Classification Algorithms for Detecting Duplicate Bug Reports in Large Open Source Repositories

Sarah Ritchey (Computer Science and Mathematics) sritchey@student.ysu.edu - student

Bonita Sharif (Computer Science) bsharif@ysu.edu – faculty advisor

Alina Lazar (Computer Science) alazar@ysu.edu – faculty advisor

Youngstown State University

Introduction

Today any open source software projects, as well as any commercial software projects, use bug tracking or issue tracking software deal with all the defect reports. Software maintenance activities, including fixing software defects, have become an important part of the software development process. The triage and processing of bug reports are time consuming tasks since in general more bugs are reported for larger systems. Also, many of the bug reports in a large-scale software project are often identified as being duplicates of known bugs, invalid or incomplete reports, for some project more than 30% [11]. In the past, the triage, filtering and prioritization has been done by hand, but lately many automated techniques have been proposed [1, 4, 6, 7, 9, 10, 12, 14]. To perform the filtering manually the person doing it, must rely on her/his memory and on basic textual searching that may not provide very accurate results. A fast and accurate duplicate bug report identification process can help the overall process of improving the software system.

It is expected that an automated process of duplicate bug identification will never be 100% accurate, so instead of simply eliminating duplicates a better approach will be to provide for each new incoming bug report a list of similar reports [13]. This procedure considerably reduce the number of bug reports, a triager should inspect. Only the top-k most similar bug reports to an existing report returned by the systems need to be manually checked.

One category of approaches for automatic duplicate bug report identification is based on natural language processing and the analysis of the summary and textual description of each bug report. Runeson et al. [17] were the first to apply natural language processing techniques such as tokenization, stemming and stop word removal to the textual part of the bug reports. To represent the words in a multi-dimensional vector space model the following formula, based on term frequency (*tf*), is used $weight = 1 + log(tf(word))$ plus the cosine similarity measure. The study was performed on bug reports collected for the Sony Ericson Mobile software. Another study [16] that extended the work of Runeson et al. changed the formula to compute the vector space to use not only *tf*, but also the *inverse document frequency* (*idf*). Some additional execution information was included in this study, which also used the cosine similarity measure and it was performed on the Firefox bug report dataset with good results. Jalbert and Weimer [4] proposed yet another formula for deriving the vector space $weight(word) = 3+2 * log2(tf(word))$, used the same cosine similarity measure, but employed graph clustering technique. They report the findings on a subset extracted for the Firefox bug report dataset.

The research project proposed here extends earlier work by Sun et al. [13]. They were the first to propose and use an innovative method [8] to create the instances to be used in the classification model. Instead of just using a vector space derived directly from the actual

samples, the data examples are created by concatenating textual parts of two different bug reports. For each example 54 features are computed, 27 based on unigrams and 27 based on bigrams. The newly created instances are then used to train a linear Support Vector Machine (SVM) [2, 3] to build a classification model. This study was performed on three bug repositories: Firefox, Eclipse and Open Office. For our project we are going to investigate other possible features that can be derived. We are also going to explore the performance of the classification algorithm, by comparing several linear classification algorithms such as SVM and Stochastic Gradient Decent [5]. Finally, we are going to extend the study to other large open-source repositories [11] such as Thunderbird, NetBeans and Android.

Research Questions and Hypotheses

Based on a the review of existing work that is related to a) the information retrieval from bug reports and b) the automated classification of duplicate bug reports, we identify the following open research questions which will be addressed by our project:

- What features extracted from the textual part of the bug reporters can be used to predict which bug reports refer to duplicate bugs?
- What are the best machine learning algorithms, classification and regression, for this problem?
- Does the proposed method works equally well on any one-source project bug repository?

Methods

The duplicate bug report retrieval problem can be approached as a binary classification problem. The dataset contains all the existing bug reports plus a label to identify the class: duplicate or non-duplicate. This dataset is divided into two subsets training and testing. The classification model is build based on the training set and evaluated on the testing set. Once the classification model is ready, new bug reports can be classified and also a list of potential duplicate bugs can be returned.

The positive instances, labeled “duplicate”, in the training set are derived from the bug report list by paring two reports where one is considered the duplicate of the other. In the same way the negative instances pair reports that are not duplicates. Each bug report has two textual parts: the summary and the description. The actual classification features are generated by computing textual similarity measures such as the inverse document frequency (idf) or the term frequency (*tf*) between combinations of summaries and descriptions.

Linear classification has been proven to be one of the most promising machine learning techniques, especially for solving problems with a huge number of instances and/or features. Based on their use on a variety of different classification problems, we already know that linear SVMs, especially Liblinear [2, 3] provide good prediction accuracy when used for document classification. Another linear classification approach is implemented in the Vowpal Wabbit [5] software. This implementation uses an online gradient decent algorithm.

Performance Metrics for Classification

Especially in the case of unbalanced datasets, classification accuracy alone is not the best metric to evaluate a classifier. Several other performance metrics can be used in order to obtain a more comprehensive picture about the classifier's capabilities, especially when the successful detection of one class (duplicate reports) is considered more significant.

