But Do Commit Messages Matter? An Empirical Association Analysis with Technical Debt

Chien Lu

Tampere University

Abstract. An empirical analysis is conducted to investigate the association of the content of commit messages and technical debt. The analysis is based on 33 open-source Apache JAVA projects. Structural Topic Modelling, a recently developed text mining technique is employed for sophisticated analysis. The result shows that the certain content of commit messages such as empty messages are potentially associated with Technical Debt.

Keywords: commit messages · technical debt · text mining

1 Introduction

This research investigates the relationship between the content commit messages and Technical Debt (TD) issues based on 33 real-world, open-source Apache JAVA projects. Although the two above-mentioned research topics have been discussed separately in related fields, the author argues that an integrated analysis for exploring the relationship between commit messages and TD is able to provide novel insights. Therefore, the research questions of this work are:

1. Do commit messages make a difference when it comes on the TD ?
2. What kind of commit messages can potentially increase (or decrease) TD ?
3. Does empty commit message increase TD ?

1.1 Commit Messages

Commit messages writing plays an important role in software development for it records, or documents the changes in natural languages during the progress of the project. It is believed that well-written commit messages can provide useful information to other developers in the life-cycle and eventually contribute to enhancing the software quality in the end ([11] and [22]). It is almost a common sense that a good teamwork oriented developer takes not only excellent coding skills but also the ability to write informative commit messages.

However, to the author's best knowledge, there has not been an empirical study based on real-world projects which challenges or confirms the above mentioned "common sense". Besides, many related issues still haven't been properly discussed, for instance, does empty message influence the software quality? Or, what kind of content in the commit messages are really helpful to the software quality? The purpose of this research is to provide suggestions to developers based on empirical analysis while writing commit messages .

1.2 Technical Debt

Technical Debt (TD) [1] is a metaphor to describe the issues (including coding issues or documentation issues) generated in the development process that cause potential problems and will have to be solved by "paying back" extra efforts in the future. For example, code smells (e.g. nested, too complicated code structure) is one type of TD issues which will potentially result in difficulties in maintenance ([6] and [15]).

Instead of analyzing the software with human efforts e.g. code reviews, to find out the potential TD issues, there are several Technical Debt measurement tools available such as Better Code Hub [1], Coverity Scan [2] and SonarQube [3]. Among those mentioned tools SonarQube is one of the most commonly used tool in both industry and research community [14]. One effective functions of SonarQube is that it can automatically detect TD issues in the software with rule-based algorithms.

TD has attracted considerable attentions in research communities (e.g. [4], [16] and [21]). Recently one open access dataset [13] has been proposed and it provides more opportunities of TD related data analysis. One recent work [19] has provided basic analysis including diffuseness and distribution of TD of the collected projects based one the above-mentioned dataset.

1.3 Topic Modelling

When it comes to text analysis, one common approach is topic modelling ([2], [3] and [18]). A topic model is a probabilistic model in which, a topic is defined as a distribution over words, and it is assumed that when the writing is going on, the author first draw a topic label in an "topic urn" and then draw a word from the topic corresponding "word urn".

Based on the above mentioned assumption, in each document, topics are occurring with different strength (prevalence), so that some documents may be composed mostly of a particular topic whereas others are a mixture of several other topics.

Thus, topic modelling is capable of not only capturing the underlying topic content but also modelling the topic prevalence among a set documents. This research employees STM [18] , a recently developed topic model to explore the relationship between commit messages and potential TD issues.

2 Related Works

There have been research works applying topic modelling to analyze commit messages. For example, Hindle et al. [9] analyzed the commit comments with LDA. Their work has performed the analysis on repositories of three database

[1] Better Code Hub, https://bettercodehub.com
[2] Coverity Scan. https://scan.coverity.com
[3] SonarQube. https://www.sonarqube.org

systems *PostgreSQL*, *MaxDB* and *Firebird*. They have found that the commit messages in three different database system emphasize different concepts. In the *PostgreSQL*, the most prevalent topic is related to the external developers Dal Zotto and Dan McGuirk, in the *MaxDB*, the most prevalent topic is related to build system files whereas in the *Firebird*, the most prevalent topic is related to commonly used terms in commit messages such as "added", "fixed", and "updated".

