1-2019

A first look at unfollowing behavior on GitHub

Jing JIANG

David LO
Singapore Management University, davidlo@smu.edu.sg

Yun YANG

Jianfeng LI

Li ZHANG

DOI: https://doi.org/10.1016/j.infsof.2018.08.012

Citation

JIANG, Jing; LO, David; YANG, Yun; LI, Jianfeng; and ZHANG, Li. A first look at unfollowing behavior on GitHub. (2019). *Information and Software Technology*. 105, 150-160. Research Collection School Of Information Systems.
Available at: https://ink.library.smu.edu.sg/sis_research/4361

A first look at unfollowing behavior on GitHub

Jing Jiang[a], David Lo[b], Yun Yang[a], Jianfeng Li[a], Li Zhang[a,*]

[a] *State Key Laboratory of Software Development Environment, Beihang University, Beijing China*
[b] *School of Information Systems, Singapore Management University, Singapore*

A R T I C L E   I N F O

Keywords:
Unfollow
Relationship dissolution
GitHub

A B S T R A C T

Context: Many open source software projects rely on contributors to fix bugs and contribute new features. On GitHub, developers often broadcast their activities to followers, which may entice followers to be project contributors. It is important to understand unfollowing behavior, maintain current followers, and attract some followers to become contributors in OSS projects.
Objective: Our objective in this paper is to provide a comprehensive analysis of unfollowing behavior on GitHub.
Method: To the best of our knowledge, we present a first look at unfollowing behavior on GitHub. We collect a dataset containing 701,364 developers and their 4,602,440 following relationships in March 2016. We also crawl their following relationships in May 2013, August 2015 and November 2015. We conduct surveys, define potential impact factors, and analyze the correlation of factors with the likelihood of unfollowing behavior.
Results: Our main observations are: (1) Between May 2013 and August 2015, 19.8% of active developers ever unfollowed some users. (2) Developers are more likely to unfollow those who have fewer activities, lower programming language similarity, and asymmetric relationships.
Conclusion: Our results give suggestions for developers to reduce the likelihood of being unfollowed by their followers, and attract researchers' attention on relationship dissolution.

1. Introduction

In social networks, *relationship dissolution* refers to the breaking up of relationships by the voluntary activity of at least one partner. Relationship dissolution is the basic process of relationship change and evolution in personal networks [1]. Relationship dissolution in the real world has been studied in a variety of contexts such as romantic love [2] and student friendships [3]. Relationship dissolution in online social networks has been analyzed for Twitter [1,4–6] and Facebook [7–9], which may help in the design of a variety of social-networking tools. For example, results may be used to distinguish strong and weak relationships [10–12], and improve new friend recommender systems [13,14]. Research results may also help a tool that fine-tune the broadcasting of user updates [15] (e.g., daily updates could be widely broadcasted to people with whom one has persistent relationships and could be selectively shared with people with whom one has relationships that are bound to break).

Social coding sites integrate social networks with code management tools, and strengthen collaborations among developers [16]. GitHub is a famous social coding site, and builds social networks to connect developers [16]. On GitHub, users follow developers to receive activity updates, discover new projects and trends, learn from developers, and show respect or support [17]. Developers often broadcast their activities to followers, which may entice followers to be project contributors [18]. Similar to social networks, social coding sites also have relationship dissolution. On GitHub, *unfollowing* developers means that users stop following some developers and receiving their updates.

Previous works studied reasons of following behavior [17] and analyzed following network structure [19–21] on GitHub. However, there has been no study that investigates unfollowing behavior on GitHub. In a similar vein like studies on Twitter/Facebook [1,4–9] about relationship dissolution, studying relationship dissolution on GitHub may provide suggestions for a variety of tools which improve collaborations among developers. Since GitHub is different from Twitter and Facebook – it is a social coding site instead of a pure social network site – some factors identified in prior work may not be relevant and additional factors may influence relationship dissolution. Extending this line of prior work on network science, we view our effort to contribute to the software science aspect of software engineering research. By studying relationship dissolution, we can get a more complete insight about developer relationships on GitHub, and understand impacts of development activities on relationship dissolution. These insights can be beneficial in the construction of new tools to help GitHub community, just like those that have been built for Twitter and traditional social media, e.g., recommenda-

* Corresponding author.
E-mail addresses: jiangjing@buaa.edu.cn (J. Jiang), davidlo@smu.edu.sg (D. Lo), ayonel@qq.com (Y. Yang), powerfaster@163.com (J. Li), lily@buaa.edu.cn (L. Zhang).

tion service on who-to-follow, fine-tuned broadcasting of user updates, etc. Indeed, there have been some recent effort to specialize work done in social network community to cater for software engineering settings and needs [22–24]. Additionally, some developers are influenced by notifications from followees, and join OSS projects which their followees participate in [18]. It is important to understand unfollowing behavior as it can help us to gain insight into how to maintain current followers, influence followers to potentially attract them to become contributors to the OSS projects he/she participates in.

In this paper, we make a comprehensive analysis of unfollowing behavior based on a large number of developers. We investigate 701,364 developers, and collect their 2,182,845 following relationships, 4,045,101 following relationships, 4,292,558 following relationships and 4,602,440 following relationships in May 2013, August 2015, November 2015 and March 2016, respectively (Section 2). We study the percentage of active developers who ever unfollowed some users, and send a survey to understand some potential reasons for unfollowing behavior (Section 3). Following survey responses and previous works, we provide research hypotheses and corresponding impact factors for unfollowing behavior (Section 4). Following a previous work [5], we use one-wave snowball sampling to extract closely connected developer groups and build a sample of 63,311 developers. We crawl detailed activities of 63,311 developers, and analyze the correlation of various factors with the likelihood of unfollowing behavior (Section 5). In particular, our study aims to answer two key questions:

RQ1 Does unfollowing behavior substantially exist on GitHub?
RQ2 What factors are associated with unfollowing behavior?

Our study provides a number of insights into unfollowing behavior on GitHub which is its main contributions:

- Between May 2013 and August 2015, 19.8% of active developers ever unfollowed some users. Substantial unfollowing behavior exists on GitHub.
- We find that some factors are associated with unfollowing behavior: Developers are more likely to unfollow those who have fewer activities; Developers prefer to unfollow users who have lower programming language similarity; Mutual following relationships make relationships stronger with lower chance of dissolution.

2. Background and data collection

Before diving into detailed analysis of unfollowing behavior, we firstly provide background information about GitHub, and then we introduce data collection.

2.1. Unfollowing behavior on GitHub

GitHub is a web-based hosting service for software development repositories. Nowadays, GitHub plays an increasingly important role in OSS communities. More and more developers join GitHub, and develop some open source software projects.