Recall is the metrics that gives the accuracy on the positive instances and can be calculated as TruePositive / (TruePositive + FalseNegative). It actually measures the fraction of positive examples correctly predicted by the classifier. Precision is the accuracy on the negative instances and can be calculated as TruePositive / (TruePositive + FalsePositive). The higher the precision is, the lower the number of false positive errors is going to be.

For our problem, given the size of a candidate list of bug reports, the *recall rate* can be interpreted as the percentage of duplicates whose related bug reports are successfully retrieved in the list. In terms of this measure, the result can show that one approach has remarkable improvement over others.

Impact on the Goal of CREU

The foremost goal of the CREU project is to encourage females and minorities to pursue graduate work and study in the field of computer science. This project will provide a realistic research experience for the female undergraduate, by active involvement in the planning, execution and interpretation of scientific research. Well-developed research projects can significantly enrich the educational experience for undergraduate students. Working on this research project, the students will be able to enhance their computer and programming skills, apply those skills to investigate scientific problems, learn how to formulate questions and problems and to participate in the discovery of new knowledge. A good research experience can foster an enthusiasm for lifelong learning and a desire to continue education beyond the bachelor's degree. Successful scientific instruction should develop in students a sense of wonder and curiosity about the world. The students will be exposed to both sides of the scientific investigation: hypothesis testing and development of theoretical explanations of observations. No science education is complete without research related activities, technical writing and oral presentations.

Rachel Turner considers this project a continuation of the CREU program she has been part of for the last year. As an undergraduate female with intentions of pursuing graduate work in computer science, this project will give her, a useful introduction to the practical applications of her studies for research. This is a project that can potentially be a foundation for her senior thesis, and also it will be a wonderful basis for a research poster. Her experience will be very helpful to the other student in the group.

Sarah Ritchey is junior student pursuing a double major in Mathematics and Computer Science with a 4.0 GPA. She is really interested in pursuing a research project in computer science. Her mathematical background will be ideal for this type of research project.

The students intend to present this project at next year's QUEST at Youngstown State University (a program highlighting student research). The students will likely also submit findings at regional and/or national scientific meetings (ICPC and ICSM) within their fields. Results generated by this project will be included in one or more future manuscripts, with

participating student afforded full opportunity to share the responsibilities and rewards of authorship.

Student Activity and Responsibilities

Specific tasks for the participant student will include: literature search and review, reading and discussing research articles, designing and implementing the experimental studies, data processing, data analysis and interpretation, summarizing and preparing results for presentations and publications, YSU QUEST 2014 participation and writing the final report. The students will also be mentored to prepare a conference paper that will be sent to the International Conference on Program Comprehension (ICPC) and/or Mining Software Repositories (MSR).

The primary responsibility of the students is to participate in all phases of the project: proposal, development, experiments, and dissemination. The students will be required to do weekly independent work and to schedule team meetings with the faculty advisors. The faculty advisors will meet with the student every week. Email and a central repository will be used for questions, announcements and documents interchange.

Faculty Activity and Responsibilities

Dr. Sharif has conducted several empirical studies in the field of software engineering using various means of data collection such as questionnaire and eye tracking hardware and software. She will be responsible for the developing concrete research hypotheses for each of the research questions mentioned above. She will mentor the students to design the experiments that address the research questions. This will involve experimental design including selection of variables and methods before the study is implemented. The students will be part of the design and variable selection. After the data is collected, she will also be responsible for guiding the student to choose appropriate methods of analyzing the data.

As faculty advisor for the proposed project Dr. Alina Lazar will work to actively mentor the student and continuously supervise their progress during the one year period. She will meet with the student on regular basis to guide their activities and answer their questions related to the project. Dr. Lazar has extensive experience in data mining, machine learning and artificial intelligence and she has written several papers related to that field. Dr. Lazar will help the student with the data analysis, and with the final report and also with the preparation of a conference paper.

The overall guidance and mentoring will not refer only to this project but it will provide insights about how to apply and how to succeed in graduate school, about being a female scientist and what the options are after graduate school.

Budget

For the proposed project we are requesting $3000 for each student. The additional $1500 will be used to buy computer media, books, and other materials necessary for the project. While working on the project the student will be encouraged to apply for the Undergraduate Research Grant Award sponsored by the Youngstown State University and other scholarships.

We foresee that we are going to apply for the summer extension next February.

Table 1. Project Time Table

Task Name	2012				2013							
	S	O	N	D	J	F	M	A	M	J	J	A
Literature review research about studies related to duplicate bug report identification and data mining classification algorithms.	←	←	←									
Analyze existing bug repositories to understand the dataset format and the processing steps necessary.	←	←	←									
Develop and conduct experimental studies using the data mining algorithms.	←	←	←	←	←	←	←					
Statistically analyze the data generated by running the algorithms.			←	←	←	←	←	←	←	←	←	→
Dissemination of results through papers and communications at specific conferences.				←	←	←	←	←	←	←	←	→
Evaluation of the applicability of the performed studies to other areas or problems.									←	←	←	→

Role of the CREU project within the larger scope of this research

The Department of Computer Science and Information Systems at Youngstown State University has always valued undergraduate research and provided resources for projects. The faculty advisors recently created a new lab as part of an internal grant that has access to state-of-the-art machines that students can use for research projects. The students will use the lab for data analysis and running the experiments.