Hu et al. [10] have studied commit messages using Dynamic Topic Modeling (DTM) [2]to discover the underlying topics and their evolution processes over time. Their work has performed case studies on *jEdit* and *PostgreSQL*, two well-known open source software systems, each contains 12116 and 15990 commit messages respectively. The analsys has drawn some interesting topics such as "Fixing Bug and Error", "GUI" topics in the *jEdit* case and "Bug Fixing", "Building and Configuration" topics in the *PostgreSQL* case.

However, although the above mentioned works have use topic modelling techniques, they have not investigated the relationships between the extracted topics and the software quality measurements especially TD that might be potentially associated with the commit messages.

There is another group of works ([8], [12] and [20]) focus on sentiment analysis of commit logs. Among those works, the most relevant one is the work of is Islam et al. [12] which investigates bug related commits and the sentiment of the corresponding commit messages. They have analyzed more than 24000 commit messages and found that both bug-introducing and bug-fixing commit messages have higher positive emotional scores.

3 Methodology

3.1 Accumulated Debt Comuptation

In this research. The Technical Debt Dataset [13] is used. In the dataset, the variable "reliabilityRemediationEffort" is generated by SonarQube. The SonarQube automatically analyzes the changes of the current software and estimates the time spent in the future to solve or the "remediate" the detected TD issues. If the SonarQube detects that some TD issues existed before the commit are solved, the value of "reliabilityRemediationEffort" will decrease, on the other hand, the value of "reliabilityRemediationEffort" will increase if the SonarQube detects extra issues after the commit.

Using the "reliabilityRemediationEffort" variable , the "debt change" of each commit is gained by calculating the difference between the "reliabilityRemediationEffort" value before and after the commit. Moreover, an "accumulated debt" of a committer has contributed to a project can be obtained by summing up all the debt change values under a certain committer of the project.

3.2 Data Processing

The empty commit messages can potentially influence the software quality, and they also reflect the developers' style while writing commit messages. To evaluate

the impact and take it into account in the topic model, if the the commit message is empty (575 out of in total 128375 commit messages), the message is labeled as "emptymessage". The "emptymessage" is then treated as a special vocabulary. All the commit messages generated by a certain commiter under a specific project are gathered as a document. There are in total 1083 documents. To focus on the contribution of a specific developer to a project, the same developer in two different projects are considered two different developers.

The text are transformed into lower-cases and extra white space are removed. Note that, to better preserving the sophisticated writing hobbies in the commit messages, the text does not undergo some standard processing steps such stemming or lemmatizing.

The previously mentioned it the commit messages contribute to TD. Documents having positive "accumulated debt" are categorized as "debt contributed", otherwise "no debt contributed".

3.3 Text Mining

The Structural Topic Model (STM) [18] is used to analysis the relationship between commit messages topics and the TD issues. STM is a more advanced model such that, comparing with topic models such as LDA and DTM, it takes the document-level covariates into account. That is, instead of analyze the topics and their relationships to covariates in a two-stage manner, STM provides a integrated solution to the problem. The STM has been widely used in different research disciplines such as policy research [7], climate change [5].

In the model selection process, for each topic number from 6 to 20, 10 random initialized models with randomly selected 50 % of the training documents are built. The topic number with the highest average held-out likelihood value on the 50% of the testing documents is selected. After deciding the topic number, 50 models are built and the one with the best semantic coherence [17] over topics is chosen as the final model.

4 Result

Among 1083 documents (developer-project pairs), 868 of them are "no debt contributed", 215 of them are "debt contributed". The proportion of "debt contributed" documents is around 20%.

The result of model selection is shown in Figure 1. The x-axis (K) represents the number of topics and the corresponding held-out likelihood is shown on y-axis. Since the $K = 18$ reaches the highest held-out likelihood, it is thus selected.

Top words of the the extracted 18 topics can be found in Table 1. Apperently the common words such as "the", "and", "fix", "for" can be found in most of the topics. The relationship between topic prevalence and TD can be observed in Figure 2. The Topics 5, 6, 8, 9 are leading to not adding TD to the projects whereas the Topics 1, 3, 13, 15, 16 are leading to adding TD to the projects.

5 Discussion

The prevalence of common words in the extracted topics are similar to the results of the previously mentioned related works ([9] and [10]). Another reason can be due to that the stop words are not removed in the data processing steps. However, the differences between topics are still observable.