GitHub integrates social media functionality with code management tools, and builds a social network to connect developers [16,25]. Users follow some developers (*followees*) to receive activity updates, discover new projects and trends, learn from developers, show respect or support, and make collaboration [17]. On GitHub, developers are free to unsubscribe and remove any users from their following lists. A developer's *unfollowees* are users who are unfollowed by this developer.

On GitHub, developers may perform different actions to contribute to a project. Developers may write issue reports to identify bugs, document software codes, or enhance the software by writing feature requests [26]. Developers fork repositories and make changes to implement new features or fix bugs [27]. Developers submit pull requests when they want to merge code changes into the main repository [28]. Core members commit satisfactory code changes into repositories [29].

Table 1
Number of following relationships.

	Collection time	# Following relationships
Snapshot 1	May 2013	2,182,845
Snapshot 2	Aug 2015	4,045,101
Snapshot 3	No. 2015	4,292,558
Snapshot 4	March 2016	4,602,440

GitHub offers two types of project owners, including personal account and organization.[1] A personal account is intended for an individual developer, while an organization is intended for a company or a non-profit organization, such as Google and Facebook. GitHub provides some mechanisms to simplify management of many projects in organizations. Existing user accounts can be converted from personal type to organization type. When such changes are made, all followers would be deleted since organization accounts cannot be followed on GitHub.

2.2. Data collection

GitHub provides access to its internal data store through an API.[2] It allows researchers to access a rich collection of information about developers and repositories. We collected four snapshots of GitHub through GitHub API, and these snapshots are used for our analysis.

We downloaded event datasets between January 25th, 2012 to May 15th, 2012 from GHTorrent, which provided a scalable and offline mirror of GitHub data [30]. We extracted developers from event datasets, and used them as seeds of crawlers. We proceeded to perform a breadth-first traversal of social graphs through GitHub API in May 2013. In total, we collected a total of 747,107 developers and their 2,234,845 following relationships, which was our first snapshot. We studied network structure of this snapshot in our prior work [18].

We collected additional snapshots in August 2015, November 2015 and March 2016. For the second to the fourth snapshots, we focus on the 747,107 developers who are in the first snapshot. We focus on these developers since we want to identify changes in following behaviors. Some developers closed their profiles and permanently left GitHub. These developers' social relationships also disappeared on GitHub. We removed these disappeared developers, because we could not collect their complete following relationships. A total of 45,743 developers disappeared, and we could track following relationships of the remaining 701,364 developers using GitHub API. We could collect following information for 93.88% of the 747,107 developers using GitHub API, while we could only collect following information for 83.23% of the 747,107 developers using GHTorrent. Data collection through GitHub API provided following information of more developers. Therefore, we decided to analyze following relationships collected through GitHub API.

We use 701,364 developers and their following relationships to build 4 networks. Each network has nodes corresponding to the 701,364 developers and edges corresponding to the following relationships between them. Table 1 shows the number of following relationships contained in the networks. The 701,364 developers have a total of 2,182,845 following relationships, 4,045,101 following relationships, 4,292,558 following relationships, and 4,602,440 following relationships in May 2013, August 2015, November 2015, and March 2016, respectively. The second snapshot was taken more than 2 years after the first one, and thus the number of following relationships is much larger than that of the first snapshot. The time interval between the subsequent snapshots are just a few months and thus we observe smaller differences in the number of following relationships.

[1] https://github.com/blog/674-introducing-organizations.
[2] http://developer.github.com/v3/.

Table 2
Number of new and deleted following relationships.

	Time	# New relationships	# Deleted relationships	Ratio
Period 1	2013/05–2015/08	2,148,668	286,412	7.5
Period 2	2015/08–2015/11	289,513	42,056	6.88
Period 3	2015/11–2016/03	357,817	47,935	7.46

Since GitHub's API did not provide historical events of developers, we directly visited developers' profiles to crawl their issues, pull requests and commits.[3] Developers' forks were obtained through GitHub API. We directly visited 701,364 developers' profiles to crawl the number of their contribution activities between snapshots. 701,364 developers had 144,681,788 contribution activities between May 2013 and August 2015. These developers had 18,138,103 and 18,332,738 contribution activities in period between August 2015 and November 2015, and in period between November 2015 and March 2016, respectively.

3. Basic analysis of unfollowing behavior

Relationship dissolution in online social networks has been analyzed for Twitter [1,4–6] and Facebook [7–9], which may help in the design of a variety of social-networking tools. The same reason applies for GitHub case too. By understanding relationship dissolution, we can get a more complete insight about developer relationships on GitHub, and provide suggestions for a variety of tools which can potentially improve collaborations among developers. For example, results may be used to distinguish strong and weak relationships, and improve new relationship recommender systems.

In this section, we describe basic analysis of unfollowing behavior. First, we make a quantitative analysis of unfollowing behavior, and report the percentage of active developers who ever unfollowed some users. Second, we send a survey to developers, and try to understand some reasons for unfollowing behavior.

3.1. Quantitative analysis

As described in Section 2.2, we collected following relationships of 701,364 developers and built 4 networks. We compare consecutive networks, and identify new relationships and deleted relationships. We compute the number of new relationships and deleted relationships in Table 2. The first period is between the time snapshot 1 (May 2013) and snapshot 2 (August 2015), which is a period of time spanning more than 2 years. During period 1, users create 2,148,668 following relationships, while 286,412 following relationships are deleted. It shows that users indeed stop following some developers and delete some following relationships. A total of 42,056 and 47,935 relationships are deleted in period 2 and 3. Column *Ratio* in Table 2 is computed by dividing the number of new relationships with the number of deleted relationships. The ratios are 7.5, 6.88 and 7.46 for period 1, 2, and 3, respectively. It shows that on average, 1 following relationship is deleted when about 7 new following relationships are created.

A developer is considered as active, if the developer has contribution activities (e.g., commits, pull requests or issues) in a period. For an active developer, we compute the number of deleted following relationships, divided by the number of following relationships at the start time of each period. This ratio describes the percentage of following relationships which exist at the start time of each period, but are deleted at the end of each period. Fig. 1 shows complementary cumulative distribution function (CCDF) of the percentage of deleted following relationships. In period 1, 19.8% of active developers have the percentage of deleted following relationships larger than 0. It means that 19.8% of active developers deleted some following relationships and unfollowed some users between May 2013 and August 2015. Moreover, 16.03% of active developers deleted more than 10% of their following relationships. The line for period 2 almost coincides with the line for period 3, because the 2 periods have similar results. In period 2, 6.04% of active developers have unfollowing behavior. Period 2 is much shorter than period 1, and thus has a smaller percentage of deleted following relationships. Results show that unfollowing behavior substantially exists on GitHub.

