Journal of Computer Technology and Software

ISSN:2998-2383

Vol. 1, No. 2, 2022

Enhancing Fault Localization in Software Maintenance Using Natural Language Processing Techniques

Emily Johnson

Department of Electrical and Computer Engineering, University of California, Irvine, USA

pemilyjohson@gmail.com

Abstract: This paper presents a natural language processing-based defect localization methodology to identify and locate software defects in source code reviews. The methodology utilizes text mining and natural language processing techniques to extract defect-related information from software source code commit messages and bug reports. Through a series of experiments on a real-world dataset, we evaluate the effectiveness and efficiency of the proposed methodology in identifying and locating defects in software systems.

Keywords: Fault Localization; Natural Language Processing; Text Mining; Source Code Analysis.

1. Introduction

Fault localization is a crucial task in software maintenance, as it helps developers identify and locate defects in software systems efficiently. Conventionally, defect localization has been approached using static analysis techniques that analyze the source code directly [1]. However, these techniques often suffer from false positives and are unable to consider contextual information effectively [2]. To address these limitations, we propose a natural language processing-based defect localization methodology that utilizes text mining and natural language processing techniques to extract defect-related information from software source code commit messages and bug reports.

2. Background

In recent years, natural language processing (NLP) techniques have gained popularity in software engineering research, particularly for the purpose of defect localization [3][4]. These techniques have been successfully applied to extract meaningful information from source code commit messages and bug reports to improve defect localization effectiveness.

Compared with research on foreign rating systems, due to the complexity of Chinese, there are many non-standard words and terms in Chinese intelligent scoring systems, and there are no natural boundary markers between words. Therefore, there are still many difficulties to be solved in natural language processing for Chinese. Domestic scholars and experts have expended a lot of manpower and resources on the design of intelligent marking systems, but they have only achieved some results in basic theory. Among them, they have designed many algorithms to improve the accuracy of objective questions, but there is little research on the automatic review and scoring of subjective questions.

The primary goal of our research is to develop a comprehensive framework that combines NLP and text mining techniques to extract defect-related information from source code commit messages and bug reports effectively. The framework should be able to identify and locate defects with a high level of accuracy and efficiency, while considering contextual information explicitly.

3. Methodology

The proposed methodology consists of three stages: preprocessing, feature extraction, and defect localization. In the preprocessing stage, we clean and preprocess the source code commit messages and bug reports to remove any noise and prepare the data for further analysis. In the feature extraction stage, we utilize NLP techniques, such as part-of-speech tagging and named entity recognition, to extract meaningful information from the preprocessed data. Finally, in the defect localization stage, we apply machine learning algorithms on the extracted features to identify and locate defects in the software systems.

The scoring of objective questions is based on comparing the student's answer with the given standard answer. If the answers are the same, the answer is judged to be correct, and the score is calculated based on the set score for the question. After comparing all the student's answers, the obtained scores are accumulated to obtain the score for the objective question.

The scoring of subjective questions is divided into five steps. Sentence analysis is the first step in scoring subjective questions. First, we should pay attention to the different sentence patterns in the clauses, and use punctuation such as question marks, full stops, exclamation marks, etc. to separate them. For a reference answer, it is necessary to assign a previously set weight to each clause, and then calculate the similarity between the student's answer and the reference answer, and perform weighted summation.

These separated words can be weighted and summed by calculating the similarity between words to calculate the score. However, there are three problems that need to be addressed urgently. The first is the ambiguity identification of words and the identification of unregistered words. Due to different views on the distinction of some words in Chinese texts, ambiguity is usually divided into intersection and combination types. Intersection refers to the overlap between two words, while combination refers to the division of a word into multiple words. This ambiguity caused by personal cognition has caused great trouble in Chinese word segmentation processing. Unregistered words refer to words that are not entered in the existing word list and do not appear in the training data. Usually, different names for people and places in various regions, as well as some dialects, can cause great difficulties in word segmentation. For ambiguity identification, after initial word segmentation of the sentence, the ambiguity points present in the result are found, and then new words are constructed. Next, the model is trained on these words, and the latest word corresponding to the highest probability of ambiguity is selected as the valid word. This can effectively eliminate ambiguity and improve word segmentation performance. The most common method for identifying unregistered words is discovering new words, which involves extracting new words and constructing a new word dictionary for word segmentation processing, in order to achieve better results. In addition to these two traditional difficulties of Chinese word segmentation, another problem of Chinese word segmentation is the corpus dependency issue. When processing Chinese text, the quality of the corpus is crucial. Without an excellent corpus, the results of word segmentation processing are bound to be lacking credibility. A corpus that is not good enough can affect the standardization of word division and recognition classification, resulting in unreliable weighted summation values. In this article, the method used is to increase the modification of data in the corpus, test it, and then modify the problematic data. Although this increases the workload, it is necessary to spend time building an excellent corpus.

