IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN : 2278-0661, p-ISSN : 2278-8727
PP 09-11
www.iosrjournals.org

A Survey on Bug triage with Data Reduction Technique

Rucha Warade , Kanyakumari Pawar Pradnya Hande, Bhavana Ambadkar
Department of Information Technology, JSPM'S Rajarshi Shahu College of Engineering, Tathawade Pune-411033

Abstract : *Software industries spend more than 45 percent of expense in managing programming bugs. An unavoidable stride of altering bugs is bug triage, which expects to effectively allocate a designer to another bug. To diminish the time cost in manual work, content characterization procedures are connected to direct programmed bug triage. In this, we address the problem of information decrease for bug triage, i.e., how to diminish the scale and enhance the nature of bug data. We join example choice with highlight determination to at the same time lessen information scale on the bug measurement and the word measurement. To focus the request of applying example choice and highlight determination, we separate properties from verifiable bug information sets and construct a prescient model for bug information set.*

I. INTRODUCTION

In modern software development industries, the large databases are required to store bug repositories. In this days, the data mining is turn out as promising to handle data. Using arbitrage data mining techniques, real world software problems can be solved. Bug repository used for storing software bugs so they can be managed. So it plays an important role. Software industries spend more than 45 percent of expense in managing programming bugs. In a bug repository the bugs are maintained using textual description as reproduction or update of bug. To perform task on bugs like fault prediction, bug localization, etc.bug repository provides data platform.

The large scale and low quality are two challenges related to bug data which may affect task of software development. Duplicate bugs and to assign the developer is time consuming. Lack of expertise will result in time cost and low accuracy to reduce this drawback an automatic bug triage approach is used. The bug triage is converted into text classification problem and solve with classification techniques like Naive Bayes classification technique.

To improve the accuracy of text classification technique used for bug triage more techniques are investigated like tossing graph approach and collaborative filtering approach. Large scale and low quality bug data in repositories block the techniques of automatic bug triage. Our main aim is to reduce bug data to save the labor cost with improved quality.

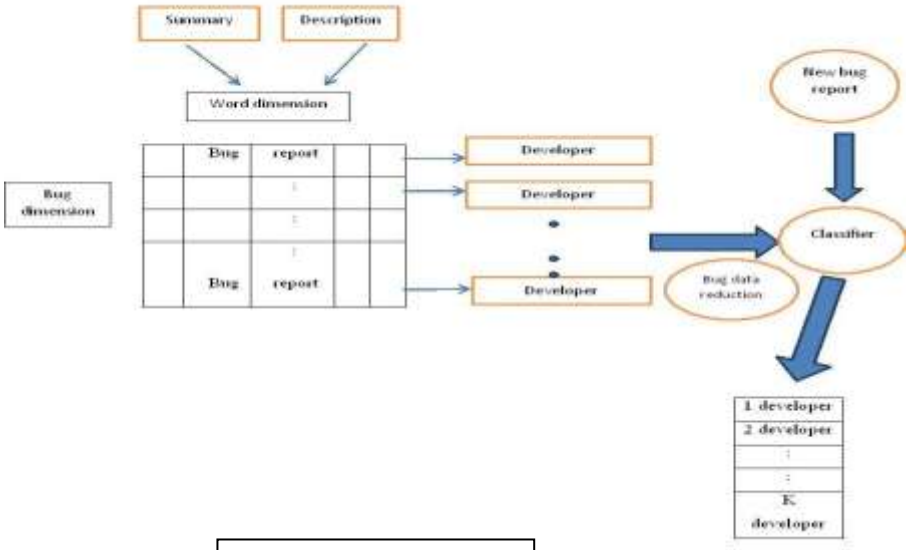
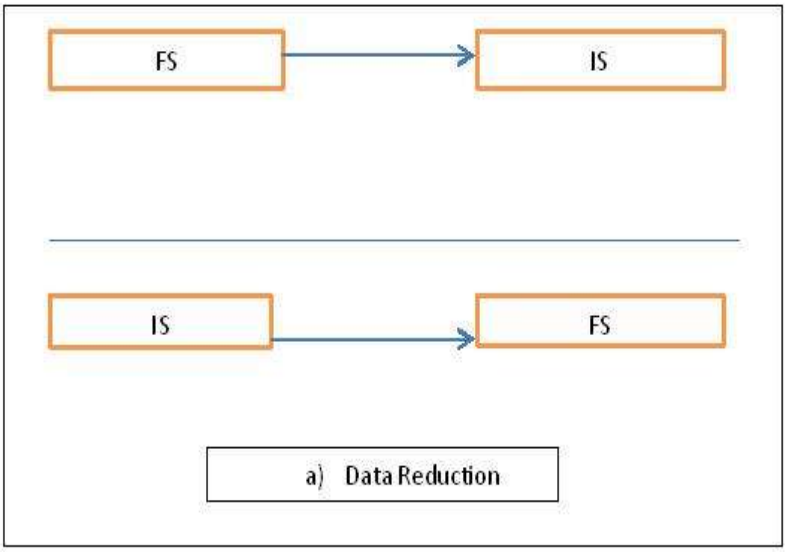