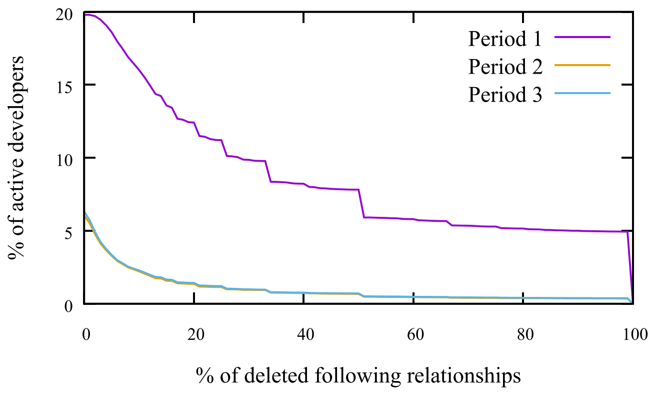

Fig. 1. Percentage of deleted following relationships.

RQ1: Unfollowing behavior substantially exists on GitHub. Between May 2013 and August 2015, 19.8% of active developers ever unfollowed some users. On average, 1 following relationship is deleted when about 7 new following relationships are created.

3.2. Survey analysis

In previous subsection, we find that some developers indeed unfollow users and stop receiving their updates. We send a survey to developers, and qualitatively analyze unfollowing behavior. We first try to understand some reasons for unfollowing behavior, and then study awareness of being unfollowed.

To get a perspective of why developers unfollow some users, we design a survey to includes 4 questions.

1. Could you kindly tell us some reasons for unfollowing users on GitHub?
2. Do you think that unfollowees are aware of being unfollowed?
3. Why do you think that unfollowees are aware of being unfollowed?
4. Why do you think that unfollowees are not aware of being unfollowed?

Questions 1, 3 and 4 are open-ended. We provide three choices for question 2, including Yes, No and Unsure. If a respondent chooses Yes in question 2, then we ask question 3; If a respondent chooses No in question 2, then we ask question 4.

We randomly selected 200 developers who ever unfollowed users in period 1 and provided email addresses on GitHub. 106,716 developers ever unfollowed users in period 1. We sent them emails, and asked the above questions. We received responses from 32 developers, and the survey response rate was 16%. 32 developers from a population of 106,716 developers yield a 95% confidence level with a 17.32% error margin.

The first question is about reasons for unfollowing behavior. The first author reads all the replies to understand reasons for unfollowing behavior. Based on this understanding, the first author builds categories for reasons why developers unfollow some users. The third and the fourth authors also independently read all 32 responses, and set up corresponding categories. Finally, the three authors compare their results and agree on the final set of categories.

[3] https://help.github.com/articles/viewing-contributions-on-your-profile-page/.

Table 3
Could you kindly tell us some reasons for unfollowing users on GitHub?

Reason for Unfollowing Behavior	Respondents
Feeling overwhelmed by too many notifications	12 / 37.5 %
Lost of interest	10 / 31.25 %
Receiving little information	6 / 18.75 %
Not working with the person anymore	4 / 12.5 %
Others	6 / 18.75 %

The first, third and fourth authors independently classify 32 responses into corresponding categories. A response is classified into a category immediately, if at least two authors make the same decision. If there are disagreements upon the classification, the three authors will discuss together to resolve conflicts and to make final classification. Some responses mention several reasons, and they are classified into multiple categories. Table 3 shows reasons for unfollowing behavior. From the table, we notice that:

1) 37.5% of respondents in survey mention that some users have too many activities, which result in too many notifications being sent to their followers. Developers receive a large number of notifications and feel overwhelmed. For example, a developer says that "When the user I follow has too many commits/statuses that floods my timeline, then I usually unfollow him/her". Another developer says that "Too much activity, making the history even less readable".

2) 31.25% of respondents mention that they unfollow some users because they lose interest of these users' projects. 3 respondents further explain reasons for losing interest. A respondent writes that "Not working in that direction anymore so not interested in those updates". Another respondent states that "they don't ship things I'm interested in or don't develop in languages I'm familiar with".

3) 6 respondents unfollow users who have few activities and seldom send notifications to their followers. Developers receive little information through following relationships. For instance, a developer writes that "inactive for a really long period". 12 respondents unfollow users with too many activities, while another 6 respondents unfollow users with too few activities. Different developers have various attitude towards their followees' activeness. In Section 4, we quantitatively analyze the correlation between users' activeness and being unfollowed.

4) 4 respondents unfollow some users since they no longer work together in the same organization/project anymore. For instance, a respondent says that "I've also unfollowed people who I only followed because I worked in the same organization with. If I no longer work with these people I don't find the need to follow them.".

5) 6 respondents mention other reasons. For example, a respondent unfollows everyone he/she knows only from the internet, rather than in real world. Another respondent unfollows users who do not follow him/her in return.

In the survey, respondents explain why they unfollow some users. We complement the survey with a historical data analysis in Sections 4 and 5 to more comprehensively understand reasons for unfollowing behavior.

In the survey, we also study the awareness of being unfollowed. In question 2, we ask whether a respondent thinks unfollowees are aware of being unfollowed. A developer's unfollowees are users who are unfollowed by this developer. We provide 3 choices for question 2, including Yes, No and Unsure. If a respondent chooses Yes, then we ask why the respondent thinks that unfollowees are aware of being unfollowed in question 3; If a respondent chooses No, then we ask a question: Why do you think that unfollowees are not aware of being unfollowed? (question 4).

Table 4 shows respondents' attitude towards unfollowees' awareness. 8 respondents choose the choice Unsure. Only 4 respondents state that unfollowees are aware of being unfollowed. In their answers to question 3, these respondents explain that unfollowees can check their profiles, and find that they are being unfollowed. Current followers and followees are shown in developers' profile. But GitHub does not provide historical information about followers and followees.

Table 4
Do you think that unfollowees are aware of being unfollowed?.

Choice	Respondents
Yes	4 / 12.5%
No	20 / 62.5%
Unsure	8 / 25%

Table 5
Why do you think that unfollowees are not aware of being unfollowed?.

Reason for unawareness	Respondents
Developers do not check their followers	9 / 45%
GitHub does not provide the notification	8 / 40%
Other	3 / 15%

62.5% of respondents think that unfollowees are not aware of being unfollowed. In their answers to question 4, they provide detailed reasons. We follow the same process that we describe in question 1. Three authors together set up categories, and classify responses into specific categories. Table 5 shows reasons for unfollowees' unawareness. From the table, we notice that: 9 respondents state that developers do not check their followers to notice who are following them and who are not. 8 respondents mention that GitHub does not notify developers when they are unfollowed. 3 respondents mention other reasons. For example, a respondent says that "We already get enough email. Who cares!". Another respondent thinks that one can successfully interact with another not being a follower.

