International Journal of Research in Engineering, Science and Management
Volume-2, Issue-1, January-2019
www.ijresm.com | ISSN (Online): 2581-5792

Bug Triage using Data Reduction Techniques

D. Patil Priyanka
PG Student, Dept. of Computer Engineering, S. S. V. P. S. B. S. Deore College of Engineering, Dhule, India

***Abstract*: Bug in software degrades quality of software unless solved with great precession and accuracy. To accomplish this process called as bug triage is conducted. Companies spend heavily on bug triage process. The data reduction is done on bug data set which will reduce the size of the data as well as improve the quality of the data. Instance selection reduces number of instance and feature selection reduces set of features. Instance selection and feature selection is also used simultaneously with historical bug data. This process is assigning of the specified bug to corresponding developer. The system will focus is this issue in software bug life cycle where lots of meetings and discussion is involved. The system will apply prediction on the history bug dataset through which intelligence can be added which will predict the most appropriate developer for the associated bug. Thus the System work will build a predictive model for bug dataset and reduce the involvement of skilled resources and which will in turn add to reduction of cost.**

***Keywords*: Bug Triage, Data Reduction, Feature Selection, Instance Selection.**

1. Introduction

Software repositories comprise valuable information about software projects. This data can help to manage the progress of these projects. One of the necessary software repositories is the bug tracking system. Many open source software projects have an open bug repository that allows both developers and users to submit faults or issues in the software. Bug tracking system is manages bug reports submitted by users, testers, and developers. Data reduction is that the transformation of numerical or alphabetical digital data derived by trial and error or by experimentation in a corrected, ordered, and simplified form. The basic concept is the reduction of multitudinous amounts of data down to the important elements. Each software projects have own bug repository [1]. A bug repository is a typical software repository for storing details of bugs. A Bug Repository is a software repository. It contains all the data related to software bugs. A software bug is a problem, which causes a program or system to crash or produce invalid output or to behave unplanned way. A bug repository provides a data platform to support many sorts of tasks on bugs, e.g., fault prediction bug localization and reopened bug analysis. Bug reports in a bug repository are known as called bug data. In bug repository, software bug is kept as bug report. It contains of textual description regarding the bug and updates related to status of bug fixing. After the creation of bug report, a human trigger assigns this bug to a developer, who will try to fix this bug. If the assigned developer cannot fix this bug, then new developer is assigned for fixing that bug. This process of assigning a correct possible developer to fix a new bug is called bug triage. A bug repository shows an important role in handling software bugs. Many 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 [2].

For open source large-scale software projects, the number of day-to-day bugs is so huge which creates the triaging process very tough and challenging. Software companies use over 45 percent of cost in fixing bugs. There are two challenges related to bug data that may affect the actual use of bug repositories in software development tasks, specifically the large scale and the low quality. Many open source software projects integrate open bug repositories during expansion and keep so that both developers and users can report bugs that they have encountered, and call for more useful features or make suggestions for revision. There are at least two important advantages of using such a bug repository. First, the bug repository allows users all around the world to be “testers” of the software, so it can increase the possibility of revealing faults and improve the quality of the software. Second, it helps the software develop according to user requests, and meet the requirements of more users [3]. Bug Triage consumes long time for handling software system bugs. In traditional software development, a human triager is used i.e. expert developers were manually triaged the new bugs. But manual Bug Triage is expensive in time and accuracy because of large number of daily bugs and the lack of expertise of all bugs. The resolution of fault triage meeting in software development process is to select the defects based on its severity, risk, product, component, priority etc. Team involved in defect triage meeting should validate severities of the defect make changes as required finalize resolution of the defects and assign resources to it. In data reduction for bug triage how to reduce the bug data to save the work cost of developers and improve the quality to simplify the process of bug triage. Data reduction for bug triage purposes to build a small-scale and high-quality set of bug data by removing bug reports and words, which are redundant or non-informative. It combines techniques of instance selection and feature selection to at the same time reduce the bug dimension and the word dimension. [4]

