessed (probability of defect, probability of false alarm and precision). For the purpose of this study outlier detection data from the LR results (all three attributes individually) including both analysis methods presented in the paper (hybrid and static) was chosen. These data sets were selected since they are used when conclusions are drawn in the original paper. In total 6 data sets with 10 data points in each was chosen. The data was available in the paper.

4.2.4 Parameter tuning or default values? An empirical investigation in search-based software engineering

Arcuri and Fraser (2013) investigates how parameter tuning impacts search results by conducting more than one million experiments. The authors claims that tuning parameters has an impact, but are not able to find settings that significantly outperforms the default search parameters. Three data sets containing information about the time budget used for the search were chosen for this study. The three different data sets have the same parameters except for population size which is set to 4, 10 and 50 for the different runs. These data sets were chosen because the authors claims that the population size could be set as a linear function of the search budget. The data from this paper was made available online via a reference in the paper.

4.2.5 An analysis of multi-objective evolutionary algorithms for training ensemble models based on different performance measures in software effort estimation

Minku and Yao (2013) analyzes how Multi-objective Evolutionary Algorithms can be used for software effort estimation. This is achieved with the help of data sets from PROMISE and ISBSG. For this study five data sets claimed to be from the PROMISE repository were chosen. However, only two out of these five data sets were available in the PROMISE repository as of today. The other three data sets were made available by the author after an email request. The data sets from ISBSG were not available for free and were therefore disregarded in this study. Furthermore, Minku and Yao (2013) conducts outlier detection on the data using k-means clustering and this is described in an extended version of the paper. The outliers were not included in the statistical analysis. Part of the data was available online via PROMISE and part of it after an email request.

4.2.6 Improving feature location practice with multi-faceted interactive exploration

In their paper Wang et al. (2013) present a tool for locating features in a system's code base. They compare their tool with Eclipse using two groups with ten participants in each. For this study the results presented in Table 2-3 in their paper was used. Furthermore, each attribute per group was considered as a variable. Performing the analysis on these sets are of interest since they are used in the significance testing carried out by the authors. In total six data sets were chosen. The data was available in the paper.

4.2.7 Transfer defect learning

Nam et al. (2013) are studying cross project defect prediction and propose an extension to Transfer Component Analysis (TCA) called Transfer Component Analysis Plus (TCA+). The authors compare TCA and TCA+ by using two already existing defect benchmarking data sets, ReLink and AEEEM. From the comparison the authors conclude that TCA+ significantly improves cross project prediction performance. For this study the F-measures were chosen since it is the measure used to compare TCA with TCA+. More specifically,

the ReLink and AEEEM data sets for both TCA and TCA+ techniques were chosen. This summed up to 4 data sets in total. The data sets were made available in the paper.

4.2.8 Do background colors improve program comprehension in the #ifdef hell?

Feigenspan et al. (2013) performed three controlled experiments to validate if background colors improve program comprehension in preprocessor-based implementations. All data sets visualized in Figure 4 in the original paper (Feigenspan et al., 2013) was chosen as well as the data used to answer their Research Hypothesis 4. In total twenty-four data sets were assessed. The data was made available by the author after a request was sent.

4.2.9 Reverb: Recommending code-related web pages

Sawadsky et al. (2013) explores how to provide useful web page recommendations to developers. The recommendations is a subset of pages already visited by the developer. The authors introduce a tool called Reverb that recommends previously visited web pages that relate to the code visible in the developers editor. The authors also conduct a field study on nine participants and find that Reverb on average can recommend a useful web page in 51% of the revisitation cases. For this study two data sets that describe the hit rates of Reverb, initial hit rate and optimized hit rate, was chosen. The two data sets were available in the paper.

4.2.10 Drag-and-drop refactoring: Intuitive and efficient program transformation

Lee et al. (2013) introduces a new approach to refactoring through a drag-and-drop tool called Drag-and-Drop Refactoring. An empirical study is conducted to validate the usefulness of the introduced tool. For this study the data collected regarding the configuration time was selected as it is used to draw conclusions about the efficiency of the new tool. More specifically, the data collections called the "Extract Method", "Move Method", "Collated Refactorings" and "Extract Class", which all contains two data sets each named "Eclipse" and "DNDR", were chosen. However, "Extract Class DNDR" was excluded since it is missing data. In total, four data sets are. The data was gathered from Table 4 in the original paper.

4.2.11 The impact of parameter tuning on software effort estimation using learning machines

In their paper, Song et al. (2013) investigates to what extent parameters affect performance of learning machines within SE estimation. Furthermore, they are also studying learning machines' sensitivity to parameter settings. Five learning machines were tested on three data sets and the parameters were varied to study the result. For the purpose of this study the data sets regarding performance for "MLPs" on the "Kitchenham" data set was chosen as the performance of "MLPs" was of particular interest of the original authors. Descriptive statistics

for these three sets are presented in Table 3a in the original paper. The data was sent by the author upon request.

4.2.12 Adoption and use of Java generics

Parnin et al. (2013) conducted an empirical investigation on how Java generics had been integrated into open source projects. They investigated this by comparing 40 open source projects and how they implement Java generics. Out of these projects, 20 of the investigated projects were started before Java generics was introduced (established projects) and 20 of the projects were started after Java generics was introduced (recent projects). This setup, with having established projects and recent projects was chosen to compare the difference in number of days between when Java generics and Java annotations was introduced. In this study two datasets were chosen. The data sets regarded the difference in days between introducing generics and annotations, for recent and established projects. This data relates to RQ3 in their study. From the dataset representing the established projects three data points were missing values and they were removed as they were removed in the original paper. The removal was carried out in order to replicate the original study to the highest possible extent. There were replication tools available online from the authors but for the intermediate data needed in this study the authors needed to be contacted.

4.2.13 Training data selection for cross-project defect prediction

In their study Herbold (2013) proposes distance-based strategies for the selection of training data used for defect prediction. Several predictor models are studied, however, only one of them (Support Vector Machine, SVM) is used for the analysis regarding the success rate, mean recall and mean precision. Furthermore, they conclude that one strategy for this model performs better than the others, neighborhood size 25 (NN-25). Therefore, for the purpose of this study the model SVM for strategy NN-25 was chosen. This summed up to three data sets in total. The data was provided by the authors after a request was sent.

4.3 Candidate algorithms from the algorithm search

Based on the criteria stated in Section 3.1.2 the following three algorithms were deemed suitable: Modified Z Score (MZS) (Garcia, 2012), LOcal Correlation Integral (LOCI) (Papadimitriou et al., 2003) and Angle Based Outlier Detection (ABOD) (Kriegel et al., 2008). MZS is an algorithm for one-dimensional data, i.e. data concerning only one attribute. The other two algorithms, LOCI and ABOD, have been created with the purpose of detecting outliers in multi-dimensional data. Multi-dimensional data is defined in this study as data that concerns two or more attributes/dimensions.

The excluded algorithms and their accompanying motivation can be found in Appendix C.

4.3.1 Modified Z Score (MZS)

The Modified Z Score algorithm measures how much a particular data point differs from the rest of the data set, using a score calculated by Equations 1 and 2.

Modified Z Score is applicable for one dimensional data and calculates the score, M_i, from the Median Absolute Deviation (MAD). Therefore, the algorithm is more robust than algorithms using the mean to score outliers (Garcia, 2012). In the exceptional case where MAD equals zero the same alternative algorithm as used by SPSS (2007) is implemented. Furthermore, to label outliers both algorithms uses the M_i score and compares it to the average M_i score of the data set. Iglewicz and Hoaglin (1993) suggests that if M_i is $> 3.5\sigma$ it should be considered as an outlier and using this cutoff value will make the method more robust. Since MZS is a robust algorithm, does not take any input parameters and does not require training data it is deemed to conform to the criteria for choosing algorithms in this study.

$$\text{MAD} = \text{median}_i(|X_i - \text{median}_j(X_j)|) \tag{1}$$

$$M_i = \frac{0.6745(|X_i - \text{median}_j(X_j)|)}{\text{MAD}} \tag{2}$$

4.3.2 Local COrrelation Integral (LOCI)

Local correlation integral method is an unsupervised and density based outlier detection algorithm. It is based on the idea of a Multi-granularity DEviation Factor (MDEF) and provides an "automatic" method for detecting outliers. MDEF can be explained as a technique that describes how much a data point p's neighborhood density, on a distance αr from p, deviates from its neighborhoods' average density. The neighborhoods chosen for comparison are the neighborhoods within r from p. A point in a very populated neighborhood will have a MDEF of 0 and points that could be suspected outliers will have a MDEF closer to 1 (Papadimitriou et al., 2003).

The LOCI algorithm is described as not being sensitive to the choice of parameters and the authors also suggests how to set the required parameters in (Papadimitriou et al., 2003). In this study a value of 20 was used for the parameter *nmin*, number of neighbors, as proposed by the authors. However, this choice requires the data set to be evaluated to consist of at least 20 data points. To determine if a data point should be considered as an outlier, or an inlier, a cutoff value is required. In this study the cutoff value is set to 3σ, again, as recommended by the authors. Furthermore, α, the scaling factor for the neighborhood, is set to 0.5 in this study, also as recommended by the authors. Finally, the parameter r_{max} can be calculated by using the definition in original paper. Therefore, LOCI does not fulfill the criterion regarding being parameterless. However, the authors propose default values for the required parameters that would make the algorithm robust. For that reason LOCI was chosen despite it not being truly parameterless.

By setting the required parameters to the proposed values, and calculating one according to the original paper, LOCI can be considered parameterless and robust. This, combined with LOCI being unsupervised, makes the algorithm fulfill the selection criteria.

