It's Not a Bug, It's a Feature: How Misclassification Impacts Bug Prediction

Kim Herzig,

Microsoft Research
Cambridge United Kingdom
kimh@microsoft.com

Sascha Just§, Andreas Zeller§

Saarland University
Saarbrücken, Germany
just@st.cs.uni-saarland.de
zeller@cs.uni-saarland.de

Abstract: *This submission presents work submitted and accepted at the International onference on Software Engineering in 2013* [Hj2013]. In empirical software engineering, it has become common to mine historic data to detect where bugs have occurred in the past, or to predict where they will occur in the future. The accuracy of such models depends on the quality of the data. For example, defect prediction models rely on the accuracy of historic data, such as bug reports. Bug reports that refer to any other than corrective development activities may cause code artefacts to be falsely marked as defective. This may have severe consequences for the resulting models and its accuracy. Earlier studies raised concerns about bug reports referring to error unrelated development activities. But how often does such misclassification occur? Further, does it actually impact analysis and prediction models? These are the questions we addressed in this paper. In a manual examination of more than 7,000 issue reports from five open-source projects, we found 33.8% of all bug reports to be misclassified threatening bug prediction models, confusing bugs and features: On average, 39% of files marked as defective actually never had a bug. The presentation will cover causes for issue report misclassification and the result of our study (some newer results not in the paper).

1 Talk Summary

Empirical studies are threatened by the quality of data analyzed and interpreted. A commong task in empirical software engineering is to separate defective from defect free code artifacts, e.g. to build defect prediction models, which relies on historic bug data. The majority of issue reports are classified as *bugs*—that is, requests for corrective code maintenance—and suggest that code changes resolving these issues should be considered as bug fixes and that the associtaed code artifacts should be considered as defective. However, it remains unclear how reliable issue report classifications are. In 2008, Antoniol et al. [Aa2008] raised concerns about bug reports referring to error unrelated development activities. If such mix-ups (which mostly stem from issue reporters and developers interpreting “bug” differently) occurred frequently and systematically they would introduce bias in data mining models threatening the external validity of any

study that builds on such data: Predicting the most error-prone files, for instance, may actually yield files most prone to new features. But how often does such misclassification occur? And does it actually bias analysis and prediction? Our study targeted the following research questions:

RQ1) Do bug databases contain data noise due to issue report misclassification, and how much?

RQ2) Which percentage of issue reports associated with a category was marked as misclassified? Which category do these misclassified reports actually belong to?

RQ3) What is the impact of misclassified issue reports when mapping issue reports to source code changes?

RQ4) How does bug mapping bias introduced by misclassified issue reports impact the TOP 5%, 10%, 15%, 20% of most defect prone source files?

To answer these research questions, we manually inspected and re-classified more than 7,000 issue reports from five open-source Java projects developed by the Apache and Mozilla foundations (we will give more details about the classification process in the talk).

Comparing the re-classified issue categories with the orginal issue report type as stated in the bug database showed that over 40% of all issue reports in the analyzed bug databases were associate to inaccurate issue report types (RQ1). Concentrating on bug reports, we showed that over 33% of all bug reports are misclassified (RQ2). During the talk, we show details of the analysis and discuss sources of misclassification, many of which refer to the fact that bug databases and bug reports provide communication platforms for different stakeholder, e.g. enginners and customers, wich have a very different perception of issues and a very different level of technical understanding.

Estimating the impact of these miscalssifactions on mappings between actual code fixes and their changed code artifacts, we show that on average 39% of all files originally marked as defective actually never had a bug (RQ3). This impact on file mapping threatens bug count and bug prediction models. In fact, we show that when identifying the top 10% most defect-prone source files, 16% to 40% of these files do not belong in this category because of issue report misclassification.

The original published paper this talk is covering can be found on the publisher's website: http://dl.acm.org/citation.cfm?id=2486788.2486840

References

[Hj2013] Herzig K.; Just S. und Zeller A. (2013): It's not a Bug, It's a Feature: How Misclassification Impacts Bug Prediction. In: *Proceedings of the 2013 International Conference on Software Engineering*. IEEE Press. S. 392--401.