International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

COPY RIGHT

IJIEMR Transactions, online available on 16th July 2017. Link :
http://www.ijiemr.org/downloads.php?vol=Volume-6&issue=ISSUE-5

Title: Towards Effective Bug Triage With Software Data Reduction Techniques.

Volume 06, Issue 05, Page No: 1854 – 1861.

Paper Authors

***MISS. SUNITA MAHADEO JADHAV, MR.JAGDISH PIMPLE.**
* Dept of CSE, Nagpur Institute of Technology.

USE THIS BARCODE TO ACCESS YOUR ONLINE PAPER

To Secure Your Paper As Per UGC Guidelines We Are Providing A Electronic Bar Code

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

TOWARDS EFFECTIVE BUG TRIAGE WITH SOFTWARE DATA REDUCTION TECHNIQUES

***MISS. SUNITA MAHADEO JADHAV,**MR.JAGDISH PIMPLE**
*PG Scholar, Dept of CSE, Nagpur Institute of Technology,Nagpur.
**Asst.Professor, Dept of CSE, Nagpur Institute of Technology,Nagpur.

ABSTRACT

In this paper, we deal with the software bugs where large software company spent lot many of their cost in the same. The step of fixing the bug is called as bug triage where we correctly assign a developer to a new bug. Here, we address the problem of data reduction for bug triage. The problem of data reduction deal with how to reduce the scale and improve the quality. Hence, we combine instance selection with feature selection both simultaneously to reduce bug dimension and word dimension. We also extract the historical bug data set and predictive model to build new data set. This work provides leveraging techniques on data processing for high quality bug data in the software development.

Keywords— bug triage,bug repositories, bug data reduction,feature selection,instance selection

I. INTRODUCTION

Software repositories are large-scale databases for storing the output of software development, e.g., source code, bugs, emails, and specifications, in the modern software development. Software repositories mining are an interdisciplinary domain which helps in data mining to deal with software engineering problem. Software analysis is very complex and not suitable for large scale and complex data in software repositories. So to handle this, data mining has emerged. Real world software problems and to uncover interesting information in software repositories, we have leveraging data mining techniques. Once a software bug is found, the bug gets reported to the bug repository. Once a bug report is formed, a human triage assigns that bug to a person who will try to fix the bug who is a developer .The developer is recorded in an item assigned-to. If the previously assigned developer cannot fix this bug, the assigned-to will change to another developer. This process of assigning a correct developer for fixing the bug is called as a bug triage. A developer starts to fix the bug based on the knowledge of historical bug fixing. Again developer has to pay efforts to understand the new bug report and examine historically fixed bugs. A human triage assigns new bugs by her or his expertise based on results of text classification techniques. We have techniques like collaborative filtering approach to improve the accuracy of text classification techniques for bug triage. Here, we address the problem of data reduction for bug triage i.e. to improve the quality to facilitate the process of bug triage and to reduce the bug data to save the labor cost. Data reduction for bug triage aims to build a small-scale and high-quality set of bug data by removing bug reports and words, which are redundant or non-informative. Here, we combine existing techniques of instance selection and feature selection to simultaneously reduce the bug dimension and the word dimension. The reduced bug data contain fewer bug reports and fewer words than the original bug data and provide similar information over the original bug data. We evaluate the reduced bug data according to two criteria: the scale of a data set and the accuracy

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

of bug triage. Our objective is, to reduce the scale of the data as well as increase the quality of the data by using instance selection and feature selection, to reduce the bug data to save the labor cost of developers and improve the quality to facilitate the process of bug triage. Also, we aim to augment the data set to build a preprocessing approach, which can be applied before an existing bug triage approach and to improve the results of data reduction in bug triage to explore how to prepare a high quality bug data set and tackle a domain-specific software task.

A. Related Work

Sandusky, Gasser and Ripoche gave the feature of the defect tracking repositories, which showed the evolvement of the bug report network which shows formal and informal relationship in the bug data and also to examine the dependency among the bug report [2].