4.3.3 Angle Based Outlier Detection (ABOD)

ABOD is an unsupervised algorithm that takes on a different approach than LOCI. Instead of using a distance measure ABOD is focusing on the angle between two distance vectors for two points seen from a point p under inspection. After the angles for each two pairs of distance vectors have been computed the total variance is calculated for the angles. This total variance is then called the Angle Based Outlier Factor (ABOF) which also represents the score for point p. The output from ABOD is a list of all points sorted by their score. The lower the score, the more likely it is that the point is an outlier. ABOD does not, however, tell you which points are outliers, it only tells you the level of 'outlierness' of each point. Deciding which point that is an outlier needs to be done by the user. An important feature of ABOD is that is does not require any input parameter, except the data, to operate. Finally, in the beginning of this section it is mentioned that distances are not used for ABOD, this however is not completely true. The distance is used but only as a weighting factor to ensure that the angle for points far away from P is contributing less to the variance (Kriegel et al., 2008).

Due to the large amount of pairs that need to be analyzed, ABOD has a time-complexity of $O(n^3)$ which results in long computation times. To solve this Kriegel et al. (2008) propose an approximation of ABOD called FastABOD. This version uses k nearest neighbors and computes the angle value only for pairs among these k neighbors. The result of this is an algorithm not as computationally intensive. However, this algorithm was disregarded in this study since it is dependent on an input parameter, i.e. k nearest neighbors, which is non-trivial to determine for an arbitrary data set. Furthermore, Kriegel et al. (2008) states that "the quality of the approximation depends on the number k of nearest neighbors" which further indicates that this algorithm is not suitable for this study.

The ABOD algorithm is parameterless in its definition (Kriegel et al., 2008), however, the implementation requires a kernel function to be supplied which is used for similarity checks. The default polynomial kernel function with a degree of 2 was chosen for this purpose.

ABOD fulfills the criteria by being unsupervised, not making any assumptions about the distribution and by not requiring any input parameters.

4.3.4 Changes to the Automated Process

As a result of the found algorithms the planned automated process was changed and those changes are defined further in this section and visualized in Appendix E.

Modified Z Score was implemented in R[4] and is available in this study's repository[5]. The LOCI and ABOD algorithms used were supplied by the ELKI Data Mining Framework[6](Achtert et al., 2013) and this software was also used to run the algorithms. Python was used to connect the different tools with each other as well as for data parsing.

[4] http://www.r-project.org/

[5] https://github.com/linqcan/odser2014

[6] http://elki.dbs.ifi.lmu.de

For a one-dimensional data set the MZS algorithm is run in R. The result is stored in a comma separated value (CSV) file. This CSV file contains the original data set and a boolean flag for each point stating if it is an outlier or not.

A data set which contains more than one dimension is evaluated with both LOCI and ABOD using ELKI. Before LOCI is run the parameter r_{max}, required by ELKI, is calculated and passed on to ELKI. The results from the outlier detection in ELKI are written to a CSV file. This file is then parsed in Python to remove strings inserted by ELKI. For LOCI this result file contains the original data points and the $MDEF_\sigma$ value for each point. Similarly, the ABOD contains the original data points and the ABOD outlier score. The CSV file for LOCI is then fed to a R script that takes the original data set and labels data points as outliers with a boolean flag if their respective $MDEF_\sigma$ value is above a specified cutoff value, 3σ. The result of this labelling is then stored in a CSV file. No further processing is to be done on the ABOD results. The CSV files with the results from the outlier detection algorithms is then used in the analysis phase.

4.3.5 Changes to the Analysis

As ABOD does not explicitly tell which points are outliers, an addition to the analysis presented in Section 3.2.1 is needed. The results from applying ABOD on a data set is compared with that of LOCI to see if the n detected outliers by LOCI are the *top* n reported by ABOD. The motivation behind this is to see if the result from an angle-based method differs much from a distance-based. Furthermore, for multi-dimensional data, no intermediate significance testing is carried out. Instead, only the conclusion, after the removal of the reported outliers from LOCI, is validated.

5 Results

This section presents the results from applying outlier detection and removal on the papers selected in the pre-study as well as a deeper analysis of a few selected papers.

5.1 Application of Outlier Detection Algorithms

The results from applying outlier detection algorithms and removing outliers are presented in Table 2 and data sets containing outliers are presented by the plots in Figure 3, Figure 4 and Figure 5.

5.2 Analysis of Selected Papers

This section describes a more in depth analyze of the papers from Section 5.1 that contained outliers. The purpose of this analysis is to determine if the removal of outliers affect the result of analysis in the original paper. Hence, the original analysis is reproduced on a modified data set that does not contain outliers. The sample sizes of the modified data sets are presented in Appendix D.2 and the R code for the analysis conducted in Appendix G.

5.2.1 Do background colors improve program comprehension in the #ifdef hell?

In this analysis the focus is on on research hypotheses RH1, RH2 and RH4 from (Feigenspan et al., 2013) since they are all regarding non-binary data and their data was made accessible by the authors.

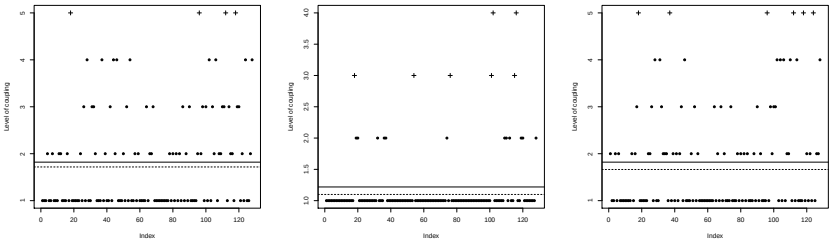
For RH1 and RH2 post-processing was carried out on all the data sets regarding the maintenance tasks (Mx) as the authors had omitted response times for questions that were answered incorrectly. This omission was mentioned in the original paper and clarified by the authors via email correspondence.

The authors answer RH1 ("In static tasks, colors speed up program comprehension compared to ifdef directives") and RH2 ("In maintenance tasks, there are no differences in response time between colors and ifdef directives") by conducting a significance test and an effect size test for the static and maintenance tasks to compare the test using colors with the test using ifdef. To reproduce these tests a Mann-Whitney U test was used for non-parametric and Welch's t-test for parametric significance testing. Only the data sets with possible outliers, S1-ifdef, M1-ifdef and M2-ifdef was considered for the reproduction. The new significance tests carried out in this study, Mann-Whitney U for S1 and M2 and Welch's t-test for M1, showed no different conclusion than those carried out by the original authors. The effect size test, calculated using Cliff's delta, for S1 was slightly altered (-0.6417 compared to -0.61) but it resulted in no different conclusions as well.

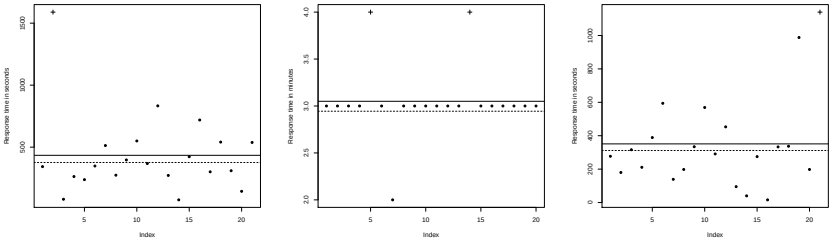
RH4 was validated using significance tests in the original study. The data sets used in RH4 consisted of results from a survey using a 1–5 Lickert scale. After outlier detection was conducted on the data sets used for this research

Table 2: Papers whose data outlier detection and removal was conducted on. Number of data sets containing outliers is presented within parentheses. 'Difference' refers to if the modified data sets were significant different to their original counterparts.

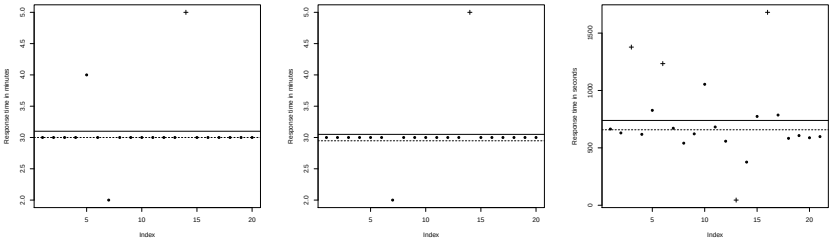
Paper	Dimensions	Outliers	Difference	Reference
An empirical study on the developers perception of software coupling	1	Yes (3/8)	No	Bavota et al. (2013)
Are test cases needed? Replicated comparison between exploratory and test-case-based software testing	1	No	No	Itkonen and Mäntylä (2013)
Mining SQL injection and cross site scripting vulnerabilities using hybrid program analysis	1	Yes (1/6)	No	Shar et al. (2013)
Parameter tuning or default values? An empirical investigation in search-based software engineering	1	No	No	Arcuri and Fraser (2013)
An analysis of multi-objective evolutionary algorithms for training ensemble models based on different performance measures in software effort estimation	1	Yes (5/5)	No	Minku and Yao (2013)
Improving feature location practice with multi-faceted interactive exploration	1	Yes (2/6)	No	Wang et al. (2013)
Transfer defect learning	1	No	No	Nam et al. (2013)
Do background colors improve program comprehension in the #ifdef hell?	1	Yes (6/24)	No	Feigenspan et al. (2013)
Reverb: recommending code-related web pages	1	No	No	Sawadsky et al. (2013)
Drag-and-drop refactoring: intuitive and efcient program transformation	1	Yes (2/4)	No	Lee et al. (2013)
The impact of parameter tuning on software effort estimation using learning machines	1	Yes (3/3)	No	Song et al. (2013)
Adoption and use of Java generics	1	Yes (2/2)	No	Parnin et al. (2013)
Training data selection for cross-project defect prediction	1	No	No	Herbold (2013)



