International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2014): 5.611

A Survey Paper on Efficient Approach of Data Reduction Techniques for Bug Triaging System

Smita Boharpi[1], Sonal Fatangare[2]

[1]M. E. Student Department of Computer Engineering, RMD Sinhgad School of Engineering, Pune, India

[2]Assistant Professor, Department of Computer Engineering, RMD Sinhgad School of Engineering, Pune, India

Abstract: ***Bug Triaging is an important part of testing process in software development organizations. It is process of assigning a correct developer for fixing a bug. Software companies spend most of cost in dealing for software bugs. Traditionally in software development, new bugs are manually triaged by expert developer i.e. human traiger. The manual bug triage is expensive in time cost and low in accuracy due to the large number of daily bugs and the lack of expertise of all bugs. To avoid the expensive cost in manual bug triage, an automatic bug triage approach is used to predict developers for bug report. For bug triage data reduction techniques is used to build a small scale and high quality set of bug data by removing bug reports and words which are redundant or non-informative. So applying instance selection with feature selection simultaneously with historical bug data sets.***

Keywords: Bug Triage, Prediction for Reduction Order, Bug Repositories, data Reduction, Feature Selection, Instance Selection, Word Dimension, Bug Dimension

1. Introduction

Data mining technology being used in software development process can not only enhances the accuracy and comprehensiveness of software development but also enhances the credibility of the software. A software bugs is an error, flaw, failure, or fault in a computer program or system that causes it to produce an incorrect or unexpected result, or to behave in unintended ways. Most bugs arise from mistakes and errors made by people in either a program's source code or its design, or in frameworks and operating systems used by such programs, and a few are caused by compilers producing incorrect code. Reports detailing bugs in a program are commonly called as bug reports, defect reports, fault reports, problem reports, and trouble reports [3].

MINING software repositories is main domain which has aims to employ data mining to deal with software engineering problems. Software repositories are large-scale databases for storing the output of software development, e.g., source code, bugs, emails, and specifications. Traditionally for large-scale and complex data in software repositories, software analysis is not completely suitable. So data mining techniques, mining software repositories can uncover interesting information in software repositories and solve real world software problems. A bug repository is also called software repository which is used storing details of bugs. The bug triage is inevitable steps for fixing a bug which are correctly assigning a developer to new bug. For open source large-scale software projects, the number of daily bugs is so large which makes the triaging process very difficult and challenging. Software companies spend over 45 percent of cost in fixing bugs. In a bug repository, a bug is maintained as a bug report, which records the textual description of reproducing the bug and updates according to the status of bug fixing. It also provides data platform to support many types of tasks on bugs, e.g., fault prediction, bug localization and reopened bug analysis. In bug repository, bug reports are called bug data. In software development tasks, there are two challenges related to bug data that may affect on bug repositories that are the large scale and the low quality. In large scale, daily-reported bugs large number of new bugs are stored in bug repositories. And low quality bugs noise and redundancy. Noisy bugs may mislead related developers while redundant bugs waste the limited time of bug handling suffer from the low quality of bug data [1].

The main aim of data reduction for bug triage to build a small scale and high quality set of bug data by removing bug reports and words which are redundant or non-informative. So instance selection and feature selection techniques are simultaneously used to reduce the bug dimension and the word dimension. The reduced bug data contain fewer bug reports and fewer words than original bug data. And they also provide similar information than original bug data. The instance selection means subset of relevant instances i.e. bug report in bug data and the feature selection means subset of relevant features i.e. words in bug data.

2. Existing System

C.Sun,D.Lo,S.C.Khoo and J.Jiang, [3] used a bug tracking system, different testers or users may submit multiple reports on the same bugs, referred to as duplicates, which may cost extra maintenance efforts in triaging and fixing bugs. In order to identify such duplicates accurately, in this paper propose a retrieval function (REP) to measure the similarity between two bug reports. It fully utilizes the information available in a bug report including not only the similarity of textual content in summary and description fields, but also similarity of non-textual fields such as product, component, version, etc. The drawbacks of that system are there is no indexing structure of bug report repository to speed up the retrieval process.

