Bug Triage with Natural Language Processing and Master-Worker Algorithm

Ning Yang, Frank Russo, Ramya Raju, Carlo Clarke, Charles C. Tappert and Avery Leider
Seidenberg School of Computer Science and Information Systems, Pace University
Pleasantville, NY 10570, USA
Email: {ny00685p, fr50558n, rr35976p, cc86686n, ctappert, aleider}@pace.edu

***Abstract*—All software written even today has bugs or defects. It is critical to fix bugs as quickly and efficiently as possible. Triaging bugs can be time consuming, especially on larger teams or with larger projects. The process of assigning bugs to the proper developer is a task that can easily be automated and is a good exercise for using Machine Learning concepts. We will be studying sample data from the Google Chromium project using Machine Learning to analyze bugs by title and description to determine the best owner of a defect. We will use Natural Language Processing to analyze the bug title and description looking for patterns to determine the correct developers to assign to bugs. We will also determine if the Master-Worker Algorithm will improve the models to successfully improve Triage times.**

***Index Terms*—Machine Learning, Bug Triage, Natural Language Processing, Master-Worker algorithm, Python.#.**

I. Introduction

Machine learning (ML) is a set of statistical techniques for problem solving. It involves applying a set of training data against models. The models are then used to predict future results against new data coming into a system. There are different types of ML, including supervised, unsupervised, semi-supervised and reinforcement learning. The flow of a typical machine learning algorithm can be seen in Figure 1.

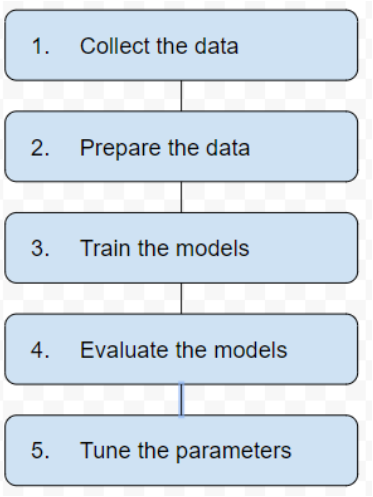

Fig. 1. Machine Learning flow.

Natural Language Processing (NLP) is considered a discipline within Artificial Intelligence (AI). It deals with computer processing of large data sets containing natural human language. It is a way analyze text based on a set of theories and a set of technologies. [10] By this it is meant that there are multiple techniques from which to choose to analyze language. The two techniques we will focus on are word2vec and Long-short-term-memory (LSTM).

NLP and ML can be used together to effectively assign bugs to developers alleviating the need for manual processing of bugs.

Thanks to the IBM Faculty Award to Dr. Tappert that supports this research.

II. Literature Review

We will be reviewing various papers on Bug Triage, Machine Learning, and Natural Language Processing.

A. Subsection example

Subsection here.

III. Project Requirements

References

[1] G. Jeong, S. Kim, and T. Zimmermann, "Improving bug triage with bug tossing graphs," in *Proceedings of the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, 2009, pp. 111–120.

[2] L. Tan, C. Liu, Z. Li, X. Wang, Y. Zhou, and C. Zhai, "Bug characteristics in open source software," *Empirical software engineering*, vol. 19, no. 6, pp. 1665–1705, 2014.

[3] S. Mani, A. Sankaran, and R. Aralikatte, "Deeptriage: Exploring the effectiveness of deep learning for bug triaging," in *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, 2019, pp. 171–179.

[4] H. Hu, H. Zhang, J. Xuan, and W. Sun, "Effective bug triage based on historical bug-fix information," in *2014 IEEE 25th International Symposium on Software Reliability Engineering*. IEEE, 2014, pp. 122–132.

[5] N. Sreenivas and S. Saritha, "Enhancement towards efficient bug triage with software data reducing methods," in *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*. IEEE, 2017, pp. 3649–3652.

[6] B. S. Neysiani and S. M. Babamir, "Improving performance of automatic duplicate bug reports detection using longest common sequence: Introducing new textual features for textual similarity detection," in *2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI)*. IEEE, 2019, pp. 378–383.

[7] C. Zhou, B. Li, X. Sun, and H. Guo, "Recognizing software bug-specific named entity in software bug repository," in *Proceedings of the 26th Conference on Program Comprehension*, 2018, pp. 108–119.

[8] M. R. Karim, A. Ihara, X. Yang, H. Iida, and K. Matsumoto, "Understanding key features of high-impact bug reports," in *2017 8th International Workshop on Empirical Software Engineering in Practice (IWESEP)*. IEEE, 2017, pp. 53–58.

[9] T. Zimmermann, R. Premraj, N. Bettenburg, S. Just, A. Schroter, and C. Weiss, "What makes a good bug report?" *IEEE Transactions on Software Engineering*, vol. 36, no. 5, pp. 618–643, 2010.
[10] E. D. Liddy, "Natural language processing," 2001.