(a) jEdit logical low (Bavota et al., 2013) **(b)** jEdit semantic (Bavota et al., 2013) **(c)** jEdit structural (Bavota et al., 2013)

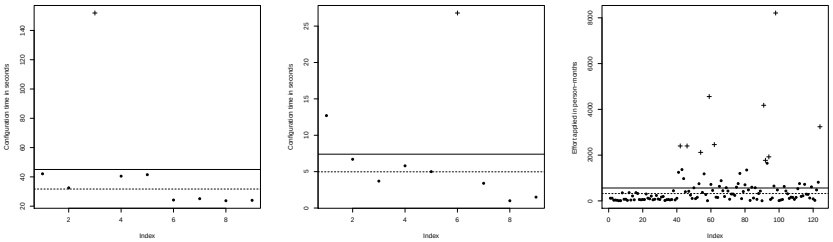


(d) M1-ifdef (Feigenspan et al., 2013) **(e)** M1-ifdef performance (Feigenspan et al., 2013) **(f)** M2-ifdef (Feigenspan et al., 2013)



(g) M2-ifdef performance (Feigenspan et al., 2013) **(h)** M3-ifdef performance (Feigenspan et al., 2013) **(i)** S1-ifdef (Feigenspan et al., 2013)

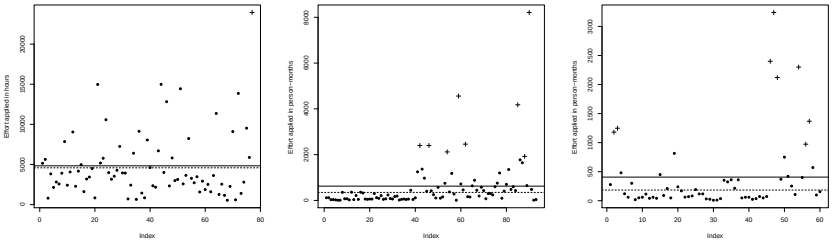
Figure 3: Plots displaying outliers for the analyzed papers. Outliers are marked with a cross. The original mean is displayed as a solid horizontal line whereas the mean after outlier removal is shown as a dotted line.



(a) Extract Class Eclipse (Lee et al., 2013)

(b) Move Method DNDR (Lee et al., 2013)

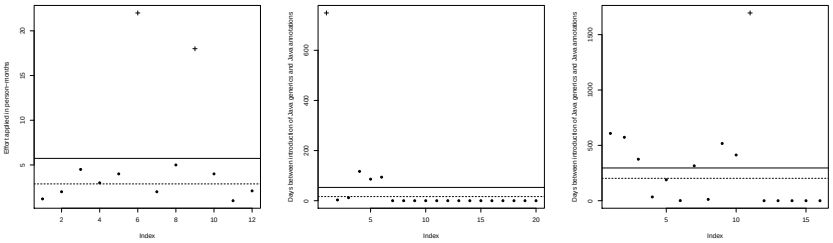
(c) Cocomo81 Effort: person months (Minku and Yao, 2013)



(d) Desharnais Effort: hours (Minku and Yao, 2013)

(e) NASA93 Effort: person months (Minku and Yao, 2013)

(f) NASA Effort: person months (Minku and Yao, 2013)

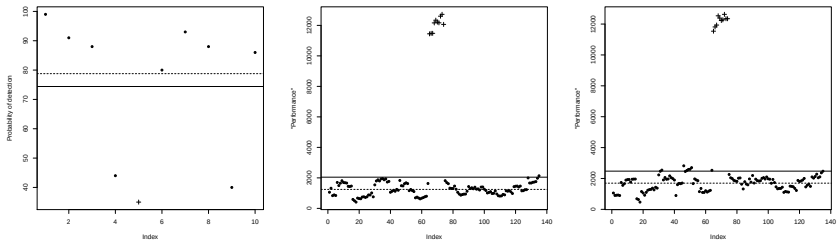


(g) SDR Effort: person months (Minku and Yao, 2013)

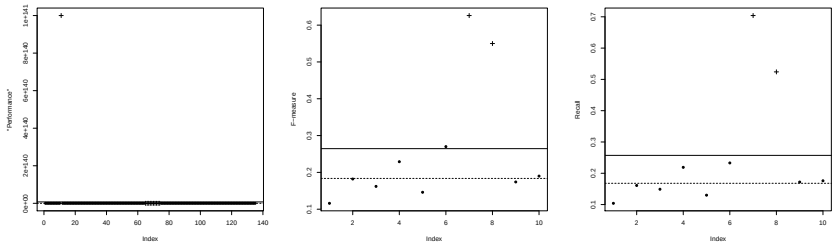
(h) Projects started after generics was introduced (Parnin et al., 2013)

(i) Projects started before generics was introduced (Parnin et al., 2013)

Figure 4: Plots displaying outliers for the analyzed papers. Outliers are marked with a cross. The original mean is displayed as a solid horizontal line whereas the mean after outlier removal is shown as a dotted line.



(a) Static defect (Shar et al., 2013) **(b)** MLP Kitchenham Best (Song et al., 2013) **(c)** MLP Kitchenham Default (Song et al., 2013)



(d) MLP Kitchenham Worst (Song et al., 2013) **(e)** Eclipse F-measure (Wang et al., 2013) **(f)** Eclipse recall (Wang et al., 2013)

Figure 5: Plots displaying outliers for the analyzed papers. Outliers are marked with a cross. The original mean is displayed as a solid horizontal line whereas the mean after outlier removal is shown as a dotted line.

question, three data sets were reported to have potential outliers: M1-ifdef performance, M2-ifdef performance and M3-ifdef performance. Since these were the only modified data sets, significance tests were only reproduced for these three data sets. The reproduction of the statistical analysis, using a Mann-Whitney U test, showed no other results than those reported in the original study.

5.2.2 Improving feature location practice with multi-faceted interactive exploration

The study by Wang et al. (2013) fits the criterions stated in the method as a suitable candidate for outlier detection as well as for further analysis. However, in the analysis, a paired t-test is used to determine if there is a significant difference in performance between using Eclipse and Multi-Faceted Interactive Explorer (MFIE). Therefore, further analysis is not possible and this will be elaborated in Section 6.2.

5.2.3 Drag-and-drop refactoring: Intuitive and efficient program transformation

Although this study matched the specified criteria for further analysis such analysis was not possible for (Lee et al., 2013). The reason for this being that the statistical analysis (significance test) conducted in the paper was conducted as pairwise analysis. Further elaboration regarding this can be found in Section 6.2.

5.2.4 Mining SQL injection and cross site scripting vulnerabilities using hybrid program analysis

Reproducing the analysis of this study, with outliers removed, was not possible, even though it matched the specified criteria, since the study uses a pair-wise statistical test. The motivation behind this decision is discussed in Section 6.2.

5.2.5 The impact of parameter tuning on software effort estimation using learning machines

In their research question RQ1 Song et al. (2013) draw conclusions about the methods studied independently from each other. However, the analysis, of determining which parameter setting that is better for each approach on each set, uses a pair-wise comparison ("Wilcoxon sign-rank test with Holm-Bonferroni corrections"). Further analysis of the impact of outlier removal on this study is therefore not possible and the reason for this is discussed in Section 6.2.

5.2.6 An empirical study on the developers' perception of software coupling

The results from the experiment are reported using a 1–5 Lickert scale and were not processed before being analyzed. In the original analysis the different coupling techniques' p-values are compared with each other. The p-values are calculated with a Mann-Whitney U test, this test is then used to determine if there is a perceived difference between the different coupling techniques. Out of the eight investigated data sets from jEdit only three of them contained outliers: Semantic-low, structural-low and logical-low. After removing all outliers from the three data sets the same Mann-Whitney U tests as used in the original study was executed. This lead to six tests and in one of those tests the p-values changed noticeable, in the comparison between structural low and logical low. However, this change does not affect the overall conclusion of the original study.

5.2.7 Adoption and use of Java generics

The study by Parnin et al. (2013) matched the specified criteria for being a suitable candidate for all except the reproducibility of the analysis as it was deemed as not being reproducible. The authors of Parnin et al. (2013) base their conclusions both on statistical significance tests such as t-tests and reasoning which makes it difficult to evaluate how the the investigated research question is affected by removing outliers. Therefore, this study will be excluded for further analysis.

5.2.8 An analysis of multi-objective evolutionary algorithms for training ensemble models based on different performance measures in software effort estimation

Minku and Yao (2013) is suitable as a candidate for further analysis on the basis that it is an empirical study which clearly describes how empirical data, suitable for outlier detection, was collected and processed. However, it was excluded from further analysis in this study since part of the data needed for evaluating the conclusions is not available for free and can not be re-published.

Including Minku and Yao (2013) would therefore hinder this report from being reproducible.

6 Discussion

In this section we will discuss and answer the previously defined hypothesis and research questions. This will be done by using the data collected during the pre-study, application of algorithms and replication of analysis. Furthermore, improvements to our review process will be discussed and finally guidelines will be presented.

6.1 Results from Pre-Study

From the pre-study we obtained data, presented in Table 1, showing the current status of outlier detection within ESE. Based on this data we can answer our research questions as follows.

RQ1: 37% of the surveyed papers did document presence of outliers and 25% of those documenting outliers document some kind of outlier detection. Unfortunately, it is not possible to see any trends or draw any conclusions regarding how documentation is conducted as all of the papers documenting outlier detection do this in different ways. This is clearly troublesome and hinders reproducibility in the long term. Additionally, we did not observe any paper that conducted outlier detection without documenting it.

