Update Frequency and Language Popularity in Open-Source: Analysis of Commit Activity in GitHub Projects

Satrio Husodo, Michael Zhang,
Jane Huffman Hayes

University of Kentucky, Lexington,
KY 40506, USA
satrio.husodo@uky.edu,
michael.zhang@uky.edu,
hayes@cs.uky.edu

ABSTRACT

Open source projects have become a staple of empirical validation for software engineering techniques and hypotheses. To better characterize such resources and learn more about the developer ecosystem, this study analyzes the activity of open-source software projects on GitHub. Trends and patterns in repository commit activity and their potential relationships with language use are examined on over 5,000 randomly selected repositories. Not so expectedly, JavaScript dominated among all sampled repositories. The study finds that almost a third of the repositories had less than 50 commits, and the majority had less than 100. In an analysis of commit activity over time, we found that approximately one-fifth of the repositories had less than two active months (months with > 1 commit). Moreover, the language of a repository is associated with commit activity levels, and repositories featuring Unix scripting and markup languages tend to result in lower commit activity levels.

CCS CONCEPTS

• **Software and its engineering** → Software notations and tools → Software libraries and repositories; **Cross-computing tools and techniques** → empirical studies

KEYWORDS

GitHub, repositories, version control, programming languages, commit activity, open-source, empirical research

1 INTRODUCTION

GitHub is an online version control platform where developers can collaborate with one another via pull requests, issue tracking, and wiki pages [5]. Developers can push code changes, or commits, that become a permanent part of the project's history. As of 2017, there are nearly 67 million repositories hosted on GitHub ranging from large open-source to small personal software projects [8] with varying levels of activity. Large, important projects with many contributors may have steady, frequent activity over time, while smaller projects may have small bursts of activity or low activity overall. GitHub may also be used for archival purposes.

Researchers routinely validate hypotheses and techniques on open-source projects; to support external validity, subject projects should be representative. Toward that end, we take advantage of the public data from GitHub to study project characteristics and developer behavior on a large scale by looking at different projects and their history. We aim to understand an important aspect of developer behavior: the frequency with which they update their projects. We detail such differences quantitatively by examining project activity as measured by commit count. We observe potential relationships between commit activity and primary language choice as such correlations may reveal which languages tend to result in repositories that are very active with many commits spanning a long period of time. The study provides insights into the productivity of developers both in the open-source setting and in their publically hosted personal projects. As GitHub is one of the most widely-used platforms, this study helps describe the nature of work and projects of a large ecosystem of developers.

2 RESEARCH QUESTIONS

The research questions focus on the relationship between commit activity and other repository properties.

RQ1: What is the distribution of commit counts among the repositories? Do most repositories have low, medium, or high activity?

We predict, based on anecdotal evidence, that most repositories will have low activity, measured using commit count.

RQ2: Can the repositories be grouped based on activity profile?
We examine commit activity, using total commits and total active months. We predict that repositories will cluster in distinct groups; most repositories will likely belong in the low activity group.

RQ3: What are the most popular languages in use from the repositories? Are there correlations between commit activity level and language choice?
For the former question, we predict that the most popular languages found in our sampled GitHub repositories will closely resemble the rankings published in a study that measured language popularity [1]. For the latter question, the null hypothesis (H_0) is that there is no association between commit activity level and language choice; the alternative hypothesis (H_A) is that there is an association. This is based on anecdotal observations that some languages have frameworks/libraries that quickly rise in popularity then decline.

