ISSN: 2321-7782 (Online)
Volume 4, Issue 4, April 2016
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

Software Data Reduction Technique for Bug Reduction Problem

Kanyakumari Pawar[1]
IT department, RSCOE, Pune – India

Archana Jadhav[2]
IT department, RSCOE, Pune – India

Pradnya Hande[3]
IT department, RSCOE, Pune – India

Bhavana Ambadkar[4]
IT department, RSCOE, Pune – India

Rucha Warade[5]
IT department, RSCOE, Pune – India

Abstract: Programming organization spend more than forty five percent of expense in overseeing programming bugs. An unavoidable stride of unravelling bugs is bug triage, which objectives to accurately allot expert to resolve the problem. To lessen the time esteem in aide work, printed content sort techniques are actualized to lead programmed bug triage. On this errand, we adapt to the inconvenience of measurements markdown for pernicious project triage, i.e., the best approach to decrease the measurements and enhance the remarkable of bug dataset. We incorporate example choice with highlight decision to at the same time lessen insights scale at the error estimation and the expression measurement. To decide the request of applying case determination and trademark decision, we separate traits from important error data sets and assemble a prescient model for another bug records set. We experimentally explore the execution of insights rebate on totally 600,000 bugs' surveys of enormous open supply activities, especially Eclipse and Mozilla. The outcomes show that our measurements diminishment can effectively decrease the insights scale and improve the exactness of malignant system triage. This test introduces a way to deal with utilizing strategies on information preparing to shape diminished and amazing bug records in programming program change and upkeep.

Keywords: Mining software repositories, application of data pre-processing, data management in bug repositories, bug data reduction, feature selection, instance selection, bug triage, prediction for reduction orders.

I. INTRODUCTION

In the mining programming archives is an interdisciplinary area, which means to utilize information mining to manage programming building issues. In present day programming advancement, programming archives are substantial scale databases for putting away the yield of programming improvement, e.g., source code, bugs, and so on. Customary programming examination is not totally suitable for the vast scale and complex information in programming storehouses. Information mining has developed as a promising intends to handle programming information. By utilizing information mining methods, mining programming vaults can reveal intriguing data in programming archives and take care of certifiable programming issues. A bug storehouse (a normal programming vault, for putting away subtle elements of bugs), assumes a critical part in overseeing programming bugs. Programming bugs are unavoidable and settling bugs is costly in programming improvement. Programming organizations spend more than 45 percent of expense in altering bugs. Substantial programming ventures convey bug stores (likewise called bug following frameworks) to bolster data accumulation and to help engineers to handle bugs. In a bug archive, a bug is kept up as a bug report, which records the printed portrayal of imitating the bug and upgrades as indicated by the status of bug altering. A bug store gives an information stage to bolster numerous sorts of errands on bugs, e.g., issue forecast. Bug confinement, and revived bug examination. In this paper, bug reports in a bug store are called bug information. There are two demanding situations associated with malicious program statistics that could have an effect on the valuable use of worm repositories in software program improvement duties, specifically the huge scale and the low satisfactory. On one hand, because of the daily-stated insects, a big range of new insects are stored in computer virus repositories. Taking an open supply venture, Eclipse, for example an average of 30 new insects are said to worm repositories in line with day in 2007; from 2001 to 2010, 333,371 insects were pronounced to Eclipse by using over 34,917 developers and users. It's miles a mission to manually study such big-scale bug statistics in software improvement. However, software strategies suffer from the low exceptional of computer virus statistics. Two ordinary traits of low-fine bugs are noise and redundancy. Noisy bugs can also deceive related builders at the same time as redundant insects waste the constrained time of computer virus handling. A time-consuming step of handling software bugs is bug triage, which aims to assign a correct developer to fix a new bug. 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. In manual bug triage in Eclipse, 44 percent of bugs are assigned by mistake while the time cost between opening one bug and its first triaging is 19.3 days on average. To avoid the expensive cost of manual bug triage, existing work has proposed an automatic bug triage approach, which applies text classification techniques to predict developers for bug reports. In this approach, a bug report is mapped to a document and a related developer is mapped to the label of the document. Then, bug triage is converted into a problem of text classification and is automatically solved with mature text classification techniques.

