ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 5, Issue 6, June 2016

Efficient Bug Tirage Approach by Data Reduction Method

D. Udaya sri, N. Deepak kumar,

Abstract: ***The process of fixing bug is bug triage, which aims to correctly assign a developer to a new bug. Software companies spend most of their cost in dealing with these bugs. To reduce time and cost of bug triaging, we present an automatic approach to predict a developer with relevant experience to solve the new coming report. In proposed approach we are doing data reduction on bug data set which will reduce the scale of the data as well as increase the quality of the data. We are using instance selection and feature selection simultaneously with historical bug data. We have added a new module here which will describe the status of the bug like whether it assigned to any developer or not and it is rectified or not.***

Key Words: Bug, Bug triage, data reduction, Instance selection, Data Mining

I. INTRODUCTION

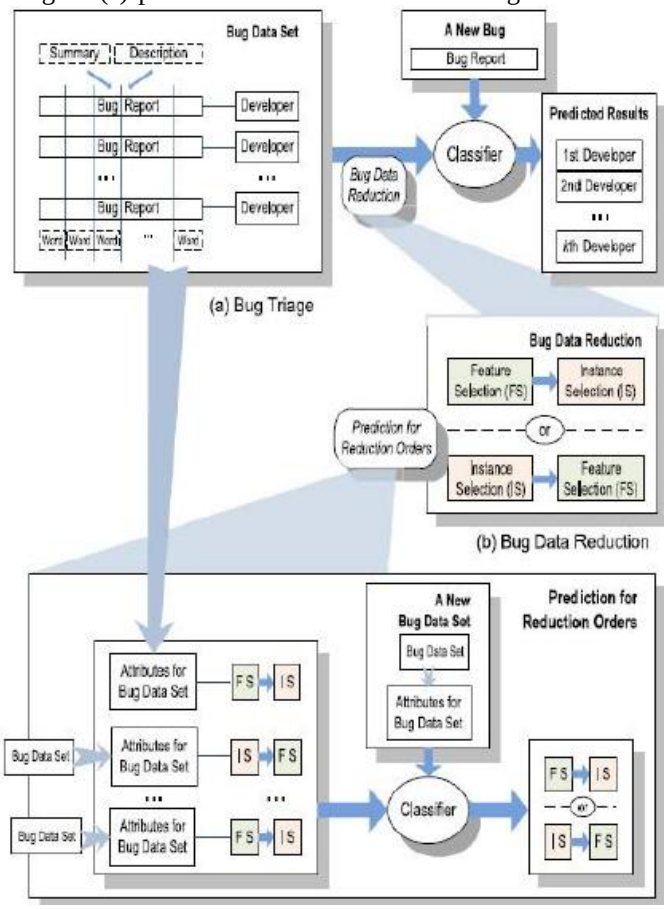
A bug repository plays an important role in managing software bugs. Many open source software projects have an open bug repository that allows both developers and users to submit defects or issues in the software, suggest possible enhancements, and comment on existing bug reports. For open source large-scale software projects, the number of daily bugs is so large which makes the triaging process very difficult and challenging [2]. Software companies spend over 45 percent of cost in fixing bugs .There are two challenges related to bug data that may affect the effective use of bug repositories in software development tasks, namely the large scale and the low quality. In a bug repository, a bug is maintained as a bug report, which records the textual description of reproducing the bug and updates according to the status of bug fixing [1]. Primary contribution of this paper is as follow:

Here in this paper we are using feature selection and instance selection with historical data for reducing the bug data in bug repository so that we get quality data as well as low scale data. We are also adding a graph module for representing the bug report's. Section II describes the architecture of the proposed system. The details of instance selection, feature selection, historical data use and graph module is given in section III and the summary is concluded in section IV.

