"This Is Damn Slick!" Estimating the Impact of Tweets on Open Source Project Popularity and New Contributors

Hongbo Fang, Hemank Lamba, James Herbsleb, Bogdan Vasilescu
Carnegie Mellon University, USA
{hongbofa,hlamba,jdh,bogdanv}@cs.cmu.edu

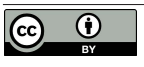
ABSTRACT

Twitter is widely used by software developers. But how effective are tweets at promoting open source projects? How could one use Twitter to increase a project's popularity or attract new contributors? In this paper we report on a mixed-methods empirical study of 44,544 tweets containing links to 2,370 open-source GitHub repositories, looking for evidence of causal effects of these tweets on the projects attracting new GitHub stars and contributors, as well as characterizing the high-impact tweets, the people likely being attracted by them, and how they differ from contributors attracted otherwise. Among others, we find that tweets have a statistically significant and practically sizable effect on obtaining new stars and a small average effect on attracting new contributors. The popularity, content of the tweet, as well as the identity of tweet authors all affect the scale of the attraction effect. In addition, our qualitative analysis suggests that forming an active Twitter community for an open source project plays an important role in attracting new committers via tweets. We also report that developers who are new to GitHub or have a long history of Twitter usage but few tweets posted are most likely to be attracted as contributors to the repositories mentioned by tweets. Our work contributes to the literature on open source sustainability.

ACM Reference Format:
Hongbo Fang, Hemank Lamba, James Herbsleb, Bogdan Vasilescu. 2022. "This Is Damn Slick!" Estimating the Impact of Tweets on Open Source Project Popularity and New Contributors. In *44th International Conference on Software Engineering (ICSE '22), May 21–29, 2022, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3510003.3510121

1 INTRODUCTION

In open-source software (OSS) development, attention can be a double edged sword. Sometimes, OSS projects receive too much attention, and maintainers have to deal with overwhelming volumes of requests and demands from users [25]; in these cases, maintainers might rather fend off new attention coming their way. Other times, even successful OSS projects are unable to attract more than a few contributors, and occasionally OSS projects are maintained by no one at all [3, 17]; in these cases, more sustained involvement from users and contributors would be welcome. Yet, for many OSS projects, gaining attention from the community, e.g., to increase adoption and attract more contributors, remains a challenge.



ICSE '22, May 21–29, 2022, Pittsburgh, P A, USA

ACM ISBN 978-1-4503-9221-1/22/05.
https://doi.org/10.1145/3510003.3510121

Several mechanisms through which OSS projects can gain attention [11, 40, 68] and attract new contributors [9, 40, 53] have been studied in the past. The literature is especially rich in recent years, in the context of social coding platforms like GitHub, because of the high level of transparency and many opportunities for project maintainers to *signal*, explicitly and implicitly, about their work [20]. For example, prior studies of OSS projects hosted on GitHub have found that how projects organize their repository homepages and README files [53], whether projects get *featured* by the hosting platform [40], whether projects have public releases [10], and how maintainers use prominent repository badges to indicate less observable project qualities [68], all have an impact on how the project is perceived by its audience and even the actions that some audience members take, e.g., joining the project.

However, prior work has, by and large, focused only on *endogenous* or "in-network" attention eliciting mechanisms, i.e., taking actions or displaying signals afforded by the code hosting platform itself. This leaves an important gap—little is known about attention eliciting mechanisms that can be considered *exogenous* from the perspective of OSS projects hosted on GitHub or similar platforms. Here we focus on one such mechanism, social media. Social media platforms, widely used by software developers [65], enable OSS maintainers to share their work with a potentially larger audience, that exists beyond their immediate connections on any code hosting platform; e.g., social media posts about an OSS project may be amplified by the authors' social networks, influential social media users, or the platform itself. Social media platforms also tend to have low barrier to participation and high viewership, which makes them actionable and potentially impactful for OSS maintainers, admirers, and evangelists looking to attract attention to projects in need. A better understanding of the effectiveness of using social media to attract attention to OSS projects could directly impact the projects' success and sustainability.

Yet, little is known about how much social media activity can contribute to OSS sustainability, if at all. The evidence from other contexts suggests that actions taken on social media platforms can have spillover, out-of-network effects; e.g., researchers have found that tweets can predict movie ratings [48] and increase citations to academic papers [39]. Can similar effects be expected for OSS?

To address this gap, in this paper, we compile a large dataset of 44,544 tweets containing links to open source GitHub repositories,[1] spanning 6 months of history, and with cross-links between user profiles on both platforms. We then apply statistical causal inference techniques to: (a) estimate the causal effect of tweets on the number of new GitHub repository stars and new committers; (b) characterize the tweets with the highest impact; and (c) characterize the OSS contributors attracted by these tweets.

[1] The title quote was part of one such tweet; see O5 in Figure 4 in the Appendix.

Among other results, we find that:

- Tweets have a statistically significant and sizable effect on attracting new *stars* to OSS GitHub projects, estimated at around 7% increase in stars on average for every set of tweets mentioning a repository around the same time.
- The effect of tweets on attracting new *committers* is present but small, around 2% more commit authors on average.
- The effect of tweets on attracting both stars and committers is moderated by multiple factors, including tweet purpose, size of tweet 'burst' (number of tweets mentioning the same repository around the same time), and tweet author affiliation with the OSS projects.
- Newly attracted contributors tend to be more active in the OSS GitHub projects when they have had prior Twitter interaction with the tweet authors.

Our replication package is online DOI 10.5281/zenodo.6321448 [27].

2 RELATED WORK

Like all software, OSS also needs a steady supply of development and maintenance effort to remain relevant, of high quality, and secure. In community-driven OSS projects, this effort comes largely from volunteers [28, 60]. And even though OSS as a whole plays important infrastructure roles in our digital economy [24], OSS maintainers' ability to attract, onboard, and retain contributors has generally not kept up with this success. For example, prior work describes how many popular OSS projects are maintained by only one or two developers [2, 4, 19, 72], how project newcomers face a swath of barriers that hinder their first contributions [63, 64], and how many of these newcomer barriers are accentuated by gender [42], which further reduces the available contributor pool. Researchers have also found that high turnover in OSS projects can have negative effects, including knowledge loss [56], longer time to fix issues [29], and decreased software quality [30]. More generally, researchers have studied the internal and external factors that contribute to the OSS projects' risk of becoming 'dormant,' 'inactive,' 'unmaintained,' or 'abandoned' [3, 17–19, 36, 70].

Although sometimes OSS maintainers are overwhelmed with the high volume of requests and demands on their time from users and contributors [25, 55], increased OSS project popularity is generally associated with desirable outcomes [46]. For example, prior work found that popular OSS projects are perceived as having higher quality and better community support [5, 20], tend to be more attractive to new contributors [10, 31, 53], and tend to be more successful at fundraising [49]. That is, they tend to be more sustainable.

Besides the intrinsic quality of the projects or the reputation and influence of their maintainers [9], which can affect project popularity and attractiveness to new contributors, various interventions have also been attempted. Some, like the *signals* providing transparency into otherwise less observable attributes, are relatively subtle, or implicit. Yet they can be effective and are abundant, with many instances being a standard part of the platform UI on social coding platforms like GitHub, or being customizable by project maintainers. For example, prior work has found that npm packages displaying quality assurance badges on their GitHub READMEs tend to be downloaded more than packages without badges [68]; moreover, adding badges to READMEs seems to encourage projects to update their dependencies [44, 68]. Similarly, the daily activity streak counters that used to be part of the GitHub user profile page UI seemed to steer users towards long, uninterrupted streaks of activity, including arguably unhealthy activity on the weekends, as a recent natural experiment has shown [45]. More generally, prior work has found that developers make rich inferences about each other and the quality of their work based on the signals available on individual profile and repository homepages on the GitHub platform [20, 41] and respond to nudges based on such signals [13, 53].

Other interventions are explicit. For example, GitHub uses an algorithm[2] to identify *trending repositories* for the day/week/month based on their recent growth in activity and popularity metrics, and features the resulting projects on a dedicated page. This can cause the OSS projects to face an "attention shock" [40] with notable effects, including a surge in new contributors. Within their control, maintainers can also actively promote their projects. Borges and Valente [11] found in a sample of 96 highly popular OSS projects that being mentioned in highly upvoted posts on the Hacker News aggregator site is associated with a statistically significant increase in the number of GitHub stars in the first three days after the publication date on Hacker News compared to the three days before.

More generally, prior work identified Twitter, blogs, in-person meetings and events, and RSS feeds as the most popular promotion channels for OSS [11]. In particular, Twitter is widely used in the software engineering community [8, 66] for a variety of purposes, including learning about new technologies, staying updated about interesting repositories, and OSS project promotion [12, 26, 62]. However, there are no studies quantitatively evaluating the effectiveness of OSS project promotion on social media, with the exception of the Hacker News study [11] above. Yet, there is evidence outside of OSS that activity on Twitter predicts popularity of offline events and other types of online content, including the online news cycle [16], gross earnings of movies [73], and popularity of academic research [37]. This effect has also been observed on other social media platforms besides Twitter, e.g., Reddit [23] and YouTube [57, 59].

3 RESEARCH QUESTIONS

The main goal of this study is to evaluate the effectiveness of tweeting about OSS projects. Focusing on GitHub, the most popular platform for hosting OSS development, we expect that tweets mentioning OSS projects could expand the projects' audience and reach beyond what they already have on GitHub through their watchers [61] or through their maintainers' direct followers [9, 38]. We seek to estimate how much of this extended audience, if any, such tweets are able to attract and convert into stargazers or project contributors, both outcomes which can drive project success and sustainability, as discussed above. We ask:

RQ1. How do tweets mentioning open-source projects impact their popularity and attractiveness to new contributors?