2. Literature Review

Predict a correct developer to a bug is important in bug triage

process. To avoid the expensive cost of manual bug triage, an automatic bug triage approach was proposed, which applies text classification techniques to predict developers for bug reports. A software bug is a problem which causes a computer program or system to crash or produce invalid output or to behave unplanned way. Many software companies have to face large number of software bugs. Bug Triage consumes more time for handling software bugs. It is the process of assigning a new bug to the correct potential developer. There are various existing techniques for bug triage. It includes Text categorization, Tossing Graph, Recommendation, Role-Based, Text Mining etc. Most of these techniques provide automatic bug triage. Some of these techniques are further classified. [5]

M. D'Ambros, M. Lanza, and M. Pinzger proposed Bug Tossing. Bug tossing is same as ticket routing method. Bug tossing is visualized the life cycle of bugs, with the assignment of developers. Patterns in bug reports; from them one was the reassignment of developers. Investigates both assignment and reassignment of developers empirically. Builds a model of bug tossing. It reduces the number of reassignment of bug reports. This system based on Markov chain approach. It captures past bug tossing history This system improves bug assignment and reduce unnecessary tossing steps. [4]

P. Bhattacharya P. and Neamtiu [6] Proposed three novel extensions techniques. It improves triaging and reduce tossing path lengths by using some techniques. There are three novel extensions to existing techniques. They achieve higher prediction accuracy using richer feature vectors. In previous work, bug title and summaries are used; but here they add attributes like to the product–component information for a bug.

D. Matter, A. Kuhn, and O. Nierstrasz [7] Proposed technique which present an approach to automatically recommend developers who have the appropriate expertise to handle a bug report. Using expertise, the vocabulary which was found in their source code contributions. Then they compare this vocabulary to the vocabulary of bug reports. They then mention developers whose contribution vocabulary is lexically like to the vocabulary of the bug report. An advantage of this technique is that, it doesn't need a record of previous bug reports. They are able to mention developers who did not work on bugs previously.

V.Akila, Dr.V.Govindasamy [1] Proposed a new framework with the additional capabilities. The reassignment of bugs is as Enriched Adaptive Bug Triaging System which is based on actual path model. The recommended graph structure captures the relationship among developers as the number of tosses. It also captures the propinquity exists among developers. So, this graph structure is enriched. The technique is based on Ant routing. Ant routing is essentially adaptive in nature. The system work gives a sub graph that consists of developers who are frequently involved in bug resolution.

John Karsten Anvik [1] proposed a machine learning approach to create triage-assisting recommenders. They mention it as MLTriage. The goal of this is to reduce the human involvement in triage. In this MLTriage process, Firstly, reports are automatically selected from a project's issue tracking system. From these selected reports, features that is specific piece of data are collected and reports with same features are grouped under a label. The label shows the category to which the features belong. These extracted data and labels area unit fed to a supervised machine learning algorithmic rule. Then, recommender is created for specific development-oriented decision. When recommender ask to make a prediction for new bug report, that time features are extracted from the new bug report and are fed to the recommender. Then, recommender gives a list of potential labels.

O. B. Michael and G. C. Robin [7] proposed a framework for automated assignment of bug. In this framework, this is able to arrange a developer's level of expertise by tracking the history of the bugs previously resolved by this developer. Preference elicitation means the problem of developing a decision support system which is capable of generating recommendations to a user, which then assist in decision making. This system employs preference elicitation to learn developer predilections in fixing bugs within a given system.

Tamrawi, T. Nguyen, J. Al-Kofahi, T. Nguyen, [8] Proposed Bugzie system a novel approach for automatic bug triaging. Bugzie is based on fuzzy set-based modeling of bug-fixing expertise of developers. Considers a system have multiple technical aspects. Bugzie uses a fuzzy set to represent the developers who are capable of fixing the bugs which are relevant to each term. The membership function of a developer in a fuzzy set is calculated via the terms extracted from the bug reports that fixed. When new fixed reports are available then the function gets updated. A new bug report, its terms are extracted and as per the terms corresponding fuzzy sets are joining together. Based on their membership scores in the joining together fuzzy set, Potential fixers will be mentioned. Bugzie achieves higher accuracy and efficiency than other approaches.

L. Neamtiu, C. R. Shelton [6] proposed a method for optimal set of machine learning techniques to improve bug assignment accuracy. This system complete set of machine learning tools as well as a probabilistic graph-based model that lead to highly-accurate predictions. The foundation for the next generation of machine learning-based bug assignment. They used methods like taking effective classifiers and features, Incremental learning, Multi-featured tossing graphs to achieve their goal.