4. Hypotheses and factors

In Section 3.2, 32 respondents in our survey describe why they unfollow some users. Their replies suggest some factors that may contribute to unfollowing behavior. Moreover, existing literature has investigated factors that may contribute to relationship dissolution in Twitter and Facebook [1,5,8]. In this section, we analyze GitHub historical data to provide a number of factors (e.g., factors identified by our survey respondents, and factors investigated in prior research on Twitter and Facebook), which may contribute to unfollowing behavior on GitHub.

We define the age of a developer to be the number of months that has passed between his/her registration and the beginning of our analysis period. The older the age is, the earlier was the account registered on GitHub. *Followee's age* is the age of a followee, and *follower's age* is the age of a follower. Control variables are predictors unrelated to the existing hypotheses but are still potentially related to the outcome. In this paper, we control for followee's age and follower's age.

4.1. Activeness

Previous work [17] observed that one main benefit of following others was to receive their activity updates. As shown in Table 3, respondents have different opinions about followees's activeness. 12 respondents unfollow users who have too many activities resulting in many notifications. However, 6 respondents unfollow users who have too few activities corresponding to few notifications. Activeness may be important in the decision of unfollowing users, but different developers have different perspectives. Based on this observation, we investigate the validity of the following hypotheses:

H1.1: Developers are more likely to unfollow those who have more activities.

H1.2: Developers are more likely to unfollow those who have fewer activities.

We consider 4 kinds of activities to measure developers' activeness: issues, pull requests, forks and commits. In OSS projects, developers write issue reports to identify bugs, and document feature requests [26]. Developers freely fork repositories, use existing codes as their own and make changes [27]. Developers submit pull requests when they want to merge code changes into main repositories [28]. Satisfactory codes are committed to main repositories [29]. For a developer who is followed, *activity amount* is the total number of activities corresponding to issues, pull requests, forks and commits in recent γ months.

Developers may perform different kinds of activities on GitHub. We consider issues, pull requests, forks and commits to measure developer activeness due to the following reasons. First, some respondents explicitly describe activities which flood their timelines in our survey. Commits are mentioned by 3 respondents, and forks are mentioned by 2 respondents. For example, a respondent says that "lots of commits lead to not seeing other project news on feed." Another respondent states that "He repeated fork projects. He make my dashboard dirty." Many commits or forks cause some developers to unfollow users, and thus we use them to measure the activeness. Second, issues and pull requests are important activities in the development of OSS projects. GitHub provides support for pull-based development, and pull requests allow developers to make contributions flexibly and efficiently [28,29,31,32]. Through issues, developers conveniently discuss bugs, feature requests and other things on GitHub [26].

4.2. Programming language similarity

People tend to associate with others having similar interests [33]. Kwak et al. (2011) report that in Twitter people unfollow users who tweet about topics that they deem uninteresting. In our survey, 31.25% of respondents mention that they unfollow users because they lose interest of these users' projects. Different interest may cause developers to unfollow some users.

The choice of programming languages reflects developers' preference towards some kinds of projects [34]. For example, Yacc is mainly used in the development of compilers, while PHP is applied in the development of web applications. Previous work finds that developers like to fork projects written in their preferred programming language [35]. In our survey, a respondent mentions that the change of programming language causes him to lose interest in some users. Another respondent states that "they don't ship things I'm interested in or don't develop in languages I'm familiar with". Therefore, we use programming language similarity as a metric of common interest between two developers. Developers may unfollow users who use different programming languages. This leads to the following hypothesis:

H2.1: Developers are more likely to unfollow those who have lower similarity of programming languages.

Before we define how similarity of programming languages can be measured, let us introduce some notations. Let us denote a developer as D_a. Let $PSet_a$ denote projects in which developer D_a submits issues, proposes pull requests, forks, or commits to in recent γ months. $PSet_a$ includes projects recently participated by the developer, and exclude those in which the developer participates a long time ago. For a project p_i ($p_i \in PSet_a$), let us denote its major programming language as $language_i$. GitHub API returns the major (i.e., primary) programming language for a project, and this is the language in which most source files are written. According to previous work [36], the major programming language is representative, and it is used in our analysis. For two developers D_a and D_b, we compute their *language similarity*, denoted as $LanguageSimilarity_{a,b}$, as follows:

$$LanguageSimilarity_{a,b} = \frac{\sum_{p_i \in PSet_a, p_j \in PSet_b} IsSameLanguage(p_i, p_j)}{|PSet_a| \times |PSet_b|} \quad (1)$$

If projects p_i and p_j have the same major programming language, the function $IsSameLanguage(p_i, p_j)$ returns 1; otherwise, the function $IsSameLanguage(p_i, p_j)$ returns 0. $|PSet_a|$ refers to the number of projects in $PSet_a$. For a project pair p_i ($p_i \in PSet_a$) and p_j ($p_j \in PSet_b$), we check whether their major programming languages are the same. Then we check each project pair belonging to developers D_a and D_b, and compute the average value. Language similarity will have a higher value, if two developers participate in projects with the same major programming languages.

4.3. Work collaboration

As shown in Table 3, 4 respondents state that they unfollow users whom they do not work together with anymore. Collaboration is the reason why some developers follow others, and their following relationships may disappear when they stop working together.

Work collaboration in open source software projects has different levels of meaning due to the hierarchical structure of artifacts. First, developers can be considered to work together when they work for projects belonging to same project owners. We consider project owner, because a respondent explicitly explains that he does not work for a particular organization anymore and unfollows some users in the organization. Organization is the project owner in this response. Organization is usually intended for a company or a non-profit organization on GitHub, such as Google and Facebook. Second, developers can be considered to collaborate when they work in same OSS projects [37]. In the following section, we define factors to measure work collaboration based on owners or projects.

Original project owner is used to measure work collaboration based on owners. Original project owner depends on the project type. If a project is created by its owner, its original project owner is the owner of this project. If a project is forked from another project, its original project owner is the owner of the project which it is forked from. When a developer forks a project, the project owner changes from its original owner to the developer. For example, a project *jamby/bootstrap* is forked from another project *twbs/bootstrap*. For the project *jamby/bootstrap*, its original project owner is *twbs*, rather than *jamby*.

For two developers D_a and D_b, we compute their *owner similarity*, denoted as $OwnerSimilarity_{a,b}$, as follows:

$$OwnerSimilarity_{a,b} = \frac{\sum_{p_i \in PSet_a, p_j \in PSet_b} IsSameOwner(p_i, p_j)}{|PSet_a| \times |PSet_b|} \quad (2)$$

If projects p_i and p_j have the same original project owner, the function $IsSameOwner(p_i, p_j)$ returns 1; otherwise, the function $IsSameOwner(p_i, p_j)$ returns 0.

Original project is used to measure work collaboration based on projects. If a project is forked from another project, its original project is the project which it is forked from; otherwise, its original project is itself. For example, a developer jamby works on a project *jamby/bootstrap*, and another developer gcbenjamin works on a project *gcbenjamin/bootstrap*. Though their projects are different, their original project is both *twbs/bootstrap*. These two developers modify codes on their own projects, and then submit pull requests to the same original project. The original project is the project to which developers really make contribution.