However, social media content, including tweets, are hardly perennial in any user's momentary view of the platform, since they are typically organized as a stream ("news feed"). On Twitter, one's timeline displays a mix of tweets from accounts they follow plus

[2] https://github.blog/2013-08-13-explore-what-is-trending-on-github/

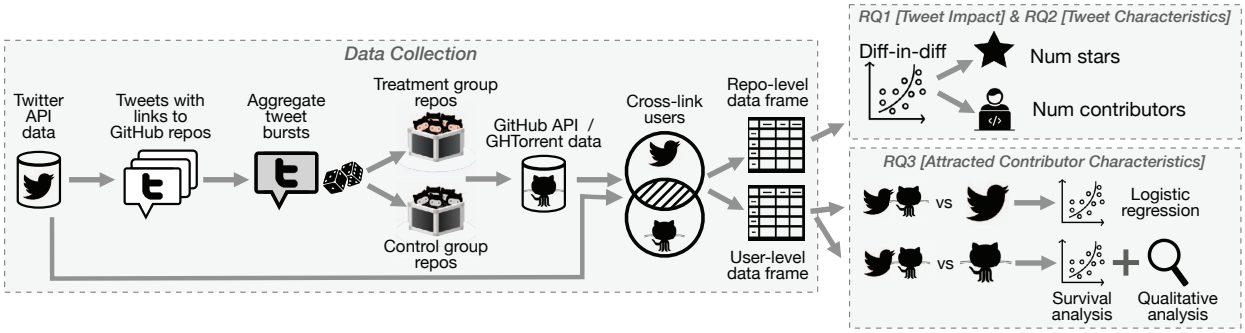


Figure 1: Overview of our data collection and analysis steps.

content suggested by the platform based on a variety of signals, e.g., whether someone one follows has interacted with that content. In addition, one's level of engagement with social media, attention span, and ability to absorb a typically overwhelming volume of content are all very limited. It follows that in order to even stand a chance at being effective, tweets mentioning open source projects should at the very least be noticed. This may depend on many factors, including their content and virality, and their authors' Twitter network span. For instance, popular tweets on Twitter are liked by many and may have stronger promotional effects, and tweets with certain hashtags, of different lengths, and with different types of content may also have different effects. It is important to understand the characteristics of successful tweets, if this OSS project promotional mechanism is to be used effectively. We ask:

RQ2. How does the impact of tweets mentioning open-source projects vary with different tweet characteristics?

Finally, we seek to better understand *who* is being attracted by these tweets. Knowing what type of audience and contributors can be attracted via Twitter is important for developers' decision to tweet or not. Prior social connections are known to impact developer engagement and retention [15, 54, 69] in open source. Twitter offers an additional modality for developers to socialize and form connections. In turn, these connections may help explain developers' engagement with the open source projects they discovered via Twitter, e.g., they may motivate people to contribute more and for a longer period. By cross-linking user accounts across the two platforms (GitHub and Twitter), we can investigate the characteristics of the users who were likely attracted by the tweets mentioning GitHub repositories, both relative to other users on Twitter who were likely exposed to those same tweets, as well as to other GitHub contributors to those same projects. In short, we ask:

RQ3. What are the characteristics of the contributors likely attracted via tweets mentioning open-source projects?

The following section gives an overview of our study design to answer these research questions.

4 STUDY OVERVIEW

We designed and carried out a mixed-methods empirical study, analyzing a novel dataset that integrates data across Twitter and GitHub. Starting from a set of tweets containing links to GitHub repositories, we collect data about two outcomes of interest—project popularity as indicated by the growth in the number of stars and project success in attracting new contributors as indicated by the growth in commit authors. We also collect characteristics of those tweets and their authors, including the tweet authors' ego networks. Finally, we use public information to cross-link user accounts across Twitter and GitHub, and collect additional data about the tweet authors' relationship to the repositories mentioned in their tweets. At a high level (Figure 1), our study consists of two main parts. We give a brief overview here and discuss details below, in Section 5.

*Part 1: Diff-in-Diff Analysis of the Causal Impact of Tweets (**RQ1**, **RQ2**).* In the first part, we *mimic an experimental research design* using the observational data we collected, by modeling the differential effect of an intervention on a 'treatment group' versus a 'control group' in a *natural experiment*.

In a true experiment, the random assignment of subjects to one of the two conditions ('treatment' and 'control'), together with the pre-test manipulation of the independent variable under study, is what enables researchers to make causal claims about the nature of the association between the independent and dependent variables, if present. Our study is observational and, therefore, more limited in its ability to support causal claims, compared to a true experimental design. Consider project popularity, one of our two main outcomes of interest, as an example. While we expect that Twitter mentions may help increase the number of GitHub stars projects receive on average, such an increasing star count trend may have already started *before* the Twitter mentions, and for different reasons. Figure 2 illustrates this point—the number of new GitHub stars received per day seems to start increasing before the project was first mentioned on Twitter on March 9th. One of the tweets mentioning the repository shortly thereafter (O8 in the Appendix) offers a clue as to why, suggesting a possible in-person event where the repository first started being promoted. Therefore, we are not sure if it is the event itself that *caused* the increase in stars, or if those tweets also played a role in the star increase. Similarly, as discussed above, being featured on the GitHub *trending* page or mentioned on platforms like Reddit, Medium, and Hacker News may have also *caused* the observed increase in stars.

To be able to make causal inferences, the key idea behind our design is to compare not the historical changes in outcome measures before and after the intervention among mentioned repositories, but rather the *difference* in these changes between a group of treated (mentioned) repositories and a carefully selected group of untreated

repositories, that acts as a control. The latter is chosen such that the pre-treatment trends in outcomes are similar between the treatment and control groups. That is, the confounding factors before the treatment (e.g., offline in-person event), if present, would have on average affected both treated and control repositories similarly since the pre-treatment trends between the two groups are parallel. Under this assumption, the difference between the observed outcome and the "normal" outcome, i.e., the difference that would still exist if neither group experienced the treatment given the same trend over time in both groups, can be seen as the true effect of the tweets assuming there were no concurrent treatments. In addition, our estimated causal effect is not subject to the influence of any other confounding variables as long as they apply to both treated and control repositories at the similar level. For example, switching accounts across commits will cause a change in the number of new committers but will not affect the estimated treatment effect if we assume developers' tendency to switch accounts is similar in both treated and control repositories. This design is known as *difference-in-differences* (diff-in-diff) estimation [71].

We use the former analysis to address **RQ1**. To address **RQ2**, we include relevant tweet characteristics as predictors in the same diff-in-diff model, which allows us to estimate their effects.

*Part 2: Mixed-Methods Analysis of Who Is Attracted By Tweets (**RQ3**).* In the second part, we report on a mixed-methods qualitative and quantitative case study of a sample of new committers to GitHub projects that were likely attracted by tweets, to better understand when this mechanism can be effective. Quantitatively, we compare developers likely attracted by tweets both to other GitHub project contributors and to others likely exposed to the same tweets but who did not start committing to the GitHub projects. Qualitatively, we analyze instances of past Twitter interaction (e.g., reply to each other's tweets) and GitHub collaboration (e.g., commit to the same repository) to better understand the reasons why those developers may have been attracted.

5 METHODS

Our analyses below are based on a dataset of 2,370 open source GitHub repositories and all the tweets mentioning them, 44,544 in total, over a span of 6 months. We detail all our operationalization and statistical modeling steps next.

5.1 Preprocessing

We start from the convenience sample of 70,427 GitHub users cross-linked with their Twitter accounts, published by Fang et al. [26]. The authors used two heuristics to identify the GitHub users' likely Twitter profiles, reportedly with 85% accuracy: "(1) mining explicit links to Twitter accounts from [all] GitHub user profile pages; (2) crawling personal websites linked from GitHub user profile pages and mining links to Twitter accounts therein" [26]. We then use the Twitter API to mine all these users' tweets, and identify a total of 331,627 tweets among these that contain links to GitHub artifacts (e.g., issue thread, repository homepage).

Next, we apply a series of filters to de-noise this starting dataset, excluding tweets that mention: (i) more than one repository (more ambiguous effects), (ii) forks rather than main repositories (confounded repository activity metrics), (iii) repositories not recorded

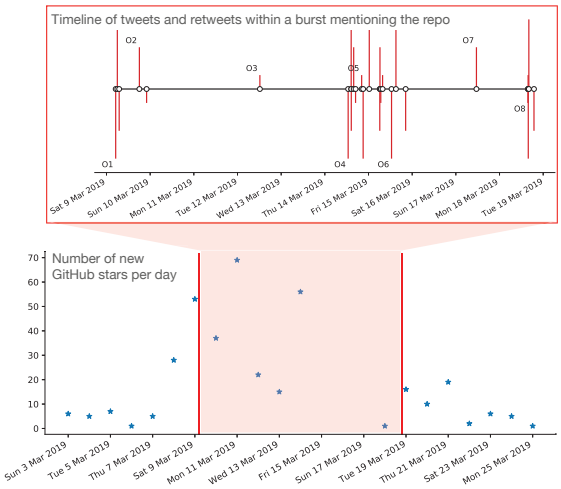


Figure 2: The number of new GitHub stars attracted by the `axa-group/nlp.js` repository per day before, during, and after the Twitter burst in the Appendix. The inset shows the timeline of tweets (labeled) and retweets (unlabeled).

in GHTorrent [34] (one of our main sources of data), (iv) repositories with multiple entries in GHTorrent (unclear which entry to choose), (v) repositories with a later recorded creation time in GHTorrent than the tweet itself, (vi) repositories deleted from GitHub (or made private) since the tweet, (vii) repositories without explicit open source licenses.[3]

We then limit our sample to tweets posted between November 1st 2018 and April 30th 2019 because: a) the copy of GHTorrent we had access to ended in May 2019; and b) the Twitter API limits the number of tweets we can obtain from any single user, therefore tweets from highly active users posted further back in the past are less likely to be retrievable (this might bias the sample towards an over-representation of tweets from less active users). The chosen period, six months, is long enough to yield a large dataset for analysis: after this and all of the above filters, we were left with 10,837 tweets mentioning 7,816 distinct repositories.