When focusing on the no debt contributing Topics 5, 6, 8 and 9, one observation worth noticing is that is seems that detailed-oriented messages can somehow reduce the TD. For example Topic 5 has the word "version", "common", Topic 6 has the word "line", "code" and "here". Besides, they both have the word "this". It is likely that the Topic 8 is related to final stage of the software development ("build", "file", "tag" and "now") so the TD have to be removed. The Topic 9 contains only common words, the reason why it reduces the TD need further analysis it can be that it does not add or reduce the TD, and it is categorized into the "no debt contributed" group.

Most of the debt contributing Topics 1, 3, 13, 15 and 16 include the term "fix". It can be due to that some code smells or TD issues are generated by over-editing the code. Topic 13 contains terms "submitted", "reviewed" and "obtained" could be related the commits in some certain stages of the software development that generates TD issues and the "emptymessage" potentially shows that empty commit messages could result in TD issues. Topic 3 and 16 could be developer or project related.

The discovered topics related to TD issues are somehow explainable, however, some more detail-oriented analysis are still required to draw a more comprehensive picture.

Another limitation of this research is that the type project and the type of the changed file of a commit is not taken into consideration. In theory, the difficulty and complexity can affect the TD. Beside, the type of changed file should affect the content of commit messages, e.g. changing ".html" files leads to more web developing related content whereas changing ".README" files is more likely to generate use user guideline messages. One practical difficulty in this analysis task is that some commits are related to not just one changed file e.g. the changed files contain ".README" files and ".html" files. Therefore the effect of the type of file is not taken into considerations. The effect of the above mentioned factors can be taken into account in future works.

6 Conclusion

This research conducted an empirical text analysis focusing on the related commit messages and TD issues. The analysis uses a sophisticated topic modelling technique to analyze a collection comprises 128375 commit messages and the corresponding estimated TD measurement across 33 different real-world open access JAVA projects. The findings show that

1. Some topics extracted from the commit messages and TD potentially associated.

2. In general, writing detailed-oriented commit messages have negative association to TD.
3. Empty commit message has positive association to TD.

However, more details and mechanisms regarding to the discovered associations still require further investigations.

Table 1. Extracted Topics. Debt contributing topics are in red, not debt contributing topics are in green, neutral topics are in black.

Topic Number	Top 6 Words
Topic 1	the, and, for, block, fixed, from, use, bug, fixing
Topic 2	fix, add, for, remove, and, update, bug, use, support
Topic 3	alexantonenko, for, via, contributed, and, fix, shwethags, the, atlas
Topic 4	the, for, closes, and, this, add, mavenreleaseplugin, prepare, from
Topic 5	the, and, for, from, build, version, adding, commons, this
Topic 6	cvs, this, the, then, from, here, and, has, line
Topic 7	the, for, jaimin, and, add, from, fix, ncole, task
Topic 8	the, for, and, added, tag, build, file, ant, now
Topic 9	add, javadoc, for, update, and, fix, from, the, use
Topic 10	the, and, for, added, fixed, from, test, with, connection, not, and, wizard
Topic 11	aonishuk, for, not, atkach, ababiichuk, onechiporenko, page, make, and
Topic 12	not, the, with, use, was, from, samples, make, and
Topic 13	from, submitted, reviewed, obtained, the, for, added, and, emptymessage
Topic 14	yusaku, via, srimanth, for, the, and, not, should, page
Topic 15	the, added, that, for, and, patch, test, fixed, with
Topic 16	via, for, ambari, swagle, not, the, upgrade, dsen, and
Topic 17	the, dlysnichenko, akovalenko, for, and, from, created, jluniya, moe
Topic 18	added, the, for, fixed, updated, removed, and, javadoc, test

References

1. Allman, E.: Managing technical debt. Commun. ACM **55**(5), 50–55 (2012)
2. Blei, D.M., Lafferty, J.D.: Dynamic topic models. In: Proceedings of the 23rd international conference on Machine learning. pp. 113–120. ACM (2006)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine Learning research **3**(Jan), 993–1022 (2003)
4. D'Ambros, M., Bacchelli, A., Lanza, M.: On the impact of design flaws on software defects. In: 2010 10th International Conference on Quality Software. pp. 23–31. IEEE (2010)
5. Farrell, J.: Corporate funding and ideological polarization about climate change. Proceedings of the National Academy of Sciences **113**(1), 92–97 (2016)
6. Fowler, M.: Refactoring: Improving the design of existing code. In: 11th European Conference. Jyväskylä, Finland (1997)
7. Gilardi, F., Shipan, C.R., Wüest, B.: The diffusion of policy perceptions: Evidence from a structural topic model. University of Zurich and University of Michigan (2015)