For two developers D_a and D_b, we compute their *project similarity*, denoted as $ProjectSimilarity_{a,b}$, as follows:

$$ProjectSimilarity_{a,b} = \frac{\sum_{p_i \in PSet_a, p_j \in PSet_b} IsSameProject(p_i, p_j)}{|PSet_a| \times |PSet_b|} \quad (3)$$

If projects p_i and p_j have the same original project, the function $IsSameProject(p_i, p_j)$ returns 1; otherwise, the function $IsSameProject(p_i, p_j)$ returns 0.

In the survey, 4 respondents unfollow some users whom they do not work together with anymore. We use the similarity of owners or projects to measure work collaboration, and give following hypotheses:

H3.1: Developers are more likely to unfollow those who have lower similarity of original project owners.

H3.2: Developers are more likely to unfollow those who have lower similarity of original projects.

4.4. Social relationship strength

Previous works explore reasons for unfollowing behavior in Twitter [1,5,6] and Facebook [8]. Xu et al. observe that strong social relationships between users reduce the likelihood of unfollowing behavior in Twitter [5]. GitHub is a social coding site, and it integrates social networks with code management tools [16]. In our datasets, 23.58% of following relationships are reciprocal. Strong social relationships between developers may reduce the likelihood of unfollowing behavior on GitHub.

We consider reciprocal following relations to measure social relationship strength. *Reciprocity* measures whether following relationships are reciprocal or not. For example, developer D_a follows another developer D_b. If developer D_b also follows developer D_a in return, then reciprocity is 1; otherwise, reciprocity is 0. An initial study shows that reciprocity creates stronger mutual ties, and increases the stability of following relations in Twitter [5]. A respondent in our survey unfollows users who do not follow him/her in return. On GitHub, mutually following relations may enable two developers to receive each other's updates, and further strengthen their relationship.

H4.1: Developers are less likely to unfollow those who follow them.

Initial study finds users with more common friends have stronger ties [38]. Therefore, we also use common followees to measure the social relationship strength. Common followees are developers who are followees of two developers. For example, $FSet_a$ includes all developers whom developer D_a follows, and $FSet_b$ includes followees of developer D_b. $FSet_a \bigcap FSet_b$ include common followees of developers D_a and D_b. *Common followee count* is the number of common followees of two developers.

H4.2: Developers are more likely to unfollow those with whom they share fewer common followees.

4.5. Social status

Social status shows the popularity and influence of an individual in a community. On GitHub, users with many followers are often considered more influential. For example, popular developers have a greater influence on their followers compared to regular developers [39]. Popular developers tend to have some special skill or knowledge. Other developers can learn from these popular developers by tracking how they are writing code, where they are giving their attention, and how they are addressing problems [16]. Therefore, developers may like to maintain relationships with users with many followers.

H5.1: Developers are more likely to unfollow those who have fewer followers.

Xu et al. observes that people with more followers are more likely to unfollow users in Twitter [5]. On GitHub, it may also be the case that high-status developers are more careful in managing their social relationships, and have a higher tendency to end unnecessary relationships.

H5.2: Developers with more followers are more likely to unfollow users.

On GitHub, users follow developers to receive activity updates, discover new projects and trends, learn from developers, and show respect or support [17]. Developers with more followees are more active, and they may be less likely to be unfollowed.

H5.3: Developers are more likely to unfollow those who follow fewer users.

Some active developers follow more users, and they may be more likely to find new users and unfollow current users.

H5.4: Developers with more followees are more likely to unfollow users.

The number of followers meas the number of developers who follow a user. Hypotheses H5.1 and H5.2 both consider the number of followers, but they analyze followers for followers and followees, respectively. The number of followees meas the number of developers who are followed by a user. Hypotheses H5.3 and H5.4 both consider the number of followees, but they analyze followees for followers and followees, respectively. For example, developer D_a follows another developer D_b. Developer D_a is the follower, and developer D_b is the followee. Related to H5.1, we define *followee's follower count* as the number of followers of a developer who is followed by another developer. Related to H5.2, we define *follower's follower count* as the number of followers of a developer who follows another developer. Related to H5.3, we define *followee's following count* as the number of followees of a developer who is followed by another developer. Related to H5.4, we define *follower's following count* as the number of followees of a developer who follows another developer. In the above example, follower's follower count is the number of followers of developer D_a, while followee's follower count is the number of followers of developer D_b. Follower's following count is the number of followees of developer D_a, while followee's following count is the number of followees of developer D_b.

5. Quantitative analysis and results

In Section 4, we list some research hypotheses about factors that may influence unfollow behavior. In this section, we use a logistic regression model, and analyze how these factors correlate with the unfollowing behavior. These quantitative results support some hypotheses and fail to support some others, shedding light to factors that significantly influence developers' decision to stop following some users.

5.1. Data collection and analysis setup

In order to analyze the correlation of various factors with the likelihood of unfollowing behavior, we mainly study developers who ever create or delete following relationships. Socially inactive developers may stop using social functions on GitHub, and their following relationships remain unchanged, which affects analysis results. Following a previous work [5], we used one-wave snowball sampling to extract closely connected developer groups. We randomly selected 5000 active users who ever created or deleted following relationships. Then we identified all their followees to build a sample of 63,311 developers. These 63,311 respondents from a population of 701,364 developers yield a 95% confidence level with a 0.37% error margin. We collected 63,311 developers' activities in issues, pull requests, commits and forks between February 2015 and November 2015. For 63,311 developers, we collected 15,989,032 commits, 607,292 pull requests, 584,545 issues, and 219,190 forks. Typically, a developer only forks a project once, but submits multiple commits, pull requests or issues. Therefore, the number of forks is much smaller than those of other activities.

Logistic regression is used to model binary outcome variables, in which the log odds of the outcomes are modeled as a linear combination of the predictor variables. It has been used in a number of studies. For example, Tsay et al. [32] used a logistic regression model to analyze factors influencing the acceptance of pull requests. Their outcome variable (acceptance) was dichotomous. Keep following or stop following is also a binary outcome variable. Previous works [4,6] used a logistic regression model to study factors influencing unfollowing behavior in Twitter. We also choose a logistic regression model, and analyze the correlation between various factors and the likelihood of unfollowing users. In order to ensure normality, every continuous variables in the model is log transformed and then centered such that the mean of each measure is 0 and standard deviation is 1. We compute Pearson's correlation between measures [40], and all of these measures have Pearson's correlation lower than 0.6, which suggests there is no multi-collinearity problem [32].

Table 6
Statistical results.