Fig. Flow of the system



II. LITERATURE SURVEY

Development of open source project which generally support open bug repository, in such a development the semi-automated approach is well intentioned, as it applies algorithm of machine learning which helps to learn kinds of reports each developer resolves. From this, when new report arrives the classification of machine learning suggest small number of developers, desirable for resolve the problem. While finding bugs in web applications dynamic test generation technique is small useful so for that tool Apollo is gives better result. Reducing effort in bug report triage, the repository of bug is need to be triaged. A triager specifies whether the report is meaningful? For this the machine learning approach is also useful to create recommendations which assist with variety of decisions.

III. EXISTING SYSTEM

Fixing a bug is very time consuming process as it require correct developer to fix a bug. In traditional development bug triage done manually called as human triage. Large number of bugs reported daily as compared to lack of expertise which leads to increase in time and cost and minimizes accuracy. So that manual bug triage required text classification technique to assign developer for bug reports. Then the related developer assign label of document.

Naive Bayes classification technique is used for conversion of bug triage to text classification which assigns to expertise developer. To facilitate the application, it is necessary to data should be free from bug data. To minimize time and cost in manual bug triage, Automatic bug triage approach is proposed. There are some new technique like tossing graph approach and collaborative filtering approach.

IV. PROPOSED SYSTEM

By using the automatic bug triage approach the time and cost is saved. This system is implemented by using the Naive bayes classification technique. The assignment of bug to developer is done automatically. Quality of bug triage is improved using this new system, because here we use classifier for implementation. We are using instance and feature selection for data reduction. In instance selection we are using ICF algorithm and in feature selection CH algorithm with Naïve Bayes classification for bug triage.

Advantages

Word dimension: We use feature selection to remove noisy duplicate words in a data set. By removing uninformative words, feature selection improves the accuracy of bug triage. This can recover the accuracy loss by instance selection.

Bug dimension: Instance selection can remove uninformative bug reports; meanwhile, we can observe that the accuracy may be decreased by removing bug reports.

V. MODULE DESCRIPTION

1. **Bug triage**
2. **Classification**
3. **Data Reduction with instance and feature selection**

1. Bug triage:

In bug triage, a bug data set is converted into a text matrix with two dimensions, namely the bug dimension and the word dimension. Each row of the matrix indicates one bug report while each column of the matrix indicates one word.

2. Classification:

We use Naive Bayes classification algorithm.

3. Data reduction:

We combine feature selection with instance selection to reduce the scale of bug data sets as well as improve the data quality. We extract attributes of each bug data set and train a predictive model based on historical data sets

VI. Conclusion

By using this system, we effectively decrease labor cost and time cost needed for bug triage with instance and feature selection. This system reduces scale of bug data set and improves quality of data.

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]. 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.
[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]. A. E. Hassan, "The road ahead for mining software repositories," inProc. Front. Softw. Maintenance, Sep. 2008, pp. 48-57.
[7]. J. W. Park, M. W. Lee, J. Kim, S. W. Hwang, and S. Kim, "Costriage: A cost-aware triage algorithm for bug reporting systems," inProc. 25th Conf. Artif. Intell., Aug. 2011, pp. 139–144.
[8]. J. Xuan, H. Jiang, Z. Ren, and W. Zou, "Developer prioritization in bug repositories," in Proc. 34th Int. Conf. Softw. Eng., 2012, pp. 25–35.