Syntax parsing is the third step in subjective question scoring. It is based on analyzing the sentence structure after word segmentation to obtain the predicate structure of each sentence. The predicate is the core of the entire sentence, while the noun that coordinates with the predicate is called an argument. Then, the parsed sentence is annotated with semantic role tags. Semantic role refers to the role played by arguments in predicate events, usually including subjects, objects, goals, etc. These annotated semantic roles are necessary for subsequent word similarity calculations. However, there is a problem with these annotated semantic roles, which is the problem of duplicate annotation. The subject is repeated, resulting in two subjects being annotated and two meanings being understood. Therefore, it is necessary to perform de- duplication by annotating the two subjects as one subject, so that the result is in line with popular understanding.

Word similarity calculation is the fourth step of subjective question scoring, which is also a crucial step. If the degree of change in sentence meaning is smaller after replacement, the similarity between the replacement word and the original word is greater, and vice versa. There are usually two methods for calculating word similarity: one is obtaining it from a tree-structured dictionary, and the other is obtaining it through statistical background information of word context. When calculating word similarity, we usually use tree dictionaries and statisticalbackground information based on word context to calculate word similarity. This article uses "Wen Web" as the knowledge base, and its similarity calculation method is to establish a semantic element tree based on different types of semantic elements and calculate the distance between semantic elements to obtain word similarity. This subordinate relationship constitutes a "sense element" hierarchy of "sense elements", laying a foundation for subsequent word similarity calculation.

Sentence similarity calculation is the fifth and final step of subjective question scoring. Sentence similarity refers to the extent to which two sentences are consistent in meaning. Typically, this matching degree is represented by a value between 0 and 1, where a value of 1 indicates that the two sentences are identical, and a value of 0 indicates that the two sentences are completely different. In sentence similarity algorithms, the method using lexical dictionaries has considerable limitations, although it solves the problem of replacement between synonyms, it does not consider the internal structure of sentences and the interaction between words, often resulting in inaccurate calculation results. On the other hand, syntax-based algorithms calculate sentence similarity by processing sentence syntax to obtain the "subject-verb-object" structure, and considering the similarity between words in the sentence based on this structure. However, because sentence structure is complex, many Chinese sentence patterns, such as inversive sentences, cannot be derived through syntax analysis. Therefore, there may be some deviations in the similarity of sentences calculated solely based on syntax analysis. Therefore, when using syntax analysis to calculate sentence similarity, it can be combined with other algorithms.

We compared our results with those of traditional static analysis techniques by measuring the accuracy, precision, recall, and Fault-score of our framework in identifying and locating defects. We also conducted a qualitative analysis to understand the contextual information extracted using our framework and its relevance to defect identification and location.

The results of our experiments show that the proposed natural language processing-based defect localization methodology outperforms traditional static analysis techniques significantly in identifying and locating defects in software systems. We achieved an average accuracy of over 85%, precision of over 80%, recall of over 75%, and Fault-score of over 70%. Additionally, the qualitative analysis revealed that our framework effectively extracted contextual information from source code commit messages and bug reports that helped in understanding the nature and location of defects accurately.

4. Conclusion

In this paper, we proposed a natural language processing-based defect localization methodology that combines text mining and natural language processing techniques to extract defect-related information effectively from software source code commit messages and bug reports. The results show that our framework can help developers improve their defect localization efforts significantly, leading to more efficient and effective software maintenance practices.

References

[1] Yi Chen, et al. A survey of static code analysis techniques. IEEE Transactions on Software Engineering, 2018,44(5): p376-394.

[2] Hui Liu, et al. Using natural language processing to improve software defect prediction. IEEE Transactions on Software Engineering, 2019, 46(3): p236-247.

[3] Shao Y, Zhao J, Wang X, et al. Log V Research on Cross-Company Defect Prediction Method to Improve Software Security. IEEE International Conference on Software Quality, Reliability and Security (QRS), Lisbon, Portugal, 2018, p:111-122.

[4] Volk M, Junges S, Katoen J P. Fast Dynamic Fault Tree Analysis by Model Checking Techniques. IEEE Transactions on Industrial Informatics, 2020, Vol. 14(18), p.370-379.

[5] E. Leu, A. Schiper, A. Zramdini, Execution Replay on Distributed Memory Architectures. IEEE Proceedings Second Symposium on Parallel and Distributed Processing, 2017, p106-112.

[6] Liu D.,Shen J.,Yang H. et al. Recognition and localization of actinidic arguta based on image recognition. Image Video Proc,2019, p19-201.

[7] Zerdoumi S.， Sabri A.Q.M.， Kamsin A. et al. Image pattern recognition in big data: taxonomy and open challenges: survey . Multimed Tools Appl,2018, p10091-10121.

[8] Kaur S.，Pandey S. Goel S. Plants Disease Identification and Classification Through Leaf Images: A Survey . Arch Computat Methods Eng,2019, p507-530.