Q. Hong, S. Kim et.al gave theory for understanding large software development and maintenance demand participation of group of developers for improving development and maintenance quality and reduction cost[3]. J. Xuan, H. Jiang et.al identified the developer prioritization which can distinguish developers and assists task in software maintenance [4]. Thomas Zimmermann and Silvia Breu suggested that many follow-up questions are needed to be posed to the reporters of bug. So, there is high need of efficient and effective communication with the teams in open source project. These gave the high results of bug fixing activities and have updates of the bug. Integration and active participation of users in bug tracking will result in efficient bug tracking.[11].

The software development, bug provides vital information to developer. But, they differ in quality. Zimmerman and Battenberg identified the quality of bug data, the designing questionnaires to developers and users. Based on this questionnaires, we can characterize what makes a good bug report and what classifier are to be trained to identify the quality of bug and strategies to improve them[5].

To detect the duplicate bug report which weakens the quality of bug, Sun and Jaing described duplicate bug detection approach by optimizing a retrieval function on multiple features [6].

D. Cubranic and G. C. Murphy solved the problem of assigning text document into one or more categories or classes, we apply the text classification on this [7].

J. Anvik and G. C. Murphy recommended applying bug triage which improves the software development process. A triage determines, if the report are meaningful, these report are then organized for integration of the project's development process [8].

A. Lamkanfi, S. Demeyer predicted that the severity of reported bug is a critical factor and decides that soon it needs to be fixed. We can do this by textual description using text-mining algorithms[9].

Rogati and Yang reported a four well-known classification algorithms a Naïve Bayesian(NB) approach, a Rocchio-style classifier, a k-Nearest Neighbour(kNN) method and a Support Vector Machine(SVM) system. This gave analysis of four classification algorithms and mapped the performance measures[12].

Above result gave that the Naïve Bayes classification improves with smaller training set. The data set was divided into a test set and train set by randomly selecting a percentage of a documents from the data set. The data set are to be placed into train set and classification is performed. J.A. Olvera-Lopez and J.A. Carrasco reviewed with the instance selection algorithms and we have selected Learning Vector Quantization for the instance selection [14].

We address the problem of data reduction for bug triage. In contrast to bug triage, defect prediction is a binary classification problem,

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

which aims to predict a software artifact contains flaws according to extracted features. So, T. M. Khoshagotkar, K. Gao, N. Seliya examine the techniques on feature selection to handle imbalanced data [10].

II. PROBLEM STATEMENT

A. Existing System

Once a software bug is initiate, a reporter or a developer records this bug to the bug depository. A recorded
bug is called a bug information, once a bug report is produced, a human triager assigns the bug to a developer, who will fix this bug. The assigned-to will search for another developer if the before assigned developer cannot fix this bug. This assigning of bug or fixing the bug is called bug triage. Typically, the developer pays efforts to recognize the new bug report and to inspect traditionally fixed bugs as a reference (e.g., searching for similar bugs and applying accessible solutions to the new bug. Existing work uses text categorization to assist bug triage. In such approaches, the summary and the explanation of a bug report are extracted with the textual content. It gives low accuracy.

Existing approach is based on text classification to assist bug triage, in such approaches; the summary and the description of a bug report are extracted as the textual content while the developer who can fix this bug is marked for classification which gives low accuracy.

B. The Lifecycle of a Bug Report

Bug has to from several states. When a bug report is submitted to the repository, its status is set to NEW. Once a developer has been either assigned to the status is set to ASSIGNED. When a report is closed its status is set to RESOLVED. It may further be marked as being verified (VERIFIED) or closed for good (CLOSED). If the resolution resulted in a change to the code base, the bug is resolved as FIXED. When a developer determines that the report is a duplicate of an existing report then it is marked as DUPLICATE. If the developer was unable to reproduce the bug it is indicated by setting the resolution status to WORKSFORME. If the report describes a problem that will not be fixed, or is not an actual bug, the report is marked as WONTFIX or INVALID, respectively. A resolved report may be reopened at a later, and will have its status set to
REOPENED.

