

Bug Triage System with Bug Data Analysis

Renu Deorankar¹ Prof. Shubhangi Vairagar²

¹ME Student ²Assistant Professor

^{1,2}Department of Computer Engineering

^{1,2}University of Pune, Maharashtra

Abstract— Software bugs are the major areas where most of the IT companies spend its 45% of efforts. The most efficient way of reducing bugs in the software is bug triage, which is appropriately assigning the new but to a developer. Automatic Bug triage is conducted by using text classification methods to minimize the manual work. In this paper, the problem of huge data for bug triage is addressed i.e. reducing the amount of data in assigning the bugs to developers. This technique combines feature selection and instance selection for simultaneously minimizing the bug dimension and word dimension. To identify the application order of instance selection and then feature selection, extraction of various attributes from previous bug track record is carried out to build new bug track or dataset. The proposed system tests its performance on the sample dataset of the open source software i.e. Eclipse. The proposed system works on novel data processing techniques to generate small amount of but highly informative bug report data in software maintenance and development.

Key words: Feature Selection, Instance Selection, Data Management For Bug Repositories, Bug Triage

I. INTRODUCTION

Mining various software repositories in these days has become essential and hence employment of data mining tools for mining data for software maintenance is necessary [2]. In the current software development environment, the output of software development is stored into large software repositories for maintaining large scale bug data, e.g. codes, emails, specifications and bugs. Conventional was of software analysis is not convenient for complex and huge data [5]. Software data has been efficiently handled by the data mining techniques in current digital age. (e.g., [7]). In proposed approach for bug data reduction, we scale out the data by analyzing and categorizing the bug data which will indirectly increase the quality of the data. Proposed approach is a combinational process of feature selection along with instance selection simultaneously. Also the proposed system comprised of a novel module to determine the status of the big checking the current state of the bug whether it's rectified or not or it's assigned to some developer or not.

Software bug management is carried out with an important entity names as bug repository. An open bug repository is available in many open source software, which not only allow users to report about the issues or defects found in the software, but also suggest future enhancements as per user's needs, and also comment or resolve the existing bug reports. The quantity of bug reports have exponentially expanded which is making bug triaging for prominent open source software [2]. Two noteworthy difficulties being confronted in software bug archives are gigantic measure of bug report information and the nature of bug report. A bug archive keeps up as a bug report in a

literary depiction shape and is redesigned by status pecking order of the bug settling [1].

The proposed system makes use of feature selection and instance selection with along with historical data for minimizing the bug sizes in the bug repository in order to obtain high quality and small size informational data. The proposed system also will show the graphical analysis of the proposed system and existing system efficiency.

II. ARCHITECTURE

A. Bug Triage:

Vital role of bug triage is to appropriately assign a developer to the corresponding bugs. The developer will reopen or rectify the bug only if the bug is assigned to the developer. Contingent on the status upgraded by the [1]. Bug triage plans to allot a proper designer to alter another bug, i.e., to figure out who ought to settle a bug.

B. Data Reduction:

Here the feature selection and instance selection is applied to reduce the bug report data size so as to get low scale data but a higher quality data. In the proposed work, to relieve the work expense of designers, bug triage information diminishment has two basic points, 1) information scale decrease and 2) precision change of bug information. In contradict to printed information displaying like as in existing framework, we point in building the subset of the first dataset for pre-processing, material before a current bug triage system.

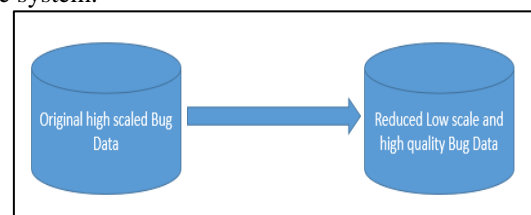


Fig. 1: Data Reduction

C. Instance Selection

Instance selection technique is corresponding with mining of data tasks such as clustering and classification :

- It's an uncommon process of identifying valid, novel, and efficiently important, and finally understandable patterns in data mining. Selecting the small subset of the data from the entire data as if the whole data is being processed.
- The standard outcome of instance selection is independent of a model:

$$P(M_s) == P(M_w).$$

1) Evaluation measures:

- 1) Direct Measure: Maintaining the similarity between subset of data and original data. Ex) Entropy, histograms, moments.

- 2) Indirect Measure: For an example, Instance selection is better or worse in predictive accuracy is done on basis of a classifier. Traditional evaluation technique in sampling, clustering and classification, can be used in performance assessment of instance selection. Ex) Recall, Precision.

D. Feature Selection

- It select a minimum set of features such that the probability distribution of different classes given the values for those features is as close as possible to the original distribution given the values of all features [1].
- Avoid Reduce # of patterns in the patterns, easier to understand.
- Do create new attributes that can capture the important information in a data set much more efficiently than the original attributes.
- Use the smallest representation which is enough to solve the task.

1) Heuristic methods:

- Step-wise forward selection
- Step-wise backward elimination
- Create new attributes that can capture the important information in a data set much more efficiently than the original attributes:

Three general methodologies:

- 1) Feature extraction.
- 2) Domain-specific.
- 3) Mapping data to new space (see: data reduction)

III. ALGORITHM

Let $X = \{x_1, x_2, x_3, \dots, x_n\}$ be the set of data points and $V = \{v_1, v_2, \dots, v_c\}$ be the set of centers.

