Paper Two Summary

Matthew Neal, Joseph Sankar, and Alexander Sobran

September 8, 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.

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 authors mentioned that most previous studies only examined the structure of the code when constructing fault-proneness prediction models. In the few studies that did not, they failed to systematically measure the cost-effectiveness of the models with factors with different data collection costs. Very few studies have looked at the methods and criteria for evaluating the models. Finally, it is very difficult to draw specific conclusions using confusion matrices since they do not provide any insight into determining the cost-effectiveness of the models under test.

Motivational Statements The motivation was to provide a systematic methodology for comparing modeling techniques. The authors state that little to no effort was put forth to prior to this paper in that area.

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 a good deal of 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 that 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.

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.