3 METHODOLOGY

We randomly sampled 100,000 public repositories that were not forks of other repositories using a Python script that made calls to the GitHub API [6]. For each of these repositories, we were able to retrieve the full commit history as well as the distribution of programming languages included in the repository in bytes. However, many repositories that we randomly sampled included repositories that were very young (less than one year since repository creation), insignificant repositories that include "Hello World" and other toy programs, and repositories that are one of a kind with an extremely high number of commits (such as the Linux kernel which has over 600,000 commits [7]). Such repositories are not entirely representative of meaningful open-source software. To exclude those repositories, we filtered the 100,000 randomly sampled repositories using the following rules:

- Repositories must be at least one year old
- Repositories must have between 30 and 5,000 total commits

After applying the filter to the 100,000 initially sampled repositories, we obtained a total of 5,298 repositories to use in our analysis. While obtaining the data in a single uninterrupted step would be ideal, we had to randomly sample the repositories first and then filter in order to comply with the 5,000 requests per hour limit imposed by GitHub API [6].

The commit activity profile of a GitHub repository can be complex. Some projects may have bursts of activity over time or constant low/high level activity. One way to evaluate project activity in addition to looking at total commit count is to observe the time frames in which commits are made. For RQ2, we define a time frame here to be months, and an active month is one that has at least one commit.

For RQ3, we define popular languages as those that appeared frequently as the primary language of repositories in our sample space. In addition, we define commit activity as commits per day - total commits divided by days elapsed from first to last commit timestamps. This was computed for each of the 4,736 repositories featuring a primary language found with one of the 15 most popular languages. Activity levels were assigned to qualifying repositories depending on the percentile ranking of the commits per day metric: *low activity* have commits per day less than or equal to the 33rd percentile, *medium activity* repositories have commits per day strictly greater than the 33rd percentile and less than or equal to the 66th percentile, and *high activity* repositories have commits per day strictly greater than the 66th percentile.

4 RESULTS AND DISCUSSION

This section presents the results for each research question.

4.1 RQ1

Fig. 1 contains two histograms (one is an enlarged view of the other). In the smaller histogram, the bins have the size of 1,000 commits. Not surprisingly, most repositories (5,128 out of 5,298) belong in the lowest bin of 0 to 1,000 commits. The trend decreases except at the highest two bins, where there are more repositories with 4,000 to 5,000 commits than 3,000 to 4,000. We expect that the trend continues to flat line, but this could be due to the noise or variability from the sampling because at these bins, there are as few as 40 repositories. If one were to gather data from all GitHub repositories, it is possible that "unicorn" projects, or those with an extraordinarily high number of commits, would be more evident, but our maximum commit limit masks this observation.

Since most of the repositories belong in the lowest bin, the larger histogram shows a more detailed view of these repositories. Again, low activity repositories dominate, with 1,731 repositories (32.7% of the total) having fewer than 50 commits. Our observation underestimates the number of repositories that have low activity because our selection criteria filters out those with < 30 commits. If these repositories were not filtered out, they would have made up an even higher proportion of the sample repositories.

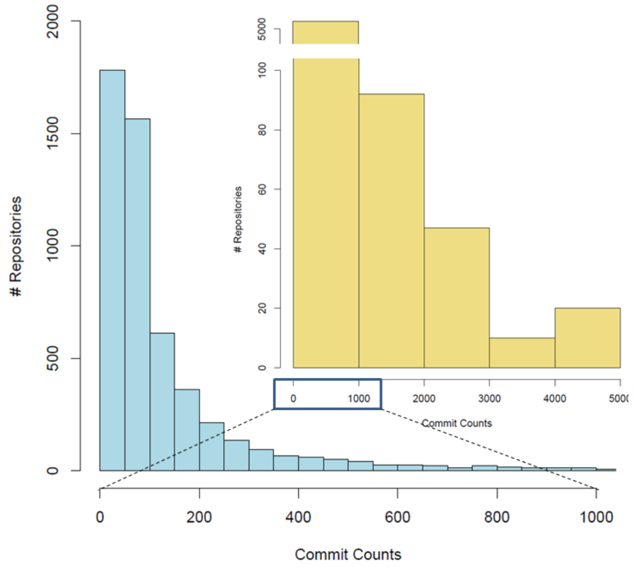

Figure 1: Histogram of commit counts. The commit counts of the sampled repositories are displayed in a histogram with bins of size 1,000. 5,128/5,298 repositories fall have < 1,000 commits. To show a more detailed view of the majority of repositories, the commit counts of the sampled repositories with lower than 1,000 commits are displayed in a histogram with commit count bins of size 50.