II. LITERATURE SURVEY

J. Anvik, L. Hiew, and G. C. Murphy present a semi-automated approach intended to ease one part of this process, the assignment of reports to a developer. S. Artzi, A. Kie_zun, J. Dolby, F. Tip, D. Dig, A. Paradkar, and M. D. Ernst presents Apollo's algorithms and implementation, and an experimental evaluation . Apollo generates test inputs for a Web application, monitors the application for crashes, and validates that the output conforms to the HTML specification. J. Anvik and G. C. Murphy shows that recommenders for which developer should fix a bug can be quickly configured with this approach and that the configured recommenders are within 15% precision of hand-tuned developer recommenders. Shivkumar Shivaj, E. James, Ram Akella, Sunghun Kim investigates multiple feature selection techniques that are generally applicable to classification-based bug prediction methods. The techniques discard less important features until optimal classification performance is reached. Ahmed Lamkanfi, Serge Demeyer, Emanuel Giger, Bart Goethals investigate whether we can accurately predict the severity of a reported bug by analyzing its textual description using text mining algorithms. Weiqin Zou ,Yan Hu, Jifeng Xuan, He Jiang propose the training set reduction with both feature selection and instance selection techniques for bug triage. We combine feature selection with instance selection to improve the accuracy of bug triage.

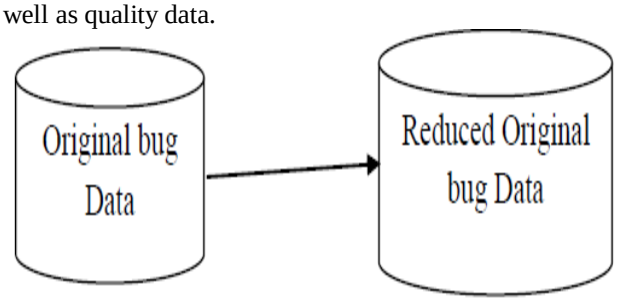
Fig 1: Illustration of reducing bug data for bug triage. Sub-figure (a) presents the framework of existing work on



bug triage. Before training a classifier with a bug data set, we add a phase of data reduction, in (b), which combines the techniques of instance selection and feature selection to reduce the scale of bug data. In bug data reduction, a problem is how to determine the order of two reduction techniques. In (c), based on the attributes of historical bug data sets, we propose a binary classification method to predict reduction orders.

a) Bug Triage Aim of bug triage is to assign a developer for bug fixing. Once a developer is assigned to a new bug report he will fix the bug or try to rectify it. He will give the status related to bug whether it is rectified or not [1].

b) Data Reduction Here we are reducing the bug data by using instance and feature selection so that we get low scale as well as quality data.



III. SYSTEM ARCHITECTURE

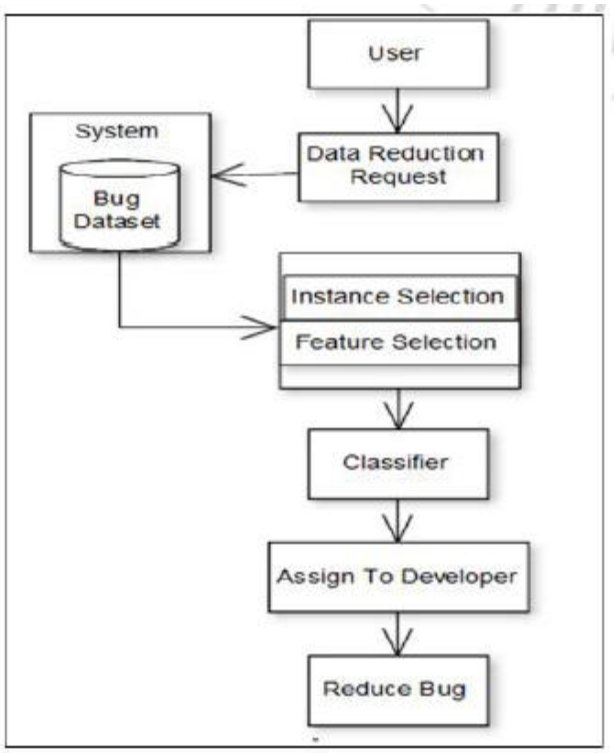


Fig 2. System architecture

Instance Selection: Instance selection and feature selection are widely used techniques in data processing. For a given data set in a certain application, instance selection is to obtain a subset of relevant instances (i.e., bug reports in bug data) while feature selection aims to obtain a subset of relevant features (i.e., words in bug data). In our work, we employ the combination of instance selection and feature selection.