However, these 10,837 tweets are only from (a subset of) the 70,427 developers in cross-linked GitHub-Twitter public dataset we started from, while potentially many other people could have tweeted about those same repositories; their tweets would go undetected if they were not part of the cross-linked dataset we started from. To capture all other tweets mentioning those repositories during our observation period, posted by people that were not part of our starting cross-linked user dataset, we further query the Twitter API for all tweets containing links with the format `https://github.com/owner/repo_name`. Note that we exclude replies, i.e., tweets posted explicitly "in reply to" other tweets, as they are generally less visible in one's timeline and therefore expected to have less attraction effect. We do, however, include retweets, for both these and all of the earlier tweets in our dataset.

[3]We used the Open Source Initiative list https://opensource.org/licenses/alphabetical

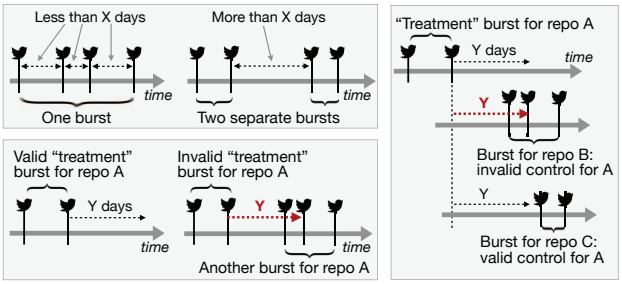


Figure 3: Operationalization of tweet *burst* and diff-in-diff data setup. Top left: Tweets mentioning the same repository within X days of each other are considered part of the same burst. Bottom left: Two bursts mentioning the same repository must be at least Y days apart. Right: Control group repositories must not have experienced any bursts of their own at least Y days after the end of the corresponding treatment group repository burst.

Finally, we expand the set of heuristics used by Fang et al. [26] to link users across the two platforms. Specifically, we further cross-link users based on: (i) similarity of their display names and usernames / logins, as Bird et al. [7] did originally for commit logs and email archives; (ii) similarity of their profile pictures on GitHub and Twitter—we use average per-channel (RGB) histogram distance between two images for comparison. For validation, we manually checked random samples of GitHub-Twitter user pairs suggested by the heuristics against other public information online. See our replication package for scripts and more details.

5.2 Aggregating Tweets into Bursts

Twitter mentions of GitHub repositories may occur closely together in time, as part of coordinated or coincidental *bursts* of social media activity. Taking the example in Figure 2, note how eight tweets by different authors (O1–O8) and many retweets of these, all mentioning the same repository, occurred within a short period. In this case, it is unclear how to reason about which tweet caused a possible increase in GitHub stars shortly thereafter—it could be any subset of those tweets. Rather than reasoning about tweets separately, we therefore first aggregate tweets into *bursts* and then reason about the effect, if any, of a burst as a whole.

There are two important operationalization decisions here. First is *identifying the start and end of a burst* (top left in Figure 3). To this end, we defined a hyper-parameter X representing the maximum allowable time gap (measured in days) between any consecutive two tweets or retweets mentioning a given repository, before they are considered to be part of different bursts.

Second is *dealing with neighboring bursts* (bottom left in Figure 3). It may take some time before the effects of a burst, if any, become visible. During this time, it is possible for another burst mentioning the same repository to have started, creating ambiguity about which burst caused those effects. To avoid this, we define a hyper-parameter Y representing the observable effects period after the end of a burst, and conservatively discard all bursts that are "too close" to neighboring bursts, i.e., if there is another burst mentioning the same repository, either started less than Y days after the end, or ended less than Y days before the start of this burst.

This decision has implications also for the control group repositories (more details about how we identified them below). A key assumption behind our diff-in-diff causal inference analysis is that the repositories acting as controls have not been treated, i.e., mentioned in Twitter bursts, around the same time. This implies that we necessarily also require that a potential burst mentioning a control group repository must not have started less than Y days after the end of a burst mentioning the corresponding treatment-group repository (right side of Figure 3).

Our results below are computed with $X = 3$ and $Y = 3$ days.[4] We performed sensitivity analyses for $X, Y \in \{3, 7\}$ and found the conclusions after regression analysis to be consistent. Given the chosen values, our resulting dataset contains 6,981 bursts in total (15,975 original tweets and 28,569 retweets), mentioning 2,370 unique repositories.

5.3 Compiling a Control Group

As discussed in Section 4, to test the causal relationship between tweets mentioning GitHub repositories and the number of GitHub stars and the number of new committers, we adopt a diff-in-diff design [1]. Specifically, we consider a burst of tweets mentioning a particular GitHub repository as an *intervention* and contrast the change in the two outcome variables post intervention between the treated repositories and an appropriate control group.

To compile our control group, we adopt a *stacked diff-in-diff* design [33], i.e., we identify suitable control group repositories also *from among the set of repositories mentioned in tweets*. This is possible because even if all repositories considered are mentioned in at least one Twitter burst, *they are not all mentioned at the same time*; thus, for any repository treated at time t, other repositories treated at different times can serve as controls for time t. This approach has several advantages, including an implicit way to control for some confounding factors (potential ways in which the repositories ever mentioned in tweets are different from those never mentioned) and computational efficiency (some variables that need to be computed for the control group will have been computed anyway).

Specifically, we use propensity score matching [14] to sample as controls, for every treated repository, up to five other repositories that were not mentioned in tweets around the same time,[5] but which show *parallel trends* in the outcome variables leading up to the intervention time, compared to the treated repository. We chose the 1:5 ratio (as opposed to 1:1) to increase robustness in our statistical conclusions while keeping the modeling sample relatively balanced. See Appendix for more implementation details of the matching approach.

5.4 Estimating Tweet Burst Causal Effects and Their Moderators (RQ1, RQ2)

As per Section 5.2, our unit of analysis is a tweet burst mentioning a particular GitHub repository. For every such burst, we record its time and mentioned GitHub repository, variables capturing

[4] The attentive reader may notice that in this case the bottom-left condition in Figure 3 becomes superfluous, unlike instances when $Y \neq X$.
[5] Recall also the constraints in Section 5.2 and Figure 3.

Table 1: The definitions of the variables in our models.

Outcome Variables	
Number of new stars	The number of stars received by the mentioned repository during the observation period.
Number of new committers	Analogous to stars. New committers are defined as developers never made commit or sent pull request to the focal repository before the tweet mention.
Tweet Burst Characteristics	
Number of likes	Sum of Twitter likes across all tweets in a burst.
Number of original tweets	Tweet count in a burst, excluding retweets.
Number of retweets	Retweet count in a burst.
Average tweet length	Measured characters; longer, more descriptive tweets can convey more information and may have stronger effects. The VIF test revealed collinearity with Number of original tweets and it was thus removed in the final model.
Has hashtags	True if more than half of the tweets in the burst contain hashtags (we also use 'any tweet in the burst contains hashtags' as alternative construction). Hashtags are known to impact information diffusion [58] and are expected to increase the visibility of tweets.
Is from committers	True if more than half of the tweets in the burst is authored by a current or past project commit author (we also use 'at least one tweet authored by current or past project committers' as alternative construction). In general, the identity of the creators has a big impact on the popularity of social media content [6].
Is promotional	True if more than half of the tweet burst contain an URL pointing to a repository homepage. Following Fang et al. [26] we assume that such tweets are primarily intended for promotional purposes, while tweets with links to other GitHub artifacts, most often specific issues, tend to serve other purposes. See Appendix for the validation of this approach.
Controls for Concurrent Events	
Had official release	True if the repository had an official release on GitHub during the period of tweet burst or within 24-hour after the start of the burst.
Is GitHub trending	True if the repository appeared on the GitHub trending page during the period of tweet burst or within 24-hour after the start of the burst. We collect historical records of trending repositories from a public dataset [52].
Number of Google search results	The number of Google search results containing the exact repository slug and originating outside of the `twitter.com` domain during the period of tweet burst; serves as a proxy for the scale of promotion repositories may have received outside of Twitter.
Burst duration	The duration of tweet burst (in hours). The longer the duration, the longer the period under observation.
Repository Characteristics	
Project age	Number of days since the first commit.
Project contribs	Total number of commit authors, excluding obvious bots [21].
Project commits	Total number of commits by the previous authors.
Attracted Developer Characteristics	
GitHub tenure	Number of days since the new contributor's first commit anywhere on GitHub.
GitHub commits	Total number of commits by the new contributor anywhere on GitHub.
Has GitHub collab	True if the new contributor ever committed to the same repository as any of the focal project's (i.e., the project mentioned by tweets) committers in the past.
Twitter tenure	Number of days since the new contributor's first tweet.
Num. tweets	Total number of original tweets (i.e., excluding retweets) authored by the new contributor in the past year.
Ratio original tweets	The fraction of original tweets to all tweets authored by the new contributor in the past year.
Has Twitter interaction	True if the new contributor ever @-mentioned any of the authors of tweets in the focal burst in the past year.

different characteristics of the burst and tweet authors, variables capturing different characteristics of the repository, and measurements of the two dependent variables — number of GitHub stars and number of committers — computed immediately before and Y days after the burst; see Table 1 for the complete list. We also record analogous measures for the up to five control group bursts (repositories) assigned to every treatment group burst (recall Section 5.3). We refer to all the pre- and post-treatment observations for a given treatment and corresponding control group bursts as a *cohort*.

