Poster: Forks Insight: Providing an Overview of GitHub Forks

Luyao Ren
Peking University

Shurui Zhou
Carnegie Mellon University

Christian Kästner
Carnegie Mellon University

ABSTRACT

Fork-based development allows developers to start development from existing software repository by copying the code files. However, when the number of forks grows, contributions are not always visible to others, unless an explicit merge-back attempt is made. To solve this problem, we implemented Forks Insight (*www.forks-insight.com*) to help developers get an overview of forks on GitHub. The current release version focuses on simple analytics for the high level overview which is lightweight, scalable and practical. It has a user-friendly interactive web interface with features like searching and tagging.

KEYWORDS

Fork-based development, Open-source, Overview of forks, GitHub

ACM Reference Format:
Luyao Ren, Shurui Zhou, and Christian Kästner. 2018. Poster: Forks Insight: Providing an Overview of GitHub Forks. In *ICSE '18 Companion: 40th International Conference on Software Engineering Companion, May 27-June 3, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3183440.3195085

1 INTRODUCTION

Fork-based development allows developers to start development from existing software repository by copying the code files, and gives developers the freedom and independence to make modifications on their own fork [2, 3, 5, 10]. Even though it has been widely used in both open source communities and industry, when the number of forks grows, it becomes difficult for developers to keep track of decentralized development activities in many forks. Some of the developers on GitHub that we have interviewed previously reported 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]. Both Duc et al. and Berger et al. found that this problem also appears in industry. It is hard for individual teams to know who is doing what and what code changes are made in other forks [1, 4].

Even though GitHub supports a network view to visualize the commit history across all branches and forks of a repository, it is difficult to gain a straightforward overview of specific activities in 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"*.[1] A lack of an overview of forks leads to several additional problems: (1) redundant development: developers may re-implement a functionality that has already been developed elsewhere; (2) lost contributions: the contributions developers made are easily lost to the larger community unless they contribute those changes back to the original repository; (3) suboptimal forking point: developers might not fork from the codebase that is closest to their intended goals [3, 9, 11]. We therefore argue that it's necessary to give developers a panoramic view to help them better understand activities among various forks.

Forks Insight (http://www.forks-insight.com) is a more lightweight and accessible solution of our prior academic prototype INFOX [11], which identifies cohesive code changes from non-merged code within forks for C/C++ projects. Forks Insight offers a web service for all GitHub repositories. It provides an overview of each repository in fork-level granularity and delivers insights to developers who are interested in the repositories.

2 FORKS INSIGHT

Forks Insight provides facilities to explore unintegrated changes to find opportunities for reuse, to find inspirations for further development, to potentially connect developers working on similar topics. It analyzes each active fork of a repository by taking the diff between the latest commit of both the upstream and the fork to get the commits that only exist in forks, and extracting keywords from corresponding code changes, comments and commit messages. Besides, Forks Insight presents statistical data of unintegrated changes at different granularities, such as commits, changed files, lines of code. The user interface of Forks Insight is shown in Fig. 1. Users could log in with their GitHub accounts, and subscribe repositories on GitHub that they are interested in. Fork Insight supports importing repositories from user's public repositories list or searching by a repository url.

2.1 Extracting Keywords

To give developers a quick summary of what code changes have been made in each fork, we extract a list of representative keywords from text that are related to the code changes, such as source code, comments, and corresponding commit messages, by using the well-known natural language processing technique TF-IDF [8]. Specifically, we first preprocess the text: removing all the numeric strings; splitting word into subtokens for *Pascal Case* and *Camel Case*; lemmatizing words into a normal form. Then, we extract keywords from the text by calculating TF-IDF weight of each token. Though TF-IDF could effectively filter out some stop words like "or", "and", there are still some words with high weight like "public", "private" that are meaningless for code summary. To improve the result, we manually add a list of reserved words for different programming languages as stop words.