RQ2: 37% of the papers collected had data available and 26% of those offered their data in the paper or online. In total, 16% of the papers in the sample offered data in the paper and 9% online. The result presented here is far from surprising for an immature research field such as SE. There could be many reasons for the low outcome and we propose two reasons which we believe are more important:

First, there is a lack of consensus on how to treat and make raw data available within SE. This is, according to us, a major issue and something that the field of SE needs to address. In our guidelines, in Section 6.5, we elaborate more on this.

The second reason, proposed by us, for the low outcome is that replication is not kept in mind by researchers while conducting their research. This might have to do with replication not being common within SE as it is a fairly immature research field (compared to physics and medicine). Also, we have observed that the replication mentioned in SE research literature concerns full experiment replication and does rarely mention replication of the data analysis, so called re-analysis. However, we believe that re-analysis is of importance as well since it can be used to validate if conclusions based on, for example, significance tests are correct. As an example of this, one of the papers included in this study was found by us to have incorrect calculations in the analysis. This error was later confirmed and corrected by the author who stated that it fortunately had no impact on the conclusions in the paper. In addition, being able to conduct re-analysis would further encourage meta-analysis in SE research, since access to data and statistical analysis procedures would be readily available.

During the course of the pre-study we encountered some issues, except those mentioned earlier, regarding data handling and analysis worth mentioning. We will present our proposal for solutions for the following issues in Section 6.5:

- Some authors use data from repositories such as the PROMISE database. However, they do not clearly state how the data was extracted from the repository which makes replication tedious and in some cases impossible. For example, in one of the investigated papers, the data sets were divided into subsets and it was not described how the division was done. This hindered us from recreating the post processing of the data and we had to exclude the paper.

- Even though we were given access to raw data, this data was at times 'too' raw. Meaning that to get the data needed to reproduce their analysis we also needed to perform some kind of data extraction. This is both a difficult and time consuming procedure (thus prone to errors) which further complicates the replication of a study.

- The connection between the raw data and the paper was not obvious for some of the studies we reviewed. For example, one data set used column names which were a combination of abbreviations and words in the author native language (not english). This led to some confusion and we needed to contact the author several times in order to understand the published data.

- For some of the papers analyzed in Section 5.2 we had to contact the authors to have them explain their analysis and motivation behind the choices they made. For example, some papers mention that they use a "Wilcoxon" significance test without specifying if it is one or two-sided and/or a paired test. This created unnecessary uncertainty about the analysis conducted and made replication more difficult.

When taking into account all the above items it is clear that journals and conference should require authors to be more explicit in describing study execution and analysis.

6.2 Results from the Application of Algorithms

After applying outlier detection with the parameters described in Section 3 on the candidate papers from Section 4 we found that 24 out of the 77 data sets contained outliers. As an additional test we removed the data points suggested as outliers by the detection algorithm and compared the modified data set with the original one. This comparison, using a significance test, showed that there was no significant difference between the two sets in any of the 24 cases. As we did find outliers using our outlier detection method we reject our null-hypothesis $\mathbf{H_{0}}_{\text{Outliers}}$ in favour for our alternative hypothesis $\mathbf{H_{1}}_{\text{Outliers}}$. As described in Section 3, we chose a robust method for detecting outliers since we wanted a method that fitted a wide range of data sets in order to implement an automatic process. However, this probably led to less outliers being found in comparison to if we would have chosen a suitable outlier detection method for each data set. This fact should be taken into account when interpreting our results and is mentioned in our threats to validity in Section 7.

Two of the analyzed papers in Section 5.2, (Itkonen and Mäntylä, 2013) and (Minku and Yao, 2013), did document some kind of outlier detection. After conducting our own outlier detection on these papers we noticed that the number

of detected outliers differed. However, this difference is probably due to the type of method used and the accuracy of it. Itkonen and Mäntylä (2013) uses a standard box plot which has a narrower span for the definition of an inlier than our MZS method. Minku and Yao (2013) uses a k-means clustering-based method which is fundamentally different from MZS; but as MZS reported more outliers than theirs we assume that our MZS uses a more narrow span for defining an inlier than their clustering method.

For the analysis of the effect on the conclusion after removing outliers, 6 papers had matched our criteria (outliers exist, analysis explained) and were candidates for being further analyzed. However, we had to disregard 4 papers due to their use of a pair-wise test in the analysis. The motivation behind this decision was that if we remove data points from one of the sets we will break pairs. A countermeasure would be to remove the whole pair and this method would work in some cases but not in this case. If we were to remove pairs we would also remove potential non-outliers from a data set. This would introduce uncertainty regarding the effect of the outlier detection and removal we have conducted. While researching the area we did not find any other methods for handling outliers in data sets used for pair-wise testing. In Section 7 we present the method of removing a whole pair as a possible threat to validity. The two remaining papers did not provide results from which we could reject our second hypothesis, $\mathbf{H_0}_{\text{Concl}}$. Though, it should be noted that we did have a small sample size. We do, however, believe that further expanding the sample size would be a difficult task as our results show that it is difficult to obtain data from recently published studies as well as conducting a re-analysis.

Even though the removal of outliers did not affect any conclusions, the algorithm used (MZS) did identify outliers in some cases. As the algorithm is easy to use it could still be of interest for researchers to use this method to quickly identify outliers. However, even though the algorithm proposes a data point to be an outlier, this alone is not enough to exclude it from a data set. Researchers are encouraged to use simple detection algorithms, such as MZS, and use the result from these to discuss the inclusion or exclusion of data points in their study.

6.3 Regarding the Review Process

During the pre-study two papers were deemed to fit our initial criteria but were later disregarded. The reason for this was that the data sets found in these papers did not have enough samples to conduct outlier detection. In hindsight, a criterion stating the minimum amount of samples needed for a paper to be a candidate would have been an improvement to our review process.

Another improvement to our review process would be a criterion stating that all data used to draw conclusions has to be free of charge and that republishing of it is allowed. Since this criterion did not exist (Minku and Yao, 2013) was excluded late in the process.

6.4 Multi-Dimensional Outlier Detection

During the set up of this study we planned for having outlier detection algorithms for both one-dimensional and multi-dimensional data. However, as the pre-study turned out we did not encounter any multi-dimensional data sets. As

our sample was a 'current' sample one could say that ESE today does not involve analysis of multi-dimensional data, according to our sample. However, we believe that the field will progress into handling larger data sets on which one would like to perform multi-dimensional analysis. As an example, one could assess how a developer is performing based on several attributes over several projects (open source contribeutors for example). The use of large data sets will also be facilitated through the access of big data storage and computing clusters. So, we propose that the field of SE starts discussing if/how multi-dimensional outlier detection should be conducted within the field in the future and how this should be documented. Our pre-study regarding algorithms can be used as a small and limited introduction to multi-dimensional algorithms and their issues. To conclude, there is still a lot of work to be done in regards to multi-dimensional outlier detection within SE.

6.5 Guidelines

In the beginning of this study we sat out to propose guidelines for conducting outlier detection. However, as our study included replication, and this was easier said than done, we have come across more challenges. In this section we propose solutions for those challenges as well since they can be a contribution to the body of knowledge of SE.

Guidelines for Outlier Detection

- Outlier detection algorithms are only tools to help suggest what data points could be outliers based on the specification of the algorithms. Therefore, researchers should view these suggestions critically and not remove data points without reflecting over the suggestions.
- A motivation to why a data point is an outlier should always be provided no matter if detection algorithms with cutoff values or scoring is used.
- When conducting outlier detection, one should present which data points were regarded as outliers by the algorithm. This can be done with the help of a scatter plot for example. Also, motivate which potential outliers that were disregarded and excluded from the statistical analysis.
- Reflect and argue for the usefulness of outlier removal before conducting it. As in the case with pair-wise tests (discussed in Section 6.2), it might not always make sense to remove deviating data points.
- Always document. It is important that no tacit knowledge is needed to replicate the outlier detection and removal conducted.

Guidelines for Facilitating Replication

- When using already available data, such as data from the PROMISE database, it is important to present how the information was extracted. This is proposed by us to be done by giving instructions or, preferably, providing extraction scripts. Furthermore, we propose that the data from the extraction and post-processing steps are made available to ease the verification of the replication.

- Use online data storage solutions such as the PROMISE database, Figshare, Dropbox or Github instead of hosting data on personal university pages that might be terminated when the authors leave.
- The information presented in the paper should clearly correspond to the information in the raw data set. Preferably, the authors should provide a key stating the mapping between the paper and the data set.
- The kind of significance test conducted should be clearly stated together with a motivation of why this test was chosen. If the test is conducted using a statistical software tool, mentioning which tool and providing all parameters used for the test is of importance.

7 Threats to Validity

7.1 Internal

Internal threats to validity involves factors, such as instrumentation settings, that could have affected the outcome.

- For conducting outlier detection robust methods were used. These methods used parameters settings that were meant to fit a wide range of data sets and were not specialized. Having used specialized settings for each data set we might discover more/less outliers than we did in this study.
- Removing a full pair in data sets used for pair-wise testing could be a validity threat as we could potentially remove non-outliers from one data set. This would then make it difficult to conclude anything about the effect of the outlier detection we did initially. However, we did not conduct this ourselves in this study but we like to underline the threat in conducting such elimination of data points.

7.2 External

External threats to the validity are factors that affect the ability to generalize the results outside the scope of this study.

- We only collected a current sample from the last one and a half year. This limited sample might not be representable for studies conducted earlier, but we deemed them to be a representative sample of current SE research.

7.3 Conclusion

Threats to the conclusion validity are factors that might affect how the conclusions are drawn.