T. M. Khoshgoftaar, K. Gao, and N. Seliya [7] purpose attribute selection and imbalanced data: Problems in software defect prediction. To handle imbalanced defect data.

J.Xuan, H.Jiang, Z.Ren, J.Yan, and Z.Luo[4] propose a semi-supervised text classification approach for bug triage to avoid the deficiency of labeled bug reports in existing supervised approaches. This new approach combines Naive Bayes classifier and expectation maximization to take advantage of both labeled and unlabeled bug reports. This approach trains a classifier with a fraction of labeled bug reports. Then the approach iteratively labels numerous unlabeled bug reports and trains a new classifier with labels of all the bug reports. In that weighted recommendation list for the semi-supervised approach provide a weighted recommendation list for augmenting the semi-supervised approach using probabilistic labels of unlabeled bug reports. Based on this weighted recommendation list, improve the classification accuracy for the semi-supervised approach. The drawbacks of these approach for automatic bug triage with a bug repository there is no bug triage plug-in component combining with the bug repository in real-world applications.

P. S. Bishnu and V. Bhattacherjee [8] purpose software fault prediction using quad tree-based k-means clustering algorithm. In that paper process the defect data with quad tree based K-means clustering to assist defect prediction. In that software metrics predict a value for individual software artifact (e.g. Source code file, a class or module). The software artifact contains fault according to the extracted features of the artifact.

S.Shivaji, E.J.Whitehead, Jr. R. Akella, and S.Kim [9] purpose reducing features to improve code change based bug prediction. In that paper a framework to examine multiple feature selection algorithms and remove noise feature in classification-based defect prediction. Its does not contain how to measure the noise resistance in defect prediction and how to defect noise data.

3. Proposed System

An automatic bug triage approach which applies text classification techniques to predict developers for bug reports. In proposed system combine instance selection with feature selection to simultaneously reduce data scale on the bug dimension and word dimension [1].

In that system when a fault or a new bug report is encountered we have to assign a developer to fix the bug. We use a classifier to know to which developer we need to assign the bug. This can be done by retrieving the information from the history table where we can extract the details like which developer has fixed the bug efficiently, how much time each developer has taken to fix each bug and how many fixers were needed to fix a particular bug etc. Now we reduce the data scale in the bug data sets by using instance selection and feature selection algorithms.

First use the text classification technique to classify the words in the bug triage. Then propose a predictive model to determine the reduction order. After which the bug data reduction process takes place. Uses the instance selection and the feature selection algorithms. The Feature selection aims to obtain a subset of relevant features by removing the uninformative words in the bug reports and the instance selection are used to obtain a subset of relevant instances (bug reports in the bug data). Hence by combining both these algorithms we get a reduced bug data set and replace it with the original bug data. Now this information is sent to the classifier and when a new bug report is encountered the classifier correctly assigns a developer.

4. System Architecture

4.1 Bug Triage

The aim of bug triage is to assign a developer for bug fixing. Once a developer is assigned to a new bug report he will fix the bug or try to rectify it. In bug report consist of two key items the summary and description about the information of bug which is recorded in natural languages. The summary denotes a general statement for identifying a bug and description denotes the details for reproducing the bug. Fig 1.Shows that before training a classifier with a bug data set. It consists of phase of data reduction.