C.Proposed System

Here, present the problem of data reduction for bug triage. We have two aims of the system with the data set of bug triage 1) to simultaneously reduce the scales of the bug dimension and the word dimension and 2) to improve the accuracy. We apply the existing techniques of instance selection and feature selection to remove certain bug report and words.

The main objective of proposed system is to reduce reassignment of the bug triage. The below figure shows the system architecture of proposed system.

Bug triage consists of the bug data set, classifier and the predicted results. In bug data reduction, we are using the instance selection and feature selection. With the help of instance selection and feature selection we get low scale as well as good quality data. Based on the attributes of the bug data set that are historical bug data set, which are for the prediction of reduction order.

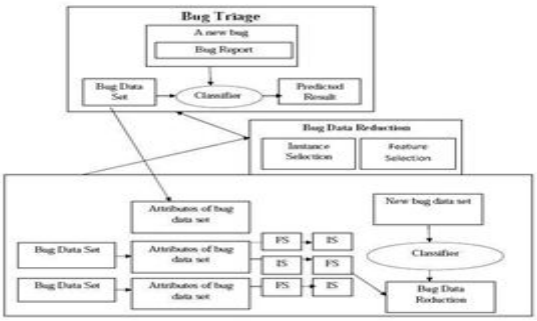

Fig.1 System Architecture

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

III. DATA REDUCTION FOR BUG TRIAGE

Here, we present an algorithm which shows how to apply instance selection and feature selection which is data reduction of bug triage.

 Algorithm for Data Reduction Data reduction based on FS->IS :

Input: training set T with n words and m bug reports, reduction order FS->IS final number n_F of words, final number m_I of bug reports,
Output: reduced data set T_{FI} for bug triage

1) Calculate objective values n for all the words by applying FS to n words of T;
2) Generate a training set TF by selecting top n_F *words* ;
3) Apply IS to m_I bug reports of T_F ;
4) When the number of bug reports is equal to or less than m_I, terminate IS and generate the final training set T_{FI} .

A. Applying instance selection and feature selection

In bug triage, a bug data set is converted into a text matrix with two dimensions which are called the bug dimension and the word dimension. In our work, we take advantage of the combination of instance selection and feature selection technique which will generate a reduced bug data set. We then replace the original data set with the reduced data set for bug triage. These techniques are most popular and frequently used in data processing. For a given data set, instance selection is to give a subset of relevant instances (i.e., bug reports in bug data) while feature selection aims to give a subset of relevant features (i.e., words in bug data).

We give the following denotation to distinguish the orders of applying instance selection and feature selection. Given an instance selection algorithm IS and a feature selection algorithm FS, we use FS->IS to denote the bug data reduction, which first we have to apply FS and then IS; on the other hand, IS->FS denotes first applying IS and then FS. In Algorithm, we briefly present how to reduce the bug data based on FS-> IS. Given a bug data set, the output of bug data reduction is a new and reduced data set. Two algorithms FS and IS are applied sequentially. That is in Step 2, some of bug reports may be blank during feature selection because all the words in a bug report are removed. Such blank bug reports are also removed in the feature selection. Here, FS-> IS and IS -> FS are viewed as two orders of bug data reduction. We introduce these algorithms as follows:

B. Instance Selection

Instance selection is a technique to reduce the number of instances by removing noisy and redundant instances. Reduced data set by removing non-representative instances. There are four instance selection algorithms, namely Iterative Case Filter (ICF), Learning Vectors Quantization (LVQ), Decremented Reduction Optimization Procedure (DROP), and Patterns by Ordered Projections (POP).

C. Feature Selection

Feature selection is a preprocessing technique for selecting a reduced set of features for large-scale data sets. The reduced set is considered as the representative features of the original feature set. We focus on the feature selection algorithms in text data. There are four well algorithms in text data and software data, Information Gain, x2 statistic, Symmetrical Uncertainty attribute evaluation (SU), and Relief-F Attribute selection (RF). Bug reports are sorted according to their feature values also on given number of words with large values is selected as representative features based on feature selection.