T. Zhang, G. Yang, B. Lee, I. Shin [9] proposed a new bug triage algorithm that mentions right developers to fix the given bugs. Analyzes different roles that the developers play in the bug fixing process for extracting the further characteristic features. Four roles of developer as: a reporter, a triager, an assigned, a commenter. Then they evaluate the developers experience for understanding and fixing submitted bugs.

3. System architecture

An unavoidable step in fixing bug is bug triage. The bug

triage has a large dataset of past bugs and their explaining developers. The system focuses on the problem of data reduction for bug triage. In bug triage the instance selection is a technique to reduce the number of instances and feature selection is reducing set of features. The system focuses on data reduction by pre-processing the dataset by removing the noisy and redundant data from the set which reduces the size of the bug dimensions. To address the accuracy, the system builds a classifier model using naive Bayes using instance selection and feature selection [10].

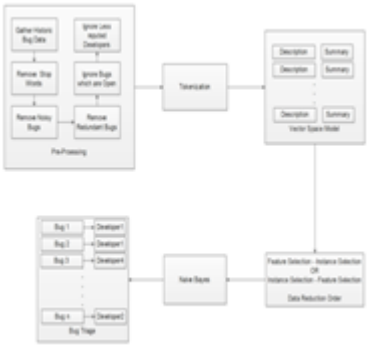

Fig. 1. System architecture

The figure1 describes the framework of the system. In system architecture contains four main block namely preprocessing, vector space model, Instance and feature selection techniques and bug triage. In data reduction contain preprocessing and vector space model. In prediction part contains instance and feature selection techniques and probability based algorithm apply on it. The system aims to predict developer to whom the corresponding bug should be assigned based on the analysis on history dataset. The system performs preprocessing on the bug description to remove noisy, redundant and unsolved bugs to achieve better accuracy. Then a vector space model is generated of bug and its short summary. Then the prediction phase is applied to the preprocessed information in order to predict the developer for the bug under consideration. The output of the system is a score of for each developer against the bug under consideration. System selects the best of the lot and gives the final predicted output. Preprocessing apply on original dataset. In preprocessing, remove stop words, remove noisy bugs and redundant bugs, ignore bugs which are open, and ignore less repeated developers. After applying preprocessing on dataset then generate vector space model.

In vector space model contains bug description and its short summary related bug data. After data reduction process applies instance selection and feature selection. Then use naïve bayes algorithm for predict correct developer. The naïve bayes algorithm is probability based algorithm. The data is reduced from original bug data. Here decreasing the bug records by means of instance and feature selection so that acquire low scale as well as eminence data. The data reduction for bug triage has two goals decreasing the data scale and enhancing the accuracy of bug triage. Figure2 shows flow chart of system. First part of preprocessing in that input is dataset. Second part is developer prediction. Developer prediction contains feature selection and instance selection. Then apply naïve bayes algorithm. Using algorithm predict correct developer for bug data. When preprocessing applies on original data set then parse the reduced data set. Using remove stop words, and remove duplicate bugs reduce large data set. Then build a vector space model. Vector model space contain short summary of bug data means its description and summary related bug data. In developer prediction, bug input give to the system. Then apply instance selection and feature selection. Using instance selection reduces number of instance and feature selection reduces a set of feature. Apply naïve bayes algorithm on bug data. Then compare probability and also find maximum probability of developer. On the basis of maximum probability of developer predict a correct developer.

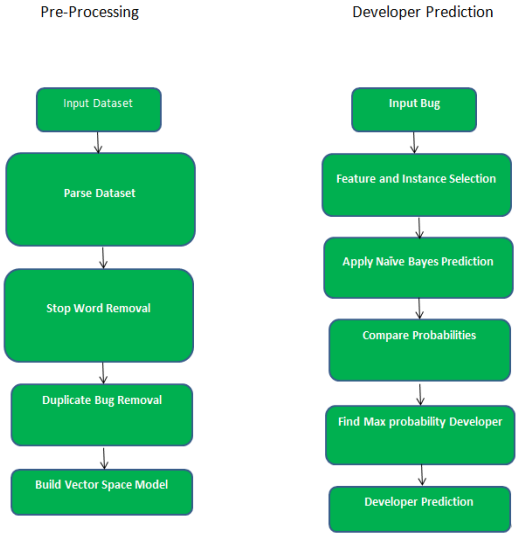

Fig. 2. Flow chart of Bug Triage System

A. *Prediction for reduction order*

Reduction order as- (FS⟶IS) OR (IS⟶FS)

