ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Towards Effective Bug Triage with Software Data Reduction Techniques

Ch.Ramesh Kumar
Associate Professor & HOD,
Department of CSE,
Malla Reddy Engineering College & Management Sciences, Kistapur, Medchal, Hyderabad.

Arige Sairam
M.Tech Student,
Department of CSE,
Malla Reddy Engineering College & Management Sciences, Kistapur, Medchal, Hyderabad.

Abstract:

The process of fixing bug is bug triage that aims to properly assign a developer to a brand new bug. Software system firms pay most of their value in managing these bugs. To reduce time and price of bug triaging, we present an automatic approach to predict a developer with relevant expertise to resolve the new coming report. In proposed approach we have a tendency to do data reduction of bug data set which can reduce the dimensions of the data still as increase the standard of the data .We are using instance selection and feature selection technique at the same time with historical bug data.

We have added a new module here which will describe the status of the bug like whether it assigned to any developer or not and it is rectified or not.. In addition, the load between developers based on their experience is re-balanced. The experimental results show that the planned theme will effectively improve the detection performance compared with previous methods.

Index Terms:

bug data reduction, feature selection technique, instance selection technique, prediction for reduction orders, bug triage, machine learning techniques.

INTRODUCTION:

A bug repository plays main position in managing software bugs. Many open source software projects have an open bug repository that makes it possible for each developer and users to publish defects or issues in the software, suggest possible enhancements, and remark on existing bug reports. In large open source software project have the bug repository that store the details of the bug. For large open source software project, the quantity of every day bugs is so substantial which makes the triaging process more challenging and difficult . There are two challenges associated with bug data that will have an effect on use of bug repositories in software development tasks, specifically the large scale data and low quality data. In a bug repository, a bug is kept up as a bug report, which record in the form of text that reproducing the bug and update as per the status of bug fixing.

In software companies spend their time and cost in dealing with the bug. The process of assigning a potential developer for fixing bug is the bug triage. Handling the software bug by bug triage is time consuming. Manual bug triage is time consuming and low in accuracy. To reduce the cost and increase the accuracy of manually bug triaging in this paper proposed an automatic bug triage. In this paper an effective bug triage approach is proposed to reduce the bug data to save the labor price of developer and enhance the quality of bug data by eliminating the repetitive and non useful bug reports.

LITERATURE SURVEY:

In [1] this paper, Software firms spend their time and cost in dealing with software bugs. So that bug triage use for fixing the bug, the goal of bug triage is bug assign the potential developer. To reduce the time and cost in manual work, proposed the automatic bug triage. In this paper used data reduction technique to reduce the data set and improve the quality of data set. Here we combine two reduction techniques, namely instance selection and feature selection. These techniques are used to reduce the data scale on the bug dimension and the word dimension. To focus the request of applying instance selection and feature selection, we extract attributes from historical bug data sets and develop prescient model for new bug data set. The outcome shows that data set can effectively reduce by the data reduction techniques and also increase the accuracy of bug triage. Our work gives approach to leveraging techniques on data processing to form reduced and high-quality bug data in software improvement.

In [2] they mention Open supply development projects most of the time aid an open bug repository to which both developers and users can re-port bugs. The experiences that appear on this repository ought to be triaged to verify if the record is one which requires attention and whether it is, which developer will be assigned the responsibility of resolving the report. Massive open supply developments are pressured by using the rate at which new bug reports appear within the bug repository. In this paper, we present a semi-automated approach intended to ease one part of this method, the assignment of reports to a developer.

In this paper used machine learning algorithm that produced the classifier to classify the developers which is potentially solve the report. With this procedure, reach the accuracy level of 57% and 64% on the Eclipse and Firefox development projects. We have now additionally applied our method to the GCC open source development with much less positive outcome.