D. Benefit of Data Reduction

We have two benefits of the Data Reduction method namely, reducing the data scale and improving the accuracy. 1. Reducing the data scale.We save the labor cost of developers by using reducing the scale of data. Bug dimension: The aim of bug triage is to assign developers for bug fixing. Word dimension:

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

We use feature selection to remove noisy or duplicate words in a data set. The reduced data set can be handled more easily by automatic techniques (e.g., bug triage approaches) than the original data set based on feature selection. Besides bug triage, the reduced data set can be further also be used for other software tasks after bug triage.

2. Improving the Accuracy

. Data reduction removes noisy or duplicate information in data sets. This can help in improving accuracy.Bug dimension: Instance selection technique provides an ease in removal of uninformative bug reports; on the contrary, removal of bug reports may results in decreased accuracy. Word dimension: Removing uninformative words can help in improving accuracy of bug triage. This can recover the accuracy loss by instance selection.

IV. IMPLEMENTATION

Instance selection and feature selection are widely used techniques in data processing. Here first we apply instance selection on the given data set. This instance selection is used to reduce the number of instances by removing noisy and redundant instances. According to existing algorithms, here we are applying Learning Vectors Quantization (LVQ).

Learning Vectors Quantization (LVQ) is the higher version of vector quantization.This uses the class information to reposition the Voronoi vectors slightly,so as to improve the quality of the quality of the clasifier decision regions. In vector quantization, we assume there is a codebook which is defined by a set of M prototype vectors. (M is chosen by the user and the initial prototype vectors are chosen arbitrarily).

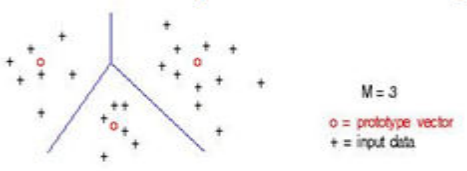

Fig.2. Codebook of vectors

An input belongs to cluster i if i is the index of the closest prototype (closest in the sense of the normal euclidean distance). This has the effect of dividing up the input space into a Voronoi tesselation.

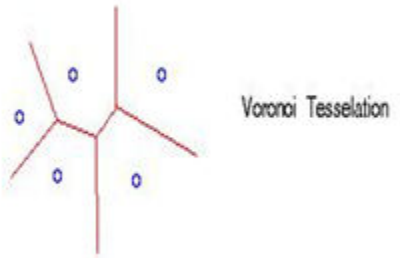

Fig.3.Voronoi Tessellation

A. Algorithm for LVQ

1. Choose the number of clusters M.
2. Initialize the prototypes $w*_1 \ldots w*_m$ (one simple method is to choose *m* vectors from the input data).
3. Repeat until stopping criteria is satisfied:

- Randomly pick input x,
- Determine the "winning" node k by finding the prototype vector that satisfies,

$|W* k - x| <= |W * i|$ ……. *(For all i)*

4. Update only the winning prototype weights according to,

$W * k (new) = w*k (old) + \mu(x - W*k (old))$

This is called standard competitive learning rule.

V. MATHEMATICAL MODEL

A. Set Theory:

Let,

$S = \{I, P, IO, O\}$

Where,

S = Set of bug report (text file)

I = Set of inputs

P = Set of processes

IO = Intermediate output

O = Set of outputs/ final outputs

1. $I = \{i\}$

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

Where, i = Folder containing bug file in text format

2. $P = \{p_1, p_2, p_3, p_4\}$ Where, $p1$ = Load data set,
p_2=Vector quantization and applying LVQ algorithm (instance selection),
p_3 = Feature selection,
p_4 = Result in the cluster file (new report).
3] $IO = \{io_1, io_2, io_3\}$
Where, io_1 = Check the file format, io_2 = Data reduction,
io_3 = Checking the path for output file. 4] $O = \{o\}$
Where, o=cluster of document (new bug report).

B. Venn Diagram

Venn diagram shows the mapping of the input, process and output relation of the system. It also represent the interaction between different processes along with input and output.