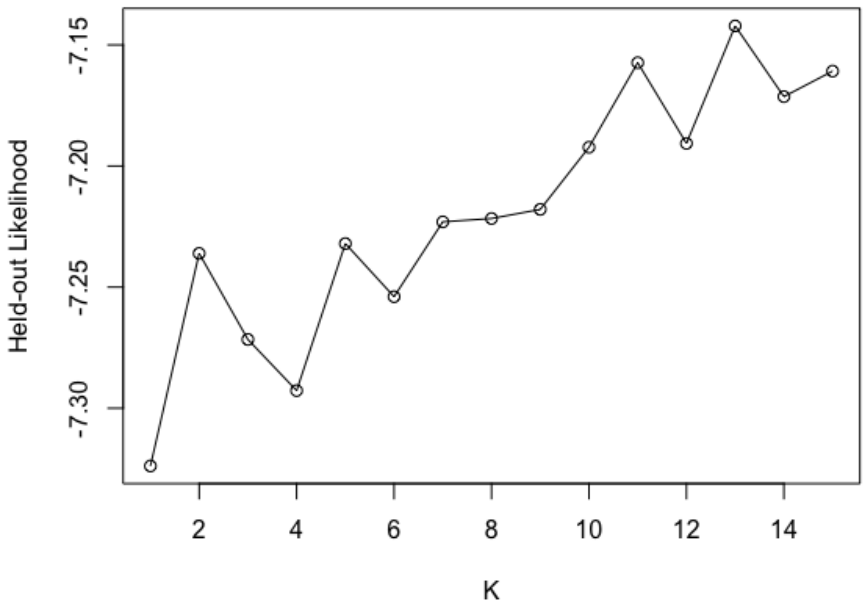

Fig. 1. Held-out Likelihood of Different Topic Numbers.