4.2 RQ2

To see the distribution of active months for RQ2, we plotted the number of active months versus the total number of commits (Fig. 2). As expected, the more total commits a project has, the more active months it will have. Linear regression of the scatter plot reveals a weak positive correlation (slope = 0.022, $R^2 = 0.425$). The clustering in the lower left corner are repositories consisting of low total commits and few active months. For the most active projects (> 2,500 commits), the positive correlation is even weaker (slope = 0.006, $R^2 = 0.019$), indicating that as projects get large, the rate of commits over time does not necessarily scale. This makes sense in that large projects may hit a saturation point in the number of new significant contributors.

4.3 RQ3

JavaScript is the most popular language being the primary language of nearly 20% of our 5,298 sampled repositories. Java, Python, and Ruby follow in popularity, and these four most popular languages are featured as primary languages in more than half of the repositories in our sample (Fig 3). Our language popularity rankings do not match exactly with the rankings from another study claiming that C, JavaScript, C++, and PHP were the four most popular languages [1]. However, the languages listed were close. It seems the typical or "medium-sized" OSS in GitHub may be JavaScript, not Java or C which are often empirically studied.

To examine repository commit activity levels and language choice, we plotted the distribution of low, medium, and high activity repositories for each of the 15 most popular languages in Fig. 4. Using Pearson's chi-squared test, we obtained a highly significant difference between the observed and expected values ($p < 0.001$). Thus, we reject H_0 which states that there is no association between commit activity level and choice of language, and we accept H_A. We also observed that the use of Unix scripting languages, such as Perl and VimL, tended to result in mostly low and medium activity repositories. Intuitively, such scripts are usually only changed to accommodate updates on dependencies or to support new releases of software. Interestingly, HTML also exhibits this phenomenon which could be due to repositories hosting code for static GitHub Pages websites.

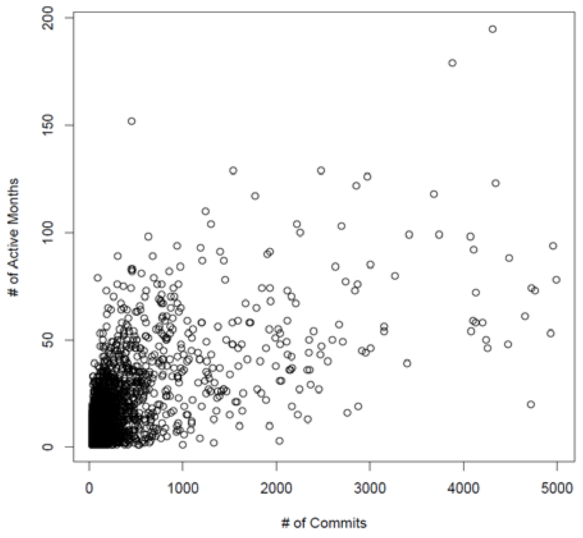

Figure 2: Scatter plot of active months vs. total commits. Each circle represents a repository. The number of active months is obtained by looking at the commit timestamps; if a month has at least one commit in the repository, it is deemed an active month.

5 THREATS TO VALIDITY

There are threats to validity of our study. A threat to external validity of this study is that our sample may not be representative of all GitHub projects. GitHub has almost 67 million repositories [8]. Our initial sample of 100,000 and filtered sample of 5,298 are nowhere near that amount. However, the language distribution that we found follows fairly closely with the findings from prior work [1]. Another threat to external validity is that public GitHub repositories are not representative of all software projects as many are closed-source and/or have their code-base stored elsewhere.

