IJITKM 0973-4414 Volume-10, Number-1 Jun-Dec 2016 pp. 6-9 **Impact Factor 2.8** available online at www.csjournalss.com

Review on Bug Detection in Text Based Data Mining

Ms. Hina[1], Gurdev Singh[2]

[1] M.Tech student, Department of CSE Jind Institute of Engineering & Technology, Jind (Haryana)

[2]Assistant Professor, Department of CSE, Jind Institute of Engineering & Technology, Jind (Haryana)

Abstract: Data mining requires data preparation which could uncover information or patterns which might compromise confidentiality & privacy obligations. A common way for this to occur is through data aggregation. Data aggregation involves combining data together (possibly from various sources) in a way that facilitates analysis (but that also might make identification of private, individual-level data deducible or otherwise apparent). One such application of Text Data Mining is bug detection. Bugs are hard to find & usually winds up programmer. In order to alleviate overhead in debugging, we present an approach to detect bugs in C programs via matching & mining techniques. Input for system is a text file containing errors, principally syntax errors from a C program which is matched for similar but slightly different code fragments in database that acts a repository of errors aptly classify them & generate an analysis report predicting solutions for same.

[1] INTRODUCTION

Humpherey [1] introduced about software bug. It is a major bug in coding implementation because without correct code we do not found correct result. Software engineering teams have bug report in which these type bugs mentioned. In absent of perfect (required) quality of software, customer does not satisfy. Help of software tracker software engineer easily detect error as a software defect & its type. Software bug report is known as problem report but by help of data mining easily we detect & analyzed bug. general idea for achieving error detection is to add some redundancy to a message, which receivers could use to check consistency of delivered message, & to recover data determined to be corrupted. Error-detection schemes could be either systematic or non-systematic: In a systematic scheme, transmitter sends original data, & attaches a fixed number of check bits which are derived from data bits by some deterministic algorithm. If only error detection is required, a receiver could simply apply same algorithm to received data bits & compare its output within received check bits; if values do not match, an error had occurred at some point during transmission. In a system that uses a non-systematic code, original message is transformed into an encoded message that had at least as many bits as original message. Based complexity program error would different but each language had predefined syntax & rules & regulation to type code using that compiler trace code line by line. Suppose if any problem occurs in object code, it won't execute properly. To avoid these problems they use an online solution to overcome existing problem.

Code which contains problem would take as input for searching after which text mining techniques were applied to find solutions.

[2] ERROR DETECTION

The general idea for achieving error detection is to add some redundancy to a message, which receivers could use to check consistency of delivered message, & to recover data determined to be corrupted. Error-detection schemes could be either systematic or non-systematic: In a systematic scheme, transmitter sends original data, & attaches a fixed number of *check bits* which are derived from data bits by some deterministic algorithm. If only error detection is required, a receiver could simply apply same algorithm to received data bits & compare its output within received check bits; if values do not match, an error had occurred at some point during transmission. In a system that uses a non-systematic code, original message is transformed into an encoded message that had at least as many bits as original message.

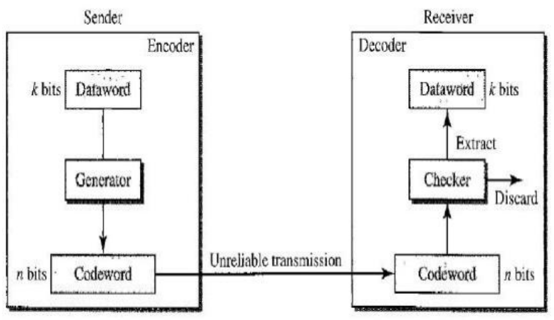

FIG 1 Error Detection

Good error control performance requires scheme to be selected based on characteristics of communication channel. Common channel models include memory-less models where errors occur randomly & within a certain probability, & dynamic models where errors occur primarily in bursts. Consequently, error-detecting & correcting codes could be generally distinguished between *random-error-detecting* & *burst-error-detecting*. Some codes could also be suitable for a mixture of random errors & burst errors.

[3] Error Reporting Framework

Our error reporting framework consists of generic operation types to capture information about progressing stages of every operation supported by different protocols.

The main purpose of classifying data transfer operation in several categories is to better understand at which stage an error has occurred. File transfer protocol such as GridFtp [3], will generate error codes & error messages. However, proposed error reporting framework will help both users & higher level planners to recognize error condition such that in respect of stage where error occurred different actions could be taken. As an example, a directory transfer operation could fail since used file transfer protocol is not supporting directory listing. In such a case, error will fail in status phase be-fore proceeding to transmit phase.

We propose to examine availability of remote server & functionality of file transfer service before initiating transfer. As it has been discussed in previous section, we could easily test whether we could access remote host over network. By using network exploration techniques, we could also detect available services running on remote site. One further step is to examine functionality of file transfer protocol such that we ensure it is responding as expected before starting actual data transfer job. This could be accomplished by some simple file transfers or by executing basic functions in client interface of protocol.

A data transfer operation which passed all initial tests & which has been started, could fail after transferring some amount of data. In order to better understand reason behind failure, we go further & perform initial tests again after an error occurred.

