International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 04 Issue: 06 | June -2017 www.irjet.net p-ISSN: 2395-0072

# Bug Triage: An Automated Process

**Fareen Sayed[1], Hetvi Shah[2], Namrata Kharat[3] , Prajakta Ohal[4], Gopal Deshmukh[5]**

[1,2,3,4] *UG Student, M.E.S. College of Engineering, Pune, Maharashtra,India*
[5]*Dept. Of Computer Engineering, M.E.S College of Engineering,Pune,Maharahtra,India*

---

**Abstract -** *Bug triage means to transfer a new bug to expertise developer. The manual bug triage allocation process is opulent in time and poor in accuracy. There is a need to automatize the bug triage process. In order to automate the bug triage process, text classification techniques are applied using stopword removal and stemming. In the proposed paper, NB-Classifiers is used to predict the appropriate developers to whom the particular bug can be allocated. The data reduction techniques like instance selection and keyword selection are used to obtain bug report and words related to bugs. The proposed system will predict only those developers who are expertise in solving the particular bug. The status of bug will be tracked by the proposed system and if the bug is solved then the bug status will be updated as solved in the database. The solves bug will not be allocated to any developer in the future but if a particular developer fails to solve the bug then the bug will be allocated to another set of appropriate developers. The proposed system is built with intention to suggest or recommend the bug and not to manually assign it to any developer randomly. The proposed system will provide an efficient window to handle real time crisis that come up during project development lifecycle***.***

***Key Words*: *Data Reduction, Bug triage, Bug Report, Instance Selection, Keyword Selection, Text Classification.***

## 1. INTRODUCTION

An error or faults that occur in a system and causes unexpected results is nothing but a software bug. Software organizations spend a large amount of expense to handle software bugs. Bug fixing is an important part in every software companies. Current software frameworks consist of repositories that contain immense databases for storing the results of software frameworks. The old frameworks aren't entirely suitable for immense and complicated data in repositories. Data mining has developed encouraging ways to deal with software data. By using data mining methods, digging repositories can discover charismatic data and can solve realistic problems.

A bug repository contains bug reports and act as a crucial part in handling bugs. Bug reports or record consists of textual details of a bug. Large projects set out bug tracking systems to collect information and to aid developers to manage bugs. Challenges arise as there are large amount of bugs disclosed each day. A process of managing bugs is bug triage. In traditional organizations, new bugs are assigned manually to an expert developer. Due to manual assessment of bug, this process has low accuracy and also takes an inadequate amount of time. Also, sometimes the new bug is assigned to a developer who doesn't have much knowledge about that bug. Since the human/manual bug triage system had many disadvantages, existing work has proposed automatic bug triage approach.

In this approach, a bug report undergoes text classification techniques. In this paper, we are going to use two text classification techniques namely, stopword removal and stemming. Stopword removal will help to remove unnecessary words (e.g. 'a', 'the', 'this', and', etc.) from the bug report.

Stemming is useful to reduce words into its root form (e.g. words like 'engineering', 'engineered' share a common base- 'engineer'). After text classification, data reduction is used to address quality of data. In this paper, we will combine keyword selection algorithm and instance selection algorithm to obtain word dimension and bug dimension. This new bug report will be then given to NB- Classifier for prediction of developers. NB- Classifier will predict the top five developers based on their history and profile (skills).

The remainder of paper comprises of following sections. Section II shows literature survey. Section III contains proposed system followed by algorithms in Section IV. Section V includes system results and Section VI concludes.

## 2. RELATED WORK

[1] Jiefng Xuan et al. marks the issue of noisy and low quality of data. They employ data reduction strategies by combining instance selection and feature selection. They made use of predictive model to decide the order of instance selection and feature selection. They used text classification techniques and solved the bugs. In this paper, they have used Naïve Bayes algorithm to predict developers. Their work provided an approach to leveraging techniques on data processing to form reduced and high-quality bug data in software development and maintenance.

[2] In this paper they investigated whether we can accurately predict the severity of a reported bug by analysing its textual description using text mining algorithms. Based on three cases drawn from the open-source community (Mozilla, Eclipse and GNOME), they concluded that given a training set of sufficient size (approximately 500 reports per severity), it is possible to predict the severity with a reasonable accuracy (both

precision and recall vary between 0.65-0.75 with Mozilla and Eclipse; 0.70-0.85 in the case of GNOME).

[3] In this paper, they performed an empirical study of bug-fixing time for three CA Technologies projects. They proposed a Markov-based method for predicting the number of bugs that will be fixed in future. For a given number of defects, they proposed a method for estimating the total amount of time required to fix them based on the empirical distribution of bug-fixing time derived from historical data. They evaluated these methods using real maintenance data from three CA Technologies projects. The results show that the proposed methods are effective.

[4] in this paper, they dealt with data reduction problems. They combined feature selection with instance selection for data diminution to construct the reduced data set and improve the quality of bug data. The results showed that this data diminution technique will provide good quality of data and decrease scale.

[5] In this paper, they have presented a comparative analysis of different available information retrieval and machine learning methods in order to automate the assignment of bug reports to developers in an optimal fashion. The target was to find a combination of methods, which is most suitable to finally develop a high performance bug triage system. They used these five data sets and apply multiple machine learning algorithms. All of these seven ML algorithms belong to the class of supervised machine learning methods, and are used for single-label multi-class classification. The experimental results indicated that classification based on LSI and support vector machine has the best accuracy, precision, and recall values.

[6] proposed the concept for text representation and processing. They made used of distance graph to represent the documents in terms of distance between the distinct words. This provided a much richer representation in terms of sentence structure of the underlying data.

[7] they proposed four broad areas for improvements. To demonstrate the potential of this idea they conducted an initial study in which they simulated an interactive bug tracking system. This helped them to speed up the process of resolving bugs.

[8] they introduced a novel dataset mined from the Firefox bug repository (Bugzilla) which contains information about the temporal alignment of developer interactions. They presented a Firefox Temporal Defect Dataset (FTDD) and the rationale for selecting the key activities which it captures. Also they have described the FTDD schema and data sources. They presented an exploratory analysis on the dataset that we mined.

[9] The aim of this paper was to attain a data set reduction in bug triage by including the representative values along with the statistical values of the bug data set. They considered the dataset from the open source project Eclipse. They focused on reducing the data scale and thereby improving the accuracy. They achieved their aim by building a representative model for prediction of reduction orders by including the summary, metadata. Their proposed work attains an accuracy result of 96.5% that is better when compared with existing work.

[10] this paper presents a model for the developer prioritization in bug repositories by extending a socio-technical approach. They analysed three problems of the developer prioritization, namely the characteristics in products, the evolution, and the tolerance of noises. Based on the analysis, they investigated the ways to leverage the developer prioritization to improve three typical tasks in bug repositories. The results were studied on over 900000 bug reports in Eclipse and Mozilla.

## 3. PROPOSED SYSTEM

In traditional software establishment, new bugs are human triaged by a skilled developer. This approach has disadvantages, as there are large number of bugs and a shortage of expertise to fix all bugs. Also, it has poor accuracy and is time consuming. To desist from high cost of manual triage, an automatic way for bug triage process is proposed.

Fig. 1, illustrates the basic framework of bug triage based on text classification. A bug data set contains bug reports with respective developers. On this bug data set, the bug data reduction is applied as a phase in data preparation of bug triage. The proposed architecture combines existing techniques of instance selection and keyword (feature) selection to remove certain bug reports and words. A problem for reducing the bug data is to discover the order of instance selection and keyword selection, which is denoted as the prediction of reduction orders.

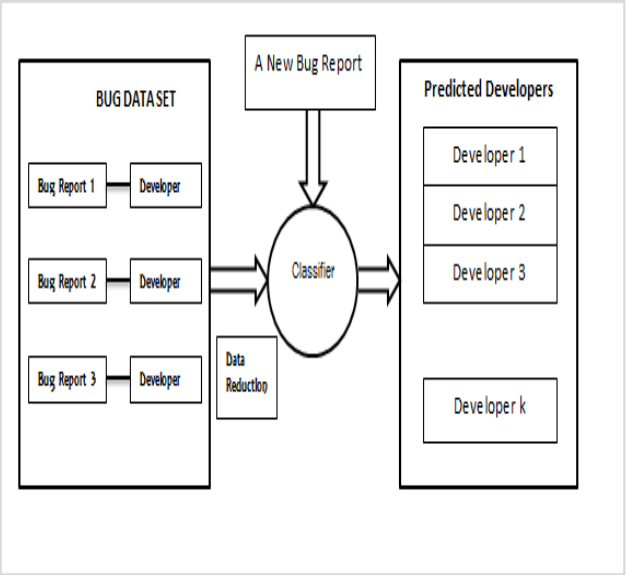



Fig. 1 Framework of Bug Triage Based On Text Classification.

In bug triage, a bug data set is converted into text matrix with two dimensions, namely the bug dimension and the word dimension. The combination of instance selection and keyword selection is used to generate a reduced bug data set. The original data set is replaced with the reduced data set for bug triage. For given data set in a certain application, instance selection is to obtain a subset of suitable instances while keyword selection aims to obtain a subset of suitable words in bug data set. Given an instance selection algorithm IS and keyword selection algorithm KS, we use KS!IS to denote the bug data reduction, which first applies KS and then IS; on the other hand, IS!KS denotes first applying IS and then KS.

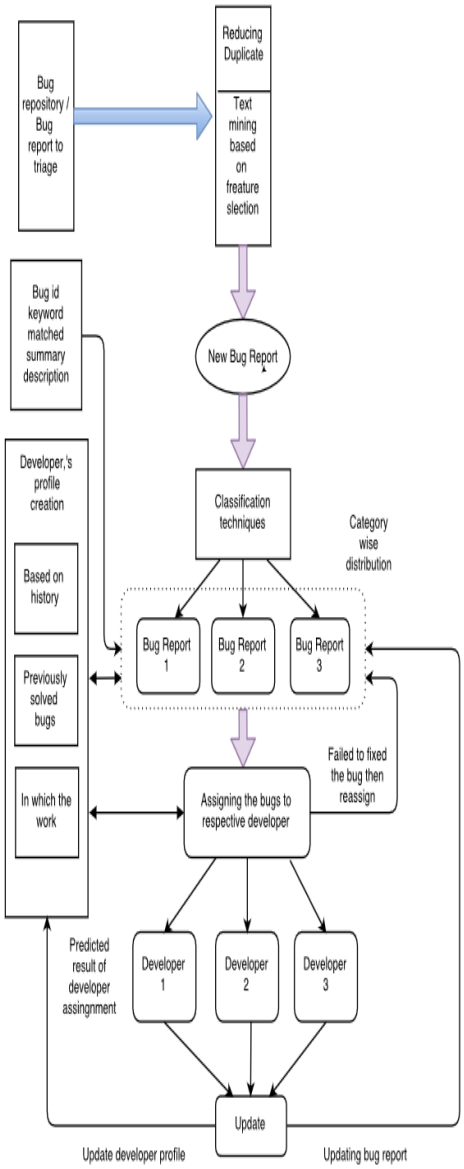



Fig. 2 System Architecture

## 4. ALGORITHMS

Proposed work consists of various algorithms that helps in predicting developers. The algorithms used are as follow:

### 4.1 Text-Classification

In order to automate the process of bug triage, text classification techniques are used. Text classification allots one or more classes to document based on their content. In our work, we are using stopword removal followed by stemming.

| | |
|---|---|
| Input | Bug Report:-<br>Bug ID : Unique Id<br>Summary : Description<br>Creation Time : YYYY/MM/DD<br>Status : New |

Table No.1 Input data for text classification

a) Stop word Removal

Words which don't import peculiar knowledge in bug report are known as stop words. For eg.,the, an, who, how,where,etc.

Steps for stop word removal algorithm:

i. Prepare stop word list using Word Net dictionary.
ii. Get statement from user.
iii. Convert statement into words.
iv. for each word in statement check if word is present in Stop WordsList.
v. if yes, then continue the loop
vi. else add word to preprocessedStatement
vii.end for
viii. preprocessedStatement is the final statement

b) Stemming

The preprocessedStatement is given to stemming for further process. Each term arriving is reduced into its root form through stemming process.

For example, 'danced','dancing' all share a equivalent root base :'dance'.

### 4.2 Data Reduction Techniques

After applying text classification via stopword removal and stemming the bug report undergoes data reduction for further process. Data reduction includes keyword selection(KS) and instance selection(IS) algorithm to obtain word dimension and bug dimension respectively. Data reduction techniques helps to improve scale and quality of bug report. Our work combines KS and IS.

a) Keyword Selection Algorithm

In keyword selection algorithm, we use CHI-square to determine whether there is a significant association between two variables. Similarly, the proposed system uses CHI-square test to determine the similarities between two developers. The developer profiles which matches the bug more appropriately is kept and the other profile is discarded. Thus, CHI-square test is used to reduce the number of developers that could be assigned to a particular bug. It includes four steps in determining the association and they are as follow:

1. State of Hypothesis: Suppose that variable A has keywords with r levels and variable B has developer's with c levels then,

Null hupothesis($H_o$): States that keywords and developers are independent.

Alternative hypothesis($H_a$): States that keywords and developers are not independent.

2. Formulate Analysis Plan: It describes how to use sample data to accept or reject the null hypothesis. The plan should specify significance level i.e any value between 0 and 1.

3. Analyze Sample Data:

Using sample data, we find degree of freedom, expected frequencies, test statics and P-value associated with test statistics. The illustration through formulas is as follow:

(a) Expected Frequency ($E_{r,c}$) = $(n_r * n_c)/n$ where,

$E_{r,c}$= expected frequency count for level r of variable A and level c for variable B.

$n_r$= total number of sample observed at level r of variable A.

$n_c$= total number of sample observed at level c of variable B.
n= total sample size.

(b) Test statistics: Chi-square random variable $X^2$ defined by the following equation:

$$X^2 = [(O_{r,c}-E_{r,c})^2 / E_{r,c}]$$

where,

$O_{r,c}$ = Observed frequency count at level r of variable A and level c of variable B.
$E_{r,c}$ = Expected frequency count at level r of variable A and level c of variable B.

4. Interpret Results

This involves comparing P-value to significance level and rejecting the null hypothesis when the P-value is less than significance level.

b. Instance Selection Algorithm

Our work uses cosine similarity to find how similar the text are purely in terms of word counts ignoring the words order. It compares two sets of bug reports. Set A contains new bug report and Set B contains existing bug report of the developer. Then using the formula:

$$\text{Cos } \theta = \frac{A.B}{|A|.|B|}$$

We check the similarity between two sets of report.If the similarity is greater than the threshold value, then the report is stated as duplicate and is given as a reference of previously solved report. Else it is passed to NB Classifier.

### 4.3 NB- Classifier

NB classifier algorithm is used to predict developers. Table No.2 shows learning data set which is given input to NB Classifier.

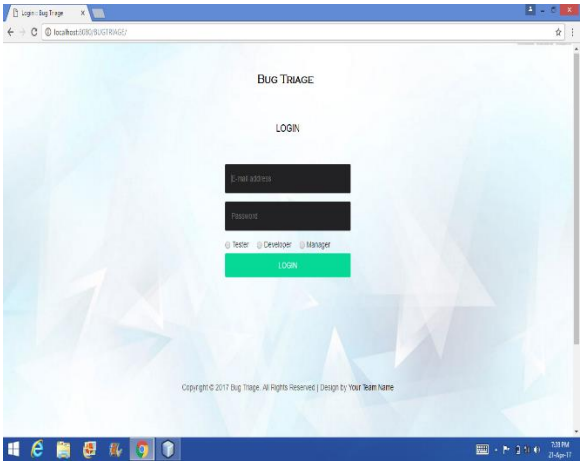
| | |
|---|---|
| Dev 1 | [Keyword Count] |
| Dev 2 | [Keyword Count] |
| . | |
| . | |
| . | |
| Dev n | [Keyword Count] |

Table No.2 Input for NB Classifier

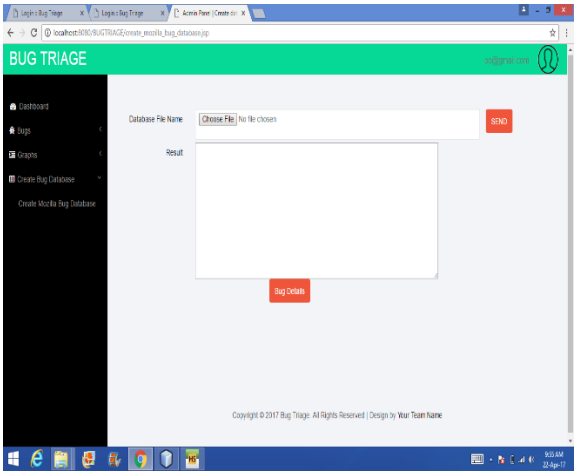
Along with this data the input also contains keyword count of new bug report. By using these counts we calculate prior, likelihood and posterior probability respectively. Based on maximum posterior probability the prediction order of developer is decide.

## 5. SYSTEM RESULTS

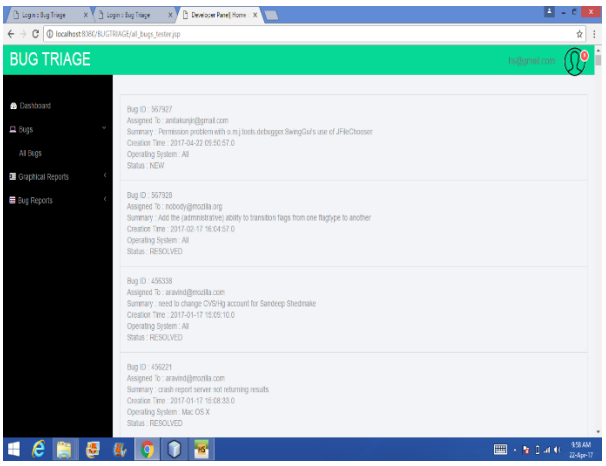
1] Screenshot showing the login screen




2] Screenshot showing the functionality of bug data set creation.

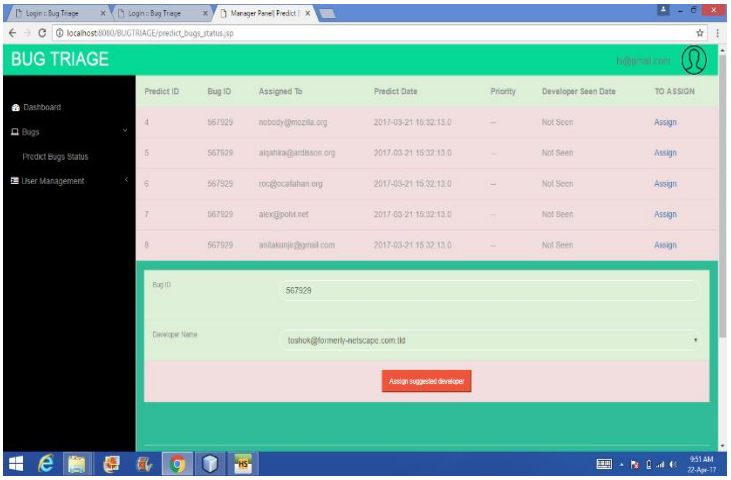



3] Screenshot showing the information of all bugs added by tester.




4] Screenshot showing the information of all the bugs in the system.


5] Screenshot showing the list of predicted developers for a particular Bug Id.




## 6. CONCLUSION

Bug fixing is an elegant part of software organization. One of the major challenges in process of bug triage is to assign a skilled developer to fix a new bug. In this paper, we presented the complete abstraction of an automatic bug triage approach.

We proposed a framework that removes the issue of data reduction to a huge extent. By combining KS and IS techniques, better quality of data can be obtained. In this paper, we used NB-Classifier which will help in suggesting a list of expert developers for fixing the bug.

In future work, we will evaluate the results of bug triage on open source project namely Bugzilla. Bugzilla is a testing tool developed and used by Mozilla. Also, we will explore and experiment on more advance methods to improve the existing functioning of our bug triage approach.

**Reference**

[1] Fareen Sayed, Hetvi Shah, Namrata Kharat,Prajakta Ohal,Gopal Deshmukh, "Automatic Bug Triage With Data Reduction", International Journal Of Engineering And Techniques, vol 2 Issue 6,2016.

[2] JifengXuan, He Jiang. Yan Hu, ZhileiRen, WeiqinZou, ZhongxuanLuo, and Xindong Wu,"Towards Effective Bug Triage with Software Data Reduction Techniques" IEEE Transaction on Knowledge and Data Engineering, vol. 27, no. 1,January 2015.

[3] 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.

[4] H. Zhang, L. Gong, and S. Versteeg, "Predicting bug-fixing time: An empirical study of commercial software projects," in Proc. 35th Int. Conf. Softw. Eng.,May 2013.

[5] Prof. A. Gadekar, N. Waghmare, P. Taralkar, R. Dapke, "Detecting and Reporting bugs by using Data Reduced technique", August 2015.

[6] S. N. Ashan, J. Ferzund and F. Wotawa, "Automatic Software Bug Triage System (BTS) Based on Latent Semantic Indexing and Support Vector Machine",2009.

[7] W. Zou, Y. Hu, J. Xuan, and H. Jiang, "Towards training set reduction for bug triage," in Proc. IEEE 35th Annual CS and Application Conference, Washington, DC, USA: IEEE Computer Society, 2011.

[8] Thomas Zimmermann,Rahul premraj, Jonathan Sillito,Silvia Brell, "Improving Bug Tracking Systems", ICSE'09,May 2016.

[9] Mayy Habayeb, Andriy Miranskyy, Syed Shariyar Murtaza, Leotis Buchanan, Ayse Bener," The Firefox Temporal Defect Dataset",12th Working Conference on Mining Software Repositories, IEEE ,2015.

[10] V Govindasamy, V Akila, "Data Reduction For Bug Triage Using Effective Prediction Of Reduction Order Techniques", International Conference on Computation of Power, Energy Information and Communication (ICCPEIC),2016.

[11] Jifeng Xuan ,He Jiang, Zhilei Ren ,Weiqin Zou," Developer Prioritization in Bug Repositories",ICSE,2012.

[12] E. Murphy-Hill, T. Zimmermann, C. Bird, and N. Nagappan, "The design of bug fixes," in Proc. Int. Conf. Softw. Eng., 2013, pp. 332–341.

[13] Stefan R¨abiger, Ataman Giris¸ken, and Cemal Yilmaz," How to Provide Developers only with Relevant Information?", 27th International Workshop on Empirical Software Engineering in Practice, IEEE, 2016

[14] C. C. Aggarwal and P. Zhao, "Towards graphical models for text processing",Knowl. Inform. Syst., vol. 36, no. 1, pp.1-21, July 2012.s

[15] 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.

[16] J. Anvik,"Automatic bug report assignment ," in Proc 28th International Conference on Software Engineering. ACM, 2006.