Data Reduction: In our work, to save the labor cost of developers, the data reduction for bug triage has two goals. 1) Reducing the data scale. 2) Improving the accuracy of bug triage. In contrast to modeling the textual content of bug reports in existing work, we aim to augment the data set to build a preprocessing approach, which can be applied before an existing bug triage approach.

INPUT:- Let S is the Whole System Consist of S= {U, FS, IS, ICF, DROP, POP, O}

Where, IS = instance selection.
FS = feature selection.
ICF = Iterative Case Filter.
DROP = Decremental Reduction Optimization Procedure.
POP = Patterns by Ordered Projections.

Procedure: Instance and Feature Selection.
FS → IS = Bug data reduction.
which first applies FS and then IS.
On the other hand, IS → FS denotes first applying IS and then FS In Algorithm 1, we briefly present how to reduce the bug data based on FS → IS.
Given a bug data set, the output of bug data reduction is a new and reduced data set.
Two algorithms FS and IS are applied sequentially Note that in Step2, some of bug reports may be blank during feature Selection In our work, FS → IS and IS → FS are viewed as two orders of bug data reduction. To avoid the bias from a single algorithm, we examine results of four typical algorithms of instance selection and feature selection, respectively.
1) Iterative Case Filter (ICF)
2) Learning Vectors Quantization (LVQ)
3) Decremental Reduction Optimization Procedure (DROP)
4) Patterns by Ordered Projections (POP)

IV CONCLUSION

Bug triage is a costly stride of software maintenance in both work cost and time cost. In this paper, we combine feature selection with instance selection to diminish the size of bug information sets and additionally enhances the information quality. To decide the request of applying instance selection and feature selection for another bug data set, we extract attributes of each bug data set and train a predictive model based on historical data sets. Our work gives a way to deal with utilizing strategies on data processing to form reduced and high-quality bug data in software development and maintenance. In future work, we plan to propose a unified approach to merge the tasks of feature selection and instance selection. Our future plan also includes an empirical study of the use of the approach by bug triagers on an open source system, an investigation of additional sources of information.

REFERENCES

[1] Jifeng Xuan, He Jiang, Yan Hu, Zhilei Ren, Weiqin Zou, Zhongxuan Luo, and Xindong Wu," Towards Effective Bug Triage with Software Data Reduction Techniques" ieee transactions on knowledge and data engineering, vol. 27, no. 1, january 2015
[2] Mamdouh Alenezi and Kenneth Magel, Shadi Banitaan "Efficient Bug Triaging Using Text Mining" © 2013 academy publisher
[3] Francisco Servant "Supporting Bug Investigation using History Analysis" 978-1-4799-0215-6/13 c 2013 IEEE
[4] Pamela Bhattacharya, Iulian Neamtiu, Christian R. Shelton, "Automated, Highly-

Accurate, Bug Assignment Using Machine Learning and Tossing Graphs", May 2, 2012

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

[6] 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.

[7] H. Brighton and C. Mellish, "Advances in instance selection for instance-based learning algorithms," Data Mining Knowl. Discovery, vol. 6, no. 2, pp. 153–172, Apr. 2002. A. K. Uysal and S. Gunal, "A novel probabilistic feature selection method for text classification," *Knowledge-Based Systems*, vol. 36, no. 0, pp. 226–235, 2012.

[8] S. Kim, H. Zhang, R. Wu, and L. Gong, "Dealing with noise in defect prediction," in Proc. 32nd ACM/IEEE Int. Conf. Softw. Eng., May 2010, pp. 481–490. [9] A. Lamkanfi, S. Demeyer, E. Giger, and B. Goethals, "Predicting the severity of a reported bug," in Proc. 7th IEEE Working Conf. Mining Softw. Repositories, May 2010, pp. 1–10.

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

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

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

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

[13] K. Balog, L. Azzopardi, and M. de Rijke, "Formal models forexpert finding in enterprise corpora," in Proc. 29th Annu. Int. ACMSIGIR Conf. Res. Develop. Inform. Retrieval, Aug. 2006, pp. 43–50.

D. Udaya sri,*M.Tech,*
Siddharth Institute of Engineering and Technology
N. Deepak kumar, *M.Tech,(Ph.D),*
Associate Professor, Department of CSE,
Siddharth Institute of Engineering and Technology