Variable	Odds ratio (Period 2)	Odds ratio (Period 3)
(Intercept)	2.13E−05 ***	1.31E−05 ***
Activity amount	0.65 ***	0.73 ***
Language similarity	0.91 *	0.76 ***
Owner similarity	0.93	0.86
Project similarity	1.03	0.91
Reciprocity	0.62 ***	0.33 ***
Common followee count	1.03	0.91
Followee's follower count	1.52 ***	1.21 **
Follower's follower count	1.37 *	1.46
Followee's following count	1.05 **	0.94
Follower's following count	0.13	0.38 **
Followee's age	0.68 ***	0.83 ***
Follower's age	1.03	0.88

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

Based on above datasets, we compare snapshots and judge whether following relationships in the snapshot of August 2015 are deleted or maintained in the snapshot of November 2015. Then we use logistic regression model to study unfollowing behavior in period 2 (between August 2015 and November 2015). Our dependent variable is whether a developer unfollows a user. If a developer unfollows a user, the value of the dependent variable is 1; otherwise, the value of the dependent variable is 0. Independent variables are factors described in Section 4. Activity amount, language similarity, owner similarity and project similarity are related to developers' activities in recent γ months. We set the temporal window length $\gamma = 3$ by default, and discuss impact of various settings of γ in Section 5.4. Since we do not know exact time of unfollowing behavior, we study developers' activities within the last $\gamma = 3$ months before the beginning of period 2, namely August 2015. It ensures that activities occur before relationship dissolution. We repeat the analysis for period 3 and check whether any finding we find for period 2 holds for period 3 too. We do not study period 1 between May 2013 and August 2015, because the time interval is too long.

5.2. Analysis results

Table 6 summarizes statistical results for unfollowing behavior in period 2 and period 3. We characterize the correlation between a factor and unfollowing behavior in terms of odds ratio, which is the increase or decrease of the odds of unfollowing per unit change of a particular factor. For example, an odds ratio of 0.6 suggests that as the value of a factor (aka. a variable) increases by one unit, the odds of unfollowing decreases by 40%; an odds ratio of 1.2 suggests that as the value of a factor increases by one unit, the odds of unfollowing increases by 20%.

5.2.1. Activeness

H1.1: Developers are more likely to unfollow those who have more activities.

H1.2: Developers are more likely to unfollow those who have fewer activities.

We test H1.1 and H1.2 by examining the correlation of activity amount with the likelihood of unfollowing behavior. In Table 6, the odds ratio of activity amount is 0.65 in period 2. The activity amount is negatively associated with the likelihood of unfollowing behavior, and each unit of activity amount decreases the odds of unfollowing by 35% in period 2. A similar finding can be made for period 3. Though respondents in the survey have different views, statistical results show that in general developers with fewer activities are more likely to be unfollowed. We find support for H1.2 and lack of support for H1.1.

5.2.2. Programming language similarity

H2.1: Developers are more likely to unfollow those who have lower similarity of programming languages.

In order to test H2.1, we examine the correlation between programming language similarity and the likelihood of unfollowing behavior. The likelihood of unfollowing decreases by 9% with an increase of each unit of programming language similarity in period 2. The likelihood of unfollowing also decreases as programming language similarity increases in period 3. Developers prefer to unfollow users who have lower programming language similarity, which supports H2.1.

5.2.3. Work collaboration

H3.1: Developers are more likely to unfollow those who have lower similarity of original project owners.

H3.2: Developers are more likely to unfollow those who have lower similarity of original projects.

Table 6 shows that p-values for owner similarity are larger than 0.05 for both period 2 and 3. Thus there is no statistically significant correlation between owner similarity and unfollowing behavior. Therefore, H3.1 is not supported. P-values for project similarity are also larger than 0.05. There is no statistically significance that project similarity is correlated to the likelihood of unfollowing behavior. H3.2 is not supported.

5.2.4. Social relationship strength

H4.1: Developers are less likely to unfollow those who follow them.

H4.2: Developers are more likely to unfollow those with whom they share fewer common followees.

The odds ratio values of the models shown in Table 6 highlight that reciprocal relationships decrease the odds of unfollowing by 38% in period 2. Period 3 has similar results. Thus, there is a statistically significant evidence supporting that mutual following relationships make relationships stronger with lower chance of dissolution. Results support H4.1.

P-value for common followee count is larger than 0.05 in period 2 and period 3. People prefer to unfollow users who share fewer common followees in Twitter [5]. However, our results do not support that the number of common followees have a negative impact on unfollowing on GitHub.

5.2.5. Social status

H5.1: Developers are more likely to unfollow those who have fewer followers.

H5.2: Developers with more followers are more likely to unfollow users.

H5.3: Developers are more likely to unfollow those who follow fewer users.

H5.4: Developers with more followees are more likely to unfollow users.

In Table 6, p-values of followee's follower count are smaller than 0.05, and thus results are statistically significant. However, odds ratios are 1.52 and 1.21 in period 2 and period 3, and followee's follower count has a positive impact on unfollowing, H5.1 is rejected, and developers are more likely to unfollow those who have more followers. Users with more followers are less likely to be unfollowed in Twitter [5], while developers with more followers are more likely to be unfollowed on GitHub. We find that some popular users change from personal accounts to organizations, and automatically lose all followers. On GitHub, only personal accounts have followers, and organizations cannot be followed by developers [17]. In Section 5.3, we discuss how users with account type changes affect statistical results.

We contact GitHub and describe the problem of losing followers in account type changes. GitHub Support team replies us that "It's not currently possible to follow organizations, so an account would lose its followers if it was converted from a user to an organization. There's currently no way for organizations to send messages to interested developers in the way you're describing, but I can definitely add your +1 to our internal feature request list for the ability to follow organizations. I can't promise if or when it will be implemented, but your suggestion is definitely in the right hands!" GitHub support team has acknowledged this issue and added it to their internal feature request list.

Table 7
Statistical results without users who change account type.

Variable	Odds ratio (Period 2)	Odds ratio (Period 3)
(Intercept)	1.51E−05 ***	1.16E−05 ***
Activity amount	0.68 ***	0.69 ***
Language similarity	0.77 ***	0.70 ***
Owner similarity	0.94	0.86
Project similarity	1.04	0.92
Reciprocity	0.49 ***	0.26 ***
Common followee count	1.27 ***	1.12
Followee's follower count	1.52	1.26
Follower's follower count	1.47 **	1.51
Followee's following count	1.05 **	1.04
Follower's following count	0.13	0.38 **
Followee's age	0.73 ***	0.87 **
Follower's age	0.99	0.80

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

The odds ratio value of follower's follower count is 1.37 in period 2. It shows that the follower's follower count has a positive impact on unfollowing on GitHub. P-value of follower's follower count is larger than 0.05 in period 3, and there is no statistically significance. H5.2 is accepted for period 2, but we have an inconclusive result for period 3.

