ISSN 2395-1621

Efficient Software Issue Tracking System Using NB Classifier

[#1]Ashwini Rakshe, [#2]Sayali Suryawanshi, [#3]Shubhangi Nighul,
[#4]Bhavana Patil, [#5]U.H.Wanaskar

[1]ashwinirakshe98@gmail.com,
[2]ssayali58@gmail.com,
[3]nighulshubhangi@gmail.com,
[4]bhavanapatilmahi0395@gmail.com,
[5]ujwalaw.267@gmail.com

[#12345]Department of Computer Engineering,
P.V.P.I.T., Pune, India - 411021.

ABSTRACT

Software companies spend over 45 percent of cost in dealing with software bugs. An inevitable step of fixing bugs is bug triage, which aim to correctly assign a developer to a new bug. To decrease the time cost in manual work, text classification techniques are applied to conduct automatic bug triage. To address the problem of data reduction for bug triage, i.e., how to reduce the scale and improve the quality of bug data. To simultaneously reduce data scale on the bug dimension and the word dimension by combining instance selection with feature selection. To determine the order of applying instance selection and feature selection. An extract attributes from historical bug data sets and build a predictive model for a new bug data set. This data reduction technique will effectively reduced the data and improve the accuracy of bug triage.

Keywords: Bug triage, Training set reduction, Feature selection, Instance selection, Software quality.

ARTICLE INFO

Article History

Received: 25th December 2016

Received in revised form :

25th December 2016

Accepted: 31st December 2016

Published online :

4th January 2016

I. INTRODUCTION

Software change requests, such as bug fixes and new features, are an integral part of software evolution and maintenance. Effectively supporting software changes is essential to provide a sustainable high-quality evolution of large-scale software systems. One of the change management issues that has gained a wide attention in the last few years is the automatic support for recommending expert developers to address change requests. Mining software repositories aim to employ data mining to deal with software engineering problems. In modern software development, software repositories are large-scale databases for storing the output of software development, e.g., source code, bugs, emails, and specifications. Traditional software analysis is not completely suitable for the large-scale and complex data in software repositories. Data mining has emerged to handle software data. By leveraging data mining techniques, mining software repositories can uncover interesting information in software repositories and solve real- world software problems. A bug repository plays an important role in managing software bugs. Software bugs are unavoidable and fixing bugs is expensive in software development. There are two challenges related to bug data that may affect the effective use of bug repositories, namely the large scale and the low quality. A large number of new bugs are stored in bug repositories, due to the daily-reported bugs. Taking an open source project, Eclipse , as an example, an average of 30 new bugs are reported to bug repositories per day in 2007; from 2001to 2010, 333,371 bugs have been reported to Eclipse by over 34,917 developers and users. It is a challenge to manually examine such large-scale bug data in software development. On the other hand, software techniques suffer from the low quality of bug data. Two typical characteristics of low-quality bugs are noise and redundancy.

II. MATERIALS AND METHODS

1)Bug Triage: The process of assigning a correct developer for fixing the bug is called bug triage. We propose bug data reduction to reduce the scale and to improve the quality of data in bug repositories.

2) Bug Data Reduction: To combine existing techniques of instance selection and feature selection to remove certain bug reports and words,

- Reducing the Data Scale-Bug Dimension, Word Dimension
- Improving the accuracy

3) Prediction for reduction order: A problem for reducing the bug data is to determine the order of applying instance selection and feature selection, which is denoted as the prediction of reduction orders,

- Reduction Orders-To apply the data reduction to each new bug data set, need to check the accuracy of both two orders (FS ->IS and IS->FS) and choose a better one.
- Attributes for a bug data set-To build a binary classifier to predict reduction orders, Extract 18 attributes to describe each bug data set. Such attributes can be extracted before new bugs are triaged.

II. ALGORITHM

1) Data Reduction For Bug Triage-

Feature selection to remove certain bug reports and words, Data reduction based on FS & IS

Input: training set T with n words and m bug reports, reduction order FS->IS, final number nF of words, final number mI of bug reports.

Output: apply FS to n words of T and calculate objective values for all the words, select the top nF words of T and generate a training setT F, apply IS to mI bug reports of T F, terminate IS when the number of bug reports is equal to or less than mI and generate the final training set T FI .

