A Novel Approach on Reducing the Bugs in Repository

G. Sindhura
PG Scholar,CSE
Sri Vasavi Engineering College
Tadepalligudem

A. Lakshmi Lavanya
Assistant Professor,CSE
Sri Vasavi Engineering College
Tadepalligudem

***Abstract*-In modern software development, software repositories are like large-scale databases for storing the output of software development. Initial step in bug repository is to manage software bugs. Bug fixing is an important and time-consuming process in software maintenance. Open source development projects typically incorporate an open bug repository to which both software developers and users can report bugs or defects or issues in the software, suggest possible enhancements, and comment on existing bug reports. The advantage of an open bug repository is that it may allow more bugs to be identified and solved, improving the quality of the software product. In the proposed system, we join existing techniques of instance selection and feature selection at the same time to decrease the bug dimension and the word dimension. The reduced bug data contain less bug reports and fewer words than the original bug data and also provides similar information over the original bug data. After that applying clustering technique on resulted reduce bug data set to generate different domain wise clusters. Text classification is used in bugs data set to classify all bugs into different classes. It is evaluating the reduced bug data by two criteria which are size of a data set and the correctness of bug triage.**

Keywords: Bug triage, data reduction, Instance selection, Feature selection, Data Mining, Mining software repositories.

I.INTRODUCTION

In modern software development, software repositories are like large-scale databases for storing the output of software development. Initial step in bug repository is to manage software bugs. Bug fixing is an important and time-consuming process in software maintenance. Open source development projects typically incorporate an open bug repository to which both software developers and users can report bugs or defects or issues in the software, suggest possible enhancements, and comment on existing bug reports. The advantage of an open bug repository is that it may allow more bugs to be identified and solved, improving the quality of the software product.

Bug triage is most vital step for bug fixing, is to allocate a new bug to a relevant developer for further handling. For open source software projects, large number of bugs are produced daily which makes the triaging process very difficult and challenging. Main objective 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. The motivation 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.

Data reduction is the process of reducing the bug data by using two techniques namely, instance selection and feature selection which intends to get low scale as well as quality data.

II. RELATED WORKS

kNN classification process. J. Anvik, L. Hiew, and G. C. Murphy, [2] author present a semi-automated approach intended to simplify one part of this process, the assignment of reports to a developer for further handling. Bug triage aims to allocate an appropriate developer to fix a new bug that is to determine who should fix a bug. Author first proposes the problem of automatic bug triage to reduce the cost of manual bug triage. Presented approach is based on a supervised machine learning algorithm that is applied to information available in the bug repository. When a new report arrives, the classifier produced by the supervised machine learning technique offered a small number of developers suitable to resolve the report.

C.C.Aggarwal and P.Zhao [3], author introduced a new paradigm for text representation and processing called distance graph representations. Distance graph representations keep information about the relative ordering as well as distance among the words in the graphs and provide a much more affluent representation in terms of sentence structure of the provided data. Knowledge discovery from text is possible with help of distance graph representation which is not possible with the use of a pure vector-space representation. Use of the distance graph representation provides significant advantages from an effectiveness perspective.

S. Kim, H. Zhang, R. Wu, and L. Gong [4], author proposes two schemes to deal with the noise present in defect data. Author introduced a method to measure noise conflict in software defect prediction and also proposed a new method called CLNI for identifying noisy instances in defect data. Noise detection and elimination algorithms are proposed to address this problem. Proposed algorithm can identify noisy data with accuracy. In addition, after eliminating the noises using proposed algorithm, defect prediction accuracy is improved. For the machine learners that do not have strong noise resistant ability, the noise-eliminated training sets produced by CLNI can improve the defect prediction performance.

G. Jeong, S. Kim, and T. Zimmermann [5], author proposed bug tossing graph model can be easily incorporated into existing bug triaging systems. Find out that over 37 percent of bug reports have been reassigned in manual bug triage to other developers specifically in case of Mozilla and Eclipse. Proposed the model increased the prediction accuracy as compared to traditional bug triaging approaches. Main objective of proposed method is to reduce reassignment in bug triage.