We describe the gives approach to leveraging techniques on data processing to form reduced and high-quality bug data in software improvement. stipulations below which the method is applicable and also report on the lessons we learned about applying machine learning to repositories used in open source development. In [3] proposed the process of fixing the bug is called bug triage which aim to assign the new coming bug to the corrected developer. The existing bug triage approach used machine learning algorithms, which construct classifiers from the training sets of bug reports. In observe, these strategies suffer from the large-scale and low-quality training sets. In the proposed work we used both instance selection and feature selection techniques to reduce bug triage. In this paper we studied the combination of feature selection algorithm χ 2-test, instance selection algorithm Iterative Case Filter. We use the eclipse to calculate training set reduction on bug data. For the training set, 70% phrases and 50% bug reports are eliminated after the training set reduction. The experimental results exhibit that the new and small training sets can provide higher accuracy.

In [4] paper, As of late, machine learning classifiers have risen as an approach to anticipate the presence of a bug in a change made to a source code document. The classifier is initially prepared on software history data, and afterward used to foresee bugs. Two drawbacks of existing classifier-based bug expectation are conceivably lacking precision for handy utilize, and utilization of countless. These extensive quantities of components antagonistically affect adaptability and exactness of the methodology. This paper proposes feature selection technique pertinent to classification based bug forecast. This method is connected to foresee bugs in software changes, and execution of Na¨ıve Bayes and Support Vector Machine (SVM) classifiers is characterized. In [5] this paper, we propose a semi-supervised text classification approach for bug triage to stay away from the lack of labeled bug reports in existing supervised approaches.

This new methodology joins naïve Bayes classifier and desire augmentation to exploit both marked and unlabeled bug reports. This methodology prepares a classifier with a small amount of labeled bug reports. At that point the methodology iteratively labels various unlabeled bug reports and trains a new classifier with marks of all the bug reports. We additionally utilize a weighted proposal rundown to help the execution by forcing the weights of multiple developers in training the classifier. Trial results on bug reports of Eclipse demonstrate that our new methodology outflanks existing supervised approaches in terms of classification exactness.

PROBLEM DEFINITION:

For reducing the affluent cost of manual bug triage we used automatic bug triage method. To build a predictive model for a new bug data sets that present the problem of data reduction for bug triage.

EXISTING SYSTEM:

- To investigate the relationships in bug data, Sandusky et al. form a bug report network to examine the dependency among bug reports.
- Besides studying relationships among bug reports, Hong et al. build a developer social network to examine the collaboration among developers based on the bug data in Mozilla project. This developer social network is helpful to understand the developer community and the project evolution.
- By mapping bug priorities to developers, Xuan et al. identify the developer prioritization in open source bug repositories. The developer prioritization can distinguish developers and assist tasks in software maintenance.
- To investigate the quality of bug data, Zimmermann et al. design questionnaires to developers and users in three open source projects. Based on the analysis of questionnaires, they characterize what makes a good bug report and train a classifier to identify whether the quality of a bug report should be improved.
- Duplicate bug reports weaken the quality of bug data by delaying the cost of handling bugs. To detect duplicate bug reports, Wang et al. design a natural language processing approach by matching the execution information.

LIMITATIONS:

- Traditional software analysis is not completely suitable for the large-scale and complex data in software repositories.
- In traditional software development, new bugs are manually triaged by an expert developer, i.e., a human triage. Due to the large number of daily bugs and the lack of expertise of all the bugs, manual bug triage is expensive in time cost and low in accuracy.

PROPOSED SYSTEM:

- In this paper, we address the problem of data reduction for bug triage, i.e., how to reduce the bug data to save the labor cost of developers and improve the quality to facilitate the process of bug triage.
- 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.
- In our work, 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 of bug triage.
- In this paper, we propose a predictive model to determine the order of applying instance selection and feature selection. We refer to such determination as prediction for reduction orders.
- Drawn on the experiences in software metrics,1 we extract the attributes from historical bug data sets. Then, we train a binary classifier on bug data sets

with extracted attributes and predict the order of applying instance selection and feature selection for a new bug data set.

