REVIEW OF DESIGN AND DEVELOPMENT OF AUTOMATIC AND EFFECTIVE BUG TRIAGING TECHNIQUE USING DATA MINING

[1]Ms. Pranoti D. Mude, [2]Ms. Vaishali G. Bhujade
[1]M.Tech Student, [2]Asst.Professor, Dept of Computer Science Engineering, BDCOE, Wardha,
Email: [1]pmpranotimude@gmail.com, [2]vaishali.bhujade@rediffmail.com

Abstract— Bug triaging is the important step of bug fixing. Bug triaging aims the assignment of a new bug to the correct developer. The automatic bug triage used to decrease the time and cost in manual work. In this paper, we focus on the efficient technique to overcome problem of data reduction for bug triage with reducing the scale and improve the quality of bug data. With the combination of instance selection and feature selection the data scale is reduced on the bug dimension and the word dimension and determined the order of applying instance selection and feature selection. The survey shows that data reduction can effectively reduce the data scale and improve the accuracy of bug triage.

Index Terms—bug triage, effective bug triage, instance selection, feature selection.

I. INTRODUCTION

As new software systems are getting larger and more complex every day, software bugs are becoming inevitable phenomenon. Bugs can occur from a various reasons such as ranging from ill-defined specifications, to carelessness of programmers and misunderstanding of the problem and technical issues [16]. Bugs are the programming errors that cause significant performance degradation. A software bug is an error, flaw, mistake, failure, or fault in a computer program or system that produces an incorrect or unexpected result, or causes it to behave in unplanned ways. Software repositories are a large-scale database that stores the output of software development such as source code, bugs, emails, and specifications. A bug repository plays an important role in managing software bugs. Fixing bugs is expensive in software development. In a bug repository, a bug is maintained as a bug report, which consists of textual description of bug details and updates according to the status of bug fixing. Selection of the most appropriate developer to fix a new bug report is one of the most important step in the bug triaging process and it has a significant effect in decreasing the time taken for the bug fixing process and the cost of the projects. Bug triage aims to correctly assign a potential developer to a new bug.

Fixing bug reports with the traditional bug triage system i.e manual bug triage is very time consuming and costly process [1].

Bug triager is the person who assigns the bug to a developer. Bug triager, must be aware of the activities of all the developers in the project. If the developer, to whom the bug report is assigned, could not resolve it, then it is assigned to another developer. This would consume both time and money. Thus, it is important to assign the bug report to a developer who could successfully fix the bug without need of any tossing. Hence, the job of bug triager is really crucial [1]. Figure 1.1 shows the life cycle of bug. It shows that by how many stages a single bug have to pass for getting fixed.

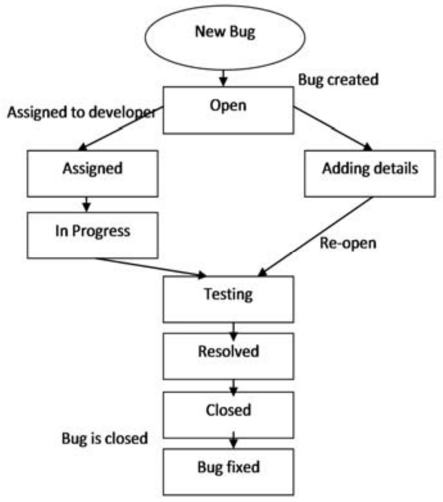

Fig. 1 Life cycle of bug

II. MOTIVATION AND RELATED WORK

A. Motivation

Open bug repository to which both developers and users can report, typically support an open source development projects. The reports that appear in this repository needs to be assigned to the proper developers for resolving the report. Noisy and redundant data always appears in real-world. Noisy data may produce problems in the data analysis techniques while redundant data may increase the cost of data processing. In bug repositories, all the bug reports are filled by developers in his own language. The low-quality bugs i.e. bugs with redundant data and noisy data accumulate in bug repositories with the growth in scale. Such large-scale and low-quality bug data may reduce the effectiveness of fixing bugs [1].

