Poster: Forks Insight: A Tool Overviewing Forks

Luyao Ren
Peking University
fancycoder@pku.edu.cn

Shurui Zhou
Carnegie Mellon University
shuiruiz@cs.cmu.edu

Christian Kästner
Carnegie Mellon University
kaestner@cs.cmu.edu

Andrzej Wąsowski
IT University of Copenhagen
wasowski@itu.dk

ABSTRACT

(Luyao: TODO) This paper provides a sample of a LaTeX document which conforms, somewhat loosely, to the formatting guidelines for ACM SIG Proceedings. This paper provides a sample of a LaTeX document which conforms, somewhat loosely, to the formatting guidelines for ACM SIG Proceedings. This paper provides a sample of a LaTeX document which conforms, somewhat loosely, to the formatting guidelines for ACM SIG Proceedings. This paper provides a sample of a LaTeX document which conforms, somewhat loosely, to the formatting guidelines for ACM SIG Proceedings.

KEYWORDS

(Luyao: TODO) word, word, word, word, word, word, word, word, word, word, word, word

1 INTRODUCTION

To start development from an existing codebase, developers could fork the repository to take a copy of existing source code, which is common in open source and industry. Developers have the freedom and independence to make modifications on their own fork [2, 3, 5, 10]. However, many forks of a repository not merge their changes back into the original code base which makes it difficult to keep track of decentralized development activity of all the forks when the number of forks grows. (Luyao: this word comes from INFOX paper originally? should I add cite? It can be easily prove by my tool, or should we prove it ?) Some of the developers on Github we have interviewed previously confirmed the problems they faced in terms of losing overview of forks. For example, one said: *"I do not have much visibility of the forks. They are too many and it is overwhelming to keep track of them."* [11] It shows that the demand for an overview of forks is really existed on software development platforms like GitHub. The network view and members view was supported on GitHub, but it is difficult to gain an overview of specific activities in many forks. As another developer said: *"The network view is helpful for seeing how active a fork is, but often you have to scroll back a lot to find the fork point and then you have to go to the end again for seeing what changed since then in the parent and in the fork, by reading the tooltips of each commit. And the Members view is just a big list of projects, only helpful for finding the parent of all forks, and for opening all of them in browser tabs to read the descriptions."* [1] Other related work shows that for fork-based development in industrial contexts it is hard for individual teams to know who is doing what and what code changes are made in other forks [1, 4]. Lack of an overview of forks leads to several problems like redundant development, lost contributions and suboptimal forking point [3, 9]. (Luyao: add more explaination on these problem?)So it's necessary to give developers a panoramic view for the various forks.

We implement Forks Insight[2] which sets out to understand activities in various forks and provide insights to interested developers including repository's maintainers, ac-hoc users who are interested in the repositories and want to see others' secondary development to prevent redundant development. Based on the related work S. Zhou did before [11], our tool is a light-weight release version and has no limit to programming language.

2 FORKS INSIGHT

The user interface of Forks Insight is shown in Figure 1. By comparison of the fork with upstream(Here we only consider the unmerged branch, namely, the code which has already merged into master branch will not be included in our tool)(Luyao: add reason here?), every row in table is on behalf of one fork included its commits, changed files, lines of code changed, representative keywords, last commit time and created time.

Users could login with their own GitHub accounts and load their public repositories(or any other repository they are interested) on GitHub. Our tool will start crawling and analyzing data if the repository is not in our database yet and remind the user when it's finished. (Luyao: delete this line?)

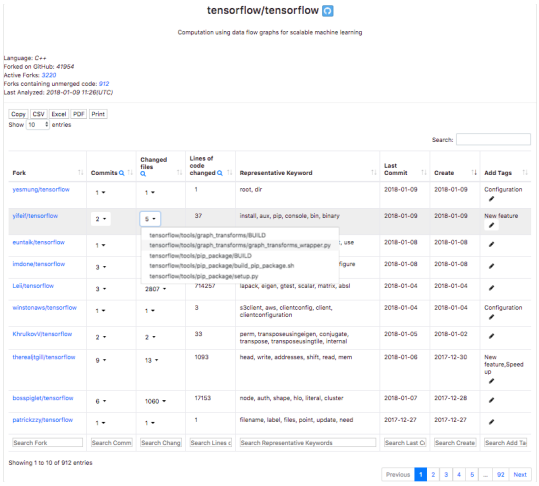

Figure 1: User Interface of Forks Insight.

[1] https://github.com/dear-github/dear-github/issues/175

[2] http://www.forks-insight.com

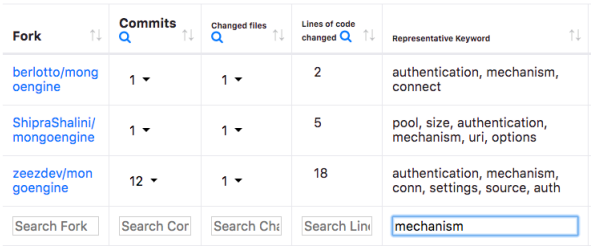

Figure 2: An example of searching for similar forks.