Model Specification. To answer **RQ1**, we estimate the regression:

$$O_{itc} = \beta_0 p_{tc} * t_{ic} + \beta_1 p_{tc} * S_{ic} + \beta_2 p_{tc} + \beta_3 t_{ic} + \delta_{tc} + \alpha_{ic} \quad (1)$$

Here, O_{itc} represents the outcome variable within the period starting at time t, for a given repository i in cohort c, where a cohort refers to a treated repository and all its corresponding controls. p_{tc} is a flag indicating whether time t is in the pre- or post-treatment period of cohort c, and t_{ic} is a flag indicating whether repository i is in the treatment or control group for cohort c. S_{ic} denotes concurrent non-tweet treatments potentially experienced by the same repository (recall the discussion of the GitHub trending page and similar 'treatments' in Section 4). Finally, δ_{tc} and α_{ic} are "time-cohort" and "repository-cohort" random effects, necessary given the inherent nested structure of our data—e.g., the same repository may appear as part of different cohorts at different times, violating the independence assumption expected in regression analyses [51]. This specification gives an intuitive interpretation of the estimated coefficient β_0—it is *the average treatment effect of the tweet burst*, since p_{tc} and t_{ic} capture the average difference between the treated and control groups, and between the pre- and post-treatment periods.

To answer **RQ2**, we extend the model to incorporate X_{ic}, the characteristics of the tweets mentioning a given GitHub repository in a given cohort, following a model specification by Duflo [22]:

$$O_{itc} = \beta_0 p_{tc} * t_{ic} * X_{ic} + \beta_1 p_{tc} * S_{ic} + \beta_2 p_{tc} + \beta_3 t_{ic} + \delta_{tc} + \alpha_{ic} \quad (2)$$

The interaction term $p_{tc} * t_{ic} * X_{ic}$ ensures that the tweet burst characteristics will only affect the value of fitted outcome variables

post-treatment. The estimated β_0 should be interpreted as the *moderating effect of those characteristics on the outcome variable*. Table 1 lists the definitions of the variables used across both models.

Model Estimation. When estimating the regressions, we take the standard precautions (see replication package), including filtering out outliers (top 1% most extreme values) [50], log-transforming variables with skewed distributions to reduce heteroscedasticity [32], and checking for multicollinearity using the variance inflation factor [67]. As indicators of goodness of fit, we report a marginal (R^2_m—the variance explained by the fixed effects alone) and a conditional (R^2_c—fixed and random effects together) coefficient of determination for generalized mixed-effects models [35, 47].

5.5 Identifying Characteristics of Developers Likely Attracted By Tweets (RQ3)

For the second part of our study, answering **RQ3**, we focus on developers placing their first contributions to the different GitHub projects soon after the repositories were mentioned in tweets, and who were plausibly attracted by those tweets.

To better understand the underlying mechanisms, we start by qualitatively studying the relationship between attracted developers and Twitter authors. For every repository and tweet burst we extract the authors of commits recorded in the main branch within 30 days of the end of the burst, that had not committed to the project before; both direct push commits and indirect merged pull request commits are captured this way. We then label those new committers as *likely attracted by tweets* if they (i) are part of our Twitter-GitHub cross-linked dataset, (ii) retweeted one of the tweets mentioning the GitHub repository within 3 days after the tweet was posted; and (iii) starred the mentioned repository in the same period. Retweeting ensures the user is exposed to the tweet, while starring the repository after the tweet increases the likelihood that the committer was attracted by the tweet.

With a set of 81 such committers likely attracted by tweets (and corresponding tweet authors who plausibly attracted them), one author coded and analyzed all instances of Twitter interaction and GitHub activity between the tweet authors and attracted committers using thematic analysis. This was an iterative process that involved discussing with another author the different codes and examples thereof, resolving disagreements and recoding, where needed. In each case, we coded on three dimensions: i) the frequency and directionality of past Twitter interactions; ii) the apparent intent of the current tweet and interaction; iii) the frequency of past GitHub collaboration and each other's roles in the respective repositories. The coded interaction is then used to infer the relationship between tweet authors and attracted committers, and try to understand the reason developers are attracted.

Following the qualitative analysis we estimate three regressions. First, we run a logistic regression model to **compare developers who contributed to the GitHub projects to others who did not**, conditioning on both plausibly having been exposed to the tweets. Here we use past Twitter interaction between the new committers and the tweet authors to identify users exposed to tweets, as users with frequent past interaction are more likely to be exposed to tweets posted by each other.[6] We define interaction as explicit *@*-mentioning the tweet author and we use at least three past Twitter "interactions" as the threshold of "tweet exposure."

With the outcome variable being whether the exposed developer is attracted as committer (i.e., made a first commit within 30 days after the tweet burst), GitHub collaboration and other developer-level co-variates are included as independent variables and the estimated coefficient reflects the difference between developers attracted versus not. Because of the low number of committers attracted and high volume of "exposed users," we randomly downsample the exposed users to make the data frame relatively balanced with respect to the outcome variable.

Finally, we estimate regression and survival models to test how the new committers likely attracted by tweets differ from other new contributors during the same period in terms of (i) their **total number of commits** 30 days after their first contribution (linear regression);[7] and (ii) their **total length of engagement with the project** (Cox proportional-hazards regression)—we follow prior work [43, 54] to detect disengagement as the start of 12 months of inactivity. See Table 1 for definitions of variables and the results in Section 6 for the complete set of independent and control variables included in each model.

6 RESULTS

6.1 Tweet Effects on Project Popularity and New Contributors (RQ1)

We present a series of nested diff-in-diff regression results answering our first two research questions in Table 2.

We begin here by presenting the results for **RQ1** and the **Number of new stars** outcome variable. Model I estimates the average causal effect of tweet bursts mentioning GitHub repositories on the number of new GitHub stars gained within 3 days after the end of the burst. Interpreting the regression results, we first note statistically significant effects for all control variables: having official releases, being featured on the GitHub trending page, and otherwise showing up in Google search results, are all associated with an increase in the number of GitHub stars gained, as expected.

Turning to the main treatment effect, captured by the interaction term **Is treated group? * Is post-treatment** as per the model specification in Section 5.4, we find a statistically significant positive effect of the tweet burst on the number of GitHub stars gained: on average each tweet burst mentioning the repository corresponds to approximately 7% increase in stars (note the dependent variable is log-scaled, coefficient should be interpretted as the percentage of increase). Considering that the average number of stars gained for treated repositories in the pre-treatment period in our sample is 16.53 and the median is 9, a 7% increase corresponds to more than one star gained via tweets on average *for every tweet burst*. Moreover, we note a positive effect of the **Burst duration**—the longer the bursts (and thus the exposure of those tweets), the more stars are gained.

Next we turn to Model IV, which estimates the tweet bursts' effect on the **Number of new committers** to the GitHub projects. Similarly to Model I, we observe a statistically significant positive

[6] https://help.twitter.com/en/using-twitter/twitter-timeline
[7] See replication package for results with 90, 180, and 360 days.

Table 2: Summaries of diff-in-diff regressions estimating the effect of tweets mentioning GitHub repositories on attracting new stars and new committers. We report the coefficient estimates together with their standard errors in parentheses.

	Number of new stars (log)			Number of new committers (log)		
	Model I	Model II	Model III	Model IV	Model V	Model VI
Main Treatment Effect						
Is post-treatment	$-0.24(0.01)^{***}$	$-0.24(0.01)^{***}$	$-0.24(0.01)^{***}$	$-0.08(0.00)^{***}$	$-0.07(0.00)^{***}$	$-0.07(0.00)^{***}$
Is treated group?	$0.24(0.02)^{***}$	$0.22(0.02)^{***}$	$0.12(0.02)^{***}$	$0.01(0.00)^{*}$	$0.01(0.00)^{*}$	$0.01(0.01)$
Is treated group? : Is post-treatment?	$0.07(0.02)^{***}$			$0.02(0.00)^{***}$		
Other Treatments and Controls						
Number of Google search results (log)	$0.60(0.01)^{***}$	$0.59(0.01)^{***}$	$0.61(0.02)^{***}$	$0.09(0.00)^{***}$	$0.09(0.00)^{***}$	$0.09(0.01)^{***}$
Had official release	$0.08(0.03)^{*}$	$0.06(0.03)$	$0.07(0.05)$	$0.11(0.01)^{***}$	$0.10(0.01)^{***}$	$0.10(0.01)^{***}$
Is GitHub trending	$0.90(0.07)^{***}$	$0.86(0.07)^{***}$	$0.71(0.10)^{***}$	$0.07(0.02)^{**}$	$0.06(0.02)^{**}$	$0.05(0.03)$
Burst duration (log)	$0.15(0.00)^{***}$	$0.14(0.00)^{***}$	$0.13(0.01)^{***}$	$0.03(0.00)^{***}$	$0.02(0.00)^{***}$	$0.03(0.00)^{***}$
Tweet Burst Characteristics[1]						
Has hashtags		$-0.09(0.02)^{***}$	$-0.11(0.04)^{**}$		$0.01(0.01)$	$-0.01(0.01)$
Is promotional		$0.01(0.02)$	$-0.07(0.03)^{*}$		$-0.07(0.01)^{***}$	$-0.07(0.01)^{***}$
Number of likes (log)		$0.03(0.02)$	$0.09(0.02)^{***}$		$0.00(0.00)$	$0.01(0.01)$
Number of original tweets (log)		$0.11(0.03)^{***}$	$0.19(0.04)^{***}$		$0.06(0.01)^{***}$	$0.06(0.01)^{***}$
Number of retweets (log)		$0.05(0.03)$	$-0.03(0.04)$		$0.00(0.01)$	$0.02(0.01)$
Is from committers			$-0.38(0.05)^{***}$			$0.00(0.02)$
Num. obs.	65,354	65,354	32,050	65,354	65,354	32,050
R^2_m (R^2_c)	0.10(0.49)	0.10(0.49)	0.09(0.47)	0.05(0.41)	0.05(0.41)	0.05(0.42)

[1] All are 3-way interaction terms with *Is treated group? : Is post-treatment?*, omitted for clarity. *Note:* $^{*}p < 0.05$; $^{**}p < 0.01$; $^{***}p < 0.001$

effect of tweeting about a repository on the number of new committers to the project in the following 3 days post burst. However, compared to the number of stars, the effect on new committers is considerably smaller—the estimated coefficient of 0.02 for the interaction **Is treated group? * Is post-treatment** corresponds to an average of 2% increase in new committers attracted per tweet burst. In absolute terms, considering that the mean number of new committers gained by treated repositories in the pre-burst period is 0.21 (median 0), given the size of our dataset one can expect this effect to translate to only approximately one out of every 250 repositories gaining a new committer as a result of the tweet burst, on average.

