**Optimized Data Processing Method and Classification Method for Cross-project Bugs Prediction**

Xinyi He, Zhenjie Jia

# 1.Introduction

Software has an increasingly influence on the world. It is very essential to build durable software with low cost. Statistically, the effort in finding and fixing bugs in a software will consume near 80% of the budget of a software development [1]. Therefore the defect prediction is very important to improve the software quality and reduce the effort to find the bugs as the size of the software is getting larger.

The software defect or bug prediction can be seen as a classification problem of whether a class in the software has bugs or not. The process of software defect predication requires the following steps:

1. Feature extraction: To represent the status of software of different languages, we need software’s feature matrix to be used as the train set.
2. Model training: Using different ML methods like random forest, logistic regression, neural networks, etc to train the model.
3. Results comparison: Testing on different software or same software’s latest version.

This field of research is now facing the following challenges:

1. The lack of the dataset. Only a few open source software have public and clear bug tracker that can be used to generate the data set.
2. The dataset is imbalanced. The number of data with “bugs” label in data set is about 15%, while the rest of the data in a dataset all have the “zero bugs” label.
3. Different projects’ feature metrics are different.
4. The data in the dataset vary a lot. Many features are zero, some features’ numbers are big while some are extremely small

This project focus on

1) discussing some ML techniques.

2) the challenge of software defect predication.

3) the analysis of the raw dataset and the comparison of the dataset after different data preprocessing methods.

4) trying different Machine Learning methods on the software defect model and the comparison of the results.

The project goal is find an improved process to predict the bugs in the software.

# 2. Related works

Describe other peoples’ methods and result on the same dataset above

Zhang Feng, etc used spectral Random Forest classifier which calculate the laplacian matrix and perform the eigendecomposition on the matrix to normalize the matrix. They tested it in *D'Ambros’ dataset* and achieved 81%, 78% of accuracy within projects and 87%, 71% of accuracy when crossing the projects for JDT and PDE. [2].

CesarCouto, etc built a model that will use some alarm threshold functions to select the variations in the metrics that may have contributed to the occurrence of defects. They remove the classes with zero defects and got 61% precision, 53% recall for JDT, 27% precision 54% of recall for PDE. [3]

The results of Haidar Osman, Mohammad Ghafari, and Oscar Nierstrasz reveal that tuning model hyperparameters has a statistically signiﬁcant positive effect on the prediction accuracy of the models. The prediction accuracy is improved by up to 20% in KNN and by up to 10% in SVM with the dataset<http://bug.inf.usi.ch/download.php>.[4]

The paper by Yasutaka Kamei, Shinsuke Matsumoto, Akito Monden, Ken-ichi Matsumoto, Bram Adams and Ahmed E. Hassan shows that package-level predictions are not more effective than ﬁle-level predictions and the effectiveness of package-level predictions can improve if we perform our predictions at the ﬁle-level then lift it to the package-level instead of collecting all metrics at the package-level. However the new model still does not outperform ﬁle-level predictions when considering the quality assurance efforts.[5]

The paper by Shivkumar Shivaji, E. James Whitehead and Jr., Ram Akella proposes a feature selection technique applicable to classiﬁcation-based bug prediction.[6]

# 3. Machine Learning algorithms

## 3.1 briefly introduce the classifier we will use and why we will use them:

the random forest, the logistic regression, the neural network

These are top three classifier after trying all the classifiers.

# 4. Experiments

## 4.1 the dataset used in the software bug prediction:

The dataset used in this research includes:

1. Dataset from NASA: KC1, KC2, PC1, CM1, JM1: <http://promise.site.uottawa.ca/SERepository/datasets-page.html>
2. Dataset from *Marco D'Ambros, Michele Lanza, Romain Robbes* : <http://bug.inf.usi.ch/index.php>
3. Dataset from Zimmermann : <https://www.st.cs.uni-saarland.de/softevo/bug-data/eclipse/>

We used the first two datasets to test the model on different software projects and the last dataset to test the model on different versions of the same software.

## 4.2 the different preprocessing methods

Over sample

Under sample

Syn sample

PCA

Standardization

And describe how the data is distributed after these methods.

## 4.3 trying different ML methods and describe the result

## 4.4 compare the result between preprocessed data and raw data?

# 5 conclusion

# References

|  |  |
| --- | --- |
| [1] | G. Tassey, "The economic impacts of inadequate infrastructure," Technical Report Planning Report, Vols. 02-3, May, 2002. |
| [2] | Q. Z. Y. Z. A. E. H. Feng Zhang, "Cross-project Defect Prediction Using a," in IEEE/ACM 38th IEEE International Conference on Software Engineering, 2016 . |
| [3] | b. P. P. M. T. V. R. S. B. Cesar Coutoa, "Predicting software defects with causality tests," The Journal of Systems and Software, vol. 24, p. 14, 2014. |

[4] Osman, Haidar , M. Ghafari , and O. Nierstrasz . "Hyperparameter Optimization to Improve Bug Prediction Accuracy." MaLTeSQuE IEEE, 2017.

[5] Kamei, Yasutaka , et al. " [IEEE 2010 IEEE 26th International Conference on Software Maintenance (ICSM) - Timi oara, Romania (2010.09.12-2010.09.18)] 2010 IEEE International Conference on Software Maintenance - Revisiting common bug prediction findings using effort-aware models." Proc of IEEE International Conference on Softw Maintenance 2010:1-10.

[6] Shivaji, Shivkumar , et al. "Reducing Features to Improve Bug Prediction." IEEE/ACM International Conference on Automated Software Engineering IEEE Computer Society, 2009.

# 