2.1 Keywords Extract

As for the column of representative keywords, we use natural language processing on source code, commit messages we fetched. First, we do some preprocessing: remove all the numeric strings; split for underline-separated and Camel-Case cases; lemmatize words to a normal form(for example, the word "dogs" will turn to "dog"). Then, we use TF-IDF [8] to extract keywords from changed code. Though TF-IDF can effectively filter some stop words like "or", "and", there are still some words with high weigh like "public", "private" which is common in language like Java/C++. To improve our result, we manually added the stop words list for different programming language corresponding to their different characteristic. Fortunately, our method runs fast and has no limit for repository's programming language. Forks Insight can successfully analyze the most popular repositories on Github in hours for the first time.

2.2 Tagging

Developers did different developments after forking for different reasons: add new features, fix bugs, change the configuration and so on [3, 6, 7, 9]. Since tagging is a simple and intuitive way to summary the characteristic of forks and also convenient to cluster similar forks. So we add a column at the end of each row to allow user to annotate the tags manually on forks. We hope user's contribution on tags can not only help themselves maintain and understand each fork which may reduce the redundant development, but also help the whole open source community, especially for the new users who are interested in this repository but not familiar with repositories. The data of tagging will also be useful for future research work.

2.3 Retrieval

With the help of the functions like searching, sort, filter in Forks Insight, user can retrieve the forks to dig out useful information. For example, in Figure 2, searching for "mechanism" in forks of MongoEngine[3], will get three forks which are all related to "mechanism of authentication during connection with database", and contain the same changed file. The example shows that by using keyword search in Forks Insight, user could find out some similar forks which implement or improve the similar thing can effectively (Luyao: or probably?) cut down the possibility of the redundant development.

[3]http://www.forks-insight.com/project/MongoEngine/mongoengine

3 CONCLUSIONS AND FUTURE WORK

We implemented a tool to help developers get an overview of the forks to quickly scan and understand them. The current release version focuses on simple analytics for the high level overview which is lightweight, scalable and practical. In order to improve the usability of Forks Insight, we plan to design a user study. And we would like to add more interactive elements and powerful functions on our tool. There are several directions we are considering to move forward: use more visualization to show the meaningful data; identify features in the fork; use natural language to summarize the fork.

ACKNOWLEDGMENTS

TODO

REFERENCES

[1] Thorsten Berger, Divya Nair, Ralf Rublack, Joanne M Atlee, Krzysztof Czarnecki, and Andrzej Wąsowski. 2014. Three cases of feature-based variability modeling in industry. In *International Conference on Model Driven Engineering Languages and Systems*. Springer, 302–319.

[2] Jürgen Bitzer and Philipp JH Schröder. 2006. The impact of entry and competition by open source software on innovation activity. *The economics of open source software development* (2006), 219–245.

[3] Yael Dubinsky, Julia Rubin, Theodore Berger, Slawomir Duszynski, Matthias Becker, and Krzysztof Czarnecki. 2013. An exploratory study of cloning in industrial software product lines. In *Software Maintenance and Reengineering (CSMR), 2013 17th European Conference on*. IEEE, 25–34.

[4] Anh Nguyen Duc, Audris Mockus, Randy Hackbarth, and John Palframan. 2014. Forking and Coordination in Multi-platform Development: A Case Study. In *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '14)*. ACM, New York, NY, USA, Article 59, 10 pages. https://doi.org/10.1145/2652524.2652546

[5] Neil A Ernst, Steve Easterbrook, and John MyLOPOULOS. 2010. Code forking in open-source software: a requirements perspective. *arXiv preprint arXiv:1004.2889* (2010).

[6] Tommi Mikkonen and Linus Nyman. 2011. To Fork or Not to Fork: Fork Motivations in SourceForge Projects. *Int. J. Open Source Softw. Process.* 3, 3 (July 2011), 1–9. https://doi.org/10.4018/jossp.2011070101

[7] Gregorio Robles and Jesús M. González-Barahona. 2012. A Comprehensive Study of Software Forks: Dates, Reasons and Outcomes. In *Open Source Systems: Long-Term Sustainability - 8th IFIP WG 2.13 International Conference, OSS 2012, Hammamet, Tunisia, September 10-13, 2012. Proceedings*. 1–14. https://doi.org/10.1007/978-3-642-33442-9_1

[8] Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management* 24, 5 (1988), 513–523.

[9] Ştefan Stănciulescu, Sandro Schulze, and Andrzej Wąsowski. 2015. Forked and Integrated Variants in an Open-Source Firmware Project. In *31st International Conference on Software Maintenance and Evolution (ICSME'15)*.

[10] Greg R Vetter. 2007. Open Source Licensing and Scattering Opportunism in Software Standards. *BCL Rev.* 48 (2007), 225.

[11] Shurui Zhou, Ştefan Stănciulescu, Olaf Leßenich, Yingfei Xiong, Andrzej Wąsowski, and Christian Kästner. 2018. Identifying Features in Forks. In *Proceedings of the 40th International Conference on Software Engineering (ICSE)*. ACM Press, New York, NY.