6.2 Characteristics of Impactful Tweets (RQ2)

Now we turn to Models II-III and V-VI, which add interaction terms between the various tweet burst characteristics and the previous **Is treated group? * Is post-treatment** effect on the **Number of new stars** (Models II-III) and the **Number of new committers** (Models V-VI). The estimated coefficients for these variables can be interpreted as the moderating effect of tweet burst characteristics on attracting new GitHub stars and new committers, since the tweet burst characteristics are only defined for repositories in the treatment group. Recall, since the addition of the **Is from committers** variable requires cross-linking of user accounts across the two platforms, we run Models III and VI on subsets of our dataset, where that information was available (cf. Section 5.1).

Interpreting the estimated coefficients, we make the following observations. First, the number of original tweets has statistically significant positive interaction effects with the treatment: the more original tweets, the stronger the effect of the burst on the **Number of new stars**. One can expect that doubling the number of original tweets in the burst, will lead to an average increase of 1.8 stars (i.e., 11%). The effect of tweet likes is partially confirmed by model III and no effect is found for the number of retweets. One possible explanation is that the relatively strong co-linear relationship among the three variables related to burst popularity (i.e., the number of original tweets, retweets, and likes in a burst) makes the effect less obvious for some variables. Model V (VI) confirms the overall moderating effect of tweet burst popularity on the **Number of new committers**, but only in terms of **Number of original tweets**—one can expect that doubling the number of original tweets in a burst can lead to an average of approximately 0.013 new committers per project, or one new committer for every 80 repositories.

Second, we turn to the effect of the **Is promotional** variable, capturing the tweet intent; recall, following Fang et al. [26] we consider tweets pointing to a GitHub repository homepage, as opposed to other targets like issue discussions, as promotional. We observe that the tweet intent has a statistically significant moderating effect on the **Number of new committers** per Model V (VI), but not much moderating effect on the **Number of new stars** per Model II (III) (only significant at model III and the variance is high). That is, while the effect of tweet bursts on attracting new stars does not vary much with tweet intent on average, tweet bursts that typically point to issue threads or pull requests (non-promotional per our operationalization) are expected to attract more new committers compared with bursts pointing to repository homepages.

Finally, the affiliation of tweet authors with the respective projects being mentioned in the tweets, i.e., whether the tweet burst **Is from committers**, has a statistically significant moderating effect for the **Number of new stars**—Model III reveals that tweet bursts from project contributors can be expected to attract 38% fewer new stars on average, controlling for the number of tweets in the burst. The effect of the tweet burst on the **Number of new committers** does not vary with the affiliation of tweet authors with the project, per

Model VI. We found a negative effect to attract new stars by using hashtags, and no effect on new committers.

6.3 Characteristics of New Contributors (RQ3)

Recall, we used thematic analysis to characterize the types of relationships between the new committers to the repositories mentioned in tweets and authors of those tweets (see Section 5.5). We stopped after qualitatively analyzing a sample of 19 such developers' GitHub and Twitter activity histories since we did not observe any new themes. Across our sample, we observe three main themes:

- **Repeated past Twitter interaction & GitHub collaboration** (6 instances, 32%): The new contributors and the tweet authors appear close socially and they share a long history of collaboration. They have had intense two-way Twitter interactions (i.e., retweeting and replying to each other's tweets), and they usually have also committed to the same other GitHub repositories before as well.
- **Following community leaders or influential developers** (7 instances, 36%): The new contributors appear either interested in a specific project or the work of an influential developer. They posted many tweets or retweets about the project or from the influential developer. In one case, the influential Twitter account was a bot.
- **Weak ties** (6 instances, 32%): The new contributors have little or no past Twitter interaction with the tweet authors and no traces of past collaboration on GitHub, but they tend to follow the tweet authors on Twitter or are in the same broad software community on Twitter, which gives them exposure to each other's tweets.

The qualitative analysis suggests the existence of past social ties plays an important role to attract new committers, but promotion on GitHub repositories can also diffuse through weak ties and reach developers with little prior interaction.

To further characterize the attracted developers, we estimate the three regressions in Table 3. Model VII is a logistic regression testing the association between the presence of past GitHub interaction and the likelihood of a developer placing their first commit to a GitHub project mentioned in tweets (i.e., **is attracted**), among developers in the cross-linked dataset who were possibly exposed to those tweets given their past Twitter interactions with the tweet authors. The model shows that developers who collaborated with focal project member in the past are more likely to be attracted (indicated by the positive effect of **Has GitHub collab**). The attracted developers tend to be newer to the GitHub platform (indicated by the negative effect of **GitHub tenure**) but not any more or less experienced otherwise outside of the focal project (no effect of **GitHub commits**); and they tend to be less active on Twitter (negative effect of **Num. tweets**), with more of their activity being original tweets than retweets (positive effect of **Ratio original tweets**). We fit another model with nominal variables representing individual developers, entered as a random effect, to assess the relative effect of individual-level variables. The associations reported above are statistically insignificant in this model, showing that individual-level random effects explain the majority of the variance, indicating whether developers being attracted are more affected by user-level characteristics not included in the model. We suggest including more detailed user-level variables (e.g., the kind of project users have committed to in the past) in future research.

Table 3: Characterizing the developers attracted by tweets

	Model VII (is attracted)	Model VIII (30-day commits)	Model IX (disengagement)
Project controls			
Project age (log)		$-0.03\ (0.01)^{**}$	$0.13\ (0.03)^{***}$
Project contribs (log)		$-0.08\ (0.01)^{***}$	$0.11\ (0.03)^{***}$
Project commits (log)		$0.07\ (0.01)^{***}$	$-0.17\ (0.02)^{***}$
Developer Characteristics			
GitHub tenure (log)	$-0.36\ (0.13)^{**}$	$-0.00\ (0.01)$	$-0.02\ (0.04)$
GitHub commits (log)	$0.01\ (0.05)$	$-0.02\ (0.01)^{**}$	$0.01\ (0.01)$
Has GitHub collab.	$1.47\ (0.13)^{***}$	$0.06\ (0.02)^{**}$	$-0.34\ (0.05)^{***}$
Has Twitter interact.		$0.07\ (0.03)^{*}$	$0.01\ (0.07)$
Twitter tenure (log)	$0.15\ (0.10)$		
Num. tweets (log)	$-0.44\ (0.07)^{***}$		
Ratio original tweets	$0.84\ (0.28)^{**}$		
Num. obs.	1,914	2,192	2,281

$^{*}p < 0.05$; $^{**}p < 0.01$; $^{***}p < 0.001$

Model VIII summarizes the estimated linear regression testing the association between the short-term activity levels of new project contributors (**30-day commit counts** after their first project commit) and the presence of past Twitter and GitHub interaction and collaboration. Controlling for project age, project size, and overall amount of past GitHub activity outside of the focal project, we observe statistically significant positive effects for both **Has GitHub collaboration** and **Has Twitter interaction**—on average, past connections are associated with higher levels of contribution in the first 30 days.

Finally, Model IX summarizes the Cox proportional hazards survival regression testing how past Twitter and GitHub interaction and collaboration associate with the risk of disengagement from the project (note the reverse coding of the outcome variable—negative estimated coefficients imply a lower risk of disengagement). The model shows that controlling for the same variables as above, prior **GitHub collaboration** with focal project member is associated with lower disengagement risk. However, the model does not reveal any statistically significant effect of past **Twitter interaction**.

7 DISCUSSION

We now summarize our main results and discuss their implications.

Twitter can be an effective mechanism to popularize open source software projects. Our study provides robust empirical evidence that tweets mentioning GitHub repositories likely *lead to* an increase in the repositories' number of GitHub stars beyond what can be explained by other observable promotional mechanisms such as being featured on the GitHub trending page or otherwise online. These results suggest that Twitter can be a useful tool for open source maintainers to promote projects, and perhaps even better than other promotional mechanisms, since anyone can tweet at any time, unlike e.g., the GitHub trending page, which maintainers have no control over and which requires the project to be popular already before being featured.

Not all tweets are created equal. Our models suggest that the popularizing effect of tweets varies with different tweet characteristics. First, we find that the more tweets there are in a burst, the stronger the effect of that burst on increasing a project's start count. This reflects, intuitively, the increased exposure and attention the repositories get when mentioned by multiple tweets around the same time. However, we find little evidence that the purpose of the tweets matters. Unlike prior work [26] suggesting that tweets that point to issue discussions or pull requests when mentioning the GitHub repositories may have less promotional effect, we find that all tweets help attract attention to the project similarly, irrespective of intent, and some of this attention translates into new GitHub stars. We do, however, find evidence that the effect varies with the tweet authors' affiliation with the GitHub projects—tweets by existing project contributors or maintainers, i.e., project 'insiders,' are seemingly less impactful than tweets by others. One possible explanation is that tweets from project insiders may be considered less objective (self-promotion) and may not be taken to reflect the true value or quality of the project. Another possible explanation is that tweets posted by others may bring this project to a different, wider audience, compared to tweets from project insiders whose Twitter followers may already know this project well. This result suggests that obtaining an 'endorsement' from a trustworthy third-party on Twitter may further benefit the project's popularity, and it also suggests the importance to promote outside one's own social circle. Multiple tweets about a repository by the same set of users may have decreasing value to attract new stars, because most of their audience have already considered the project before.

Tweets can help to attract new committers, but only under certain condition. The effect of tweets mentioning GitHub repositories on attracting new contributors is weaker than for stars. However, it is still statistically significant and *causal* given our study design. Comparing the stars and committers models, the latter indicate that more focused attention is needed. In particular, it seems important that tweets not be generic but rather point to specific repository elements, typically specific issue or pull request discussions, and that tweet bursts contain more original tweets than retweets to increase the magnitude of the effect.