II. Related Work

In this paper [1], gift a semi-automatic technique intended to ease one a part of this manner, the mission of news to a developer. Our method applies a system mastering set of rules to the open worm repository to study the types of reviews every developer resolves. Whilst a new document arrives, the classifier produced through the system getting to know technique indicates a small range of builders appropriate to remedy the document. With this approach, they got reached precision degrees of fifty seven percent and sixty four percent on the Eclipse and Firefox improvement initiatives respectively. They got additionally carried out our method to the gcc open supply development with much less advantageous consequences. System describe the situations under which the approach is applicable and also document on the classes we learned about applying gadget studying to repositories utilized in open source improvement .

This paper [2], presents a dynamic test generation technique for the domain of dynamic Web applications. The technique utilizes both combined concrete and symbolic execution and explicit-state model checking. The technique generates tests automatically, runs the tests capturing logical constraints on inputs, and minimizes the conditions on the inputs to failing tests so that the resulting bug reports were small and useful in finding and fixing the underlying faults. Our tool Apollo implements the technique for the PHP programming language. Apollo generates test inputs for a Web application, monitors the application for crashes, and validates that the output conforms to the HTML specification. This paper presents Apollo's algorithms and implementation, and an experimental evaluation that revealed 673 faults in six PHP Web applications.

A key collaborative hub [3], for many software improvement projects was the bug file repository. Although its use can enhance the software program improvement process in some of methods, reports introduced to the repository want to be triaged. A triage determines if a record was meaningful. To assist triages with their work, this article offers a device getting to know method to create recommenders that assist with a ramification of selections aimed at streamlining the improvement method. The recommenders created with this technique are accurate; as an instance, recommenders for which developer to assign a file that they got created using this method have a precision among 70% and ninety eight% over five open source projects. Because the configuration of a recommender for a selected mission can require sizeable effort and be time consuming, we also present method to assist the configuration of such recommenders that appreciably lowers the cost of placing a recommender in place for a task. System display that recommenders for which developer must restoration a bug can be fast configured with this method and that the configured recommenders are inside 15% precision of hand-tuned developer recommenders.

On this paper [4], were able to introduce the idea of distance graph representations of text statistics. Such representations preserve facts approximately the relative ordering and distance between the words inside the graphs and offer a far richer illustration in phrases of sentence shape of the underlying facts. This technique permits knowledge discovery from textual content which isn't always viable with using a natural vector-area representation, because it loses an awful lot much less records approximately the ordering of the underlying phrases. Furthermore, this illustration does not require the improvement of latest mining and management strategies. That was because the method also can be converted right into a structural version of the vector-space representation, which lets in using all current equipment for text. Further, present techniques for graph and XML records may be at once leveraged with this new illustration. System can apply this method to a variety of mining and management programs and display its advantages and richness in exploring the structure of the underlying textual content documents.

In this system [5], present two widespread techniques to professional searching given a record collection which is formalized the use of generative probabilistic models. The primary of those directly fashions a professional's knowledge primarily based on the documents that they're related to, while the second one locates documents on topic, and then unearths the related expert. Forming dependable institutions is essential to the overall performance of professional finding structures. Consequently, in our assessment we examine the specific procedures, exploring an expansion of institutions alongside other operational parameters.

III. Implementation Details

In this system, we address the issue of information extraction for bug triage, i.e., how to lessen the bug information to spare the work expense of designers and recoup the quality to encourage the procedure of bug triage. Information lessening for bug triage intends to fabricate a little scale and removing so as to amaze arrangement of bug information bug reports and words, which are pointless or non-instructive. In our system, we consolidate existing systems of example choice and highlight choice to at the same time lessen the bug measurement and the word measurement. The diminished bug information contain less bug reports and less words than the first bug information and give comparable data over the first bug information. We assess the decreased bug information as per two criteria: the size of an information set and the precision of bug triage. To keep away from the predisposition of a specific calculation, we experimentally analyze the consequences of four example choice calculations and four component determination calculations.