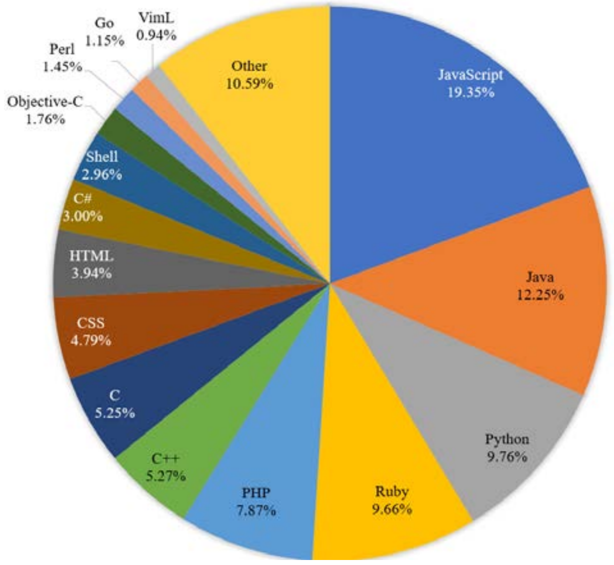

Figure 3: Overall language choice. The proportions of all languages used by the repositories are shown. Only the most-used language is counted.

There are many ways to gauge repository activity, and our method only looks at two ways to do this; this poses a threat to internal validity. Other methods include examining pull requests and changes in terms of lines of code. Due to the potential size of the data, we would then need to limit the sample size in order to analyze the data in a timely manner.

Another limitation with the activity analysis is in the use of commit totals and active months to represent activity profile because they are broad. An alternative is to conduct clustering analyses based on the shapes of the commit graphs. All of these methods have the caveat that the projects may vary greatly in size and lifetime, possibly rendering comparisons inaccurate.

Another threat to internal validity is in our language analysis. We looked at the most-used language per repository, which might not fairly represent repositories that use multiple languages. GitHub provides a distribution of all languages used in terms of bytes for each repository. In the future, we will examine this information.

There is also a threat to replication validity as we randomly sampled repositories. Researchers wishing to replicate our work would obtain a different sampling of repositories and therefore possibly obtain different results. However, Cosentino et al. performed a meta-study on research using data from GitHub and found that many of the studies do not have rigorous sampling methods to ensure that the samples are representative of the population [9].

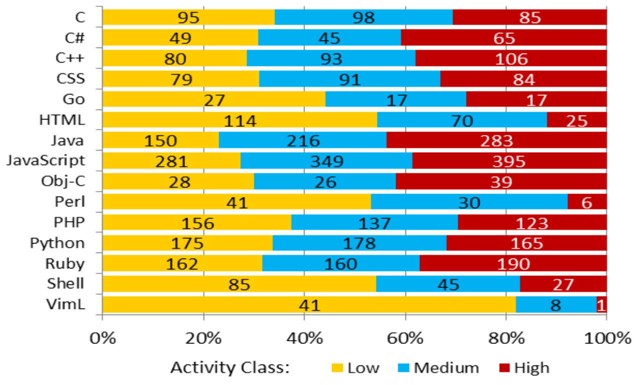

Figure 4: Language choice in low, medium, and high activity repositories. The repositories are divided into classes of activity levels, and the language choice in each class is shown. Numbers within the bars are total repository counts.

6 RELATED WORK

Ours is the first work, to our knowledge, to examine commit frequency from a repository-based perspective as well as associate that with language use, whereas Kolassa et al. examined time between commits from a user-based perspective using data collected on over 5,000 large open-source projects indexed by Ohloh [2]. They discovered that committers usually have a few commits back-to-back within hours followed by a longer period of inactivity. This differs from our study in that we analyzed commit activity in both small and large GitHub repositories across all repository contributors. Another difference is the group's use of commit intervals, or the time between commits, to indirectly measure activity. This is similar to, but an inverse of, our approach; to measure activity by commit interval is to look at the idle time between commits, while our approach does not look at this time range and instead looks at the non-idle or active time.

