ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 6, Issue 11, November 2017

Software Data Reduction Technique for Effective Bug Triage: An Overview

Prof. Pooja Thakre, Parmeshwar Pawar, Nitin Nandankar, Kiran Nikam, Swati Shirsat

Department of Computer Engineering, Genba Sopanrao Moze College of Engineering, Pune, India

ABSTRACT: Bug triage is a fundamental step during bug fixing. Bug triage is the way toward fixing bug whose primary target is to accurately apportion a designer to another bug additionally taking care of. Many software organizations spend their majority of cost in managing these bugs. To reduce the time cost in manual work and to improve the working of programmed bug triage, two procedures are connected in particular content characterization and double arrangement. In writing different papers address the issue of information diminishment for bug triage, i.e., how to lessen the scale and enhance the nature of bug information. By joining the example determination and the component choice calculations to at the same time reduce the information scale and upgrade the correctness of the bug reports in the bug triage .According to writing, need to build up a powerful model for doing information lessening on bug information set which will decrease the size of the information and increment the nature of the information., by minimizing the time and cost. System produces bug report and assigns that bug to appropriate developer.

KEYWORDS: Bug triage, data reduction, Instance selection, Feature selection, Data Mining.

I. INTRODUCTION

1.1 Overview

Many software companies spend most of the money in fixing the bugs. Large software projects have bug repository that collects all the information related to bugs. In bug repository, each software bug has a bug report. The bug report consists of textual information regarding the bug and updates related to status of bug fixing [1]. Once a bug report is formed, a human triager assigns this bug to a developer, who will try to fix this bug. This developer is recorded in an item assigned-to. The assigned to will change to another developer if the previously assigned developer cannot fix this bug. The process of assigning a correct developer for fixing the bug is called bug triage [2]. Bug triage is one of the most time consuming step in handling of bugs in software projects. Manual bug triage by a human triager is time consuming and error-prone since the number of daily bugs is large and lack of knowledge in developers about all bugs. Because of all these things, bug triage results in expensive time loss, high cost and low accuracy [3]. The information stored in bug reports has two main challenges. Firstly the large scale data and secondly low quality of data. Due to large number of daily reported bugs, the number of bug reports is scaling up in the repository. Noisy and redundant bugs are degrading the quality of bug reports. The effective bug triage system is proposed which will reduce the bug data to save the labor cost of developers. It also aims to build a high quality set of bug data by removing the redundant and non-informative bug reports [4].

1.2 Background

Putting away points of interest of bugs, it assumes a critical part. Bug stores are broadly utilized for keeping up programming bugs, e.g., a well-known and open source bug store, Bugzilla. A recorded bug is known as a bug data, once a bug report is created, a human triager doles out this bug to a designer, who will endeavor to settle this bug [2]. A designer, who is allocated to another bug report, begins to settle the bug in view of the information of chronicled bug settling. Bug triage is a routine of passing the settling bugs to correct engineer. An engineer, who is relegated to an imaginative bug data, begins to settle the bug based on the data of past bug settling. Regularly, the engineer pays endeavors to perceive the new bug report and generally settled bugs as a kind of perspective (e.g., hunting down comparative bugs and applying available answers for the new bug) [3]. Status of a bug report is changed by late

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 6, Issue 11, November 2017

consequence of taking care of this bug until the bug is totally settled. Displayed work utilizes the methodologies in view of content demeanor to help bug triage. In such a way, the rundown of a bug report are concentrate as the printed content while the engineer who7 can join this bug is as the name for grouping. It gives low precision [4].

II. LITERATURE SURVEY

Jif et. al.eng Xuan et. Al. [1] proposed Towards Effective Bug Triage with Software Data Reduction Techniques. Bug triage is an expensive step of software maintenance in both labor cost and time cost. Here 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, to extract attributes of each bug data set and train a predictive model based on historical data sets. Empirically investigate the data reduction for bug triage in bug repositories of two large open source projects, namely Eclipse and Mozilla. The work provides an approach to leveraging techniques on data processing to form reduced and high-quality bug data in software development and maintenance.

Shay Artzi et.al. [2] proposed Finding Bugs in Web Applications Using Dynamic Test Generation and Explicit-State Model Checking. Present a technique for finding faults in PHP Web applications that is based on combined concrete and symbolic execution. The work is novel in several respects. First, the technique not only detects runtime errors but also uses an HTML validator as an oracle to determine situations where malformed HTML is created. Second, address a number of PHP-specific issues, such as the simulation of interactive user input that occurs when user-interface elements on generated HTML pages are activated, resulting in the execution of additional PHP scripts. Third, to perform an automated analysis to minimize the size of failure-inducing inputs tags.