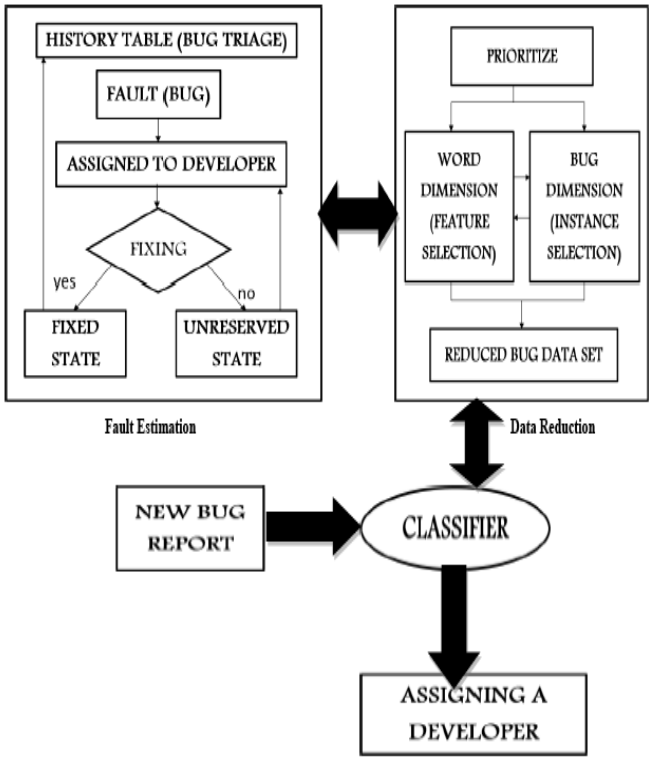

Figure -1: Architecture of Bug Triage

4.2 Data Reduction

The data reduction is used for reduce the scale and improve the quality of data in bug repositories. For data reduction applied as a phase in data preparation of bug triage. By using instance selection and feature selection reduce data, so that get high quality data.
For data processing the instance selection and feature selection are widely used. For given data sets instance selection is to obtain a subset of relevant instances that is bug

report in bug data and feature selection means to obtain a subset of relevant features that is words in bug data.

Instance selection is a technique to reduce the number of instances by removing noisy and redundant instances. An instance selection algorithm can provide a reduced data set by removing non-representative instances .Choose four instance selection algorithms such as Iterative Case Filter (ICF), Learning Vectors Quantization (LVQ), Decremental Reduction Optimization Procedure (DROP) and Patterns by Ordered Projections (POP).

Feature selection is a pre-processing 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. Since bug triage is converted into text classification, focus on the feature selection algorithms in text data. Choose four well-performed algorithms in text data and software data such as Information Gain (IG), x2 statistic (CH), Symmetrical Uncertainty attribute evaluation (SU), and Relief-F Attribute selection (RF). Based on feature selection, words in bug reports are sorted according to their feature values and a given number of words with large values are selected as representative features [1][2].

4.2.1Advantages of Data Reduction

There are many potential benefits of variable and feature selection: facilitating data visualization and data understanding, reducing the measurement and storage requirements, reducing training and utilization times, defying the curse of dimensionality to improve prediction performance.

4.2.1.1. Reducing Data Scale

In word dimension we use feature selection to remove noisy or duplicate words in a dataset. Based on feature selection, the reduced data set can be handled more easily by automatic techniques (e.g., bug triage approaches) than the original data set. Besides bug triage, the reduced data set can be further used for other software tasks after bug triage (e.g., severity identification, time prediction, and reopened bug analysis)

4.2.1.2 Improving Accuracy

In Word dimension by removing uninformative words, feature selection improves the accuracy f bug triage .This can recover the accuracy loss by instance selection.

4.3 Prediction for Reduction Order

Given an instance selection algorithm (IS) and a Feature selection algorithm (FS), FS →IS and IS →FS are viewed as two orders for applying reducing techniques. It is a challenge is how to determine the order of reduction techniques, i.e., how to choose one between FS →IS and IS →FS.

