ISSN 2395-1621

Reducing the Effort of Bug Report to Assigning Developers

[#1]Thavare Manisha G., [#2]Kute Deepali R., [#3]Khedkar Priya S., [#4]Prof. Vrunda Bhusari

[1]manishathavare56@gmail.com
[2]dipalikute9@gmail.com
[3]khedkarpiu29@gmail.com
[4]vrundabhusari82@gmail.com

[#1234]Deparment of Computer Engineering
JSPM's
Bhivarabai Sawant Institute of Technology & Research,
Wagholi, Pune – 412207

ABSTRACT

Open source development projects typically support an open bug repository to which both developers and users can report bugs. Developers build all the software artefacts in development. Existing work has studied the social behaviour in software repositories. In one of the most important software repositories, a bug repository, developers create and update bug reports to support software development and maintenance. However, no prior work has considered the priorities of developers in bug repositories. In this paper we address the feature selection and instance selection to reduce the data scale and address the problem of data reduction. the assignment of reports to a developer ,creating the bug triage .We decide the order of bug solve and assign to the developer as per the history record .and build the predictive bug report .

Keyword: **Bug data reduction, feature selection, instance selection, prediction for reduction orders, bug triage, bug report**

ARTICLE INFO

Article History

Received: 9th October 2015

Received in revised form :

10th October 2015

Accepted : 14th October 2015

Published online :

15th October 2015

I. INTRODUCTION

The Most open source software developments incorporate an open bug repository that allows both developers and users to post problems encountered with the software, suggest possible enhancements, and comment upon existing bug reports. One potential advantage of an open bug repository is that it may allow more bugs to be identified and solved, improving the quality of the software produced However, and this potential advantage also comes with a significant cost. Each bug that is reported must be triaged to determine if it describes a meaningful new problem or enhancement, and if it does, it must be assigned to an appropriate developer for further handling.

As a means of reducing the time spent triaging, we present an approach for semi-automating one part of the process, the assignment of a developer to a newly received report. This information can help the triage process in two ways: it may allow a triage to process a bug more quickly, and it may allow triages with less overall knowledge of the system to perform bug assignments more correctly.

Our approach is used to triage the bug particular user and assign the bug ,also bug reduction so performance of the user ,developer are increases reduction order and each attribute is helpful to the prediction. The primary contributions of this paper are as follows:

- We address the problem of data reduction for bug triage.
- We propose a combination approach to addressing the problem of data reduction. This can be viewed as an application of instance selection and feature selection in bug repositories.
- We build a binary classifier to predict the order of applying instance selection and feature selection. To our knowledge, the order of applying instance selection and feature selection has not been investigated in related domains. For that we use Fs->Is and Is->Fs algorithm.

II. PROBLEM STATEMENT

In A time-consuming step of handling software bugs is bug triage, which aims to assign a correct developer to fix a new bug. In traditional software development, new bugs are

manually triaged by an expert developer, i.e., a human triage. Due to the large number of daily bugs and the lack of expertise of all the bugs, manual bug triage is expensive in time cost and low in accuracy. In manual bug triage in Eclipse, percent of bugs are assigned by mistake while the time cost between opening one bug and its first triaging is 19.3 days on average. To avoid the expensive cost of manual bug triage, existing work has proposed an automatic bug triage approach, which applies text classification techniques to predict developers for bug reports. In this approach, a bug report is mapped to a document and a related developer is mapped to the label of the document. Then, bug triage is converted into a problem of text classification and is automatically solved with mature text classification techniques, e.g., Naive Bayes. Based on the results of text classification, a human triager assigns new bugs by incorporating his/her expertise. To improve the accuracy of text classification techniques for bug triage, some further techniques are investigated, e.g., a tossing graph approach and a collaborative filtering approach. However, large-scale and low-quality bug data in bug repositories block the techniques of automatic bug triage. .Since software bug data are a kind of free-form text data (generated by developers), it is necessary to generate well-processed bug data to facilitate the application

In this paper, we 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. In our work, we 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. We 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, we empirically examine the results of four instance selection algorithms and four feature selection algorithm.

Disadvantages:
We present the problem of data reduction for bug triage. This problem aims to augment the data set of bug triage in two aspects, namely

a) To simultaneously reduce the scales of the bug dimension and the word dimension.

b) To improve the accuracy of bug triage.

We propose a combination approach to addressing the problem of data reduction. This can be viewed as an application of instance selection and feature selection in bug repositories.

We build a binary classifier to predict the order of applying instance selection and feature selection. To our knowledge, the order of applying instance selection and feature selection has not been investigated in related domains.

III. PROPOSED SYSTEM

Since bug triage aims to predict the developers who can fix the bugs, we follow the existing work to remove unfixed bug reports, e.g., the new bug reports or will-not-fix bug reports. Thus, we only choose bug reports, which are fixed and duplicate (based on the items status of bug reports). Moreover, in bug repositories, several developers have only fixed very few bugs. Such inactive developers may not provide sufficient information for predicting correct developers.