Partha Sarathi Bishnu and Vandana Bhattacherjee [3] proposed a system Software Fault Prediction Using Quad Tree-Based K-Means Clustering Algorithm It reviews the problems with using simple K-Means in the classification of data sets. The effectiveness of Quad Tree based EM clustering algorithm in predicting faults while classifying a dataset, as compared to other existing algorithms such as, K-Means has been evaluated. The Quad Tree approach assigns appropriate initial cluster centers and eliminates the outliers. K-Means is considered to be one of the simplest methods to cluster data. However, the proposed EM algorithm is used to cluster data effectively. Combining the Quad Tree approach and the EM algorithm gives a clustering method that not only fits the data better in the clusters but also tries to make them compact and more meaningful. Using EM along with Quad Tree makes the classification process faster. With K-means, convergence is not guaranteed but EM guarantees elegant convergence.

Yutaro, Kashiwa Hayato Yoshiyuki et. Al. [4] proposed **A Pilot Study of Diversity in High Impact Bugs** The conducted case study on high impact bugs, which classified bugs reported to four open source projects into six types of high impact bugs. In the case study, one hundred bug reports were manually inspected for each project and are classified into six types of high impact bugs based on previous studies which focus on high impact bugs. The case study aimed to reveal distributions of high impact bugs in reported bugs and overlapped relationships among high impact bugs.

Shivkumar Shivaji et. Al. [5] proposed Reducing Features to Improve Bug Prediction. It has implemented costriage algorithm which helps to exploit the cost and accuracy of bug fixing or bug prediction. It has also implemented the feature selection technique which reduces the number of features used by a machine learning classifier for bug prediction [5].

A. Srisawat, T. et. Al. [6] proposed Reducing Features to Improve Code Change-Based Bug Prediction. They explored the use of feature selection techniques to predict software bugs. An important pragmatic result is that feature selection can be performed in increments of half of all remaining features, allowing it to proceed quickly. Between 3.12 and 25 percent of the total feature set yielded optimal classification results. The reduced feature set permits better and faster bug predictions. The process is fast performing and can be applied to predicting bugs on other projects. The most important results stemming from using the feature selection process are found in Table 5, which presents F-measure optimized results for the Naive Bayes classifier. A useful pragmatic result is that feature selection can be performed in increments of half of all remaining features, allowing it to proceed quickly. The average buggy is precision is quite high at 0.97, with a reasonable recall of 0.70. This result outperforms similar classifier-based bug prediction techniques and the results pave the way for practical adoption of classifier-based bug prediction. From the perspective of a developer

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 6, Issue 11, November 2017

receiving bug predictions on their work, these figures mean that if the classifier says a code change has a bug, it is almost always right. The recall figures mean that, on average, 30 percent of all bugs will not be detected by the bug predictor. This is likely a fundamental limitation of history-based bug prediction, as there might be new types of bugs that have not yet been incorporated into the training data. Believe this represents a reasonable tradeoff since increasing recall would come at the expense of more false bug predictions, not to mention a decrease in the aggregate buggy F-measure figure. Such predictions can waste developer time and reduce their confidence in the system.

III. PROPOSED SYSTEM DESIGN

3.1 Objectives

- Propose a combination approach to addressing the problem of data reduction
- Triage Bug to developer which having most experience on that problem.
- To generated Bug report with the bug suggestion.

3.2 Problem Statement

Bug triage is an essential step in the process of bug fixing. Bug triage is the process of fixing bug whose main objective is to correctly allocate a developer to a new bug for further handling by using two techniques are applied namely Text Classification algorithm and Binary classification algorithm. The output of system is bug allocate to appropriate developer which having more skill to solve that bug.

3.3 Project Overview

The goal of bug triage is to assign a bug to the correct potential developer. In bug triage, a bug data set is converted into a text matrix with two proportions, namely the bug dimensions and word dimension. Bug reports are in the form of summary and description. Bug data decrease to compress the scale and to improve the quality of data in bug repositories which is applied as a stage in data preparation of bug triage. Existing techniques of instance selection and feature selection are used to remove definite bug reports and words [4]. Classifiers are also used to fix new bug report. It also used to predicate developer to fix bug. It replace the original data set with the reduced data set for bug triage. Instance selection and feature selection are widely used techniques in data processing. 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. 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 [5]. In this approach, a bug report is mapped to a document and are late 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.