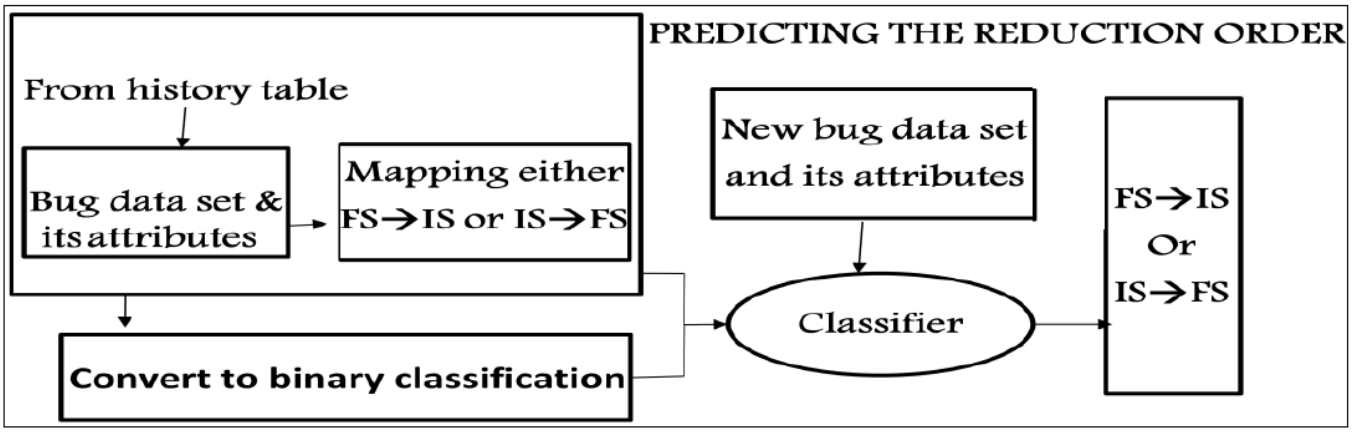

Figure 2: Prediction for Reduction Order

Each new bug data set to apply the data reduction, need to check the accuracy of both two orders (FS→ IS) and (IS→FS) and choose a better one. To avoid the time cost of manually checking both reduction orders, consider predicting the reduction order for a new bug data set based on historical data sets. As shown in Fig. 2, convert the problem of prediction for reduction orders into a binary classification problem. A bug data set is mapped to an instance and the associated reduction order (either FS → IS or IS → FS) is mapped to the label of a class of instances. Fig. 2 shows that steps of predicting reduction orders for bug triage.

5. Algorithm

Reduced data set TFI for bug triage

- Apply FS to n words of T and calculate objective values for all word
- Select the top nF words of T and generate a training set TF
- Apply IS to mI bug reports of TF
- Terminate IS when the number of bug reports is equal to or less than mI and generate the final training set TFI.

6. Conclusions

In software maintenance bug triage is an expensive step of in both labour cost and time cost. In this system focused on reducing bug data set in order to have less scale of data and quality data. The feature selection and the instance selection are used for reduce the scale of bug data sets and also improve the data quality. The order of applying instance selection and feature selection for new bug data set, extract the attributes of each bug data sets and also train a predictive model which is based on historical data sets. It provides an approach to leveraging techniques on data processing to form reduced and high quality bug data.

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" ieee transactions on knowledge and data engineering, vol. 27, no. 1, January 2015.

[2] 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, pp. 323–332.

[3] 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, pp. 253–262.

[4] J. Xuan, H. Jiang, Z. Ren, and W. Zou, "Developer prioritization in bug repositories," in Proc. 34th Int. Conf. Soft. Eng., 2012, pp. 25–35.

[5] T. Zimmermann, N. Nagappan, P. J. Guo, and B. Murphy, "Characterizing and predicting which bugs get reopened," in Proc. 34th Int. Conf. Softw. Eng., Jun. 2012, pp. 1074–1083.

[6] Mamdouh Alenezi and Kenneth Magel, Shadi Banitaan "Efficient Bug Triaging Using Text Mining" © 2013 academy publisher.

[7] 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,pp. 137–144.

[8] 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.

[9] S. Shivaji, E. J. Whitehead, Jr., R. Akella, and S. Kim, "Reducing features to improve code change based bug prediction," IEEETrans. Soft. Eng., vol. 39, no. 4, pp. 552–569, Apr. 2013.