Combine instance selection and feature selection to remove certain bug reports and words. 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. The instance selection reduces number instances using remove noisy and remove duplicate instance. Using feature selection reduces large number of feature in data set. The problem of prediction for reduction orders into binary classification problem. A bug data set is mapped to an instance and the associated reduction order is either(FS⟶IS) or (IS⟶FS) mapped to the label of a class of instance. To build a binary classifier to predict reduction orders, extract 13 attributes to describe each bug data set. Such attributes can be extracted before new bugs are triaged. Attributes are severities, priorities, product, components etc. [9].

B. *Prediction Algorithm for Developer Prediction*

The naïve bayes algorithm is a probability based algorithm.

Use of naïve bayes algorithm to predict developer for fixing a bug and also the improve accuracy of data reduction. The following are the steps included in algorithm.

1) Count initial frequency.
   $f(op) = n$
2) Calculate initial probability-
   $p(op_n) = \frac{frq.}{Total\ count}$
3) Calculate conditional probability.
4) Overall/ final probability
   $f(op_1)\ compare\ f(op_2)$

Get output basis on probability based algorithm and assign a correct developer for a bug.

4. Experimental set up and Result

We have evaluated the system on eclipse dataset and found the precision of the system which is shown in the below graph. The result of data reduction for bug triage can be measured in two aspects, namely the scale of data sets and the quality of bug triage. The techniques discussed in the proposed system phase shows the system is adaptive to different datasets. We have shown our working on the eclipse dataset and evaluated the precision graphs which have greater precision values then the previous system. Firstly, we take 10 results of system then its precision values 0.9. Then we take 15 results of system then its precision values 0.93. Figure shows the graph for developer prediction precision values of first 10 result precision value 0.9 and for 15 results precision values 0.93.

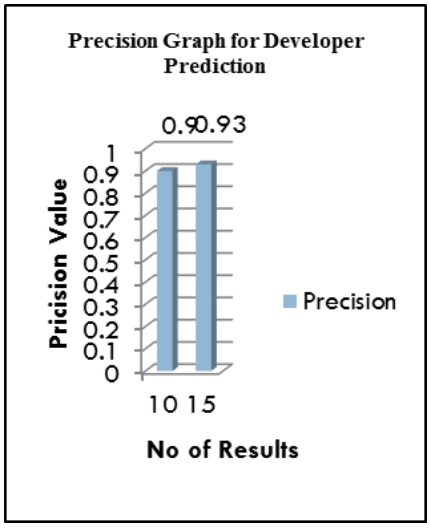

Fig. 3. Precision graph for developer prediction

We have evaluated the system on eclipse dataset and found the recall of the system which is shown in the below graph. The result of data reduction for bug triage can be measured in two aspects, namely the scale of data sets and the quality of bug triage. We have shown our working on the dataset and evaluated the recall graphs which have lowest recall values then the previous system also in previous system recall values is constant.

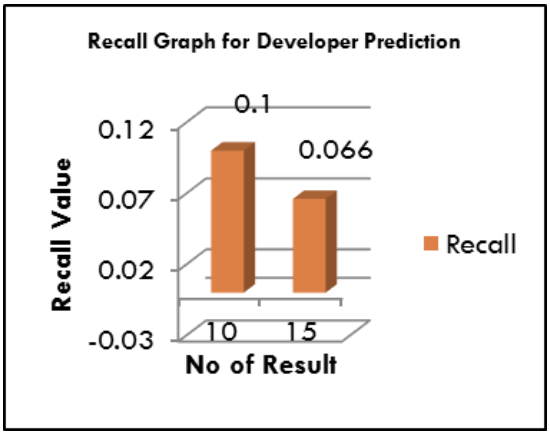

Fig. 4. Recall graph for developer prediction

Figure shows the graph for developer prediction for 10 results recall values 0.1 and for developer prediction for15 results recall values 0.066.