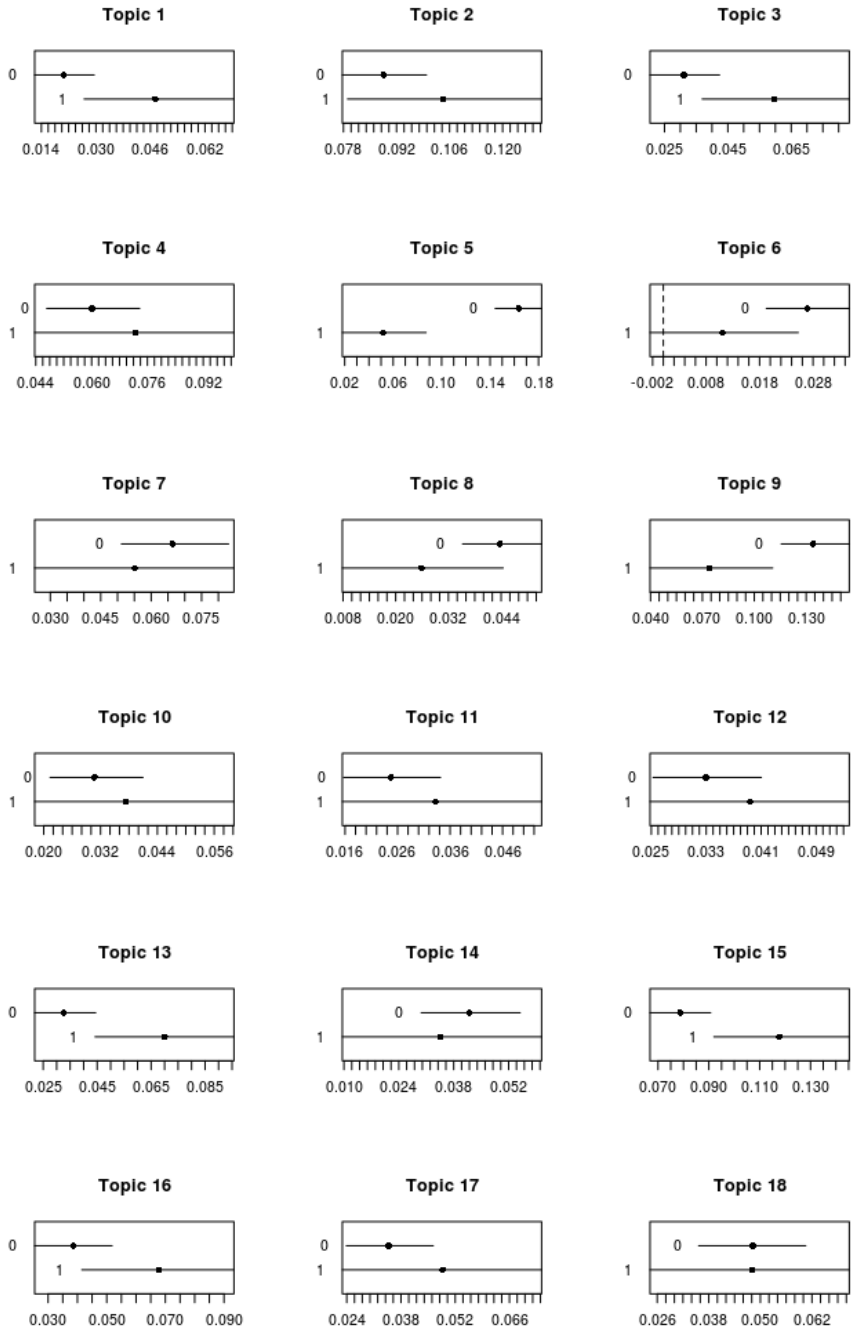

Fig. 2. Held-out Likelihood of Different Topic Numbers. "1" represents debt contributing and "0" represents no debt contributing. The length of bars represents the 95 % confidence interval.

8. Guzman, E., Azócar, D., Li, Y.: Sentiment analysis of commit comments in github: an empirical study. In: Proceedings of the 11th Working Conference on Mining Software Repositories. pp. 352–355. ACM (2014)
9. Hindle, A., Godfrey, M.W., Holt, R.C.: What's hot and what's not: Windowed developer topic analysis. In: 2009 IEEE International Conference on Software Maintenance. pp. 339–348. IEEE (2009)
10. Hu, J., Sun, X., Lo, D., Li, B.: Modeling the evolution of development topics using dynamic topic models. In: 2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER). pp. 3–12. IEEE (2015)
11. Humphrey, W.S.: A discipline for software engineering. Addison-Wesley Longman Publishing Co., Inc. (1995)
12. Islam, M.R., Zibran, M.F.: Sentiment analysis of software bug related commit messages. Network **740**, 740 (2018)
13. Lenarduzzi, V., , Saarimäki, N., Taibi, D.: The technical debt dataset. In: The Fifteenth International Conference on Predictive Models and Data Analytics in Software Engineering (PROMISE19) (Sept 2019). https://doi.org/10.1145/3345629.3345630
14. Lenarduzzi, V., Sillitti, A., Taibi, D.: A survey on code analysis tools for software maintenance prediction. In: International Conference in Software Engineering for Defence Applications. pp. 165–175. Springer (2018)
15. Li, W., Shatnawi, R.: An empirical study of the bad smells and class error probability in the post-release object-oriented system evolution. Journal of systems and software **80**(7), 1120–1128 (2007)
16. Lozano, A., Wermelinger, M.: Assessing the effect of clones on changeability. In: 2008 IEEE International Conference on Software Maintenance. pp. 227–236. IEEE (2008)
17. Mimno, D., Wallach, H.M., Talley, E., Leenders, M., McCallum, A.: Optimizing semantic coherence in topic models. In: Proceedings of the conference on empirical methods in natural language processing. pp. 262–272. Association for Computational Linguistics (2011)
18. Roberts, M.E., Stewart, B.M., Airoldi, E.M.: A model of text for experimentation in the social sciences. Journal of the American Statistical Association **111**(515), 988–1003 (2016)
19. Saarimäki, N., Lenarduzzi, V., Taibi, D.: On the diffuseness of code technical debt in java projects of the apache ecosystem. In: Proceedings of the Second International Conference on Technical Debt. pp. 98–107. IEEE Press (2019)
20. Sinha, V., Lazar, A., Sharif, B.: Analyzing developer sentiment in commit logs. In: Proceedings of the 13th International Conference on Mining Software Repositories. pp. 520–523. ACM (2016)
21. Sjøberg, D.I., Yamashita, A., Anda, B.C., Mockus, A., Dybå, T.: Quantifying the effect of code smells on maintenance effort. IEEE Transactions on Software Engineering **39**(8), 1144–1156 (2012)
22. Van Kleek, M.G., Bernstein, M., Panovich, K., Vargas, G.G., Karger, D.R., Schraefel, M.: Note to self: examining personal information keeping in a lightweight note-taking tool. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 1477–1480. ACM (2009)