C. Sun, D. Lo, S. C. Khoo, and J. Jiang [7], in this paper author propose a retrieval function (REP) to identify such duplicates accurately between two bug reports. Proposed approach is twofold, first BM25F is an effective textual duplicates measure that is designed for short unstructured queries and seconds a new retrieval function REP fully utilizing text and other information available in reports such as product.

A. Srisawat, T. Phienthrakul, and B. Kijsirikul, [8], paper proposed SV-kNNC approach for data reduction to enhance performance of kNN. Proposed algorithm is three fold approaches, first support vector machines (SVMs) are applied to select some important training data then weights are allocate to each training instance based on k-mean clustering and finally classify the query instances

III PROPOSED SYSTEM

The objective of paper is to address the problem of data reduction for effective bug triage, i.e., how to reduce the bug information to save the work cost of developers and improve the quality to facilitate the process of bug triage. Data reduction for bug triage expects to build a small-scale and high-quality set of bug data by removing bug reports and words, which are not informative and redundant. In proposed system, we join existing techniques of instance selection and feature selection at the same time to decrease the bug dimension and the word dimension. The reduced bug data contain less bug reports and fewer words than the original bug data and also provides similar information over the original bug data. After that applying clustering technique on resulted reduce bug data set to generate different domain wise clusters. Text classification is used in bugs data set to classify all bugs into different classes. Proposed system is evaluating the reduced bug data by two criteria which are size of a data set and the correctness of bug triage.

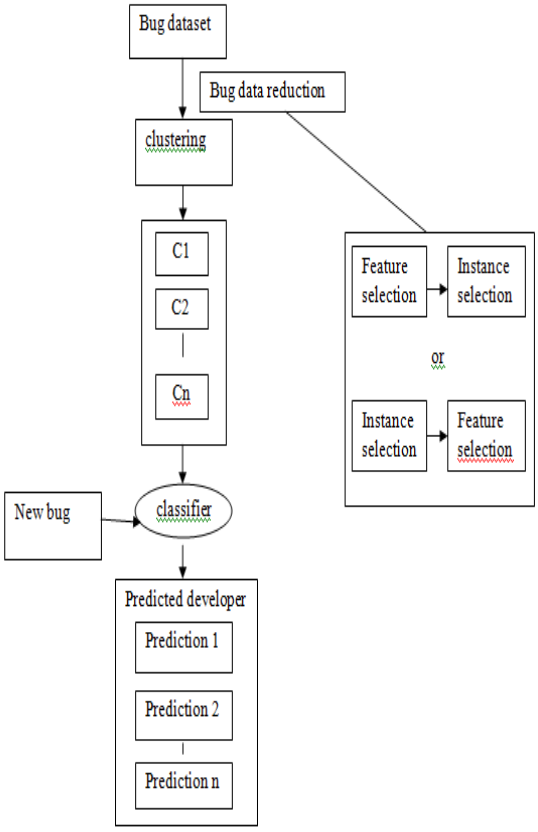

Fig. 1 System architecture

Figure shows illustration of reducing bug data for bug triage. In this figure input is bug data set. Bug data set views as a text matrix. Each row of text matrix indicates one bug report and each column of text matrix indicates one word. Instance selection and Feature selection techniques are applying on bug data set to reduce the scale of bug data. Instance selection technique is used to remove bug reports and Feature selection technique is used to remove non-informative words. After that applying clustering on resulted reduced bug data set to make different domain wise clusters. In next step Naïve Bayes classifier is used to make classification of bugs data set in to different classes. Finally bugs are assigned to specific developers.