2) Naïve Bayes Algorithm (Use to classifying data)-

Input: Training data set

Output: To classify fixed bug from data set, To classify most solved bug on which type, To assign bug to developer to most solved bugs as particular type of bug.

III. PROPOSED SYSTEM

To present the data preparation for applying the bug data reduction. An evaluate the bug data reduction on bug repositories of two large open source projects, namely Eclipse and Mozilla. Eclipse is a multi-language software development environment, including an Integrated Development Environment (IDE) and an extensible plug-in system; Mozilla is an Internet application suite, including some classic products, such as the Firefox browser and the Thunderbird email client. An address the problem of data reduction for bug triage, i.e., how to reduce the bug data to save the labor cost of developers and improve the quality to facilitate the process of bug triage. Data reduction for bug triage aims to build a small scale and high quality set of bug data by removing bug reports and words, which are redundant or non-informative. To combine existing techniques of instance selection and feature selection to simultaneously reduce the bug dimension and the word dimension. The reduced bug data contain fewer bug reports and fewer words than the original bug data and provide similar information over the original bug data. An evaluate the reduced bug data according to two criteria: the scale of a data set and the accuracy of bug triage. To avoid the bias of a single algorithm, an empirically examine the results of four instance selection algorithms and four feature selection algorithm.

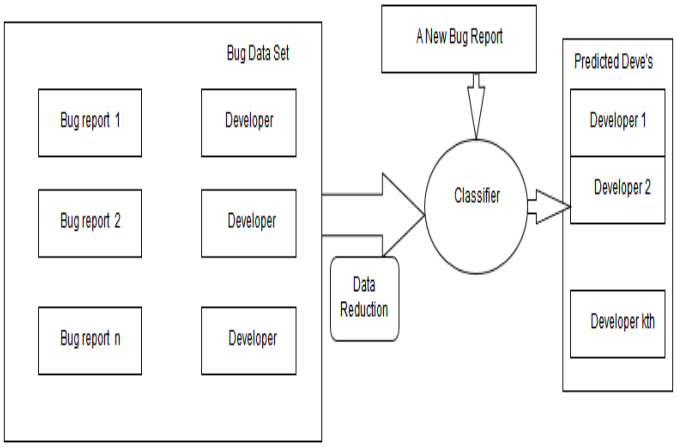

Fig 1: Proposed system

Results:

An examine the results of bug data reduction on bug repositories of two projects, Eclipse and Mozilla. For each project, an evaluate results on five data sets and each data set is over 10,000 bug reports, which are fixed or duplicate bug reports. To check bug reports in the two projects and find out that 45.44 percent of bug reports in Eclipse and 28.23 percent of bug reports in Mozilla are fixed or duplicate.

IV. CONCLUSION

Bug triage is an expensive step of software maintenance in both labor cost and time cost.To work provides a techniques on data processing to form reduced and high-quality bug data in software development and maintenance.The results of data reduction in bug triage to explore how to prepare a high quality bug data set and tackle a domain specific software task.To find out the potential relationship between the attributes of bug data sets and the reduction orders using predicting reduction orders.

REFERENCES

[1] Jifeng Xuan,Yan Hu,Zhilei Ren,Weiqin Zou,Zhongx uan Luo,and xindong,"Towards Effective Bug Triage with Software Data Reduction Techniques",IEEE Transactions on knowledge and Data Engineering ,Jan.2015

[2] S. Artzi, A. Kie_zun, J. Dolby, F. Tip, D. Dig, A. Paradkar, and M. D. Ernst, "Finding bugs in web applications using dynamic test gen- eration and explicit-state model checking," IEEE Softw., vol. 36, no. 4, pp. 474–494, Jul./Aug. 2010.

[3] J. Anvik and G. C. Murphy, "Reducing the effort of bug report tri- age: Recommenders for development-oriented decisions," ACM Trans. Soft. Eng. Methodol., vol. 20, no. 3, article 10, Aug. 2011.

[4] C. C. Aggarwal and P. Zhao, "Towards graphical models for text processing," Knowl. Inform. Syst., vol. 36, no. 1, pp. 1–21, 2013.

[5] Bugzilla, (2014). [Online]. Avaialble: http://bugzilla.org/