Who is the most suitable developer to fix the bug? Every day high numbers of new bugs being found and large numbers of developers working on the same software project development teams needs to dedicate the time to decide who is the most suitable person to fix each one of them. The bug assignment with very limited information such as the contents of the bug description in the bug report, the error message provided by the bug and their experimental knowledge about the expertise of each developer makes complexity in bug triage. [6]

B. Related work

The problem of data reduction for bug triage, i.e., how to reduce the scale and improve the quality of bug data, Xuan et al. [1] combine instance selection with feature selection to simultaneously reduce data scale on the bug dimension and the word dimension. To determine the order of applying instance selection and feature selection, they extract attributes from historical bug data sets and build a predictive model for a new bug data set. Jeong et al. [2] have proposed a graph model based on Markov chains, which captures bug tossing history. They analyzed 445,000 bug reports and their detailed activities from the Eclipse and Mozilla projects. They found that it takes a long time to assign and toss bugs. Zou et al. [1, 3] used different techniques for finding bugs in web applications using dynamic test generation and explicit-state model checking. To reduce time and cost of bug triaging, Alenezi and Magel present an automatic approach to predict a developer with relevant experience to solve the new coming report. [4]. John Anvik et al. [5] present a semi-automated approach intended to the assignment of reports to a developer. Francisco Servant proposed an automated technique to support bug investigation by using a novel analysis of the history of the source code [6]. Akila et al. [7] preserves the work done by all the intermediate software developers. Further the system uses actual path model instead of goal oriented path model. Thus an efficient system which captures the knowledge of the shortest paths has been highlighted. [8] Proposed new approach for selecting the developers who have appropriate expertise in the related area for handling the bug reports. A profile is created for each developer based on his previous work. This profile is mapped to a domain mapping matrix which indicates the expertise of each developer in their corresponding area. In order to evaluate our approach, they have experimented with bug reports of chromium dataset. Their experimental evaluation shows that proposed approach is able to achieve an efficiency of 86% for top-10 and 97% for top-20 developer ranking list. Kumar et al. [9] introduced a bug tracking system tools that analyze the bugs in two different ways, which help to classify bug reports. In first ways they introduce Naïve Bayes classification process by which they find probability of bugs categories on the basis of attributes of category of bug dataset. And in second ways they use natural language processing in summary attributes of bug dataset. By compare the results of two methods they were able to classify more accurately and efficiently. Xindong Wu et al. [10] present a HACE theorem that characterizes the features of the Big Data revolution, and proposes a Big Data processing model, from the data mining perspective. This data-driven model involves demand-driven aggregation of information sources, mining and analysis, user interest modeling, and security and privacy considerations. They analyze the challenging issues in the data-driven model and also in the Big Data revolution. Herzig et al. [11] discuss the impact of this misclassification on earlier studies and recommend manual data validation for future studies. Tian et al. [13] compare the effectiveness of seven state-of-the-art POS taggers on bug reports. They build a ground truth set that contains 21,713 tagged words from 100 sampled bug reports from Eclipse and Mozilla project. [14] have classified the bugs in different labels on the basis of summary of the bug. Multinomial Naïve Bayes text classifier is used for classification purpose.

III. APPROACH

To reduce the time spent in triaging, we present an approach for automatic triaging by recommending one experienced developer for each new bug report. To improve the accuracy of bug triage, bug data reduce with reducing the scales of the bug dimension, word dimension. A combination approach to addressing the problem of data reduction can be viewed as an application of instance selection and feature selection in bug repositories [1]. The order of applying instance selection and feature selection is predicted by binary classifier.

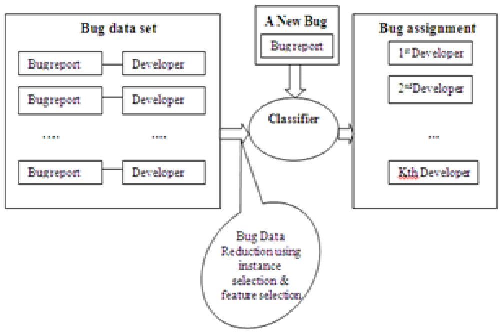

Fig.2 Block diagram of bug triage system

A. Instance Selection and Feature Selection

In bug triage, a bug data set is converted into a text matrix with two dimensions, namely the bug dimension and the word dimension. The combination of instance selection and feature selection use to generate a reduced bug data set. The original data set is replaced with the reduced data set for bug triage. Instance selection and feature selection are widely used techniques in data processing. Instance selection is to obtain a subset of relevant instances (i.e., bug reports in bug data) [1, 15] while feature selection aims to obtain a subset of relevant features.