Table 6 shows that followee's following count has a positive impact on unfollowing behavior for period 2. P-value for followee's following count is smaller than 0.05 for period 2, but it is not the case for period 3. This indicates that there is a statistically significant correlation between followees following count and unfollowing behavior for period 2 but it is not the case for period 3. Thus, H5.3 is rejected for period 2, but we have an inconclusive result for period 3.

P-value for follower's following count is larger than 0.05 for period 2, and thus we have an inconclusive result for period 2. The odds ratio value of follower's following count is 0.38 in period 3 and the P-value is less than 0.01. Thus, H5.4 is rejected for period 3 but we have an inconclusive result for period 2.

5.3. *Impacts of account type changes*

In Section 5.2.5, we observe that some popular users change from personal accounts to organizations, and automatically lose all followers. In this subsection, we omit users who change account types, rerun experiments, and describe results in Table 7. We compare Tables 6 and 7, and analyze result difference with and without users who change account type. In Tables 6 and 7, activity amount, language similarity, and reciprocity are all negatively correlated with the likelihood of unfollowing behavior. Results of followee's follower count are statistically significant in Table 6, but results of followee's follower count are not statistically significant in Table 7. Omitting users with account type changes affects results of followee's follower count. It further proves that account type changes cause the positive correlation between followee's follower count and the likelihood of unfollowing behavior. When we remove users with type account changes, this positive correlation is not supported by results.

5.4. *Effect of temporal window length*

The temporal window length γ is used in computing activity amount, language similarity, owner similarity and project similarity. By default, we set the temporal window length γ as 3. In this subsection, we investigate the effect of temporal window length γ on statistical results. We increase γ values from 1 to 5 with an interval of 1, and compare statistical results for the different γ values. Results for period 2 and period 3 are similar, and thus we only plot results for period 2.

Table 8 shows statistical results with different temporal window length γ in period 2. For different values of γ, p-values of activity amount are all smaller than 0.001, and odds ratios are between

Table 8
Statistical results with different temporal window length γ for period 2.

Variable	Odds ratio $\gamma = 1$	Odds ratio $\gamma = 2$	Odds ratio $\gamma = 3$
(Intercept)	2.07E−05 ***	2.06E−05 ***	2.13E-05 ***
Activity amount	0.71 ***	0.67 ***	0.65 ***
Language similarity	0.85 **	0.89 *	0.91 *
Owner similarity	0.9	0.91	0.93
Project similarity	1.09 *	1.06	1.03
Reciprocity	0.56 ***	0.56 ***	0.62 ***
Common followee count	0.87 *	0.86	1.03
Followee's follower count	1.38 ***	1.42 ***	1.52 ***
Follower's follower count	1.37 *	1.37 *	1.37 *
Followee's following count	1.3 ***	1.31 ***	1.05 **
Follower's following count	0.13	0.13	0.13
Followee's age	0.65 ***	0.65 ***	0.68 ***
Follower's age	1.04	1.04	1.03
Variable	**Odds ratio** $\gamma = 4$	**Odds ratio** $\gamma = 5$	
(Intercept)	1.98−E05 ***	1.95−E05 ***	
Activity amount	0.6 ***	0.59 ***	
Language similarity	0.88 *	0.91 *	
Owner similarity	0.93	0.91	
Project similarity	1.04	1.03	
Reciprocity	0.56 ***	0.55 ***	
Common followee count	0.86 *	0.86	
Followee's follower count	1.51 ***	1.53 ***	
Follower's follower count	1.38 *	1.38 *	
Followee's following count	1.35 ***	1.36 ***	
Follower's following count	0.13	0.14	
Followee's age	0.65 ***	0.65 ***	
Follower's age	1.04	1.04	

$^{***}p < 0.001$, $^{**}p < 0.01$, $^{*}p < 0.05$.

0.59 and 0.71. For all γ values considered, we find that developers with fewer activities are more likely to be unfollowed. For different values of γ, activity amount, language similarity, and reciprocity are negatively correlated with the likelihood of unfollowing behavior. Followee's follower count, follower's follower count and followee's following count are positively correlated with the likelihood of unfollowing behavior. Therefore, we set the temporal window length γ as 3 by default, which does not affect statistical results.

RQ2: Developers are more likely to unfollow those who have fewer activities, lower programming language similarity, and asymmetric relationships.

6. Implications

Automatically receiving notifications from followees is an important mechanism for information transparency across developer social networks [16]. Previous work [18] found that some developers were influenced by notifications from followees, and joined OSS projects which their followees participated in. In this work, we analyze how developers stop following developers and cease to receive notifications from followees. Future works may explore how developers manage their notifications by unfollowing users, and study impacts of unfollowing behavior on developers. For example, a targeted survey or interview can be conducted to ask individual developers how the notifications that they receive and their activities change after unfollowing users.

Some previous works analyzed network structure and studied relationship formation in social coding sites [19,20]. In this study, results of RQ1 shows that substantial unfollowing behavior exists on GitHub. It implies that relationship dissolution is the basic process of relationship change and evolution in social coding sites. Researchers may study co-evolution of development activities (e.g., commits, pull requests, issues, etc.) and relationship dissolution. Future works may explore whether users reduce contributions, and submit fewer pull requests or issues, after they unfollow core developers of these projects.

Previous works studied social networks' effects on pull request evaluation [32] and project dissemination [18]. Tsay et al. found that developers used both technical and social information when evaluating potential contributions to open source software projects. In previous works [18,32], following relationships were identified at the time of data collection. In this work, we observe that some developers delete following relationships and unfollow some users on GitHub. Prior research ignores following relationships which *ever exist* between developers but break up later. These historical links may add value in estimating the real effect of social networks; the effect may be more significant than results reported in previous works [18,32]. Future researchers may continuously collect following relationships, obtain complete developer relationships in a period, and analyze how current and historical social networks influence project development.

Lim et al. proposed solutions that could revive dormant ties in an online social network [41]. Reviving such ties can make a social network more active and contribute to its health [42]. In this work, we highlight that relationship dissolution often happens in GitHub. This points to yet another avenue for future work in the development of automated techniques and intervention mechanisms that can ensure the health of GitHub overall network or its subnetwork by preventing relationship dissolutions and even reviving severed ties.

According to relationship dissolution in Twitter and Facebook [1,4–9], our results may help in the design of a variety of tools, which improve social collaborations among developers. For example, in our investigation of RQ2, we find that developers are more likely to unfollow those with fewer activities and lower programming language similarity. Based on this finding, if GitHub introduces a new service to recommend accounts to follow, our results suggest that that system needs to be designed with a preference for active developers that share similar programming languages with the target account for whom the recommendation is given.

Many OSS projects rely on contributors to fix bugs and contribute new features, and having a large pool of potential contributors is important to the health of these projects [17]. Maintaining current followers helps projects to attract new contributors. Results of RQ2 show that on GitHub, some popular users change from personal accounts to organizations, and automatically lose all followers. In future work, researchers may further explore how changing account type and losing followers affect projects ability in maintaining current contributors and attracting new contributors. Future studies may study whether changing from personal accounts to organizations benefits projects or not.