- Some of the data sets gathered during the pre-study did have a small sample size to begin with (see Appendix D.1). The size became even smaller when we removed data points suggested as outliers. The initial small sample size is a validity threat to the original study, but the new smaller size is a validity threat to our study and how we draw conclusions regarding the significant difference of a data set before and after outliers are removed.

7.4 Construct

Threats to the construct validity can be design errors in the study which could lead to the wrong phenomena being studied. These errors could be caused by social factors.

- Only having one researcher reviewing each study might have caused a bias. To try and mitigate this risk we discussed issues regarding the studies among us.

8 Conclusions

In this study we have investigated the presence of outliers in ESE studies as well as the effect outliers have on conclusions drawn. In order to conduct this study we needed the original data, documented post-processing and documented analysis as input. Therefore, this study also investigates to what extent the presence of outliers is documented, outlier detection is conducted and data is available from ESE studies.

An automated process was created to conduct outlier analysis and create the results used to answer our hypotheses. This process is available for download[7] and all results can be reproduced for replication and verification purposes.

Based on the results gathered from the application of an outlier algorithm we can conclude that outliers do exist in ESE studies.

From the information we collected while preparing our replications we found that 37% of the investigated studies document outliers in some way and that 25% out of those do conduct some kind of outlier detection. Regarding the data availability, we found that 26% of the studies have their data directly available either in the paper or online. Additionally, 12% of the studies' authors replied with data after we sent out an email request. In total, 37% of the studies investigated had data available. From this we conclude that the state of replication, in regards to replicating data analysis, is less than desirable within ESE and we think it is in need of improvement.

In order to help the research field improve our study provides the following contributions to the body of knowledge:

- Outliers exists within recently published ESE studies and can be found with robust methods.
- The extent to which recently published ESE studies document outliers and conduct outlier detection.
- The extent to which recently published ESE studies make their data available and how it is made available.
- Guidelines for conducting and presenting outlier detection for ESE.
- Guidelines for how to improve the reproducibility of ESE studies.
- An analysis of recently published results and reproducibility within ESE.

[7] `https://github.com/linqcan/odser2014`

9 Future Work

For future work we recommend not to conduct more studies regarding outliers and outlier detection on already published studies. Instead, we propose that this should be done on to-be-published studies by journal and conference authors and reviewers through the use of mandatory outlier detection. For example, a journal could make it mandatory for all submitted papers to include information about how outliers were handled and "did not handle outliers" can not be an answer. This information would include information about the outlier detection method used, actions taken based on the result from the detection and motivations for the method used as well as the actions taken. The reason we propose to not conduct more studies on already published studies is that our results shows that it is difficult to obtain data from recently published studies. Hence, we assume that trying to get hold of data from older studies could be even more difficult.

A Software Requirements for Using Developed Tools

This section lists which software packages that were used to develop the tools for the pipeline. In order to recreate the results of this study the software dependencies mentioned in this section needs to be fulfilled. The pipeline and generation of results have been verified to work on Linux and OS X.

- Python 2.7
- R 2.15.1 or greater
  - xtables
  - rjson
- TEX Live 2013 (required packages are listed below)
  - caption
  - float
  - graphicx
  - hyperref
  - longtable
  - natbib
  - pdflscape
  - subcaption
  - tabu
  - tikz
- ELKI 0.6.0 (for multi-dimensional data sets)
  - OpenJDK 7

B Papers Reviewed in the Pre-Study

Table B.1: All papers used in the study. *Post-process* refers to if replication of the data post-processing was possible. *Analysis* refers to if replication of the data analysis was possible.

Paper	Data available	Post-process	Analysis	Excluded	Has outliers	Analyzed	Reference
Software bertillonage	Online	Not evaluated	Not evaluated	Yes	No	No	Davies et al. (2013)
Are test cases needed? Replicated comparison between exploratory and test-case-based software testing	Online	Yes	Yes	No	No	No	Itkonen and Mäntylä (2013)
Training data selection for cross-project defect prediction	From Author	Yes	No	No	No	No	Herbold (2013)
Adoption and use of Java generics	From Author	Yes	No	No	Yes	No	Parnin et al. (2013)
Effectiveness for detecting faults within and outside the scope of testing techniques: an independent replication	Paper	No	No	No	No	No	Apa et al. (2014)
Drag-and-drop refactoring: intuitive and efcient program transformation	Paper	Yes	Yes	No	Yes	Yes	Lee et al. (2013)
Program transformations to fix C integers	Paper	Not evaluated	Not evaluated	Yes	No	No	Coker and Hafiz (2013)

Table B.1: All papers used in the study. *Post-process* refers to if replication of the data post-processing was possible. *Analysis* refers to if replication of the data analysis was possible.

Paper	Data available	Post-process	Analysis	Excluded	Has outliers	Analyzed	Reference
Software defect prediction using bayesian networks	No response	Yes	No	No	No	No	Okutan and Yıldız (2014)
A replicated quasi-experimental study on the influence of personality and team climate in software development	No response	No	No	No	No	No	Gómez and Acuña (2013)
How do open source communities blog?	No response	Not evaluated	Not evaluated	No	No	No	Pagano and Maalej (2013)
How (and why) developers use the dynamic features of programming languages: the case of smalltalk	No response	Not evaluated	Not evaluated	No	No	No	Callaú et al. (2013)
Performance and reliability prediction for evolving service-oriented software systems	Confidential	Not evaluated	Not evaluated	No	No	No	Koziolek et al. (2013)

Table B.1: All papers used in the study. *Post-process* refers to if replication of the data post-processing was possible. *Analysis* refers to if replication of the data analysis was possible.

Paper	Data available	Post-process	Analysis	Excluded	Has outliers	Analyzed	Reference
Parameter tuning or default values? An empirical investigation in search-based software engineering	Online	Yes	Yes	No	No	No	Arcuri and Fraser (2013)
The limited impact of individual developer data on software defect prediction	No response	Not evaluated	Not evaluated	No	No	No	Bell et al. (2013)
Using error abstraction and classification to improve requirement quality: conclusions from a family of four empirical studies	No response	Not evaluated	Not evaluated	No	No	No	Walia and Carver (2013)
An empirical study of the state of the practice and acceptance of model-driven engineering in four industrial cases	Confidential	Not evaluated	Not evaluated	No	No	No	Mohagheghi et al. (2013)

Table B.1: All papers used in the study. *Post-process* refers to if replication of the data post-processing was possible. *Analysis* refers to if replication of the data analysis was possible.

Paper	Data available	Post-process	Analysis	Excluded	Has outliers	Analyzed	Reference
Using tabu search to configure support vector regression for effort estimation	No response	Not evaluated	Not evaluated	No	No	No	Corazza et al. (2013)
Do background colors improve program comprehension in the #ifdef hell?	From Author	Yes	Yes	No	Yes	Yes	Feigenspan et al. (2013)
X-PERT: accurate identication of cross-browser issues in web applications	No response	Not evaluated	Not evaluated	No	No	No	Roy Choudhary et al. (2013)
Can traditional fault prediction models be used for vulnerability prediction?	No response	Not evaluated	Not evaluated	No	No	No	Shin and Williams (2013)
Building a second opinion: learning cross-company data	No response	Not evaluated	Not evaluated	No	No	No	Kocaguneli et al. (2013)
Effort estimation of FLOSS projects: a study of the Linux kernel	No response	Not evaluated	Not evaluated	No	No	No	Capiluppi and Izquierdo-Cortázar (2013)

Table B.1: All papers used in the study. *Post-process* refers to if replication of the data post-processing was possible. *Analysis* refers to if replication of the data analysis was possible.

Paper	Data available	Post-process	Analysis	Excluded	Has outliers	Analyzed	Reference
Mining SQL injection and cross site scripting vulnerabilities using hybrid program analysis	Paper	Yes	Yes	No	Yes	Yes	Shar et al. (2013)
An analysis of multi-objective evolutionary algorithms for training ensemble models based on different performance measures in software effort estimation	From Author	Yes	Yes	No	Yes	No	Minku and Yao (2013)
An empirical study on the developers perception of software coupling	Online	Yes	Yes	No	Yes	Yes	Bavota et al. (2013)
Are comprehensive quality models necessary for evaluation software quality	No response	Not evaluated	Not evaluated	No	No	No	**?**
A learning-based method for combining testing techniques	No response	Not evaluated	Not evaluated	No	No	No	Cotroneo et al. (2013)

Table B.1: All papers used in the study. *Post-process* refers to if replication of the data post-processing was possible. *Analysis* refers to if replication of the data analysis was possible.

Paper	Data available	Post-process	Analysis	Excluded	Has outliers	Analyzed	Reference
The impact of parameter tuning on software effort estimation using learning machines	From Author	Yes	Yes	No	Yes	Yes	Song et al. (2013)
POPT: a problem-oriented programming and testing approach for novice students	No response	Not evaluated	Not evaluated	No	No	No	Neto et al. (2013)
Human performance regression testing	No response	Not evaluated	Not evaluated	No	No	No	Swearngin et al. (2013)
Guided test generation for web applications	No contact info	Not evaluated	Not evaluated	No	No	No	Thummalapenta et al. (2013)
Reverb: recommending code-related web pages	Paper	Yes	Yes	No	No	No	Sawadsky et al. (2013)
Comparing multi-point stride coverage and dataow coverage	No	Not evaluated	Not evaluated	No	No	No	Hassan and Andrews (2013)
Predicting bug-fixing time: an empirical study of commercial software projects	No response	Not evaluated	Not evaluated	No	No	No	Zhang et al. (2013a)

Table B.1: All papers used in the study. *Post-process* refers to if replication of the data post-processing was possible. *Analysis* refers to if replication of the data analysis was possible.

