Paper Two Summary

Matthew Neal, Joseph Sankar, and Alexander Sobran

September 11, 2015

Reference

Arisholm et al [1] listed below.

Important Keywords

Fault-proneness The number of defects detected in a software-component. A measurement of the likelihood of software or software component error.

Object-oriented (OO) code measures Measurement of the structural properties of the source code. For example cyclomatic complexity and LOC.

Delta measures Measurement of the amount of change (churn) in a file between two successive releases.

Process measures Measures from a configuration management system. Includes developer experience, the number of developers that have edited a file, the number of faults in the previous releases, the number of line added, the number of lines removed.

Feature Extraction

Motivational statements The motivation of this study was to compare multiple feature sets, machine learning techniques, and evaluation criteria to determine the most useful combination at predicting fault-proneness in a continually changing codebase. The authors leverage both structural measurements inherent to the code and process measurements derived from configuration management systems. Due to the lack of evidence for the economic viability of fault-proneness detection, the authors introduce a novel evaluation criteria, cost-effectiveness (CE) measure, which incorporates the verification costs of detected faults.

Patterns The authors provide an overview of a number of different techniques and different metric sets in an attempt to prove which provides the best results across comparison techniques. The results seem to show that the

ROC curve combined with cost effectiveness comparisons provide more insight into the performance of techniques and matrices as compared to using the confusion matrix approach.

Future Work The authors used the default parameters for the statistical techniques they employed in the study. They have stated that they hope to tweak the parameters and observe the results to see if there are optimizations that can be made while at the same time preventing overfitting. The authors also stated they are going to work on a large-scale evaluation of the costs and benefits of various prediction models in the COS project.

Informative Visualizations Figure four on page 12 of the paper provides a really interesting visualization of the distribution of cost efficiency of prediction models using Process and Object Oriented metrics. The figure provides two lines, one for OO and one for Process metrics, but they also provide clouds around those lines that demarcate the 25th and 75th percentiles. The authors use this figure to show the fact that the performance of OO metrics based models is poor compared with Process. The baseline for this study was a line such that $y = x$ so as is shown in the figure, OO has very close to the baseline performance while OO outperforms the baseline by a significant margin.

Possible Improvements

- In the Introduction section, the authors reference a paper published in 1989 to cite a percentage of effort spent on testing. Is there a more recent paper the authors could have chosen to cite? If not, why quote a figure that is 26 years old and quite possibly outdated?

- While the way the authors combined data in an attempt to consolidate it to a smaller number of tables was very clever, it produces tables that are very difficult to interpret. They have split the data into a standard set of stats (mean, median, Q1, Q3) on half of the table and then created a diagonal split on the other half of the table into two sets of data, one for effect size and the other for the Wilcoxon test data. It seems that they had so much data to report that it would have been impossible to report it in the space they had available without this type of table, but perhaps some further graphical representation would have been preferable with an appendix of the tables and data sets split out into more easily decipherable tables.

- The authors mentioned that one threat to the validity of the study was that the cost of making the measures available and collecting them was not accounted for. It seems that such calculations should be taken into account. Additionally, they note the Process metric set has a high cost of data reporting and collection but was found to be the most cost-effective. It is hard to say if Process is indeed the most cost-effective, since all cost

measures were not taken into consideration. A future study should include as many cost factors as possible when determining the cost-effectiveness of a particular method or metric.

Connection to Other Papers

Posnett et al [3] cited this paper as well as Menzies et al [2] as evidence that prediction models designed for the aggregated level have poor performance at the disaggregated level. But moreover the Posnett paper uses this paper as a methodological foundation for the way that they went about using the ROC curve and Cost Effectiveness analysis that they used in their study.

References

[1] Erik Arisholm, Lionel C. Briand, and Eivind B. Johannessen. A systematic and comprehensive investigation of methods to build and evaluate fault prediction models. *J. Syst. Softw.*, 83(1):2–17, January 2010.

[2] Tim Menzies, Zach Milton, Burak Turhan, Bojan Cukic, Yue Jiang, and Ayşe Bener. Defect prediction from static code features: current results, limitations, new approaches. *Automated Software Engineering*, 17(4):375–407, 2010.

[3] D. Posnett, V. Filkov, and P. Devanbu. Ecological inference in empirical software engineering. In *Automated Software Engineering (ASE), 2011 26th IEEE/ACM International Conference on*, pages 362–371, Nov 2011.