Community engagement on Twitter is important. Our qualitative analysis revealed that of those repositories that do attract new committers, many of them succeeded through the bond of a community or strong interpersonal connections. Either the project itself has a vibrant community on Twitter, with many developers following the activity of the project, and major contributors and administrators of the project managing the online conversation and maintaining connection with other peer developers; or the project's maintainers are highly active and maintain intense communication with other developers on Twitter. Both suggest the importance of maintaining an active Twitter presence as a GitHub developer and managing the interaction with a set of potential collaborators on Twitter, where the latest information about developers' project can be diffused to them swiftly.

Social connections can be leveraged for work-related tasks, especially short-term ones. Our models of how the new project contributors plausibly attracted by tweets differ from other contributors, on average, confirmed the importance of maintaining ties with past collaborators on GitHub and, in addition, the added value of Twitter social ties. However, comparing the models of short- (commits in the first 30 days) and long-term outcomes (length of engagement with the project), we see weakening impact of the past Twitter connections, long-term. This suggests that open-source maintainers may tap into their Twitter social connections for on-demand, perhaps for help with specific issues, but that these resources are not necessarily sustainable.

Attention can be a double edged sword. The fact that tweets have strong effect to attract user attention, but comparatively lower numbers of new committers also raises concerns to developers tweeting about GitHub repositories. As we mentioned in Section 1, while using Twitter as a promotion platform may increase the popularity of a project, it may also simultaneously bring about requests and demands from new users without a proportional amount of contribution input. Future research can further investigate other project outcomes that may be affected by increased attention, such as the number of issue reports. Developers should also consider this factor when they decide to start a promotion campaign for their project on Twitter.

The role of Twitter in open-source development. Comparing with developers who are not evidently attracted by tweets, the plausibly attracted ones are relatively new to GitHub, and they don't post much on Twitter. We hypothesize that they are new developers looking for open-source projects to contribute to so they may not have routine collaborators or the energy and motivation to maintain a strong social media presence. We argue that Twitter is especially important for such developers, since it provides them an opportunity to receive updates or information about GitHub projects at a low cost.

Twitter can be more tightly integrated into code hosting platforms. GitHub is one of the most popular platforms for hosting open-source repositories, and it has various initiatives to promote projects (e.g., trending pages, project spotlights) which can lead to higher popularity of the promoted projects. Given that Twitter seems to be a valuable *exogenous* attention eliciting platform, we suggest that integrating Twitter access into the code hosting platform could help with broadcasting information about those projects more effectively (note that GitHub recently added an explicit Twitter field in user profile pages too[8]). This could be done, e.g., by adding a 'tweet' button on each project homepage and issue thread page. Providing ready access to Twitter could lead to easy, immediate action for developers to promote their project. This can also be helpful to other non-developer users of the project to discuss issues or promote a certain feature of the repository to their broader social circle. In addition to this, keeping in mind the moderators of the effects of tweets we uncovered, platforms could also provide tweet templates that can incorporate some of these suggestions as soon as a user tries to tweet from the project homepage.

Diff-in-diff can be a useful design in software engineering research. At a higher level, we note that our causal inference research design, while well-established in the social sciences, has hardly ever been used in software engineering research. We hope

[8] https://github.blog/changelog/2020-07-22-users-and-organizations-can-now-add-twitter-usernames-to-their-github-profiles/

that our current work will motivate empirical software engineering researchers to consider this and similar causal inference designs more frequently in the future.

8 THREATS TO VALIDITY

Despite our best efforts to carefully collect and analyze our data, we acknowledge the existence of several limitations in our study.

First, we have several missing data problems. The set of tweets we collected mentioning a repository may not be complete because of tweet deletion, which we assume is insufficiently frequent to affect our results. The commit and other activity data we extract from GHTorrent may also be incomplete; similarly, we assume that is sufficiently rare.

Second, the cross-linked Twitter and GitHub user data may not be always correct and we expect noise introduced because of potentially inaccurate Twitter-GitHub account matching. Similarly to Fang et al. [26], we selected a random sample of 100 Twitter-GitHub linked user pairs in our study (where users were either tweet authors or committers likely attracted by or exposed to tweets) for further inspection. Among these, one pair had either their Twitter or GitHub account deleted or inaccessible, which left us with 99 pairs. We then manually evaluated the accuracy of the matching by comparing the profile information and activity traces of both accounts. Among the 99 pairs, 87 appear obviously correct (87.9%), 2 obviously incorrect (2.0%), and we cannot confidently validate the accuracy of the remaining pairs (10.1%) given their public activity traces. This puts the accuracy of the linked pairs on par with the one reported by Fang et al. [26] and overall high – at least 87.9% in the validation sample, and likely higher.

9 CONCLUSION

In this paper, we empirically demonstrated that mentioning open-source GitHub repositories on Twitter can *lead to* an increase in project popularity and help to attract new developers. However, the mechanism is not equally effective for the two outcomes: while the effect of tweets on gaining new stars seems to apply to most repositories and kinds of tweets, tweeting to attract new developers is considerably less effective on average, reflective of the relatively higher bar to placing a technical contribution in an open-source project compared to simply expressing interest in the project by starring it. Still, we argue, there is hope for open-source maintainers, community managers, and evangelists, since *the effect of tweeting on both outcomes is moderated by many factors within one's control.* We conclude, optimistically, that tweeting about open source can contribute to improving open source sustainability.

ACKNOWLEDGMENTS

The authors kindly acknowledge support from the NSF awards 1546393, 1633083, 1717415, and 1901311, as well as an award from the Alfred P. Sloan Foundation. We are additionally grateful to Audris Mockus for discussions around our experimental design.

REFERENCES

[1] Alberto Abadie. 2005. Semiparametric difference-in-differences estimators. *The Review of Economic Studies* 72, 1 (2005), 1–19.
[2] Amritanshu Agrawal, Akond Rahman, Rahul Krishna, Alexander Sobran, and Tim Menzies. 2018. We don't need another hero? the impact of "heroes" on software development. In *International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*. 245–253.
[3] Guilherme Avelino, Eleni Constantinou, Marco Tulio Valente, and Alexander Serebrenik. 2019. On the abandonment and survival of open source projects: An empirical investigation. In *International Symposium on Empirical Software Engineering and Measurement (ESEM)*. IEEE, 1–12.
[4] Guilherme Avelino, Leonardo Passos, Andre Hora, and Marco Tulio Valente. 2016. A novel approach for estimating truck factors. In *International Conference on Program Comprehension (ICPC)*. IEEE, 1–10.
[5] Ali Sajedi Badashian and Eleni Stroulia. 2016. Measuring user influence in GitHub: the million follower fallacy. In *International Workshop on CrowdSourcing in Software Engineering (CSI-SE)*. IEEE, 15–21.
[6] Eytan Bakshy, Itamar Rosenn, Cameron Marlow, and Lada Adamic. 2012. The role of social networks in information diffusion. In *Proceedings of the 21st international conference on World Wide Web*. 519–528.
[7] Christian Bird, Alex Gourley, Prem Devanbu, Michael Gertz, and Anand Swaminathan. 2006. Mining email social networks. In *International Conference on Mining Software Repositories (MSR)*. 137–143.
[8] Sue Black, Rachel Harrison, and Mark Baldwin. 2010. A survey of social media use in software systems development. In *Proceedings of the 1st Workshop on Web 2.0 for Software Engineering*. 1–5.
[9] Kelly Blincoe, Jyoti Sheoran, Sean Goggins, Eva Petakovic, and Daniela Damian. 2016. Understanding the popular users: Following, affiliation influence and leadership on GitHub. *Information and Software Technology* 70 (2016), 30–39.
[10] Hudson Borges and Marco Tulio Valente. 2018. What's in a GitHub star? Understanding repository starring practices in a social coding platform. *Journal of Systems and Software* 146 (2018), 112–129.
[11] Hudson Silva Borges and Marco Tulio Valente. 2019. How do developers promote open source projects? *Computer* 52, 8 (2019), 27–33.
[12] Gargi Bougie, Jamie Starke, Margaret-Anne Storey, and Daniel M German. 2011. Towards Understanding Twitter Use in Software Engineering: Preliminary Findings, Ongoing Challenges and Future Questions. In *International Workshop on Web 2.0 for Software Engineering*. ACM, 31–36.
[13] Chris Brown and Chris Parnin. 2020. Understanding the impact of GitHub suggested changes on recommendations between developers. In *Joint Meeting on the Foundations of Software Engineering (ESEC/FSE)*. ACM, 1065–1076.
[14] Marco Caliendo and Sabine Kopeinig. 2008. Some practical guidance for the implementation of propensity score matching. *Journal of Economic Surveys* 22, 1 (2008), 31–72.
[15] Casey Casalnuovo, Bogdan Vasilescu, Premkumar Devanbu, and Vladimir Filkov. 2015. Developer onboarding in GitHub: the role of prior social links and language experience. In *Proceedings of the 2015 10th joint meeting on foundations of software engineering*. 817–828.
[16] Carlos Castillo, Mohammed El-Haddad, Jürgen Pfeffer, and Matt Stempeck. 2014. Characterizing the life cycle of online news stories using social media reactions. In *ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW)*. ACM, 211–223.
[17] Jailton Coelho and Marco Tulio Valente. 2017. Why modern open source projects fail. In *Joint Meeting on Foundations of Software Engineering (ESEC/FSE)*. ACM, 186–196.
[18] Jailton Coelho, Marco Tulio Valente, Luciano Milen, and Luciana L Silva. 2020. Is this GitHub project maintained? Measuring the level of maintenance activity of open-source projects. *Information and Software Technology* 122 (2020), 106274.
[19] Jailton Coelho, Marco Tulio Valente, Luciana L Silva, and Emad Shihab. 2018. Identifying unmaintained projects in GitHub. In *International Symposium on Empirical Software Engineering and Measurement (ESEM)*. 1–10.
[20] Laura Dabbish, Colleen Stuart, Jason Tsay, and Jim Herbsleb. 2012. Social coding in GitHub: transparency and collaboration in an open software repository. In *ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW)*. 1277–1286.
[21] Tapajit Dey, Sara Mousavi, Eduardo Ponce, Tanner Fry, Bogdan Vasilescu, Anna Filippova, and Audris Mockus. 2020. Detecting and characterizing bots that commit code. In *International Conference on Mining Software Repositories (MSR)*. 209–219.
[22] Esther Duflo. 2001. Schooling and labor market consequences of school construction in Indonesia: Evidence from an unusual policy experiment. *American Economic Review* 91, 4 (2001), 795–813.
[23] Subhabrata Dutta, Sarah Masud, Soumen Chakrabarti, and Tanmoy Chakraborty. 2020. Deep Exogenous and Endogenous Influence Combination for Social Chatter Intensity Prediction. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1999–2008.
[24] Nadia Eghbal. 2016. *Roads and bridges: The unseen labor behind our digital infrastructure.* Ford Foundation.
[25] Nadia Eghbal. 2020. *Working in Public: The Making and Maintenance of Open Source Software.* Stripe Press.
[26] Hongbo Fang, Daniel Klug, Hemank Lamba, James Herbsleb, and Bogdan Vasilescu. 2020. Need for Tweet: How Open Source Developers Talk About Their GitHub Work on Twitter. In *Proceedings of the 17th International Conference on Mining Software Repositories*. 322–326.