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

ICSE '18 Companion, May 27-June 3, 2018, Gothenburg, Sweden

ACM ISBN 978-1-4503-5663-3/18/05.
https://doi.org/10.1145/3183440.3195085

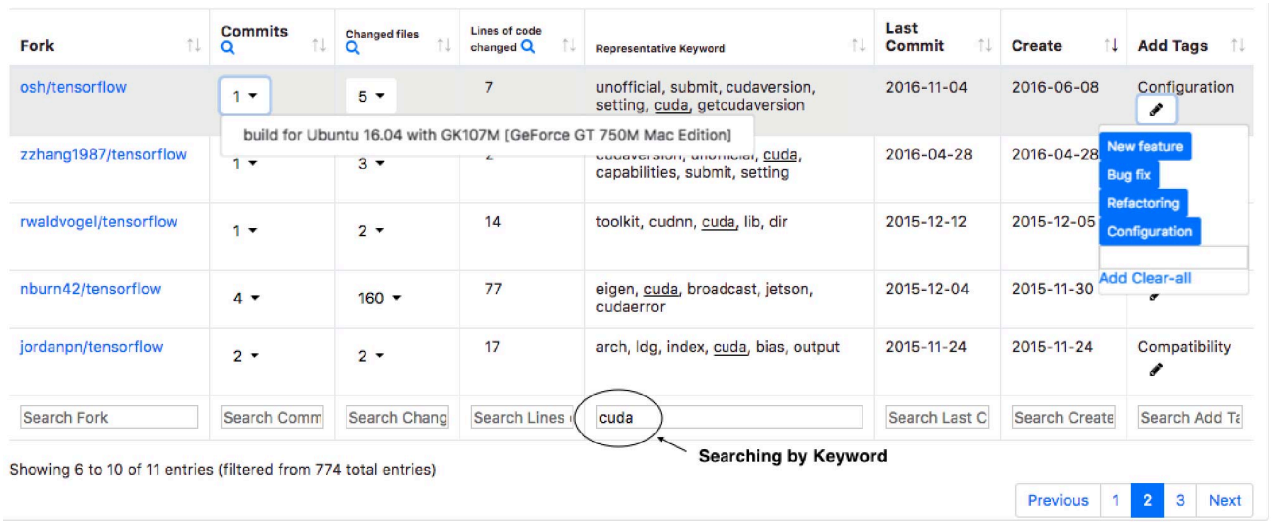

Figure 1: User Interface of Forks Insight. This example shows searching "cuda" in repository of *tensorflow/tensorflow*.

2.2 Tagging Forks

Developers fork a repository for different reasons: adding new features, fixing bugs, and changing configuration, etc. [3, 6, 7, 9]. This information could help developers quickly find specific forks they want to explore. Thus, Forks Insight allows users to tag each fork based on their understanding of the main activity (see the last column in Fig. 1). We hope user's input on tags could not only help themselves maintain and understand each fork, but also help the whole community, especially for the new users who are not familiar with this repository to get a better overview.

2.3 Searching Related Forks

The interviews we did for INFOX show that the problem of redundant development exists. For example, a developer found another fork implemented a very simple one-function as they did several years ago, and he said: *"I think they should use our code"*. Another developer said: *"I can see multiple forks are working on the similar problem. This one looks like it is adding [...] that I already added"* [11].

To identify redundant development, we design the functionality of keywords searching, which helps to find code changes that contain the same keyword. We expect to find forks that are working on the similar topics, and build the connection between the developers, which could hopefully reduce the redundant development. For example, in Fig. 1, if developers search for "cuda", which is a common keyword related to GPU configuration in *tensorflow/tensorflow*, Forks Insight will return several forks whose keywords contain "cuda".

3 CONCLUSIONS AND FUTURE WORK

We implemented Forks Insight to help developers get an overview of forks. The current version focuses on simple analytics for the high level overview. It uses the keyword extraction of INFOX and extends it with a user-friendly interactive web interface and features for searching and tagging. In order to improve the usability of Forks Insight, we plan to ask for feedback from open source developers. And we would like to add more interactive elements and powerful functions into our tool. There are several directions we are considering to move forward: using more visualization to show meaningful data; identifying features in forks; summarizing the activities of forks by natural language.

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.

[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.

[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.

[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.