Paper	Data available	Post-process	Analysis	Excluded	Has outliers	Analyzed	Reference
Automatic patch generation learned from human-written patches	Confidential	Not evaluated	Not evaluated	No	No	No	Kim et al. (2013)
Bridging the gap between the total and additional test-case prioritization strategies	No response	Not evaluated	Not evaluated	No	No	No	Zhang et al. (2013b)
Improving feature location practice with multi-faceted interactive exploration	Paper	Yes	Yes	No	Yes	Yes	Wang et al. (2013)
What good are strong specications?	No	Not evaluated	Not evaluated	No	No	No	Polikarpova et al. (2013)
Partition-based regression verication	No response	Not evaluated	Not evaluated	No	No	No	Böhme et al. (2013)
How to effectively use topic models for software engineering tasks? An approach based on genetic algorithms	No response	Not evaluated	Not evaluated	No	No	No	Panichella et al. (2013)
Transfer defect learning	Paper	Yes	Yes	No	No	No	Nam et al. (2013)

Table B.1: All papers used in the study. *Post-process* refers to if replication of the data post-processing was possible. *Analysis* refers to if replication of the data analysis was possible.

Paper	Data available	Post-process	Analysis	Excluded	Has outliers	Analyzed	Reference
How, and why, process metrics are better	No response	Not evaluated	Not evaluated	No	No	No	Rahman and Devanbu (2013)
Not going to take this anymore: multi-objective overtime planning for software engineering projects	No contact info	Not evaluated	Not evaluated	No	No	No	Ferrucci et al. (2013)

C Algorithms Reviewed in the Pre-Study

Table C.2: Algorithms reviewed in this study.

Name	Dimensions	Type	Unsupervised	Robust	Parameterless	Reference	Comment
Modified Z Score	One	Median Absolute Deviation	Yes	Yes	Yes	Garcia (2012)	See Section 4.3
ABOD	Multi	Angle based	Yes	Yes	Yes	Kriegel et al. (2008)	See Section 4.3
LOCI	Multi	Density based	Yes	Yes	No	Papadimitriou et al. (2003)	See Section 4.3
LOF	Multi	Density based	Yes	No	No	Breunig et al. (2000)	Difficult to calculate parameter needed
PCOut	Multi	Median Absolute Deviation	Yes	Yes	No	Filzmoser et al. (2008)	Difficult to estimate the parameters needed
FastABOD	Multi	Angle based	Yes	Yes	No	Kriegel et al. (2008)	k nearest neighbor
ORCA	Multi	Distance based	Yes	Yes	No	Bay and Schwabacher (2003)	k nearest neighbor

D Sample Sizes for Data Sets Used in this Study

D.1 Before Outlier Detection and Removal

Table D.3: Sample sizes for the original data sets used in this study

Data set name	Reference	Sample size
popsize10	Arcuri and Fraser (2013)	15
popsize4	Arcuri and Fraser (2013)	15
popsize50	Arcuri and Fraser (2013)	15
jEdit dynamicHigh	Bavota et al. (2013)	128
jEdit dynamicLow	Bavota et al. (2013)	128
jEdit logicalHigh	Bavota et al. (2013)	128
jEdit logicalLow	Bavota et al. (2013)	128
jEdit semanticHigh	Bavota et al. (2013)	128
jEdit semanticLow	Bavota et al. (2013)	128
jEdit structuralHigh	Bavota et al. (2013)	128
jEdit structuralLow	Bavota et al. (2013)	128
m1color	Feigenspan et al. (2013)	21
m1color performance	Feigenspan et al. (2013)	22
m1ifdef	Feigenspan et al. (2013)	21
m1ifdef performance	Feigenspan et al. (2013)	21
m2color	Feigenspan et al. (2013)	21
m2color performance	Feigenspan et al. (2013)	22
m2ifdef	Feigenspan et al. (2013)	21
m2ifdef performance	Feigenspan et al. (2013)	21
m3color	Feigenspan et al. (2013)	19
m3color performance	Feigenspan et al. (2013)	22
m3ifdef	Feigenspan et al. (2013)	21
m3ifdef performance	Feigenspan et al. (2013)	21
m4color	Feigenspan et al. (2013)	15
m4color performance	Feigenspan et al. (2013)	22
m4ifdef	Feigenspan et al. (2013)	21
m4ifdef performance	Feigenspan et al. (2013)	21
s1color	Feigenspan et al. (2013)	22
s1color performance	Feigenspan et al. (2013)	22
s1ifdef	Feigenspan et al. (2013)	21
s1ifdef performance	Feigenspan et al. (2013)	21
s2color	Feigenspan et al. (2013)	22
s2color performance	Feigenspan et al. (2013)	22
s2ifdef	Feigenspan et al. (2013)	21
s2ifdef performance	Feigenspan et al. (2013)	21
nn25 precision	Herbold (2013)	44
nn25 recall	Herbold (2013)	44
nn25 success	Herbold (2013)	44
et defect count	Itkonen and Mäntylä (2013)	51
et effort	Itkonen and Mäntylä (2013)	51
tct defect count	Itkonen and Mäntylä (2013)	51

Table D.3: Sample sizes for the original data sets used in this study

Data set name	Reference	Sample size
tct effort	Itkonen and Mäntylä (2013)	51
collatedref dndr	Lee et al. (2013)	9
collatedref eclipse	Lee et al. (2013)	9
extractclass eclipse	Lee et al. (2013)	9
extractmethod dndr	Lee et al. (2013)	9
extractmethod eclipse	Lee et al. (2013)	9
movemethod dndr	Lee et al. (2013)	9
movemethod eclipse	Lee et al. (2013)	9
coc81effort	Minku and Yao (2013)	124
desharnaiseffort	Minku and Yao (2013)	77
nasa93effort	Minku and Yao (2013)	93
nasaeffort	Minku and Yao (2013)	60
sdreffort	Minku and Yao (2013)	12
aeeem tca	Nam et al. (2013)	20
aeeem tcaplus	Nam et al. (2013)	20
relink tca	Nam et al. (2013)	7
relink tcaplus	Nam et al. (2013)	7
new	Parnin et al. (2013)	20
old	Parnin et al. (2013)	16
initial	Sawadsky et al. (2013)	9
optimized	Sawadsky et al. (2013)	9
hybrid defect	Shar et al. (2013)	10
hybrid fault	Shar et al. (2013)	10
hybrid precision	Shar et al. (2013)	10
static defect	Shar et al. (2013)	10
static fault	Shar et al. (2013)	10
static precision	Shar et al. (2013)	10
kitchenham mlp meanBest	Song et al. (2013)	135
kitchenham mlp meanDflt	Song et al. (2013)	135
kitchenham mlp meanWorst	Song et al. (2013)	135
eclipse fmeasure	Wang et al. (2013)	10
eclipse precision	Wang et al. (2013)	10
eclipse recall	Wang et al. (2013)	10
mfie fmeasure	Wang et al. (2013)	10
mfie precision	Wang et al. (2013)	10
mfie recall	Wang et al. (2013)	10

D.2 After Outlier Detection and Removal

Table D.4: Sample sizes for the data sets from which outliers were removed.

Data set name	Reference	Sample size
jEdit logicalLow	Bavota et al. (2013)	124
jEdit semanticLow	Bavota et al. (2013)	121
jEdit structuralLow	Bavota et al. (2013)	122
m1ifdef	Feigenspan et al. (2013)	20
m1ifdef performance	Feigenspan et al. (2013)	18
m2ifdef	Feigenspan et al. (2013)	20
m2ifdef performance	Feigenspan et al. (2013)	19
m3ifdef performance	Feigenspan et al. (2013)	19
s1ifdef	Feigenspan et al. (2013)	17
extractclass eclipse	Lee et al. (2013)	8
movemethod dndr	Lee et al. (2013)	8
coc81effort	Minku and Yao (2013)	114
desharnaiseffort	Minku and Yao (2013)	76
nasa93effort	Minku and Yao (2013)	85
nasaeffort	Minku and Yao (2013)	52
sdreffort	Minku and Yao (2013)	10
new	Parnin et al. (2013)	19
old	Parnin et al. (2013)	15
static defect	Shar et al. (2013)	9
kitchenham mlp meanBest	Song et al. (2013)	125
kitchenham mlp meanDflt	Song et al. (2013)	125
kitchenham mlp meanWorst	Song et al. (2013)	124
eclipse fmeasure	Wang et al. (2013)	8
eclipse recall	Wang et al. (2013)	8

E Pipeline

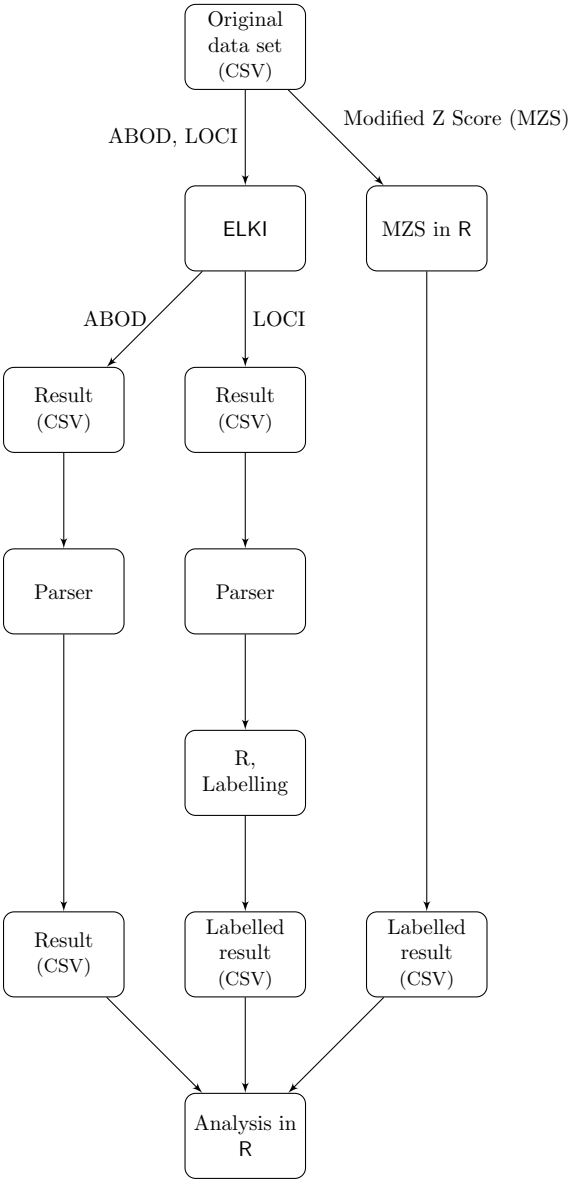


Figure E.1: Flowchart showing the steps involved in the pipeline. *Analysis in R* involves creating all the artifacts mentioned in Section 3.2.