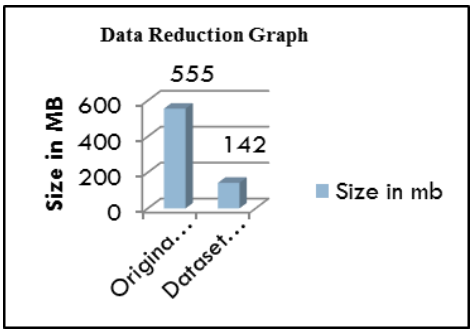

Fig. 5. Data Reduction

Figure show the data reduction graph. We evaluate the result of data reduction for bug triage on data set. Each instance selection and feature selection based on one bug triage algorithm, naïve bayes. Combine the instance selection and feature selection to data reduction on prediction algorithm. Original data is 555mb then reduced original data using preprocessing. Bug data reduction to reduce the scale and to improve the quality of data in bug repositories.

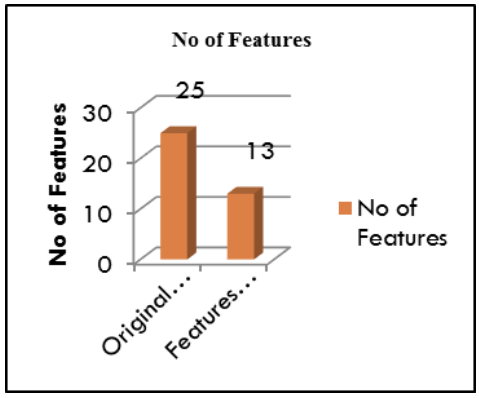

Fig. 6. Feature graph

Figure show the feature selection graph. Original features are

25 then reduced feature for predicted correct developer and for accuracy.

5. Conclusion

Study about data reduction and predict a correct developer for new bug data. Bug Triage is very important task in any software company and must handle with maximum precession. Human involvement in the bug triage process may reduce the accuracy and also will increase the time. Duplicate bug entries in bug report also can increase the processing time and can reduce the quality of results. The proposed system will work on both the aspect by removing the duplicate bug entries from the report and providing a predictive model for developer prediction provided a bug entry. Thus system will greatly add effiency in the process of bug triage in industry and will eliminate the human intervention and save the time and resource for any company.

References

[1] He Jiang Jifeng Xuan, "Towards Effective Bug Triage with Software Data Reduction Techniques," *IEEE Trans. Data Eng.*, vol. 27, no. 1, pp. 264-280, January 2015.

[11] X. Zhu and X. Wu, "Cost-constrained Data Acquisition for intelligent data preparation," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 11, pp. 1542-1556, Nov 2005.

[2] Yogita Dhole Pankaj Gakare, "Bug Triage with Bug Data Reduction," *IRJOET*, vol. 2, no. 4, July 2015.

[3] E.J.Whitehead, Jr. R.Akella and S.Kim S.Shivaji, "Reducing Features to Improve Code Change based Bug Predication," *IEEE Trans.Soft.Eng.*, vol. 39, no. 4, pp. 552-569, April 2013.

[4] Nilesh V.Alone Vaishnavi B.Sawant, "A Survey on Various Technique for Bug Triage," *IRJOET*, vol. 2, no. 9, pp. 917-920, Dec 2015.

[5] Z. Ren and Z.Luo J.Xuan, "Solving the Large Scale next Release Problem with a backbone based Multilevel Algorithm," *IEEE Trans.Softw.Eng*, vol. 38, no. 5, pp. 1195-1212, Sept/Oct 2012.

[6] T. M. Khoshgoftaar and K. Gao, "Attribute Selection and Imbalanced Data: Problems in Software Defect Prediction," *IEEE Trans.*, pp. 137-144, Oct 2010.

[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, "Advance in Instance Selection for Instance-based Learning Algorithms," *Data Mining Knowl.*, vol. 6, no. 2, pp. 153-172, April 2002.

[9] Y.Hu and H.Jiang W.Zou, "Towards Training Set Reduction for Bug Triage," *IEEE Trans.*, pp. 576-581, 2011.

[10] D. Lo and S. C. Khoo, "Mining Iterative Generators and Representative Rules for Software specification Discovery," *IEEE Trans.Knoel.Data Eng.*, vol. 23, no. 2, pp. 282-296, Feb 2011.