In another paper, the same research group, Kolassa et al., examined commit size distributions using the same dataset, concluding that commits that are smaller in size are more prevalent than large commits and that commit frequency can be modeled by a Pareto Distribution [3]. The biggest difference between this study and ours is that Kolassa et al. examined the commit sizes as measured by lines of code. We did not examine the commit size and treated all commits to be of the same unit size. Despite this difference, our findings are similar to their results in that we observed a distribution of commit counts in which the commit totals quickly level off, reflecting the predominance of small projects in the sample.

Lin et al. analyzed the commit history of four large open-source projects and uncovered five distinct zones of commit activity through clustering analyses [4]. The group also found a power law distribution for commit sizes, which supports the findings by Kolassa et al. [3]. The study design by Lin et al. contrasts with ours in their focus on five specific projects [4]. In addition, these projects are larger than the projects for which we filtered, ranging from 8,000 to 20,000 in commit totals. We argue that the contrasting approaches in looking at a few projects versus a large swath of projects are both necessary. A bird's eye approach like ours allows for possible observations of general trends, while a focused approach is akin to a case study where researchers can understand project-specific characteristics that may be missed by using the general approach.

Although there is a lack of studies looking at commit activity in GitHub projects, there are many that have mined GitHub repositories to understand other aspects of open-source software development. In 2016, a meta-study by Cosentino et al. found 243 studies published after 2010 that focus on GitHub repositories [9]. They found that 39.78% of the studies use the GitHub API, the tool we also used. However, none of the selected works investigated commit activity, so this topic is still nascent and provides emerging opportunities to characterize developer activity and project characteristics for use as empirical validation subjects.

7 CONCLUSIONS AND FUTURE WORK

Our analyses provide a snapshot of commit activity and language usage in a large sampling of open-source repositories. JavaScript was the most frequently used language, though a prior study showed that to be C [1]. Prior to the study, we posited an abundance of low activity projects; this study provides quantitative support. This abundance indicates that GitHub is a low-level barrier for people who want to host personal projects. Although there was not a plethora of high activity repositories, the occurrences were by no means insignificant. These projects show orders of magnitude higher commit activity than personal projects, evidenced by the highly collaborative nature of open-source projects. Overall, GitHub is an effective hosting platform, and the large number of projects allows researchers to study developer behavior and perform project comparison at scale. In the future, we will collect a larger sample of repositories. We will include all languages used in a project as well as incorporate code size. Future work will also examine why repositories featuring Unix scripting and markup languages yield low commit activity levels.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their insightful feedback.

REFERENCES

[1] T. Bissyande, F. Thung, D. Lo, L. Jiang and L. Reveillere, "Popularity, Interoperability, and Impact of Programming Languages in 100,000 Open Source Projects", 2013 IEEE 37th Annual Computer Software and Applications Conference, 2013.

[2] C. Kolassa, D. Riehle and M. Salim, "The empirical commit frequency distribution of open source projects", Proceedings of the 9th International Symposium on Open Collaboration - WikiSym '13, 2013.

[3] C. Kolassa, D. Riehle and M. Salim, "A model of the commit size distribution of open source", LSOFSEM 2013: Theory and Practice of Computer Science, 2013.

[4] S. Lin, Y. Ma and J. Chen, "Empirical evidence on developer's commit activity for open-source software projects", SEKE, 2013.

[5] "Features For Collaborative Coding - Developers Work Better, Together", GitHub. [Online]. Available: https://github.com/features. [Accessed: 20-Apr-2017].

[6] "GitHub API v3", GitHub Developer, 2017. [Online]. Available: https://developer.github.com/v3/. [Accessed: 20-Apr-2017].

[7] L. Torvalds, "Linux kernel source tree", GitHub Repository. [Online]. Available: https://github.com/torvalds/linux. [Accessed: 20-Apr-2017].

[8] The state of the Octoverse 2017" [Online]. Available: https://octoverse.github.com

[9] V. Cosentino, J. Luis and J. Cabot, "Findings from GitHub: methods, datasets and limitations", Proceedings of the 13th International Workshop on Mining Software Repositories - MSR '16, 2016.