F Papers Documenting Outliers

Table F.5: Papers that document the presence of outliers and outlier detection. *Used in experiment* refers to if the paper was part of the application of outliers in Section 5.1

Paper	Documents outliers	Outlier detection	Used in experiment	Reference	Comment
Are test cases needed? Replicated comparison between exploratory and test-case-based software testing	Yes	Yes	Yes	Itkonen and Mäntylä (2013)	Finds outliers and mentions that they include the outliers in the analysis.
Adoption and use of Java generics	Yes	No	Yes	Parnin et al. (2013)	Mentions outlier but does not describe how they were detected or if they exclude or include them.
Effectiveness for detecting faults within and outside the scope of testing techniques: an independent replication	Yes	No	No	Apa et al. (2014)	Does not mention if they include or exclude the outliers found.
A replicated quasi-experimental study on the influence of personality and team climate in software development	Yes	No	No	Gómez and Acuña (2013)	Does not mention how outlier detection is conducted just mentions that the "analyst" is responsible for identifying possible outliers.
Using tabu search to configure support vector regression for effort estimation	Yes	No	No	Corazza et al. (2013)	Mention that outliers can be a problem problems in some cases.
Do background colors improve program comprehension in the #ifdef hell?	Yes	No	Yes	Feigenspan et al. (2013)	Uses a box-plot but does not say if they remove the data points or not.

Table F.5: Papers that document the presence of outliers and outlier detection. *Used in experiment* refers to if the paper was part of the application of outliers in Section 5.1

Paper	Documents outliers	Outlier detection	Used in experiment	Reference	Comment
Can traditional fault prediction models be used for vulnerability prediction?	Yes	No	No	Shin and Williams (2013)	Mentions that they found outliers, but not with which method.
Building a second opinion: learning cross-company data	Yes	Yes	No	Kocaguneli et al. (2013)	Describes a reproducible process but does not show which data points they exclude.
Effort estimation of FLOSS projects: a study of the Linux kernel	Yes	No	No	Capiluppi and Izquierdo-Cortázar (2013)	Determining if a point is an outlier by using manual inspection.
Mining SQL injection and cross site scripting vulnerabilities using hybrid program analysis	Yes	No	Yes	Shar et al. (2013)	Uses a clustering algorithm, k-means clustering, but does not present any cutoff value.
An analysis of multi-objective evolutionary algorithms for training ensemble models based on different performance measures in software effort estimation	Yes	Yes	Yes	Minku and Yao (2013)	References an extended version of the paper where outlier detection is described using k-means clustering.
Are comprehensive quality models necessary for evaluating software quality?	Yes	Yes	No	Lochmann et al. (2013)	Calculates quartiles and presents thresholds.

Table F.5: Papers that document the presence of outliers and outlier detection. *Used in experiment* refers to if the paper was part of the application of outliers in Section 5.1

Paper	Documents outliers	Outlier detection	Used in experiment	Reference	Comment
The impact of parameter tuning on software effort estimation using learning machines	Yes	No	Yes	Song et al. (2013)	Mentions that outlier detection could be done as future work.
Human performance regression testing	Yes	No	No	Swearngin et al. (2013)	Does 5 runs of a experiment in order to lower the probability for any outliers present.
How, and why, process metrics are better	Yes	No	No	Rahman and Devanbu (2013)	They say that they see alot of outliers in the proximity to a box-plot but they do not elaboate on this in the text.

G R Code for the Replication of Analysis

You will find R code for the replication of the analysis in Section 5.2 in our repository[8] in the folder *report/appendix*.

[8] `https://github.com/linqcan/odser2014`

References

Ian Sommerville. *Software Engineering: (Update) (8th Edition) (International Computer Science)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2006. ISBN 0321313798.

Andrew J Ko, Thomas D LaToza, and Margaret M Burnett. A practical guide to controlled experiments of software engineering tools with human participants. *Empirical Software Engineering*, pages 1–32, 2013.

Hans-Peter Kriegel, Arthur Zimek, et al. Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 444–452. ACM, 2008.

Andrew Brooks, Marc Roper, Murray Wood, John Daly, and James Miller. Replication's role in software engineering. In *Guide to advanced empirical software engineering*, pages 365–379. Springer, 2008.

Jason W Osborne and Amy Overbay. The power of outliers (and why researchers should always check for them). *Practical assessment, research & evaluation*, 9(6):1–12, 2004.

Victoria J Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.

Peter J Rousseeuw and Bert C Van Zomeren. Unmasking multivariate outliers and leverage points. *Journal of the American Statistical Association*, 85(411): 633–639, 1990.

Yeong-Seok Seo and Doo-Hwan Bae. On the value of outlier elimination on software effort estimation research. *Empirical Software Engineering*, 18(4): 659–698, 2013.

Ke-Hai Yuan and Peter M Bentler. Effect of outliers on estimators and tests in covariance structure analysis. *British Journal of Mathematical and Statistical Psychology*, 54(1):161–175, 2001.

David N Card, Frank E. Mc Garry, and Gerald T. Page. Evaluating software engineering technologies. *Software Engineering, IEEE Transactions on*, (7): 845–851, 1987.

Victor R Basili, Forrest Shull, and Filippo Lanubile. Building knowledge through families of experiments. *Software Engineering, IEEE Transactions on*, 25(4): 456–473, 1999.

Forrest J Shull, Jeffrey C Carver, Sira Vegas, and Natalia Juristo. The role of replications in empirical software engineering. *Empirical Software Engineering*, 13(2):211–218, 2008.

Natalia Juristo and Sira Vegas. The role of non-exact replications in software engineering experiments. *Empirical Software Engineering*, 16(3):295–324, 2011.

Dag IK Sjøberg, Jo Erskine Hannay, Ove Hansen, Vigdis By Kampenes, Amela Karahasanovic, N-K Liborg, and Anette C Rekdal. A survey of controlled experiments in software engineering. *Software Engineering, IEEE Transactions on*, 31(9):733–753, 2005.

R Murray Lindsay and Andrew SC Ehrenberg. The design of replicated studies. *The American Statistician*, 47(3):217–228, 1993.

Tin W Tan, Joo C Tong, Asif M Khan, Mark de Silva, Kuan S Lim, and Shoba Ranganathan. Advancing standards for bioinformatics activities: persistence, reproducibility, disambiguation and minimum information about a bioinformatics investigation (miabi). *BMC genomics*, 11(Suppl 4):S27, 2010.

Robert Gentleman. Reproducible research: A bioinformatics case study. 2004.

Black Pyrkosz Alexis Preeyanon, Likit and C. Titus Brown. Reproducible bioinformatics research for biologists. In Victoria Stodden, Friedrich Leisch, and Roger D. Peng, editors, *Implementing Reproducible Computational Research*. Chapman and Hall/CRC, 2014. URL `http://www.crcpress.com/product/isbn/9781466561595`. ISBN 978-1466561595.

OS Gómez, N Juristo, and S Vegas. Replication, reproduction and re-analysis: Three ways for verifying experimental findings. In *Proceedings of the 1st international workshop on replication in empirical software engineering research (RESER 2010), Cape Town, South Africa*, 2010.

Claes Wohlin, Per Runeson, Martin Hst, Magnus C Ohlsson, Bjrn Regnell, and Anders Wessln. *Experimentation in software engineering*. Springer Publishing Company, Incorporated, 2012.

Per Runeson and Martin Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical software engineering*, 14(2): 131–164, 2009.

Barbara A Kitchenham and Stuart Charters. Guidelines for performing systematic literature reviews in software engineering. 2007.

Nornadiah Mohd Razali and Yap Bee Wah. Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. *Journal of Statistical Modeling and Analytics*, 2(1):21–33, 2011.

Gabriele Bavota, Bogdan Dit, Rocco Oliveto, Massimiliano Di Penta, Denys Poshyvanyk, and Andrea De Lucia. An empirical study on the developers' perception of software coupling. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 692–701. IEEE Press, 2013.

Juha Itkonen and Mika V Mäntylä. Are test cases needed? replicated comparison between exploratory and test-case-based software testing. *Empirical Software Engineering*, pages 1–40, 2013.

Lwin Khin Shar, Hee Beng Kuan Tan, and Lionel C Briand. Mining sql injection and cross site scripting vulnerabilities using hybrid program analysis. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 642–651. IEEE Press, 2013.

Andrea Arcuri and Gordon Fraser. Parameter tuning or default values? an empirical investigation in search-based software engineering. *Empirical Software Engineering*, 18(3):594–623, 2013.

