p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 13
September 2016

Bug Triage of the Software Data to Reduce the Bugs in the Application Software

Mrs. B.Mounika[1], Mr. S.Ramana Reddy[2]

ABSTRACT:

Software companies spend over 45 percent of cost in dealing with software bugs. An inevitable step of fixing bugs is bug triage, which aims to correctly assign a developer to a new bug. To decrease the time cost in manual work, text classification techniques are applied to conduct automatic bug triage. In this paper, we address the problem of data reduction for bug triage, i.e., how to reduce the scale and improve the quality of bug data. We combine instance selection with feature selection to simultaneously reduce data scale on the bug dimension and the word dimension. To determine the order of applying instance selection and feature selection, we extract attributes from historical bug data sets and build a predictive model for a new bug data set. We empirically investigate the performance of data reduction on totally 600,000 bug reports of two large open source projects, namely Eclipse and Mozilla. The results show that our data reduction can effectively reduce the data scale and improve the accuracy of bug triage. Our work provides an approach to leveraging techniques on data processing to form reduced and high-quality bug data in software development and maintenance.

INTRODUCTION

MINING software repositories is an interdisciplinary domain, which aims to employ data mining to deal with software engineering problems [22]. In modern software development, software repositories are large-scale databases for storing the output of software development, e.g., source code, bugs, emails, and specifications. Traditional software analysis is not completely suitable for the large-scale and complex data in software repositories [58].

Data mining has emerged as a promising means to handle software data (e.g., [7], [32]). By leveraging data mining techniques, mining software repositories can uncover interesting information in software repositories and solve real world software problems.

A bug repository (a typical software repository, for storing details of bugs), plays an important role in managing software bugs. Software bugs are inevitable and fixing bugs is expensive in software development. Software companies spend over 45 percent of cost in fixing bugs [39]. Large software projects deploy bug repositories (also called bug tracking systems) to support information collection and to assist developers to handle bugs [9], [14]. In a bug repository, a bug is maintained as a bug report, which records the textual description of reproducing the bug and updates according to the status of bug fixing [64]. A bug repository provides a data platform to support many types of tasks on bugs, e.g., fault prediction [7], [49], bug localization [2], and reopenedbug analysis [63]. In this paper, bug reports in a bug repository are called bug data.

The primary contributions of this paper are as follows: 1) We present the problem of data reduction for bug triage. This problem aims to augment the data set of bug triage in two aspects, namely a) to simultaneously reduce the scales of the bug dimension and the word dimension and b) to improve the accuracy of bug triage.

2) We propose a combination approach to addressing the problem of data reduction. This can be viewed as an application of instance selection and feature selection in bug repositories. 3) We build a binary classifier to predict the order of applying instance selection and feature selection. To our knowledge, the order of applying instance selection and feature selection has not been investigated in related domains 2 BACKGROUND AND MOTIVATION

2.1 Background

Bug repositories are widely used for maintaining software bugs, e.g., a popular and open source bug repository, Bugzilla [5]. Once a software bug is found, a reporter (typically a developer, a tester, or an end user) records this bug to the bug repository. A recorded bug is called a bug report, which has multiple items for detailing the information of reproducing the bug. In Fig. 1, we show a part of bug report for bug 284541 in Eclipse.2 In a bug report, the summary and the description are two key items about the information of the bug, which are recorded in natural languages. As their names suggest, the summary denotes a general statement for identifying a bug while the description gives the details for reproducing the bug

DISCUSSION

In this paper, we propose the problem of data reduction for bug triage to reduce the scales of data sets and to improve the quality of bug reports. We use techniques of instance selection and feature selection to reduce noise and redundancy in bug data sets. However, not all the noise and redundancy are removed. For example, as mentioned in Section 5.2.4, only less than 50 percent of duplicate bug reports can be removed in data reduction (198=532 ¼ 37:2% by CH ! ICF and 262=532 ¼ 49:2% by ICF ! CH). The reason for this fact is that it is hard to exactly detect noise and redundancy in real-world applications. On one hand, due to the large scales of bug repositories, there exist no adequate labels to mark whether a bug report or a word belongs to noise or redundancy; on the other hand, since all the bug reports in a bug repository are recorded in natural languages, even noisy and redundant data may contain useful information for bug fixing

CONCLUSIONS

Bug triage is an expensive step of software maintenance in both 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 invest gate 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 highquality 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.

, “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.

[15] A. K. Farahat, A. Ghodsi, M. S. Kamel, "Efficient greedy feature
selection for unsupervised learning," Knowl. Inform. Syst., vol. 35,
no. 2, pp. 285–310, May 2013.

[16] N. E. Fenton and S. L. Pfleeger, Software Metrics: A Rigorous and
Practical Approach, 2nd ed. Boston, MA, USA: PWS Publishing,
1998.

[17] Y. Freund and R. E. Schapire, "Experiments with a new boosting
algorithm," in Proc. 13th Int. Conf. Mach. Learn., Jul. 1996, pp. 148–
156.

[18] Y. Fu, X. Zhu, and B. Li, "A survey on instance selection for active
learning," Knowl. Inform. Syst., vol. 35, no. 2, pp. 249–283, 2013.

[19] I. Guyon and A. Elisseeff, "An introduction to variable and feature
selection," J. Mach. Learn. Res., vol. 3, pp. 1157–1182, 2003.

[20] M. Grochowski and N. Jankowski, "Comparison of instance selection

algorithms ii, results and comments," in Proc. 7th Int. Conf.
Artif. Intell. Softw. Comput., Jun. 2004, pp. 580–585.

Author's profile:

Mr. S.Ramana Reddy received M.Tech(CSE) Degree from School of Information Technology, Autonomous, and Affiliated to JNTUH, Hyderabad. He is currently working as Assistant Professor in the Department of Computer Science and Engineering in Nalgonda Institute of Technology and Science, Nalgonda, Telangana, India. His interests includes Object Oriented Programming, Operating System, Database Management System, Computer Networking, Cloud Computing and Software Quality Assurance.

Mrs. B.Mounika received B.Tech Degree from Swami Ramananda Tirtha Institute of Science and Technology in Nalgonda. She is currently pursuing M.Tech Degree in Computer Science and Engineering specialization in Nalgonda Institute of Technology & Science in Nalgonda, Telangana, India.