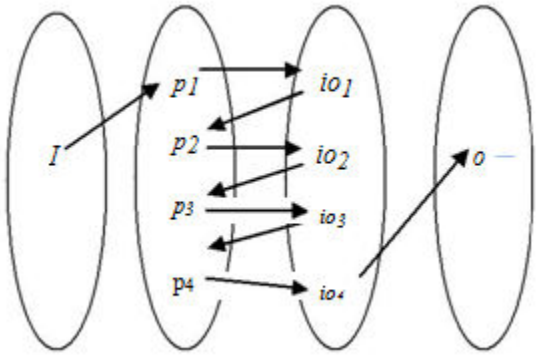

Fig.4. Venn diagram

C. Process State Diagram

Here, process p_1, p_2, p_3, p_4 are denoted by q_1,q_2,q_3 and q_4 respectively where
q_4 is a final state and q_0 is a initial state, $q1$ = Load data set,
q_2=Vector quantization and applying LVQ algorithm (instance selection),
q_3 = Feature selection,
q_4 = Result in the cluster file (new report).

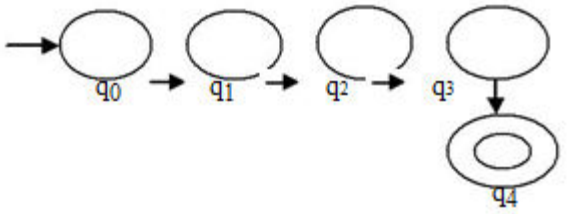

Fig.4. Process State Diagram

D. Time Complexity

The time complexity can be calculated by summing the time complexities of all the processes;
$T=\sum_i^4 T(q_i)$ $T=T(q_1)+T(q_2)+T(q_3)+T(q_4)$
T= O(n)+O(n)+O(n)+O(n) Therefore, time complexity is
T=O(n).

VI. RESULT

Here, we have implemented the module for instance selection using the LVQ algorithm. And study of the feature selection is on the track and algorithms for same have to be tested. The below diagram shows the sequence diagram for instance selection.

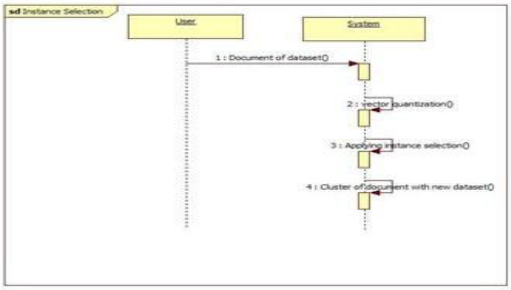

Fig 5.Sequence diagram for instance selection.

A. Testing Result Analysis

At the initial stage, we have the folder of text file which contains the multiple text file which the bug report. Then after loading the data set, we have to apply vector quantization by applying the LVQ algorithm. Then we generate the cluster of document which contains only the instances and get data reduction for the input file.

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

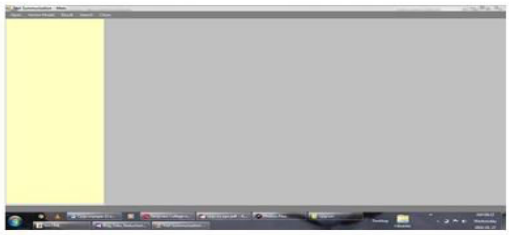

Fig 6. Initial window for instance selection

Fig 6. Shows the initial window for the instance selection which has the tab to load file and perform vector model. And the fig 7. shows the loading of data set from the specified folder