Leandro L Minku and Xin Yao. An analysis of multi-objective evolutionary algorithms for training ensemble models based on different performance measures in software effort estimation. In *Proceedings of the 9th International Conference on Predictive Models in Software Engineering*, page 8. ACM, 2013.

Jinshui Wang, Xin Peng, Zhenchang Xing, and Wenyun Zhao. Improving feature location practice with multi-faceted interactive exploration. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 762–771. IEEE Press, 2013.

Jaechang Nam, Sinno Jialin Pan, and Sunghun Kim. Transfer defect learning. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 382–391. IEEE Press, 2013.

Janet Feigenspan, Christian Kästner, Sven Apel, Jörg Liebig, Michael Schulze, Raimund Dachselt, Maria Papendieck, Thomas Leich, and Gunter Saake. Do background colors improve program comprehension in the# ifdef hell? *Empirical Software Engineering*, 18(4):699–745, 2013.

Nicholas Sawadsky, Gail C Murphy, and Rahul Jiresal. Reverb: recommending code-related web pages. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 812–821. IEEE Press, 2013.

Yun Young Lee, Nicholas Chen, and Ralph E Johnson. Drag-and-drop refactoring: intuitive and efficient program transformation. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 23–32. IEEE Press, 2013.

Liyan Song, Leandro L Minku, and Xin Yao. The impact of parameter tuning on software effort estimation using learning machines. In *Proceedings of the 9th International Conference on Predictive Models in Software Engineering*, page 9. ACM, 2013.

Chris Parnin, Christian Bird, and Emerson Murphy-Hill. Adoption and use of java generics. *Empirical Software Engineering*, 18(6):1047–1089, 2013.

Steffen Herbold. Training data selection for cross-project defect prediction. In *Proceedings of the 9th International Conference on Predictive Models in Software Engineering*, page 6. ACM, 2013.

Francisco Augusto Alcaraz Garcia. Tests to identify outliers in data series. *Pontifical Catholic University of Rio de Janeiro, Industrial Engineering Department, Rio de Janeiro, Brazil*, 2012.

Spiros Papadimitriou, Hiroyuki Kitagawa, Phillip B Gibbons, and Christos Faloutsos. Loci: Fast outlier detection using the local correlation integral. In *Data Engineering, 2003. Proceedings. 19th International Conference on*, pages 315–326. IEEE, 2003.

IBM SPSS. IBM SPSS modified z score, 2007. URL http://pic.dhe.ibm.com/infocenter/spssas/v1r0m0/index.jsp?topic=%2Fcom.ibm.spss.analyticcatalyst.help%2Fanalytic_catalyst%2Fmodified_z.html.

Boris Iglewicz and David C Hoaglin. *How to detect and handle outliers*, volume 16. ASQC Quality Press Milwaukee (Wisconsin), 1993.

Elke Achtert, Hans-Peter Kriegel, Erich Schubert, and Arthur Zimek. Interactive data mining with 3d-parallel-coordinate-trees. In *SIGMOD Conference*, pages 1009–1012, 2013.

Julius Davies, Daniel M German, Michael W Godfrey, and Abram Hindle. Software bertillonage. *Empirical Software Engineering*, 18(6):1195–1237, 2013.

Cecilia Apa, Oscar Dieste, EdisonG. Espinosa G., and EfranR. Fonseca C. Effectiveness for detecting faults within and outside the scope of testing techniques: an independent replication. *Empirical Software Engineering*, 19(2): 378–417, 2014. ISSN 1382-3256. doi: 10.1007/s10664-013-9267-7. URL http://dx.doi.org/10.1007/s10664-013-9267-7.

Zack Coker and Munawar Hafiz. Program transformations to fix c integers. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 792–801. IEEE Press, 2013.

Ahmet Okutan and Olcay Taner Yıldız. Software defect prediction using bayesian networks. *Empirical Software Engineering*, 19(1):154–181, 2014.

Marta N Gómez and Silvia T Acuña. A replicated quasi-experimental study on the influence of personality and team climate in software development. *Empirical Software Engineering*, pages 1–35, 2013.

Dennis Pagano and Walid Maalej. How do open source communities blog? *Empirical Software Engineering*, 18(6):1090–1124, 2013.

Oscar Callaú, Romain Robbes, Éric Tanter, and David Röthlisberger. How (and why) developers use the dynamic features of programming languages: the case of smalltalk. *Empirical Software Engineering*, 18(6):1156–1194, 2013.

Heiko Koziolek, Bastian Schlich, Steffen Becker, and Michael Hauck. Performance and reliability prediction for evolving service-oriented software systems. *Empirical Software Engineering*, 18(4):746–790, 2013.

Robert M Bell, Thomas J Ostrand, and Elaine J Weyuker. The limited impact of individual developer data on software defect prediction. *Empirical Software Engineering*, 18(3):478–505, 2013.

Gursimran S Walia and Jeffrey C Carver. Using error abstraction and classification to improve requirement quality: conclusions from a family of four empirical studies. *Empirical Software Engineering*, 18(4):625–658, 2013.

Parastoo Mohagheghi, Wasif Gilani, Alin Stefanescu, and Miguel A Fernandez. An empirical study of the state of the practice and acceptance of model-driven engineering in four industrial cases. *Empirical Software Engineering*, 18(1): 89–116, 2013.

Anna Corazza, Sergio Di Martino, Filomena Ferrucci, Carmine Gravino, Federica Sarro, and Emilia Mendes. Using tabu search to configure support vector regression for effort estimation. *Empirical Software Engineering*, 18(3):506–546, 2013.

Shauvik Roy Choudhary, Mukul R Prasad, and Alessandro Orso. X-pert: accurate identification of cross-browser issues in web applications. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 702–711. IEEE Press, 2013.

Yonghee Shin and Laurie Williams. Can traditional fault prediction models be used for vulnerability prediction? *Empirical Software Engineering*, 18(1): 25–59, 2013.

Ekrem Kocaguneli, Bojan Cukic, Tim Menzies, and Huihua Lu. Building a second opinion: learning cross-company data. In *Proceedings of the 9th International Conference on Predictive Models in Software Engineering*, page 12. ACM, 2013.

Andrea Capiluppi and Daniel Izquierdo-Cortázar. Effort estimation of floss projects: a study of the linux kernel. *Empirical Software Engineering*, 18(1): 60–88, 2013.

Domenico Cotroneo, Roberto Pietrantuono, and Stefano Russo. A learning-based method for combining testing techniques. In *Software Engineering (ICSE), 2013 35th International Conference on*, pages 142–151. IEEE, 2013.

Vicente Lustosa Neto, Roberta Coelho, Larissa Leite, Dalton S Guerrero, and Andrea P Mendonça. Popt: a problem-oriented programming and testing approach for novice students. In *Software Engineering (ICSE), 2013 35th International Conference on*, pages 1099–1108. IEEE, 2013.

Amanda Swearngin, Myra B Cohen, Bonnie E John, and Rachel KE Bellamy. Human performance regression testing. In *Software Engineering (ICSE), 2013 35th International Conference on*, pages 152–161. IEEE, 2013.

Suresh Thummalapenta, K Vasanta Lakshmi, Saurabh Sinha, Nishant Sinha, and Satish Chandra. Guided test generation for web applications. In *Software Engineering (ICSE), 2013 35th International Conference on*, pages 162–171. IEEE, 2013.

Mohammad Mahdi Hassan and James H Andrews. Comparing multi-point stride coverage and dataflow coverage. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 172–181. IEEE Press, 2013.

Hongyu Zhang, Liang Gong, and Steve Versteeg. Predicting bug-fixing time: an empirical study of commercial software projects. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 1042–1051. IEEE Press, 2013a.

Dongsun Kim, Jaechang Nam, Jaewoo Song, and Sunghun Kim. Automatic patch generation learned from human-written patches. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 802–811. IEEE Press, 2013.

Lingming Zhang, Dan Hao, Lu Zhang, Gregg Rothermel, and Hong Mei. Bridging the gap between the total and additional test-case prioritization strategies. In *Software Engineering (ICSE), 2013 35th International Conference on*, pages 192–201. IEEE, 2013b.

Nadia Polikarpova, Carlo A Furia, Yu Pei, Yi Wei, and Bertrand Meyer. What good are strong specifications? In *Proceedings of the 2013 International Conference on Software Engineering*, pages 262–271. IEEE Press, 2013.

Marcel Böhme, Bruno C d S Oliveira, and Abhik Roychoudhury. Partition-based regression verification. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 302–311. IEEE Press, 2013.

Annibale Panichella, Bogdan Dit, Rocco Oliveto, Massimiliano Di Penta, Denys Poshyvanyk, and Andrea De Lucia. How to effectively use topic models for software engineering tasks? an approach based on genetic algorithms. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 522–531. IEEE Press, 2013.

Foyzur Rahman and Premkumar Devanbu. How, and why, process metrics are better. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 432–441. IEEE Press, 2013.

Filomena Ferrucci, Mark Harman, Jian Ren, and Federica Sarro. Not going to take this anymore: multi-objective overtime planning for software engineering projects. In *Proceedings of the 2013 International Conference on Software Engineering*, pages 462–471. IEEE Press, 2013.

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *ACM Sigmod Record*, volume 29, pages 93–104. ACM, 2000.

Peter Filzmoser, Ricardo Maronna, and Mark Werner. Outlier identification in high dimensions. *Computational Statistics & Data Analysis*, 52(3):1694–1711, 2008.

Stephen D Bay and Mark Schwabacher. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 29–38. ACM, 2003.

Klaus Lochmann, Jasmin Ramadani, and Stefan Wagner. Are comprehensive quality models necessary for evaluating software quality? In *Proceedings of the 9th International Conference on Predictive Models in Software Engineering*, page 3. ACM, 2013.