ADVANTAGES:

- Experimental results show that applying the instance selection technique to the data set can reduce bug reports but the accuracy of bug triage may be decreased.
- Applying the feature selection technique can reduce words in the bug data and the accuracy can be increased.
- Meanwhile, combining both techniques can increase the accuracy, as well as reduce bug reports and words.
- Based on the attributes from historical bug data sets, our predictive model can provide the accuracy of 71.8 percent for predicting the reduction order.

DATA REDUCTION FOR BUG TRIAGE:

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 unnecessary. In this paper we are using feature selection and instance selection with historical data for reducing the bug data in bug repository so that we get quality data as well as low scale data.

1) APPLYING INSTANCE SELECTION AND FEATURE SELECTION:

In bug triage, a bug data set is converted into a text matrix with two dimensions; they are the bug dimension and the word dimension. In this paper, we use the combination of instance selection and feature selection to generate a reduced bug data set. We change the original data set with the reduced data set for bug triage. Instance selection and feature selection are widely used techniques in data processing. In data reduction for a given data set in a certain application, instance selection is to obtain a subset of relevant instances while feature selection is to obtain a subset of relevant features.

2) BENEFITS OF DATA REDUCTION:

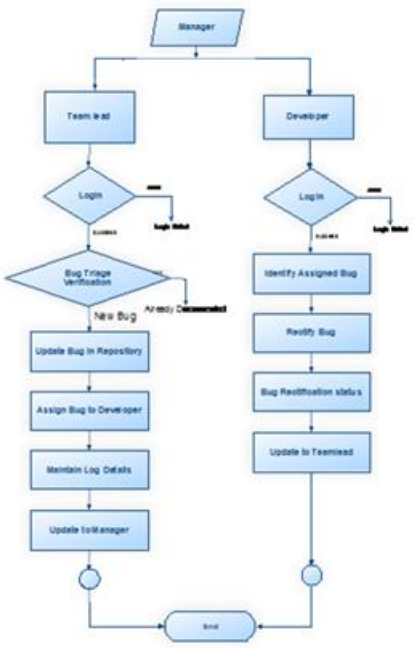
a) Reducing the data scale: In word dimension we use feature selection to remove noisy or duplicate words in a data set. Based on feature selection method, the reduced data set can be handled more easily by automatic techniques (e.g., bug triage approaches) than the original data set. In bug triage, 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).

b)Improving Accuracy: In Word dimension by removing meaningless words, feature selection improves the accuracy of bug triage .This can recover the accuracy loss by instance selection.

IMPLEMENTATION:

Dataset Collection:

To collect and/or retrieve data about activities, results, context and other factors. It is important to consider the type of information it want to gather from your participants and the ways you will analyze that information. The data set corresponds to the contents of a single database table, or a single statistical data matrix, where every column of the table represents a particular variable. after collecting the data to store the Database.



Preprocessing Method:

Data preprocessing or Data cleaning, Data is cleansed through processes such as filling in missing values, smoothing the noisy data, or resolving the inconsistencies in the data. And also used to removing the unwanted data. Commonly used as a preliminary data mining practice, data preprocessing transforms the data into a format that will be more easily and effectively processed for the purpose of the user.

Feature Selection/ Instance Selection:

The combination of instance selection and feature selection to generate a reduced bug data set. We replace the original data set with the reduced data set for bug triage. Instance selection is a technique to reduce the number of instances by removing noisy and redundant instances. By removing uninformative words, feature selection improves the accuracy of bug triage. It recovers the accuracy loss by instance selection.

Bug Data Reduction:

The data set can reduce bug reports but the accuracy of bug triage may be decreased. It improves the accuracy of bug triage. It tends to remove these words to reduce the computation for bug triage. The bug data reduction to reduce the scale and to improve the quality of data in bug repositories. It reducing duplicate and noisy bug reports to decrease the number of historical bugs.

Performance Evaluation:

In this Performance evaluation, algorithm can provide a reduced data set by removing non-representative instances. The quality of bug triage can be measured with the accuracy of bug triage. to reduce noise and redundancy in bug data sets.

