An Ensemble of Vector Space Model and Gated Recurrent Unit for Defect Localization*

Yue Cai Zhu
McGill University
Montreal, Quebec
yue.c.zhu@mail.mcgill.ca

ABSTRACT

This paper provides a sample of a LaTeX document which conforms, somewhat loosely, to the formatting guidelines for ACM SIG Proceedings.

CCS CONCEPTS

• **Computing methodologies** → **Information extraction**; **Neural networks**; **Ensemble methods**; • **Software and its engineering** → **Software defect analysis**;

KEYWORDS

Defect Localization, Vector Space Model, Gated Recurrent Unit,Ensemble method

ACM Reference Format:
Yue Cai Zhu . 2018. An Ensemble of Vector Space Model and Gated Recurrent Unit for Defect Localization. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Bug localization is one of the important steps in software maintenance. Especially in issue triaging, where triager has to predict the potential defected components based on his understanding to the issue and to the system, and assign the issue to the right developer who has the required knowledge. Automating this step helps speed up bugs fixing and reduce costs in issue triaging. Lam *et al.* [2] proposed the state of the art approach to automate bug localization by ensemble rVSM[3] and DNN. Similar as most of the existed algorithms, this approach require source code information which is not always available to triagers or the QA team in large companies with code security requirement. Analyzing the code base dramatically increase the learning time of this kind of algorithm in large projects. Moreover, the fast evolution of source code further penalizes this approach that the predictive model has to be updated frequently. In this paper, we explore the possibility to predict defected components solely based on information learned from bug reports.

Recurrent neural network(RNN) has been proved to be efficient in capturing semantic as well as syntactic information from text to build statistical language models. But for source code, Hellendoorn *et al.* show that the RNN is not better than traditional IR method in language model building[1]. Bug report is different from common text and source code in that it is generally composed by common text and source code segment or error trace message. We believe that IR method and RNN can learn different information from bug reports and could complement each other. Thus the ensemble of these two could offer better performance than using only one of them.

The contribution of this paper is in two folds:

REFERENCES

[1] Vincent J Hellendoorn and Premkumar Devanbu. 2017. Are deep neural networks the best choice for modeling source code?. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*. ACM, 763–773.
[2] An Ngoc Lam, Anh Tuan Nguyen, Hoan Anh Nguyen, and Tien N Nguyen. 2017. Bug localization with combination of deep learning and information retrieval. In *Program Comprehension (ICPC), 2017 IEEE/ACM 25th International Conference on*. IEEE, 218–229.
[3] Jian Zhou, Hongyu Zhang, and David Lo. 2012. Where should the bugs be fixed?-more accurate information retrieval-based bug localization based on bug reports. In *Proceedings of the 34th International Conference on Software Engineering*. IEEE Press, 14–24.

*A course project for COMP767

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn