Text Mining of Bug Repositories

Komal
Scholar M.Tech. Computer Science & Engineering
Department of Computer Science and Applications, Kurukshetra University, Kurukshetra

Abstract— Text mining of bug repositories is an emerging area of research. Bug repositories are main source of knowledge. Different mining algorithms can be applied on these repositories to extract useful and interesting information. Finding frequently occurring bugs information helps the developers to plan the code to overcome the occurrences of the bugs in future projects. In this paper we propose a method to mine useful information from the bug repositories. Our method considers bug summary extracted from the bug repository on which text mining techniques are applied to find frequently used relevant words from the bug summary.

Keywords: - Bug repositories, Frequent Patterns, Text Mining.

I. INTRODUCTION

The target of Software Engineering (SE) is to develop software quality and productivity. Mining software engineering data has recently appeared as a promising way to meet this target. SE data can be used to grow understanding of software development, plan, predict, understand various aspects of a project, project management activities and support future development. Types of SE data include SRS documents, bug repositories, configuration management data, source code, mailing lists. Open source projects e.g. Firefox and Eclipse have open source bug repositories. User can report any type of bugs to these repositories. Users of these repositories are basically non-technical and cannot allocate correct class to these bugs. Different data mining algorithms like classification, frequent pattern mining, clustering can be applied on this data to discover interesting information.

In this paper, we propose a method to mine frequently used relevant words from the bug repositories. Bug is reported by the test engineers and users of the software product. These bugs are stored in the bug repositories which are tracked and managed by different tools such as Bugzilla etc. Bug database can be used to extract information, which can be used by the software developers to remove such bugs in the code in the future projects. In this paper, bug summary attribute of the bug database is used and text mining techniques are applied on bug database to find frequently used relevant words in bug summary. These frequently used relevant words can be used to find the associations among the words appearing in the bug report. This paper is organized as follows. Section II includes related work. Section III, includes proposed work. Section IV shows results and discussion. Section V discusses conclusions and future scope and references are specified in Section VI.

II. RELATED WORK

Bug repositories can be used to mine different itemsets. Jaweria Kanwal and Onaiza Maqbool projected a method based on support vector machines for managing bug repositories through bug report prioritization [Kanwal et al., 2010]. Naresh Kumar Nagwani, Ashok Bhansali projected a model to calculate software bug complexity using bug clustering and estimation [Nagwani et al., 2010]. Philipp Schugerl, Juergen Rilling, Philippe Charland planned a method for quality measurement in bug repositories [Schugerl et al., 2008]. Leon Wu, et.al. planned a tool BUGMINER, which is used to gain useful information from historic bug repository using data mining. This information is used to do redundancy check and completion check on a given or new bug report, and to approximate the bug report trend using statistical analysis [Leon Wu et al., 2011]. Michael Gegick, Pete Rotella, Tao Xie projected a approach that applies text mining on NLP descriptions of bug reports to train a statistical model on previously manually-labeled bug reports to recognize security bug reports that are manually-mislabeled as not-security bug reports(NSBR) [Gegick et al., 2010]. Security engineers can use the model to computerize the classification of bug reports from big and huge bug databases to reduce the time that they spend on searching for security bug reports (SBR). Xiaoyin Wang, Tao Xie, Lu Zhang, John Anvik and Jiasu Sun developed an approach for identifing duplicate bug reports using execution information and natural language [Wang et al., 2008]. When a new bug report arrives, its execution information and natural language information are compared with the existing bug reports. Then number of existing bug reports are given to the triager (A triager is a person who decides whether a given report should be worked on and who should work on it). Amir Michail, Tao Xie developed a method to help users evade bugs in GUI applications [Michail et al.,2005]. Users would use the application normally and report bugs that they meet to prevent anyone from meeting those bugs again. Syed Nadeem Ahsan et.al proposed a technique to mine the bug-fix data from the history of developer's bug fix data [Ahsan et al., 2010]. Ahmed Lamkanfi et.al. developed a method for calculating the severity of a reported bug by examining its textual description [Lamkanfi et al., 2010]. Adrian Schroter, Nicolas Bettenburg, and Rahul Premraj recognized the support of usefulness of stack traces to ECLIPSE developers by analyzing main patterns in the resolution of bug reports that enclosed stack traces [Schroter et al., 2010]. Marco D'Ambros, Michele Lanza and Romain Robbes projected a benchmark for defect prediction, in the form of a openly available data set consisting of several software systems, and give an extensive comparison of the predictive and explanative power of well-known bug prediction techniques [D'Ambros et al., 2010]. All the above methods are centered on application of classification, text mining and clustering techniques on the bug databases. In this paper, text mining techniques are used to extract frequently used relevant words from the bug repositories.

III. PROPOSED WORK

Bug repository is the database which stores the information relating to the bugs present in the software or product.

Typical attributes of bug database are: Bug ID (A unique identification number), Product (Name of product in which bug is present), Component(part of product in which bug is present), Bug Entry Time (Time of the bug is reported), Bug Modified Time (Time of the bug is modified), Bug Type (Type of the bug present), Bug Status (Typical values for bug status are NEW, FIXED, DUPLICATE, RESOLVED etc.), Bug Severity (Bug complexity), Bug Summary (Description of the bug) etc. From this database, bug summary attribute is considered for further work. First task is to extract useful knowledge from bug summary. After then find the frequency of words in summary, remove stop words from the text summary and apply the stemming process on preprocess summary. Bug summary has the description about bugs in the projects. The open source projects like Firefox, Eclipse etc. have open source code and bug repository. Any end user can report the bug in the bug repository and developer review the bug reports to handle the bugs in the projects.

The process involved in the proposed method is depicted in the following algorithms:-

A. Algorithm 1: Bug Summary Extraction.

Input: Database D of transactions, Specific Product attributes.
Output: Summary corresponding to attribute.

1. Start
2. Accept input_Atr (Specific Attribute type)
3. For (int i=0; i<= D.size; i++)
4. If (input_Atr = = D_Atr)
5. Extract bug summary corresponding to attribute from the database
6. Return summary
7. End

This algorithm returns all the bug summary of corresponding attributes and arranges the summary in the paragraph. This summary is presented in the form of unstructured document. Now text mining technique is applied on unstructured document to extract the useful information from the bug summary.

B. Algorithm 2: For Text mining of Bug Summary:

1. Start
2. Extract the bug summary corresponding to input attributes
3. Merge the extracted summary in a paragraph
4. Remove the stop words(e.g. prepositions) from summary
5. Find the frequency of words used in bug summary
6. Apply stemming technique on the words
7. Get all the relevant terms related to the bugs.
8. End

The bug description is first divided into number of words and some of the words (stop-words) are not useful in the study. Stop-words are those words which have low importance in the study of the bug reports and so these are removed. Stop-words can be the, a, an, and, are etc. The elimination of stop words doesn't make any difference in the programming. For example, when a client or user logged into the system with incorrect identification, an error note will be displayed as "Your access is denied". By this note, the user or client will come to know that he/she has entered the incorrect identification. In the above challenge, the system shows an error note such as "Access denied", the user gets the same meaning as above. From this case, the practical result is the presence of words "your" and "is" doesn't make any alteration in the process of accessing the system.

After stop words removal, Frequency of words is found out. Apriori algorithm can be used to find frequency of words because this algorithm is used to mine the frequent pattern analysis.

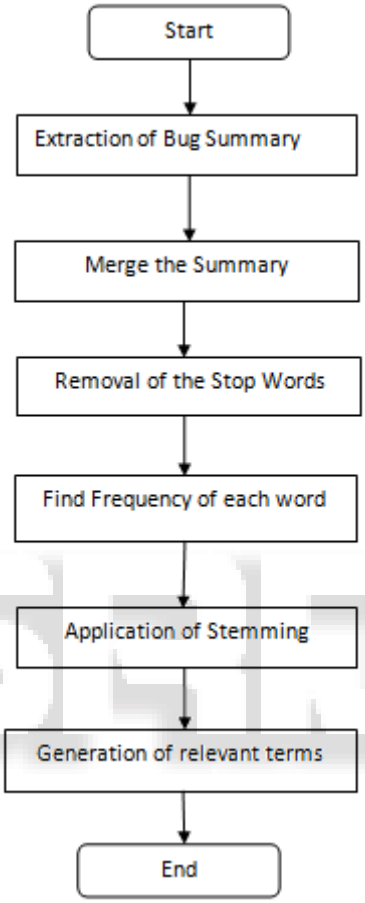

Fig. 1: Text Mining of Bug Summary

Stemming is the process of converting the derived words to their root word. When the process of stemming is applied to reading, reads etc. words, they are changed with its stem word "read" that means all the derived words are integrated to its base word. In this paper, Porter Stemmer algorithm is used to stem the bug words.

IV. Results And Discussion

Bug tracking dataset is used in this paper. Bug tracking dataset having different attribute such as product, component, status, resolution, Summary and its category stage. Product attribute has the name of open source products, components attribute has the name of components of products in which bug is present, status attribute has the information about the status of bug e.g. verified, resolved, new etc, resolution attribute has the information about resolution of bug e.g. fixed, invalid etc, summary attribute has the description of bug e.g. type of bugs and category attribute define the output class of bug if bug belongs to SBR then need to be handle soon and if bug belongs to NSBR these type of bugs can be removed later. Extract the

bug summary of respective attributes from the dataset. This summary is in unstructured format.

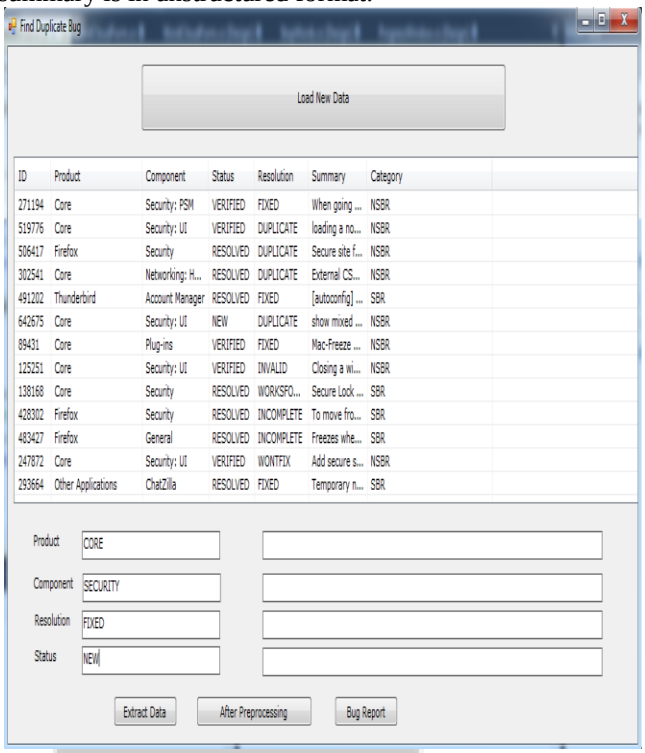

Fig. 2: Fetch the Summary of Respective Attribute

Fetch the summary for text mining in bug tracking system. Bug summary of different attributes is merged in to one paragraph.

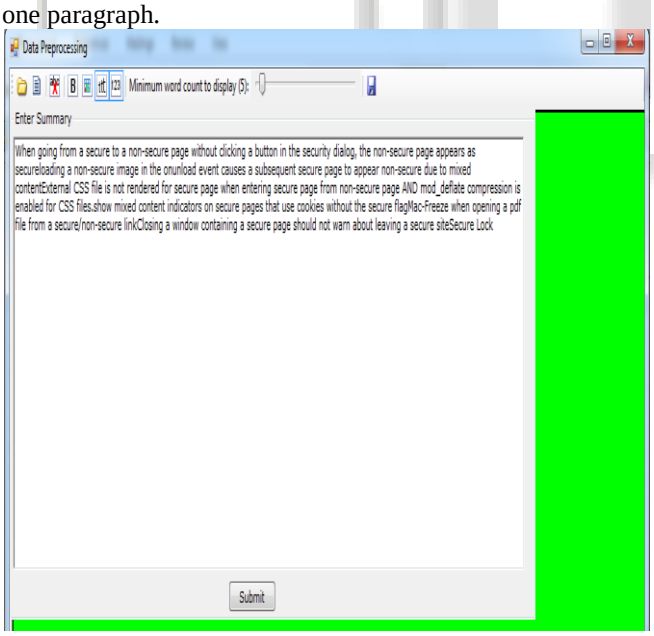

Fig. 3: Merge Summary

Now preprocessing of text is started and removes the stop words and also count the frequency of words in bug summary. Frequency for the words in whole dataset will rectify the maximum security words in dataset.

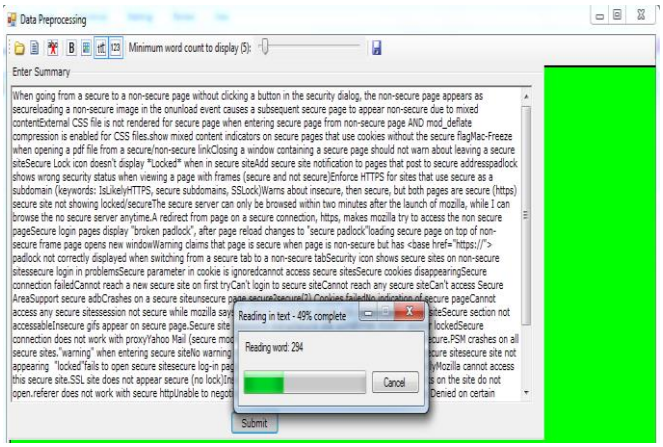

Fig. 4: Preprocessing of Summary (Stop Words Removal)

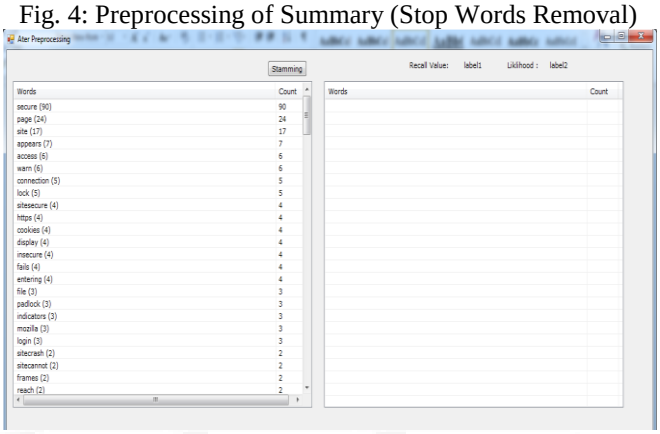

Fig. 5: Frequencies of Words

List of words is generated. After preprocessing, start the stemming process.

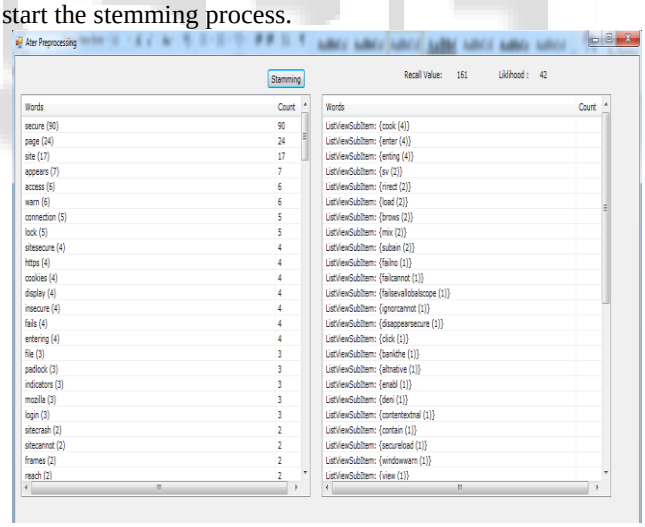

Fig. 6: after Stemming

Recall value is the total no of relevant words and these frequent relevant terms generated from the bug databases can be used to know the type of bugs generated. And this information is used by the developer to plan the code for the removal of bugs in future projects.

V. Conclusion And Future Scopes

The frequent terms generated from the bug dataset can be used to identify the type of bugs generated and this information is used by the developers to plan the code in the future software and projects, which can reduce the maintenance cost and improve the performance of the systems. These terms can be used to cluster the bug records

based on the similarity of words and also used to suggest the same resolution process for similar type of bugs.

REFERENCES

[1] Ahsan et al., 2010] Syed Nadeem Ahsan, Muhammad Tanvir Afzal, Safdar Zaman, Christian Gutel, Franz Wotawa, "Mining Effort Data From The OSS Repository of Developer's Bug Fix Activity", Journal of IT in Asia, Vol 3, pp:67-80, 2010.
[2] D'Ambros et al., 2010] Marco D'Ambros, Michele Lanza, "An Extensive Comparison of Bug Prediction Approaches", pp: 31-41, MSR 2010.
[3] Gegick et al., 2010] Michael Gegick, Pete Rotella, Tao Xie, "Identifying Security Bug Reports via Text Mining: An Industrial Case Study", pp 11-20, MSR 2010.
[4] Kanwal et al., 2010] Jaweria Kanwal, Onaiza Maqbool, "Managing Open Bug Repositories through Bug Report Prioritization Using SVMs", Proceedings of the 4th International Conference on Open-Source Systems and Technologies (ICOSST 2010) © ICOSST 2010, pp: 1-7, December, 2010.
[5] Lamkanfi et al., 2010] Ahmed Lamkanfi_, Serge Demeyer_y, Emanuel Gigery, Bart Goethalsz, "Predicting the Severity of a Reported Bug" pp: 1-10, MSR-2010.
[6] Leon Wu et al., 2011] Leon Wu, Boyi Xie, Gail E. Kaiser, and Rebecca J. Passonneau, "BUGMINER: Software Reliability Analysis Via Data Mining of Bug Reports" *SEKE*, Knowledge Systems Institute Graduate School, pp: 95-100, 2011.
[7] Michail et al.,2005] Amir Michail, Tao Xie "Helping Users Avoid Bugs in GUI Applications", ICSE'05, May 15-21, St. Louis, Missouri, USA, 2005.
[8] Nagwani et al., 2010] Naresh Kumar Nagwani, Ashok Bhansali, "A Data Mining Model to Predict Software Bug Complexity Using Bug Estimation and Clustering", International Conference on Recent Trends in Information, Telecommunication and Computing, pp:13-17, 2010.
[9] Schroter et al., 2010] Adrian Schroter, Nicolas Bettenburg, Rahul Premraj, "Do Stack Traces Help Developers Fix Bugs?" pp:118-121,MSR-2010.
[10] Schugerl et al., 2008] Philipp Schugerl, Juergen Rilling, Philippe Charland, "Mining Bug Repositories – A Quality Assessment", CIMCA, pp:1105-1110, 2008.
[11] Wang et al., 2008] Xiaoyin Wang, Lu Zhang, Tao Xie, John Anvik and Jiasu Sun, "An Approach to Detecting Duplicate Bug Reports using Natural Language and Execution Information" ICSE'08, May 10–18, Leipzig, Germany, 2008.