[4] ERROR DETECTION ANALYSIS

We had presented approach based on grouping similar reports in repository & deriving a centroid of clustered reports. An incoming report (represented as a document vector) is then compared to centroids of bug report groups to compute a similarity score for detecting duplicates based on a predefined threshold value. In general, mining algorithms fall into four main categories: Frequent pattern mining—finding commonly occurring patterns; Pattern matching—finding data instances for given patterns; Clustering—grouping data into clusters; and Classification—predicting labels of data based on already-labeled data. The final step transforms mining algorithm results into an appropriate format required to assist SE task. In [2], aims to help software developer in aspect of code debugging. Code debugging is an eminent phase of software development. This paper gives some solution to rectify code debugging problem using data mining technique. A class of bugs which is particularly hard to handle is called crash. Occasional bugs that are failures which lead to faulty results within some but not within any input data. Non-crashing bugs in general are already hard to end. This is because no stack trace of failure is available. With occasional bugs, situation is even more difficult, as they are harder to reproduce. Developers usually try to find & rectify bugs by doing an in depth code review along within testing methods & classical debugging strategies. Since such reviews are very expensive, there is a need for tools which localize pieces of code that are more likely to contain a bug. Here they have used combination of Hierarchical clustering algorithm & APRIORI algorithm. Based complexity program error would different but every language has predefined syntax & rules & regulation to type code using that compiler trace code line by line. Suppose if any problem occurs in object code, it won't execute properly. To avoid these problems they use an online solution to overcome existing problem. The code which contains problem would take as input for searching after which text mining techniques were applied to find solutions.

[5] PROBLEM STATEMENT

Our research is for less error prone classification by reducing misclassification. Misclassification is defined as when legitimate emails are categorized as junk emails or vice versa. Cost of misclassifying legitimate emails as junk is much higher than cost of junk mails as legitimate mails. Classification scheme which will provide probability for its classification decisions Cost of these two kind of misclassification errors In reveals challenges in pattern matching & shows comparison between existing & a proposed algorithm for finding out matched links within given number of links. Text mining is an important step of knowledge discovery process. It is used to extract hidden information from not-structured or semi-structured data. They had been developed a crawler for getting pattern matched within a given text by using several algorithms such as:

1. Finite automata algorithm
2. Knuth-Morris-Pratt algorithm(KMP)
3. Boyer-Moore algorithm

[5] CONCLUSION

Mining software repositories is an emerging field that had received significant research interest in recent times. Several tools & techniques based on data mining approaches had been developed to assist a practitioner in decision making & automating software engineering tasks. In this research we would apply data & text mining for task of duplicate bug report detection. We had presented approach based on grouping similar reports in repository & deriving a centroid of clustered reports. An incoming report (represented as a document vector) is then compared to centroids of bug report groups to compute a similarity score for detecting duplicates based on a predefined threshold value.

REFERENCES

1. M. O. Mansur, 2 Mohd. Noor Md. Sap 2005 Outlier Detection Technique in Data Mining: A Research Perspective
2. Wahidah Husain1, Pey Ven Low2, Lee Koon Ng3, Zhen Li Ong4, 2011 Application of Data Mining Techniques for Improving Software Engineering
3. Ashish Sureka 2011 Detecting Duplicate Bug Report Using Character N-Gram-Based Features we present an approach to identify duplicate bug
4. V. Neelima1, 2013 Annapurna Bug Detection through Text Data Mining Volume 3, Issue 5, might 2013 ISSN: 2277 128X International Journal of Advanced Research in Computer Science & Software Engineering
5. Promila Devi1, Rajiv Ranjan 2014 Enhanced Bug Detection by Data Mining Techniques
6. Safia Yasmeen 2014 Software Bug Detection Algorithm using Data mining Technique
7. Dhyan Chandra Yadav April 2015 Software Bug Detection using Data Mining International Journal of Computer Applications (0975 – 8887) Volume 115
8. Damini.V.S1, Rabindranath2, Priyashree.K3, .Jayashubhaj.K4 10, might 2016 Optimized Error Detection Analytics within Big data on Cloud International Journal of Innovative Research in Science, Engineering & Technology Vol. 5, Special Issue 10, might 2016
9. Tao Xie & Suresh Thummalapenta, North Carolina State University, David Lo, Singapore Management University, Chao Liu, Microsoft Research ―Data Mining in Software Engineering‖ , August, 2009, pp. 55-60
10. S.Suyambu Kesavan, Dr.K.Alagarsamy ―SE Code Optimization using Data Mining Aproach‖ , International Journal of Computer & Organisation – Volume 2, Issue – 3, 2012, pp. 65-68
11. Abhay Bhatia, Shashikant, Robin Choudhary ―Comparative Study of Pattern Matching Using Text Mining‖ , IJRREST: International Journal of Research Review in Engineering Science & Technology (ISSN 2278- 6643) Volume-1 Issue-1, June 2012, pp 58-61
12. P. Adragna, Queen Mary, University of London, ―Software Debugging Techniques", pp. 72-80
13. Andreas Hotho, Andreas Nurnberger, Gerhard PaaB, ―A brief of Text mining‖ , might 2005
14. Sarah K Kummerfeld & Judy Kay, ―The neglected battle fields of Syntax Errors‖ , 2006
15. Yiannis Kanellopoulos & Christos Tjortjis, ―Data Mining Source Code to Facilitate Program Comprehension:Experiments on Clustering Data Retrieved from C++ Programs‖