CONCLUSION:

Bug triage is a vital step of software maintenance to save labor cost and time cost. To decrease the expensive cost of manual bug triage we used automatic bug triage approach that appropriately assigns a developer to a new bug for additional usage.

The main aim of this work is to reduce the large scale of the training set and to remove the noisy and redundant bug reports for bug triage. In this system engrossed on reducing bug data set in order to have a fewer scale of data and also superiority data. So feature selection and instance selection are used for shrink the scale of bug data sets and also increase the data quality. Using 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. For reduced and high quality of data bug data, we used data preprocessing.

FUTURE WORK:

The future work of the proposed system is to get better the outcome of data reduction in bug triage to investigate how to organize a high quality bug data set and deal with a domain-specific software assignment. For predicting reduction orders, aim to give attempts to locate out the possible relationship among the attributes of bug data sets and the reduction orders.

REFERENCES:

[1] J. Anvik, L. Hiew, and G. C. Murphy, “Who should fix this bug?” in Proc. 28th Int. Conf. Softw. Eng., May 2006, pp. 361–370.

[2] S. Artzi, A. Kie_zun, J. Dolby, F. Tip, D. Dig, A. Paradkar, and M. D. Ernst, “Finding bugs in web applications using dynamic test generation and explicit-state model checking,” IEEE Softw., vol. 36, no. 4, pp. 474–494, Jul./Aug. 2010.

[3] 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.

[4] C. C. Aggarwal and P. Zhao, “Towards graphical models for text processing,” Knowl. Inform. Syst., vol. 36, no. 1, pp. 1–21, 2013.

[5] Bugzilla, (2014). [Online]. Avaialble:

http://bugzilla.org/

[6] K. Balog, L. Azzopardi, and M. de Rijke, “Formal models for expert finding in enterprise corpora,” in Proc. 29th Annu. Int. ACM SIGIR Conf. Res. Develop. Inform. Retrieval, Aug. 2006, pp. 43–50.

[7] 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.

[8] H. Brighton and C. Mellish, “Advances in instance selection for instance-based learning algorithms,” Data Mining Knowl. Discovery, vol. 6, no. 2, pp. 153–172, Apr. 2002.

[9] 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, pp. 301–310.

[10] V. Bol_on-Canedo, N. S_anchez-Maro~no, and A. Alonso-Betanzos, “A review of feature selection methods on synthetic data,” Knowl. Inform. Syst., vol. 34, no. 3, pp. 483–519, 2013.

[11] V. Cerver_on and F. J. Ferri, “Another move toward the minimum consistent subset: A tabu search approach to the condensed nearest neighbor rule,” IEEE Trans. Syst., Man, Cybern., Part B, Cybern., vol. 31, no. 3, pp. 408–413, Jun. 2001.

[12] D. _Cubrani_c and G. C. Murphy, “Automatic bug triage using text categorization,” in Proc. 16th Int. Conf. Softw. Eng. Knowl. Eng., Jun. 2004, pp. 92–97.

[13] Eclipse. (2014). [Online]. Available: http://eclipse.org/

[14] B. Fitzgerald, “The transformation of open source software,” MIS Quart., vol. 30, no. 3, pp. 587–598, Sep. 2006.

Authors Biography

Arige sairam completed her B.Tech degree in Potti Sriramulu College of Engg & Tech. 2012. He is pursuing M.Tech. in Computer Science & Engineering from Department of Computer Science & Engineering in Malla Reddy Engineering College & Managemet sciences, Kistapur, Medchal, Hyderabad. Affiliated to JNTUH, HYDERABAD,TELANGANA.,India.. His research interest include cloud, data mining, bigdata and networking.

Ch.Ramesh Kumar, working as Assoc.Prof & Head of the Department of Computer Science and Engineering in Malla Reddy Engineering College & Management Sciences, Kistapur, Medchal, Hyderabad. Affiliated to JNTUH, HYDERABAD,TELANGANA., India. he has several international publications to his credit. His research interests include Software reuse, Software performance, Software testing ,Data Mining and cloud computing.