[27] Hongbo Fang, Hemank Lamba, James Herbsleb, and Bogdan Vasilescu. 2022. Replication package. https://doi.org/10.5281/zenodo.6000085

[28] Yulin Fang and Derrick Neufeld. 2009. Understanding sustained participation in open source software projects. *Journal of Management Information Systems* 25, 4 (2009), 9–50.

[29] Fabio Ferreira, Luciana Lourdes Silva, and Marco Tulio Valente. 2020. Turnover in Open-Source Projects: The Case of Core Developers. In *Brazilian Symposium on Software Engineering (SBES)*. 447–456.

[30] Matthieu Foucault, Marc Palyart, Xavier Blanc, Gail C Murphy, and Jean-Rémy Falleri. 2015. Impact of developer turnover on quality in open-source software. In *Joint Meeting on the Foundations of Software Engineering (ESEC/FSE)*. 829–841.

[31] Felipe Fronchetti, Igor Wiese, Gustavo Pinto, and Igor Steinmacher. 2019. What attracts newcomers to onboard on OSS projects? tl;dr: Popularity. In *IFIP International Conference on Open Source Systems (OSS)*. Springer, 91–103.

[32] Andrew Gelman and Jennifer Hill. 2006. *Data analysis using regression and multilevel/hierarchical models*. Cambridge University Press.

[33] Todd A Gormley and David A Matsa. 2011. Growing out of trouble? Corporate responses to liability risk. *The Review of Financial Studies* 24, 8 (2011), 2781–2821.

[34] Georgios Gousios and Diomidis Spinellis. 2012. GHTorrent: GitHub's data from a firehose. In *International Conference on Mining Software Repositories (MSR)*. IEEE, 12–21.

[35] Paul CD Johnson. 2014. Extension of Nakagawa & Schielzeth's R2GLMM to random slopes models. *Methods in Ecology and Evolution* 5, 9 (2014), 944–946.

[36] Jymit Khondhu, Andrea Capiluppi, and Klaas-Jan Stol. 2013. Is it all lost? A study of inactive open source projects. In *IFIP International Conference on Open Source Systems (OSS)*. Springer, 61–79.

[37] Samara Klar, Yanna Krupnikov, John Barry Ryan, Kathleen Searles, and Yotam Shmargad. 2020. Using social media to promote academic research: Identifying the benefits of Twitter for sharing academic work. *PloS One* 15, 4 (2020), e0229446.

[38] Michael J Lee, Bruce Ferwerda, Junghong Choi, Jungpil Hahn, Jae Yun Moon, and Jinwoo Kim. 2013. GitHub developers use rockstars to overcome overflow of news. In *CHI'13 Extended Abstracts on Human Factors in Computing Systems*. 133–138.

[39] Jessica GY Luc, Michael A Archer, Rakesh C Arora, Edward M Bender, Arie Blitz, David T Cooke, Tamara Ni Hlci, Biniam Kidane, Maral Ouzounian, Thomas K Varghese Jr, et al. 2020. Does Tweeting Improve Citations? One-Year Results from the TSSMN Prospective Randomized Trial. *The Annals of Thoracic Surgery* (2020).

[40] Danaja Maldeniya, Ceren Budak, Lionel P Robert Jr, and Daniel M Romero. 2020. Herding a Deluge of Good Samaritans: How GitHub Projects Respond to Increased Attention. In *Web Conference (WWW)*. 2055–2065.

[41] Jennifer Marlow, Laura Dabbish, and Jim Herbsleb. 2013. Impression formation in online peer production: activity traces and personal profiles in GitHub. In *ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW)*. 117–128.

[42] Christopher Mendez, Hema Susmita Padala, Zoe Steine-Hanson, Claudia Hilderbrand, Amber Horvath, Charles Hill, Logan Simpson, Nupoor Patil, Anita Sarma, and Margaret Burnett. 2018. Open source barriers to entry, revisited: A sociotechnical perspective. In *International Conference on Software Engineering (ICSE)*. 1004–1015.

[43] Courtney Miller, David Gray Widder, Christian Kästner, and Bogdan Vasilescu. 2019. Why do people give up FLOSSing? a study of contributor disengagement in open source. In *IFIP International Conference on Open Source Systems (OSS)*. Springer, 116–129.

[44] Samim Mirhosseini and Chris Parnin. 2017. Can automated pull requests encourage software developers to upgrade out-of-date dependencies?. In *International Conference on Automated Software Engineering (ASE)*. IEEE, 84–94.

[45] Lukas Moldon, Markus Strohmaier, and Johannes Wachs. 2021. How gamification affects software developers: Cautionary evidence from a natural experiment on GitHub. In *International Conference on Software Engineering (ICSE)*. IEEE, 549–561.

[46] Suhaib Mujahid, Diego Elias Costa, Rabe Abdalkareem, Emad Shihab, Mohamed Ayme Saied, and Bram Adams. 2021. Towards Using Package Centrality Trend to Identify Packages in Decline. *arXiv preprint arXiv:2107.10168* (2021).

[47] Shinichi Nakagawa and Holger Schielzeth. 2013. A general and simple method for obtaining R2 from generalized linear mixed-effects models. *Methods in Ecology and Evolution* 4, 2 (2013), 133–142.

[48] Andrei Oghina, Mathias Breuss, Manos Tsagkias, and Maarten De Rijke. 2012. Predicting imdb movie ratings using social media. In *European Conference on Information Retrieval*. Springer, 503–507.

[49] Cassandra Overney, Jens Meinicke, Christian Kästner, and Bogdan Vasilescu. 2020. How to not get rich: An empirical study of donations in open source. In *International Conference on Software Engineering (ICSE)*. 1209–1221.

[50] Jagdish K Patel, CH Kapadia, and Donald Bruce Owen. 1976. *Handbook of statistical distributions*. M. Dekker.

[51] José C Pinheiro and Douglas M Bates. 2000. Linear mixed-effects models: basic concepts and examples. *Mixed-effects models in S and S-Plus* (2000), 3–56.

[52] Vitaliy Potapov. [n.d.]. GitHub Trending Repos. https://github.com/vitalets/github-trending-repos.

[53] Huilian Sophie Qiu, Yucen Lily Li, Susmita Padala, Anita Sarma, and Bogdan Vasilescu. 2019. The Signals that Potential Contributors Look for When Choosing Open-source Projects. *Proceedings of the ACM on Human-Computer Interaction* 3, CSCW (2019), 1–29.

[54] Huilian Sophie Qiu, Alexander Nolte, Anita Brown, Alexander Serebrenik, and Bogdan Vasilescu. 2019. Going farther together: The impact of social capital on sustained participation in open source. In *International Conference on Software Engineering (ICSE)*. IEEE, 688–699.

[55] Naveen Raman, Minxuan Cao, Yulia Tsvetkov, Christian Kästner, and Bogdan Vasilescu. 2020. Stress and burnout in open source: Toward finding, understanding, and mitigating unhealthy interactions. In *International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*. 57–60.

[56] Peter C Rigby, Yue Cai Zhu, Samuel M Donadelli, and Audris Mockus. 2016. Quantifying and mitigating turnover-induced knowledge loss: case studies of Chrome and a project at Avaya. In *International Conference on Software Engineering (ICSE)*. IEEE, 1006–1016.

[57] Marian-Andrei Rizoiu, Lexing Xie, Scott Sanner, Manuel Cebrian, Honglin Yu, and Pascal Van Hentenryck. 2017. Expecting to Be HIP: Hawkes Intensity Processes for Social Media Popularity. In *International Conference on World Wide Web (WWW)*.

[58] Daniel M Romero, Brendan Meeder, and Jon Kleinberg. 2011. Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter. In *Proceedings of the 20th international conference on World wide web*. 695–704.

[59] Suman Deb Roy, Tao Mei, Wenjun Zeng, and Shipeng Li. 2013. Towards cross-domain learning for social video popularity prediction. *IEEE Transactions on Multimedia* 15, 6 (2013), 1255–1267.

[60] Sonali K Shah. 2006. Motivation, governance, and the viability of hybrid forms in open source software development. *Management science* 52, 7 (2006), 1000–1014.

[61] Jyoti Sheoran, Kelly Blincoe, Eirini Kalliamvakou, Daniela Damian, and Jordan Ell. 2014. Understanding "watchers" on GitHub. In *International Conference on Mining Software Repositories (MSR)*. 336–339.