- 1) Step 1: Randomly select 'c' cluster centers.
- 2) Step 2: Calculate the distance between each data point and cluster centers.
- 3) Step 3: Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers.
- 4) Step 4: Recalculate the new cluster center using:

$$v_i = (1/c_i) \sum_{j=1}^{c_i} x_j$$

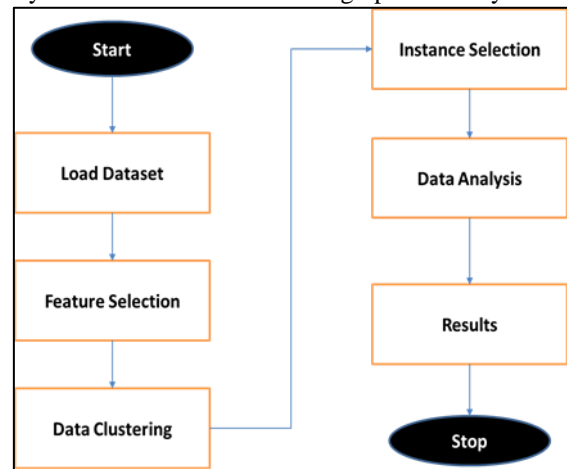
where, 'c_i' represents the number of data points in *i*th cluster.

- 5) Step 5: Recalculate the distance between each data point and new obtained cluster centers.
- 6) Step 6: If no data point was reassigned then stop, otherwise repeat from step 3.

IV. WORKING OF PROPOSED SYSTEM

- The working of the proposed system is described as below:
- Dataset Collection and dataset loading into application of bug assignment and analysis system.
- User authentication on privilege level for bug assignment operation and bug resolution operation.
- Data Preprocessing and loaded data analyzing based on differential symbols.
- New bug or old bug reassigning facility for Bug triaging.

- Data Reduction and instance selection module for differentiating instances of bug status.
- Determining the Feature selection from the instance selection process.
- View various Bug reports as per bug status.
- System evaluation module for graphical analysis.



V. EXPERIMENTAL RESULTS

The experimental results of the proposed system shows that the input random large data for bug reports is analysed under combinational implementation of instance and feature selection thereby making the bug data smaller and highly efficient for bug triaging and assigning the new tasks as per the developer history of bug fixing. The instances of the bugs are differentiated based on the bug status after assigning the bugs to the developers. Unless the bugs are assigned to any developer any data regarding bug triaging cannot be analysed for future bug status tracking. Thus the proposed system aids in bug triaging based on historical data of bug reports and thereby make software bug triaging easier with reduction in size of bug repository and bug reports.

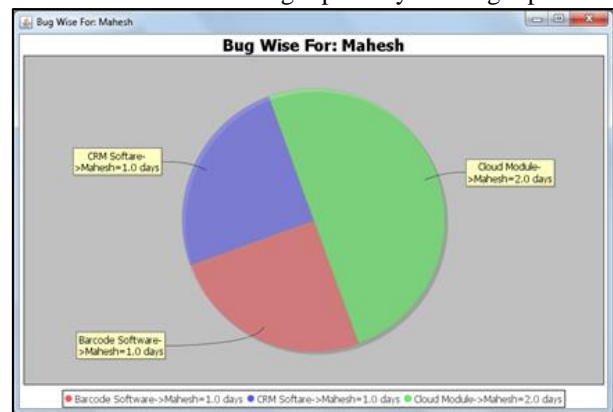


Fig. 3: Bug wise analysis of Employee

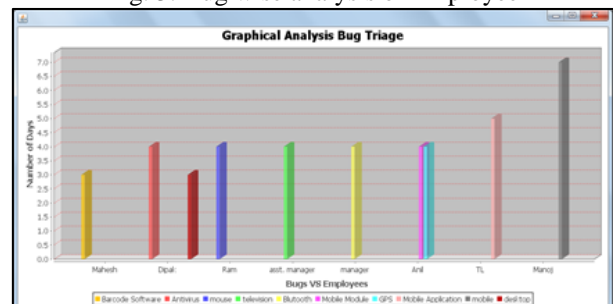


Fig. 4: Bug vs Employees Graphical Analysis

ACKNOWLEDGMENT

I would like to take this opportunity to express my profound gratitude and deep regard to my guide Prof. Shubhangi Vairagar for her exemplary guidance, valuable feedback and constant encouragement throughout the duration of the project. Her valuable suggestions were of immense help throughout my project work. Her perceptive criticism kept me working to make this project in a much better way. Working under her was an extremely knowledgeable experience for me.

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] Mamdouh Alenezi and Kenneth Magel, Shadi Banitaan "Efficient Bug Triaging Using Text Mining" 2013 academy publisher.
- [3] Francisco Servant "Supporting Bug Investigation using History Analysis" 978-1-4799-0215-6/13 c 2013 IEEE.
- [4] Pamela Bhattacharya, Iulian Neamtiu, Christian R. Shelton, "Automated, Highly- Accurate, Bug Assignment Using Machine Learning and Tossing Graphs", May 2, 2012.
- [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] P. S. Bishnu and V. Bhattacharjee, "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.
- [7] 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. A. K. Uysal and S. Gunal, "A novel probabilistic feature selection method for text classification," *Knowledge-Based Systems*, vol. 36, no. 0, pp. 226–235, 2012.
- [8] 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.
- [9] A. Lamkanfi, S. Demeyer, E. Giger, and B. Goethals, "Predicting the severity of a reported bug," in *Proc. 7th IEEE Working Conf. Mining*
- [10] *Softw. Repositories*, May 2010, pp. 1–10.