3.4 Development Methodology

The single problem can be solved by different solutions. This considers the performance parameters for each approach. Thus considers the efficiency issues. The system will have a database containing files along with encryption and decryption keys. The end user will access these files and the owner will upload the files with the key which only the authorized user can access to download the file. This solves the problem of unauthorized access. Unravel completely different components of the matter (sub problems), then mix the resolutions of the sub issues to succeed in Associate in Nursing overall solution. Usually once employing an additional naive technique, several of the sub issues are generated and solved over and over [6]. The dynamic programming approach seeks to unravel every sub downside just once, so reducing the quantity of computations: once the answer to a given sub downside has been computed, it's stored; consequent time an equivalent resolution is required, it's merely researched. This approach is particularly helpful once the quantity of continuation sub issues grows exponentially an operate of the dimensions of the input. This system to have a tendency to use this technique for authentication purpose. The user must register 1st and also the details are hold on within the information. The registered user are hold on along with his identity, whenever the authority must certify he can merely consider information for the desired details [7].

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 6, Issue 11, November 2017

3.5 System Architecture

Figure shows the architecture of the bug triaging. Aim of bug triage is to allocate a developer for bug deception. Once a developer is allocated to a new bug report they will solve the bug or attempt to fix it. They will provide the status associated to bug whether it is corrected or not [1].

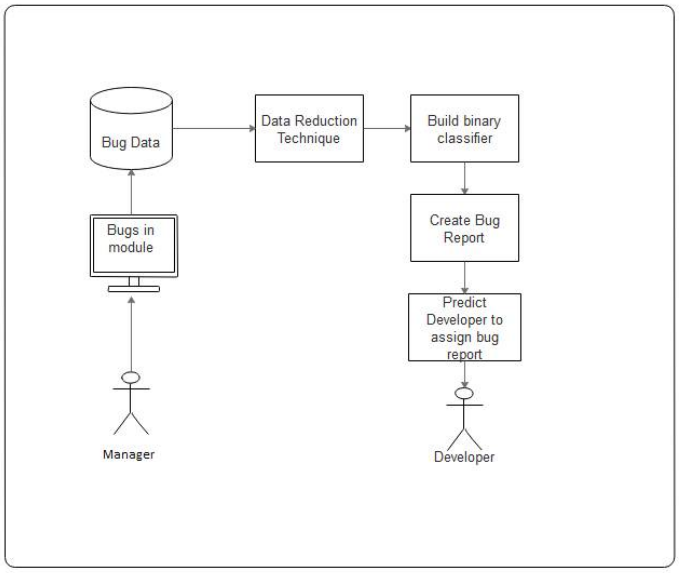

Figure 1 : Proposed System Architecture

Decreasing the Data Scale:

It reduce scales of data sets to protect the manual labor charge of developers. As mentioned above, the aim of bug triage is to allocate developers for bug deception. Once a developer is allocated to a new bug report, the developer can inspect historically fixed bugs to custom a solution to the existing bug report [1]. For example, historical bugs are checkered to detect whether the new bug is the replica of a prevailing one moreover, existing results to bugs can be searched and applied to the new bug. Thus consider reducing replica and raucous bug reports to reduce the number of historical bugs. In practice, the manual labor charge of developers (i.e., the cost of examining historical bugs) can be avoided by subsiding the number of bugs based on instance selection. Word dimension [2]. To use feature selection to remove noisy or replica words in a data set. Based on feature selection, the reduced data set can be controlled more easily by automatic methods (e.g., bug triage approaches) than the original data set. In addition to bug triage, the reduced data set can be further used for other software tasks after bug triage (e.g., severity identification, time prediction, and reopened bug analysis) [3].

Enhancing the Accuracy:

Accuracy is an essential evaluation measure for bug triage. The work, data reduction discovers and removes noisy or duplicate information in data sets [1]. Bug dimension. Instance selection can remove unhelpful bug information; meanwhile, can notice that the accuracy may be reduced by removing bug reports. Word dimension. By removing unhelpful words, feature selection recovers the accuracy of bug triage [4].