System Implementation consist of various parts described as follows:

We are implementing our project by using Java Technology and MySQL database. We are going to implement following modules for achieving our propose system:

- Assign bug to the developer
  Is a process to assign a priority to each developer in a bug repository and to rank all the contributions of developers to assist software tasks.
- Feature selection
  Feature selection returns a subset of the features. Feature selection techniques are often used in domains where there are many features and comparatively few samples (or data points).
  Remove the certain bug reports and word from the bug report.
- Instance selection
  Is to obtain a subset of relevant instances is a technique to reduce the number of instances by removing noisy and redundant instances.

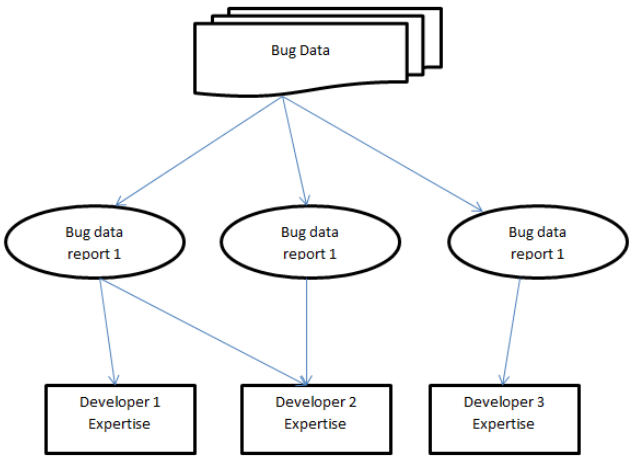

Fig 1: System Architecture

IV. ALGORITHM

Data reduction based on FS->IS

INPUT:

- Training set T with n words and m bug report
- Reduction order feature selection then instance selection
- final number nf of words
- final number mI of bug reports

OUTPUT:

Reduce dataset TFI for bug triage.

Step1

- Apply FS to n words of T and calculate objective values
- for all the words

Step2

- Select the top nF words of T and generate a training
- set T F

Step3]

- apply IS to mI bug reports of T F ;

Step4

- terminate IS when the number of bug reports is equal to
- or less than mI and generate the final training set T FI

Where

- F=Feature selection
- I=Instance selection
- TFI=final data after feature and instance selection

V. CONCLUSION

In this paper, we have presented an approach to the assignment of a bug report to a developer with the appropriate expertise to resolve the report we combine feature selection with instance selection to reduce the scale of bug data sets as well as improve the data quality. To determine the order of applying instance selection and feature selection for a new bug data set, we extract attributes of each bug data set and train a predictive model based on historical data sets. We believe that our approach shows promise for improving the bug assignment problem for open source software developments.

REFERENCE

[1] J. Anvik, L. Hiew, and G. C. Murphy, "Who should fix this bug?"inProc. 28th Int. Conf. Softw. Eng., May 2006, pp. 361–370.

[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/

[6] K. Balog, L. Azzopardi, and M. de Rijke, "Formal models for expert finding in enterprise corpora," inProc. 29th Annu. Int. ACM SIGIR Conf. Res. Develop. Inform. Retrieval, Aug. 2006, pp. 43–50.

[7] P. S. Bishnu and V. Bhattacherjee, "Software fault prediction using quad tree-based k-means clustering algorithm," IEEE Trans. Knowl. Data Eng., vol. 24, no. 6, pp. 1146–1150, Jun. 2012.

[8] H. Brighton and C. Mellish, "Advances in instance selection for instance-based learning algorithms," Data Mining Knowl. Discov-ery, vol. 6, no. 2, pp. 153–172, Apr. 2002.

[9] S. Breu, R. Premraj, J. Sillito, and T. Zimmermann, "Information needs in bug reports: Improving cooperation between developers and users," in Proc. ACM Conf. Comput. Supported Cooperative Work, Feb. 2010, pp. 301–310.

[10] V. Bol on-Canedo, N. S anchez-Maro~ no, and A. Alonso-Betanzos, "A review of feature selection methods on synthetic data,"Knowl. Inform. Syst., vol. 34, no. 3, pp. 483–519, 2013.

[11] V. Cerver on and F. J. Ferri, "Another move toward the minimum consistent subset: A tabu search approach to the condensed near-est neighbor rule,"IEEE Trans. Syst., Man, Cybern., Part B, Cybern., vol. 31, no. 3, pp. 408–413, Jun. 2001.

[12] D.Cubrani c and G. C. Murphy, "Automatic bug triage using text categorization," in Proc. 16th Int. Conf. Softw. Eng. Knowl. Eng., Jun. 2004, pp. 92–97.

[13] Eclipse. (2014). [Online]. Available: http://eclipse.org/

[14] B. Fitzgerald, "The transformation of open source software,"MIS Quart., vol. 30, no. 3, pp. 587–598, Sep. 2006.

[15] A. K. Farahat, A. Ghodsi, M. S. Kamel, "Efficient greedy feature selection for unsupervised learning,"Knowl. Inform. Syst., vol. 35,no. 2, pp. 285–310, May 2013.