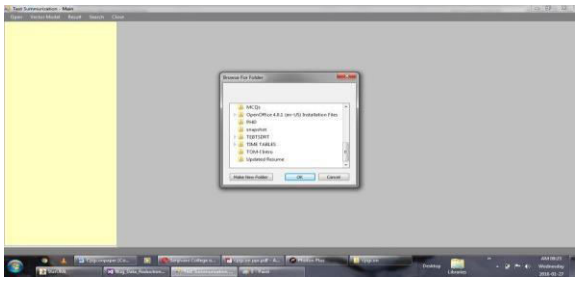

Fig 7. Loading the data set

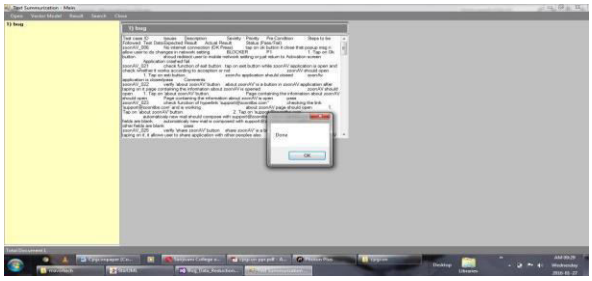

Fig 8. Text files of bug report

After selection of the text file, we perform the vector quantization for the same and we get output in text file with reduced data set.

VII. CONCLUSION

In this paper, we combine feature selection with instance selection to reduce the data scale of bug data set and also to improve the data quality. To investigate the data reduction for bug triage, we use the bug repository of the large software project and perform the data reduction and get the new bug report. Our work provides an approach to leveraging techniques on data processing to form reduced and high-quality bug data in software development and maintenance.

REFERENCES

[1] Jifeng Xuan, He Jiang, Yan Hu, Zhilei Ren, Weiqin Zou, Zhongxuan Luo, and Xindong Wu, "Towards Effective Bug Triage with Software Data Reduction Techniques" IEEEknowledge and data engineering, vol. 27, no. 1, January 2015 transactions on

[2] R. J. Sandusky, L. Gasser, and G. Ripoche, "Bug report networks: Varieties, strategies, and impacts in an F/OSS development community," in Proc. 1st Intl. Workshop Mining Softw. Repositories, May 2004.

[3] Q. Hong, S. Kim, S. C. Cheung, and C. Bird, "Understanding a developer social network and its evolution," in Proc. 27th IEEE Int. Conf. Softw. Maintenance, Sep. 2011.

[4] J. Xuan, H. Jiang, Z. Ren, and W. Zou, "Developer prioritization in bug repositories," in Proc. 34th Int. Conf. Softw. Eng., 2012.

[5] T. Zimmermann, R. Premraj, N. Bettenburg, S. Just, A. Schr€oter, and C. Weiss, "What makes a good bug report?" IEEE Trans. Softw. Eng., vol. 36, no. 5, Oct. 2010.

[6] C. Sun, D. Lo, S. C. Khoo, and J. Jiang, "Towards more accurate retrieval of duplicate bug reports," in Proc. 26th IEEE/ACM Int. Conf. Automated Softw. Eng., 2011.

[7]D. Cubrani_c and G. C. Murphy, "Automatic bug triage using text categorization," in Proc. 16th Int. Conf. Softw. Eng. Knowl. Eng., Jun. 2004.

[8] 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.

[9] A. 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.

[10] T. M. Khoshgoftaar, K. Gao, and N. Seliya, "Attribute selection and imbalanced data: Problems in software defect prediction," in Proc. 22nd IEEE
Int. Conf. Tools Artif. Intell., Oct. 2010.

[11]S. Breu, R. Premraj, J. Sillito, and T. Zimmermann, "Information needs in bug reports:Improving cooperation between developers and users," in Proc.
ACM Conf. Comput. Supported Cooperative Work, Feb. 2010.

[12] M. Rogati and Y. Yang, "High-performing feature selection for text classification," in Proc. 11th Int. Conf. Inform. Knowl. Manag., Nov. 2002.

[13] D.Cubrani_c and G. C. Murphy, "Automatic bug triage using text categorization," in Proc. 16th Int. Conf. Softw. Eng. Knowl. Eng., Jun. 2004.

[14] J. A. Olvera-L_opez, J. F. Mart_ınez-Trinidad, and J. A. Carrasco- Ochoa, "Restricted sequential floating search applied to object selection," in Proc. Int. Conf. Mach. Learn. Data Mining Pattern Recognit.,2007.

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org