[62] Leif Singer, Fernando Figueira Filho, and Margaret-Anne Storey. 2014. Software engineering at the speed of light: how developers stay current using twitter. In *Proceedings of the 36th International Conference on Software Engineering*. 211–221.

[63] Igor Steinmacher, Tayana Conte, Marco Aurélio Gerosa, and David Redmiles. 2015. Social barriers faced by newcomers placing their first contribution in open source software projects. In *ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW)*. 1379–1392.

[64] Igor Steinmacher, Marco Aurelio Graciotto Silva, Marco Aurelio Gerosa, and David F Redmiles. 2015. A systematic literature review on the barriers faced by newcomers to open source software projects. *Information and Software Technology* 59 (2015), 67–85.

[65] Margaret-Anne Storey, Christoph Treude, Arie van Deursen, and Li-Te Cheng. 2010. The impact of social media on software engineering practices and tools. In *Proceedings of the FSE/SDP workshop on Future of software engineering research*. 359–364.

[66] Margaret-Anne Storey, Alexey Zagalsky, Fernando Figueira Filho, Leif Singer, and Daniel M German. 2016. How social and communication channels shape and challenge a participatory culture in software development. *IEEE Transactions on Software Engineering* 43, 2 (2016), 185–204.

[67] Christopher Glen Thompson, Rae Seon Kim, Ariel M Aloe, and Betsy Jane Becker. 2017. Extracting the variance inflation factor and other multicollinearity diagnostics from typical regression results. *Basic and Applied Social Psychology* 39, 2 (2017), 81–90.

[68] Asher Trockman, Shurui Zhou, Christian Kästner, and Bogdan Vasilescu. 2018. Adding sparkle to social coding: an empirical study of repository badges in the npm ecosystem. In *International Conference on Software Engineering (ICSE)*. 511–522.

[69] Jason Tsay, Laura Dabbish, and James Herbsleb. 2014. Influence of social and technical factors for evaluating contribution in GitHub. In *International Conference on Software Engineering (ICSE)*. 356–366.

[70] Marat Valiev, Bogdan Vasilescu, and James Herbsleb. 2018. Ecosystem-level determinants of sustained activity in open-source projects: A case study of the PyPI ecosystem. In *Joint Meeting on the Foundations of Software Engineering (ESEC/FSE)*. 644–655.

[71] Jeffrey M Wooldridge. 2016. *Introductory econometrics: A modern approach*. Nelson Education.

[72] Kazuhiro Yamashita, Shane McIntosh, Yasutaka Kamei, Ahmed E Hassan, and Naoyasu Ubayashi. 2015. Revisiting the applicability of the pareto principle to core development teams in open source software projects. In *International Workshop on Principles of Software Evolution (IWPSE)*. 46–55.

[73] Wenbin Zhang and Steven Skiena. 2009. Improving movie gross prediction through news analysis. In *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, Vol. 1. IEEE, 301–304.

APPENDIX

A EXAMPLE TWEET BURST

Figure 4 contains the tweets mentioning the `axa-group/nlp.js` repository as part of the same burst.

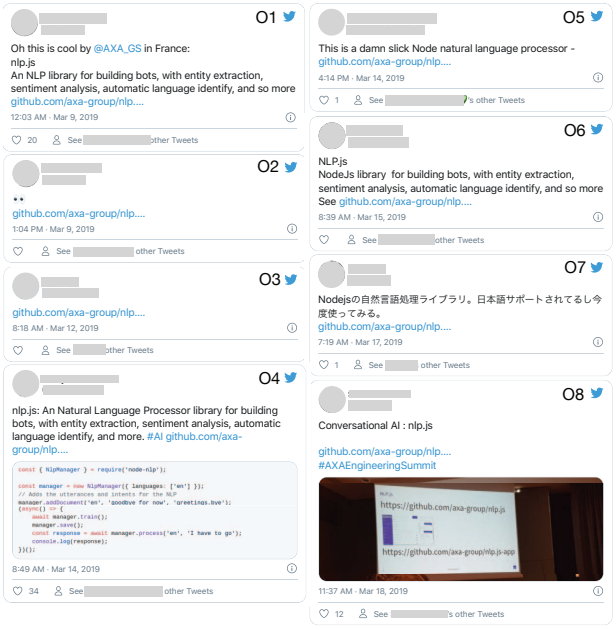


Figure 4: Tweets mentioning the `axa-group/nlp.js` repository as part of a burst of social media activity spanning 10 days.

B PROPENSITY SCORE MATCHING

As part of our DID modeling, we used propensity score matching to ensure the control repositories, on average, have the same pre-treatment trend in outcome variables (i.e., daily number of committers and stars gained) as the treatment group. We use a logistic regression model to fit the probability of a repository being mentioned by a tweet at a particular time, given a set of repository features. Following Maldeniya et al. [40], we use the *log* of the relative increase in outcome variables (i.e., stars and new contributor counts) and the absolute number of stars and new committers gained as predictors. Specifically, we compute

$$I_{it} = log(\frac{O_{it}+1}{O_{i(t-1)}+1})$$

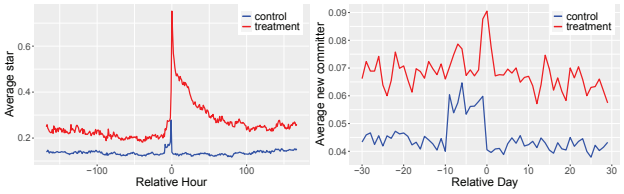
where I_{it} is the relative increase of a given outcome variable for repository i at time t, and O_{it} corresponds to the value of that outcome variable. We add one to both the numerator and the denominator to handle zero counts for stars or new committers.

Manual evaluation shows the number of new stars gained of treated repositories start to increase around nine hour before the treatment, with new committers increases nine day before. Therefore, we include the relative increase on both the number of stars and new committers gained, starting 9-hour before the treatment for stars and 9-day before for new committers, as well as the total number of stars and new committers gained within 9-hour (stars), or 9-day (new committers) before the treatment.

Our model is formally described in equation 3, where $P(t_{ic})$ corresponds to the probability repository i is treated by tweet c at time t, $IC_{i(t-j)}$ and $IS_{i(t-j)}$ stands for the relative increase on "star" and "new committers" gained for repository i on the j^{th} unit of time before the treatment, respectively. The unit time of relative increase is 1-hour for stars and 24-hour day for new committers. Similarly, $S_{i(t-j)}$ and $C_{i(t-j)}$ corresponds to the number of stars and new committers gained for repository i on the j^{th} unit before treatment. (i.e., $IS_{i(t-2)}$ corresponds to the relative star increase for repository i from the period of 2-1 hour before the treatment, to the period of 1-0 hour before the treatment. $S_{i(t-2)}$ corresponds to the number of stars gained for repository i from the period of 2-1 hour before the treatment.). σ here is the "sigmoid" function and transforms the result within $0-1$ range.

$$P(t_{ic}) = \sigma(\sum_{j=2}^{j=9} IC_{i(t-j)} + \sum_{j=2}^{j=9} IS_{i(t-j)} + \sum_{j=1}^{j=9} S_{i(t-j)} + \sum_{j=1}^{j=9} C_{i(t-j)}) \quad (3)$$

We plot the pre-treatment trend of outcome variables in both treatment and control groups in Figure 5, the outcome variable at relative time (day or hour) 0 or after is in the post-treatment period, and at time -1 or before is in the pre-treatment period. According to the graph, both the treatment and matched control group display an upward trend in both the number of stars and new committers gained before the treatment, and the trends are generally the same until the treatment. The plot indicates that our matched control repositories have similar pre-treatment trend in the outcome variables as the treated repositories, therefore the parallel trend assumption holds.



(a) (b)

Figure 5: (a) Hourly number of stars gained for control and treatment group (b) Daily number of new committers gained for control and treatment group.

C PROMOTIONAL TWEET IDENTIFICATION VALIDATION

In section 5.4, we use the target of link mentioned by tweet (i.e., repository homepage linked for promotional purposes, and issue page linked for technical discussion purposes) as a proxy for the intention of the tweet. However, this might not be the case. To validate this approach, we randomly select 20 tweets mentioning a GitHub repository homepage, and another 20 tweets mentioning any other repository related page (e.g., issue page). After randomly shuffling these 40 tweets, we asked an open source software researcher (not part of author team, and not aware of the methodology of data collection) to manually annotate the purpose of the tweet

Table 4: Descriptive data of projects tweeted

	Mean	Median	Std	Min	Max	N/P
Project commit	1030.83	279.5	5298.76	1	169,480	2,370
Project star	2393.74	632.5	4866.98	0	76,434	2,370
Project developer	40.18	13.0	113.35	1	2,076	2,370
Project age (in days)	959.60	773.0	743.59	31	4,030	2,370
Percentage project owned by organization						49.96%

Table 5: Descriptive data of tweets mentioning project

	Mean	Median	Std	Min	Max	N/P
Total tweet like in burst	16.42	1.0	104.72	0	4,647	6,981
Original tweets in burst	2.29	1.0	6.71	1	186	6,981
Retweets in burst	4.09	0.0	23.42	0	852	6,981
Burst duration in hour	22.93	0.0	55.92	0	908	6,981
Percentage tweet bursts with more than half hashtags						33.62%
Percentage promotional tweet bursts						62.38%
Percentage tweet bursts with more than half posted by project developer						5.66%
Percentage tweet bursts with unidentified author						52.56%

into three categories: (a) tweet to promote a project, (b) tweet not to promote a project, and (c) unclear purpose.

Out of the 40 tweets, the annotator marked 5 as "unclear" purpose, which we excluded from further analysis in this section. For the remaining 35 tweets, we discovered that 68.75% of tweets marked as promotional tweets by our heuristic were also marked as promotional by the annotator. Similarly, only 15.8% of tweets labelled as non-promotional tweet by our heuristic were labelled as promotional tweets by annotator, bringing the overall accuracy of our heuristic (assuming annotator to be ground truth) to be 77.1%.