Bug Reduction Algorithm

1. Take input dataset
2. Apply Data reduction
3. Apply clustering
4. Extract Features-Stored into training set
5. New bug reported
6. Classifier will take two inputs, new bug and training set
7. Output of classifier is predicted set of developers

Feature Selection

Feature selection is a standard technology to decrease the features of huge data sets in machine learning. The number of variables (or features) gathered in a dataset is typically relatively large and some of these features are not informative or can't provide high differentiating power. The objective of feature selection algorithm is to remove the irrelevant and redundant words from the selected dataset, thus optimizing the performance of the classification and/or clustering algorithms. In addition, for a particular dataset, feature selection can aid to realize which features are important as well as how they are associated. Feature selection can be defined as the procedure of choosing a smallest subset of m features from the original dataset of n features (m is less than n), so that the feature space (i.e. the dimensionality) is optimally reduced according to the evaluation criteria: The classification accuracy does not significantly reduce and The resulting class distribution, given only the values for the selected features, is as similar as possible to the original. A feature selection algorithm generally consists of four steps which are subset generation, subset evaluation, stopping criterion, and result validation. Subset generation is a search procedure which creates subsets of features for evaluation. Each subset generated is verified by some particular evaluation criterion and evaluated with the earlier best one with respect to this criterion. If a new subset is found to be better, then the earlier best subset is replaced with the new subset. In this paper, Feature selection technique is used to remove the words in bug reports which are redundant and non-informative.

Instance Selection:

As data increases the requirement for data reduction also increases for effective data mining. Instance selection is one of effective means to data reduction. The objective of Instance Selection algorithm is to diminish the size of a dataset while still sustaining the integrity of the actual dataset. In many cases, generalization correctness can raise when noisy instances are eliminated and when decision borders are smoothed to more closely equal the true core function. These instances can also be considered as outliers (or bad data). Specifically, outliers are those data points which are extremely unlikely to arise given a model of the data. In this paper, Instance selection technique is used to reduce the number of instances means bug reports by removing noisy and redundant bug reports. Due to removing a set of instances from a dataset the response time of classification decisions decreases, as some instances are examined when a query instance is presented. This purpose is primary when working with huge database and has limited storage.

IV CONCLUSION

Bug triage is a costly stride of programming upkeep in both work cost and time cost. In this paper, we join feature selection with instance selection to decrease the size of bug data sets and also enhance the data quality. To decide the request of applying instance selection and feature selection for a new bug data set, we extract attributes of each bug information set and prepare a predictive model based on recorded information sets. Our work gives a way to deal with utilizing systems on information handling to shape reduced and high-quality bug information in programming improvement and support.

REFERENCES

[1] Jifeng Xuan, He Jiang, Member, IEEE, Yan Hu, Zhilei Ren, Weiqin Zou, Zhongxuan Luo, and Xindong Wu, Fellow, IEEE, "Towards Effective Bug Triage with Software Data Reduction Techniques", IEEE Transactions on knowledge and Data Engineering Vol. 27, No. 1, January 2015.

[2] 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.

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

[4] S. Kim, H. Zhang, R. Wu, and L. Gong, "Dealing with noise in defect prediction," in Proc. 32nd ACM/IEEE Int. Conf. Softw. Eng., May 2010, pp. 481–490..

[5] G. Jeong, S. Kim, and T. Zimmermann, "Improving bug triage with tossing graphs," in Proc. Joint Meeting 12th Eur. Softw. Eng. Conf. 17th ACM SIGSOFT Symp. Found. Softw. Eng., Aug. 2009, pp. 111–120.

[6] Q. Shao, Y. Chen, S. Tao, X. Yan, and N. Anerousis, "Efficient ticket routing by resolution sequence mining," in Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, Aug. 2008, pp. 605– 613.

[7] 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.

[8] A.Srisawat, T. Phienthrakul, and B. Kijsirikul, "SV-kNNC: An algorithm for improving the efficiency of k-nearest neighbor," in Proc. 9th Pacific Rim Int. Conf. Artif. Intell., Aug. 2006, pp. 975–979.