We propose bug data reduction to reduce the scale and to improve the quality of data in bug repositories which is applied as a phase in data preparation of bug triage. We combine existing techniques of instance selection and feature selection to remove certain bug reports and words. A problem for reducing the bug data is to determine the order of applying instance selection and feature selection, which is denoted as the prediction of reduction orders. In this system we use SVM algorithm for data classification. In our system SVM algorithm classify the error according to error type.

Feature selection and instance selection algorithm reduce data redundancy steps of algorithm are given below;

```
Algorithm: Data Reduction based on FS-> IS

Input: Training set T with n words and m bug reports,

        Reduction order FS->IS

        Final number nF of words,

        Final number mI of bug reports

Output: reduced data set T FI for bug triage

1) Apply FS to n words of T and calculate objective values for all the words;
```

2) Select the top nF words of T and generate a training set T F ;

3) Apply IS to mI bug reports of T F ;

4) Terminate IS when the number of bug reports is equal to or less than mI and generate the final training set T FI .

In this system we use SVM algorithm for data classification. In our system SVM algorithm classify the error according to error type.

```
Algorithm: SVM Classification
Input: A linearly separable set S, learning rate R
w0 = 0; b0 = 0; k = 0;
R = max||xi||
1≤i≤l
While at least one mistake is made in the for loop do
For i = 1; : : : ; l do
If yi(< wk; xi > +bk) ≤ 0 then
wk+1 = wk + nyixi
bk+1 = bk + nyiR2 (updating bias1)
k = k + 1
end if
end for
end while
Return wk; bk, where k is the number of mistakes
```

SYSTEM ARCHITECTURE

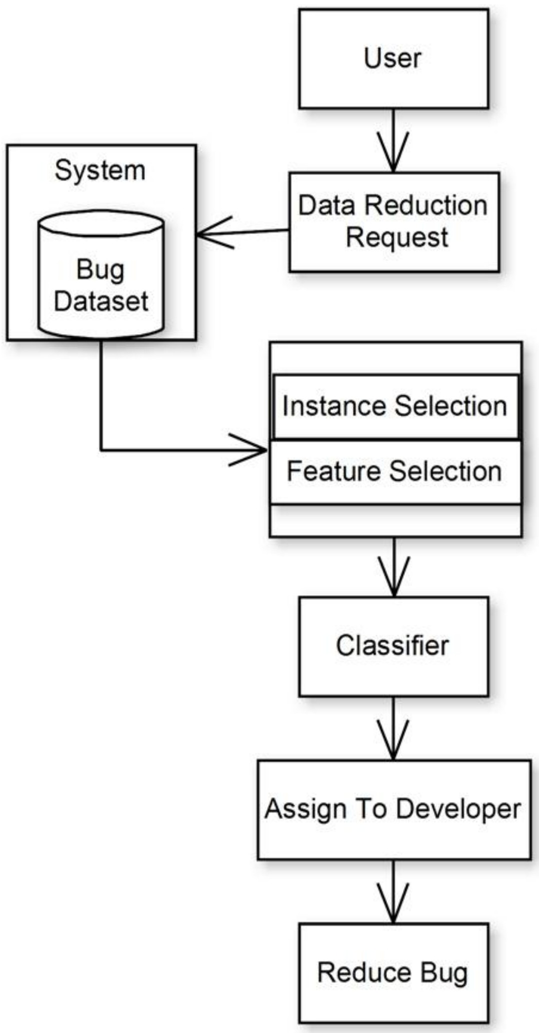

Fig.1. System Architecture

The main module of the proposed system is,

I. Data reduction For Bug Triage

II. Instance and Feature Selection

III. Attributes for a Bug Data Set

IV. Bug Triage

V. Data Quality in Defect Prediction

1. Data reduction for bug triage

We propose bug data reduction to reduce the scale and to improve the quality of data in bug repositories.