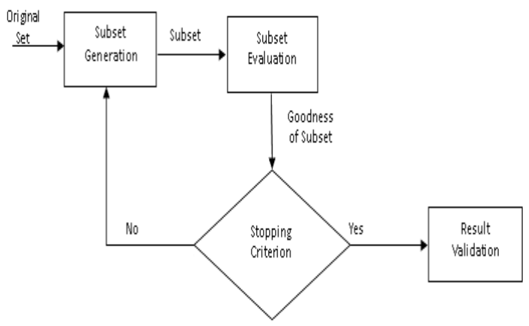

Fig.3. Four key steps for the feature selection process

IV. CONCLUSIONS

This survey provides a comprehensive overview of various techniques for bug triage used to reduce the scales of data sets and to improve the quality of bug reports. The techniques used for data reduction are instance selection and feature selection. The techniques of instance selection and feature selection are used to reduce noise and redundancy in bug data sets. The data reduction effectively reduces the bug data set into high quality bug data which can be used for effective bug triaging.

REFERENCES

[1] Lifeng Xuan, He Jiang,, Yan Hu, Zhilei Ren, Weiqin Zo, “Towards Effective Bug Triage with SoftwareData Reduction Technique”, IEEE transactions on knowledge and data engineering, vol. 27, no. 1, january 2015.

[2] Gaeul Jeong, Sunghun Kim, Thomas Zimmermann, “Improving Bug Triage with Bug Tossing Graphs”, 2009 ACM.

[3] Weiqin Zou Yan Hu, Jifeng Xuan, He Jiang “Towards Training Set Reduction for Bug Triage”, IEEE transactions on knowledge and data engineering, vol. 27, no. 1, january 2015.

[4] Mamdouh Alenezi and Kenneth Magel, “Efficient Bug Triaging Using Text Mining” , ACADEMY PUBLISHER 2013.

[5] John Anvik, Lyndon Hiew and Gail C. Murphy, “Who Should Fix This Bug?”, 2006 ACM.

[6] Francisco Servant “Supporting Bug Investigation using History Analysis”, 2013 IEEE.

[7] V.Akila, Dr.G.Zayaraz, Dr.V.Govindasamy, “Effective Bug Triage – A Framework”, International Conference on Intelligent Computing, Communication & Convergence (ICCC-2014), Procedia Computer Science 48 (2015) 114 – 120

[8] Anjali Sandeep and Kumar Singh, “Bug Triaging: Profile Oriented Developer Recommendation”, International Journal of Innovative Research in Advanced Engineering (IJIRAE) ISSN: 2349-2163 Volume 2 Issue 1 (January 2015).

[9] Pankaj Kumar and Mr.Nishant Anand, “Two way classification approache of statistical bug reports”, International Journal of Engineering Research and General Science Volume 3, Issue 3, May-June, 2015 ISSN 2091-2730

[10] Xindong Wu, Xingquan Zhu, Gong-Qing Wu and Wei Ding, “Data Mining with Big Data”, IEEE transactions on knowledge and data engineering, vol. 26, no. 1, january 2014.

[11]Kim Herzig, Sascha, “It’s Not a Bug, It’s a Feature:How Misclassification Impacts Bug Prediction”, 978-1-4673-3074-9/13/$31.00 c 2013 IEEE, ICSE 2013, San Francisco, CA, USA

[12]Pankaj Gakare, Yogita Dhole, “ Bug Triage with Bug Data Reduction”, International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056, Volume: 02 Issue: 04 | July-2015

[13]Yuan Tian and David Lo, “A Comparative Study on the Effectiveness of Part-of-Speech Tagging Techniques on Bug Reports”, 978-1-4799-8469-5/15 c 2015 IEEE, SANER 2015, Montréal, Canada

[14]Gousiya Begum, S.Vijaya Lakshmi, Shaik Irfan Babu .Nagi Reddy, “Novel Design and Implementation of an Automated Approach for Software Bug Classification”, International Journal of P2P Network Trends and Technology (IJPTT) – Volume 9 – June 2014, ISSN: 2249.

[15] Y. Fu, X. Zhu, and B. Li, “A survey on instance selection for active learning,” Knowl. Inform. Syst., vol. 35, no. 2, pp. 249–283, 2013.

[16]Anahita Alipour, Abram Hindle, Eleni Stroulia, “A Contextual Approach towards More Accurate Duplicate Bug Report Detection”, Natural Sciences and Engineering Research Council (NSERC), Alberta Innovates Technology Futures (AITF), and International Business Machines (IBM) corporation.