IV. RESULTS AND DISCUSSION

Manual Bug fixing is time consuming task and didn't get accurate result. So that proposed system is provided. There is problem of getting accurate bug solution according to domain. In existing approach, get reduced bug dataset and high

ISSN(Online): 2319-8753
ISSN (Print): 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Visit: www.ijirset.com

Vol. 6, Issue 11, November 2017

quality bug dataset. For that purpose, proposed system is provided [5]. Here used existing system instance selection and feature selection for reducing bug dataset. And additionally use Top-K pruning algorithm for improving results of data reduction quality as compared to existing system and get domain wise bug solution [7]. According to error it will create the bug report. Report is generated in form of pdf file. Pdf contains details as follows:

1. Bug Summary: To have use reduction algorithm here to reduce the result and show that in pdf.
2. Bug Deadline: It contains date by which the bug should be solved.
3. Bug Description: Here the details of bug are reported. This is nothing but the bug reported by the manager.
4. Suggestions: Developer who has already worked on some or have some knowledge have a bug can be solved then he or she can give suggestion to the user. This suggestion is also included in this pdf report.
5. Assign bug report to appropriate developer: This is done with respect to experience of each developer. Their suggestions also matters here.

V. SUMMERY AND CONCLUSION

Bug triage is an expensive step of software maintenance in both labor cost and time cost. The 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, to extract attributes of each bug data set and train a predictive model based on historical data sets. The work provides an approach to leveraging techniques on data processing to form reduced and high-quality bug data in software development and maintenance. For predicting reduction orders, plan to pay efforts to find out the potential relationship between the attributes of bug data sets and the reduction orders. To propose bug data diminution to reduce the scale and to get better the quality of data in bug repositories. Which is applied as a stage in data training of bug triage. Here to mingle existing techniques of instance selection and feature selection to eliminate certain bug information and words. A crisis 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. In this section, first how to apply instance selection and feature selection to bug data, i.e. data shrinking for bug triage. Then, to list the benefit of the data reduction. Those are Bug summery that will summarize the bug generated report in the form of pdf file. It will contain bug summery, bug deadline, bug description and suggestions by developer if any. Graphical representation for bug assigning and completion by developer is also provided. This makes the analysis easier for the higher authority to decide the developer to assign further bug to repair. In future work, to expect improving the eventual outcomes of data diminishment in bug triage to explore how to set up a first rate bug data set and handle a space specific programming task. For predicting reducing orders, plan to pay attempts to find the potential relationship between the characteristics of bug data sets and the diminishment orders. In Future System used as:

- For IT industry to manage bug solution process task.
- Used automatic bug triage in industry.

REFERENCES

[1] Jifeng Xuan, He Jiang, Yan Hu, Zhilei Ren, Weiqin Zou, Zhongxuan Luo, and Xindong Wu, IEEE VOL. 27, no. 1, January 2015
[2] 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.481490.
[3] G. Jeong, S. Kim, and T. Zimmermann, Improving bug triage with tossing graphs,"in Proc. Joint Meeting 12th Eur. Softw. Eng. Conf. 17th ACM SIGSOFT Symp. Found. Softw. Eng., Aug. 2009, pp. 111120.
[4] Q. Shao, Y. Chen, S. Tao, X. Yan, and N. Anerousis, Efficient ticket routing by resolution sequence mining,"in Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, Aug. 2008, pp. 605 613.
[5] C. Sun, D. Lo, S. C. Khoo, and J. Jiang, Towards more accurate retrievalof duplicate bug reports,"in Proc. 26th IEEE/ACM Int. Conf. Automated Softw. Eng., 2011, pp. 253262.
[6] A. Srisawat, T. Phienthrakul, and B. Kijsirikul, SVkNNC: An algorithm for improving the efficiency of k nearest neighbor,"in Proc. 9th Pacific Rim Int. Conf. Artif. Intell., Aug. 2006, pp. 975979.
[7] J. Anvik, L. Hiew, and G. C. Murphy, Who should fix this bug?"in Proc. 28th Int. Conf. Softw. Eng., May 2006, pp. 361370.
[8] 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 generation and explicit state model checking,"IEEE Softw., vol. 36, no. 4, pp. 474494, Jul./Aug. 2010.
[9] J. Anvik and G. C. Murphy, Reducing the effort of bug report triage: Recommenders for development-oriented decisions, "ACM Trans. Soft. Eng. Methodol., vol. 20, no. 3, article 10, Aug. 2011.