Prior results of CREU projects

Bonita Sharif and Alina Lazar guided student Rachel Turner during the CREU project titled: "*An Empirical Investigation on Code Debugging and Understanding: An Eye-tracking Perspective*". Rachel presented the poster "*C ++ vs. Python: An Eye-tracking Assessment*" at the OCWIC 2013 Ohio Celebration of Women in Computing conference and a talk at the YSU's QUEST 2013 A Forum for Student Scholarship. We also submitted a paper to CSEE&T 2013 and although we did not get accepted into the conference we did receive very helpful reviews and plan to further expand our research by taking the reviewer comments into account.

Alina Lazar advised two prior CREU projects in 2004-2005 and 2006-2007, and one MROW project in 2007-2008. Darcy Davis the participant student in the 2004-2005 project just finished her PhD in Computer Science and Engineering at Notre Dame University. Louise Popio who participated as a student during 2006-2007, received her Master's in Information Sciences and Technology from Pennsylvania State University in 2010. Irena Lanc participated in the 2007-2008 MROW project and received her Master's in Computer Science and Engineering from Notre Dame University this year and is currently pursuing a PhD at the same institution. Erin Pfeil the other student that did the 2007-2008 MROW project, is working on her PhD in Ecology at University of Pittsburgh.

References

[1] Bettenburg, N., Premraj, R., Zimmermann, T. and Kim, S. 2008. Duplicate bug reports considered harmful ... really? *IEEE International Conference on Software Maintenance, 2008. ICSM 2008* (2008), 337–345.

[2] Hsiang-Fu Yu, Chia-Hua Ho, Prakash Arunachalam, Manas Somaiya and Chih-Jen Lin 2012. Product title classification versus text classification. *Technical Report* (2012).

[3] Hsiang-Fu Yu, Chia-Hua Ho, Yu-Chin Juan and Chih-Jen Lin 2013. LibShortText: A Library for Short-text Classi_cation and Analysis. *Technical Report* (2013).

[4] Jalbert, N. and Weimer, W. 2008. Automated duplicate detection for bug tracking systems. *IEEE International Conference on Dependable Systems and Networks With FTCS and DCC, 2008. DSN 2008* (2008), 52–61.

[5] John Langford, Lihong Li and Alexander Strehl 2007. Vowpal wabbit open source project. *Technical Report* (2007).

[6] Kaushik, N. and Tahvildari, L. 2012. A Comparative Study of the Performance of IR Models on Duplicate Bug Detection. *2012 16th European Conference on Software Maintenance and Reengineering (CSMR)* (2012), 159–168.

[7] Mehdi Amoui, Nilam Kaushik, Abraham Al-Dabbagh, Ladan Tahvildari, Shimin Li and Weining Liu 2013. Search-Based Duplicate Defect Detection: An Industrial Experience. (San Francisco, CA, May. 2013).

[8] Nallapati, R. 2004. Discriminative models for information retrieval. *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval* (New York, NY, USA, 2004), 64–71.

[9] Nguyen, A.T., Nguyen, T.T., Nguyen, T.N., Lo, D. and Sun, C. 2012. Duplicate bug report detection with a combination of information retrieval and topic modeling. *Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering* (New York, NY, USA, 2012), 70–79.

[10] Runeson, P., Alexandersson, M. and Nyholm, O. 2007. Detection of Duplicate Defect Reports Using Natural Language Processing. *29th International Conference on Software Engineering, 2007. ICSE 2007* (2007), 499–510.

[11] Serrano Zanetti, M., Scholtes, I., Tessone, C.J. and Schweitzer, F. 2013. Categorizing Bugs with Social Networks: A Case Study on Four Open Source Software Communities. *eprint arXiv:1302.6764*. (Feb. 2013).

[12] Sun, C., Lo, D., Khoo, S.-C. and Jiang, J. 2011. Towards more accurate retrieval of duplicate bug reports. *Proceedings of the 2011 26th IEEE/ACM International Conference on Automated Software Engineering* (Washington, DC, USA, 2011), 253–262.

[13] Sun, C., Lo, D., Wang, X., Jiang, J. and Khoo, S.-C. 2010. A discriminative model approach for accurate duplicate bug report retrieval. *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1* (New York, NY, USA, 2010), 45–54.

[14] Sureka, A. and Jalote, P. 2010. Detecting Duplicate Bug Report Using Character N-Gram-Based Features. *Software Engineering Conference (APSEC), 2010 17th Asia Pacific* (2010), 366–374.

[15] Tian, Y., Sun, C. and Lo, D. 2012. Improved Duplicate Bug Report Identification. *2012 16th European Conference on Software Maintenance and Reengineering (CSMR)* (2012), 385–390.

[16] Wang, X., Zhang, L., Xie, T., Anvik, J. and Sun, J. 2008. An approach to detecting duplicate bug reports using natural language and execution information. *Proceedings of the 30th international conference on Software engineering* (New York, NY, USA, 2008), 461–470.