2. Instance and feature selection

In bug triage, a bug data set is converted into a text matrix with two dimensions, namely the bug dimension and the word dimension. In our work, we leverage 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.

3. Attributes for a bug data set

To build a binary classifier to predict reduction orders, we extract 18 attributes to describe each bug data set. Such attributes can be extracted before new bugs are triaged.

4. Bug triage

In data mining, the problem of bug triage relates to the problems of expert finding. In contrast to the broad domains inexpert finding or ticket routing, bug triage only focuses on assign developers for bug reports.

5. Data quality in defect prediction

In our project, we address the problem of data reduction for bug triage. To our knowledge, no existing work has investigated the bug data sets for bug triage. In a related problem, defect prediction, some work has focused on the data quality of software defects.

IV. Result Analysis

Algorithm	Precision	Recall
CH -> ICF	51.3	48.0
ICF-> CH	79.4	81.5
IS-> FS	81.6	85.7

Table 1. Value of Precision and Recall

The above table shows the value of precision and recall there is four term true positive (TP), true negative (TN), false positive (FP), false negative (FN). By using this term precision and recall is calculating. if both objects belong to the same class and same cluster then the pair is a true positive (TP); if objects belong to the same cluster but different classes the pair is a false positive (FP); if objects belong to the same class but different clusters the pair is a false negative (FN); otherwise the objects belong to different classes and different clusters, and the pair is a true negative (TN).

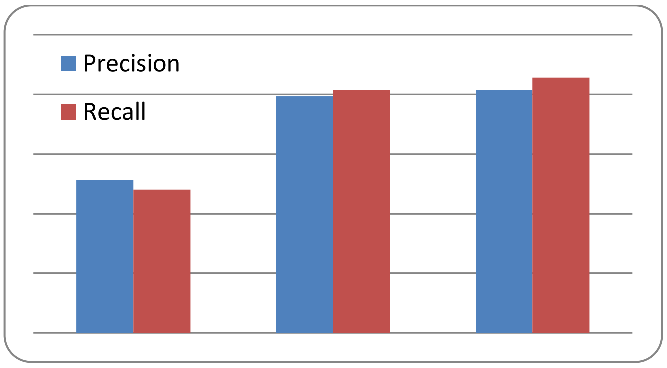

Fig 2. Comparison of algorithm

The fig 2 shows comparison of algorithm its show the value of precision and recall in graphical view . Its shows the better technique for data reduction.

V. Conclusion and Future Work

Bug triage is an expensive step of software maintenance in both manual or labor cost and time cost. In this paper, we combine feature selection with instance selection to reduce the scale of bug data sets as well as improve the data quality. To determine the order of applying instance selection and feature selection for a new bug data set, we extract attributes of each bug data set and train a predictive model based on historical data sets. We empirically investigate the data reduction for bug triage in bug repositories of two large open source projects, namely Eclipse and Mozilla. Our work provides an approach to leveraging techniques on data processing to form reduced and high-quality bug data in software development and maintenance.

In future work, we plan on improving 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. For predicting reduction orders, we plan to pay efforts to find out the potential relationship between the attributes of bug data sets and the reduction orders.

ACKNOWLEDGEMENT

The authors would like to thank the researchers as well as publishers for making their resources available and teachers for their guidance. We are thankful to the authorities of Savitribai Phule University of Pune. We are also thankful to the reviewer for their valuable suggestions. We also thank the college authorities for providing the required infrastructure and support. Finally, we would like to extend a heartfelt gratitude to friends and family members.

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. 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.
6. Bugzilla, (2014). [Online]. Avaialble: http://bugzilla.org/
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.