On GitHub, the number of followers is an important indicator of social influence on GitHub [39]. Developers with more followers have a greater influence in the dissemination of OSS projects. Our results provide insights for developers to maintain current following relationships. First, our investigation of RQ2 finds that developers with fewer activities are more likely to be unfollowed. In order to attract current followers, developers should be active in OSS project development and have updates corresponding to new issues, pull requests, forks, or commits. Second, results of RQ2 show that developers with lower programming language similarity are more likely to be unfollowed. Some followers are interested in projects written in their familiar programming languages. It suggests that if developers change their main programming languages, they may lose some followers who are not interested in projects written in the new programming languages. Third, our investigation of RQ2 finds that developers are less likely to unfollow those who follow them. Mutual following relationships indeed make relationships stronger and more cohesive. If developers want to maintain relationships with some specific users, it would be better that they follow these users in return, and receive these users' updates.

7. Threats to validity

Threats to internal validity relate to experimenter biases and errors. First, we send the survey to understand some reasons for unfollowing behavior. We receive responses from 32 developers. Their inputs are used as initial hypotheses. 32 developers from a population of 106,716 developers yield a 95% confidence level with a 17.32% error margin. To reduce bias due to the limited number of respondents, we analyze a large amount of historical data on GitHub to see if there is a statistical evidence to support the hypotheses. Second, we do not know the exact time of unfollowing behavior. We only know whether developers maintain or delete relationships in a time period. Therefore, we compute the number of deleted following relationships for periods 1, 2, 3. In future work, we will try to collect developers' following relationships every day, obtain the exact day of unfollowing behavior, and study the percentage of deleted following relationships for developers who are active within several months before a specific time. Furthermore, we study developers' activities within the last γ months before the beginning of a period. We do not consider developers' activities which occur between the beginning of a period and exact time of relationship dissolution. We plan to collect the exact day of unfollowing behavior, and study developers' activities within the last γ months before the exact day of unfollowing behavior.

Threats to external validity relate to the generalizability of our study. We analyze following relationships of 701,364 developers and activities of 63,311 developers. We believe that these large numbers can mitigate the threat to external validity. Still, we do not analyze all developers on GitHub, and we ignore developers outside GitHub. It is not clear if our results will generalize to other open source hosting site (e.g., BitBucket). In the future, we plan to study a similar set of research questions using data from other open source hosting site, and compare the results with the results that we find for GitHub.

Threats to conclusion validity relate to issues that affect the ability to draw the correct conclusion. The most probable conclusion validity threat in our work is due to the analysis of the survey. We manually analyze 32 replies, set up categories, and classify reasons about unfollowing behavior. We must admit that this process is a subjective one. In order to reduce human errors, three authors analyze replies, and independently build categories and make classification.

Threats to construct validity relate to the degree to which the construct being studied is affected by experiment settings. First, a frequently observed threat to a survey-based analysis is that designed questions may misguide responders. We have tried to reduce bias, e.g., by introducing the option Unsure. Second, we define some factors to quantitatively measure potential reasons mentioned by respondents. There may be other measures. For example, some respondents mention that they unfollow users because they lose interest of these users' projects. In this work, we measure developer interest by programming language similarity; other measures can be used instead of programming language similarity. In future work, we will investigate other measures to uncover other reasons for unfollowing behavior. Third, we have not studied all potential factors for relationship dissolution, including those that can be used as additional independent and control variables. In future work, we will study more potential factors. For example, we plan to investigate impact of workload issues (which can be detected by various automated tools [43,44]) on unfollowing behavior. Fourth, we only study developers' activities on GitHub. According to a previous work [45], active projects may not conduct all their software development activities on GitHub. Some projects may use their own defect tracking systems or email systems, which are not recorded by GitHub. Unfollowing behavior may be influenced by activities which are not recorded on GitHub. In future work, we will try to collect more information and analyze their impacts on unfollowing behavior.

8. Related work

Following behavior. On GitHub, users follow developers and they are also followed by others. Previous work [17] studied reasons of following behavior. They found that main reasons of following behavior included receiving activity updates, discovering new projects and trends, learning from developers, showing respect or support, and making collaboration. They also found that as the number of followers increased, developers influenced more followers to join projects and make contributions. Our work supplements this work and explores reasons of unfollowing behavior. We find that main reasons of unfollowing behavior include few activities, low programming language similarity, and asymmetric relationships.

Some previous works analyzed following network structure in social coding sites. Lima et al. found that distributions of degree, in-degree and out-degree of social networks exhibit a power-law scaling behavior on GitHub [19]. Yu et al. identified independence-patterns and group-patterns in following networks [20]. Blincoe et al. found that project owners in an ecosystem tended to follow the owner of the central repository [21]. Different from these previous works, our work studies unfollowing behavior and major factors which affect relationship dissolution.

Some researchers studied relationships between following behavior and software development. Badashian et al. found that as developers became increasingly engaged with the GitHub platform, i.e., joining more projects, committing more code, and contributing to more issues, they accrued more followers [46]. Lee et al. found that rockstars with thousands of followers had a greater influence on their followers compared to regular developers [39]. Tsay et al. found that developers used both technical and social information in the evaluation of potential contributions [32]. Jiang et al. observed that social relationships disseminated OSS projects to some followers, and successfully transformed some followers to contributors [18]. We study relationships between unfollow behavior and software development. We find that developers are more likely to unfollow those who have fewer activities, lower programming language similarity, and asymmetric relationships.

Relationship dissolution. Some researchers studied relationship dissolution in Twitter and Facebook [1,4–9]. Kwak et al. found that 43% of active users unfollowed at least 1 person during 51 days [1]. Swaine et al. also observed that the tie break phenomenon was significant in Twitter [6]. Xu et al. found that mutual following relations and common followees reduced the likelihood of unfollowing [5]. Sibona et al. found that both online and offline interactions were related to relationship dissolution in Facebook [8]. Sibona et al. also discovered that common emotional responses to being unfriended included surprise, bothered, amusement and sadness in Facebook [9]. The above mentioned works investigate unfollowing behavior in Twitter and Facebook. Different from their works, we study unfollowing behavior in a social coding site GitHub, and analyze impacts of developer activities on relationship dissolution.

9. Conclusion

Some developers unfollow users and stop receiving their updates. This paper investigates unfollowing behavior on GitHub which to the best of our knowledge has not been studied before. First, we analyze the existence of relationship dissolution on GitHub. Results show that 19.8% of active developers ever unfollowed some users between May 2013 and August 2015. Next, we send the survey to understand some potential reasons for unfollowing behavior, and then use a logistic regression model to analyze the correlation of various factors with the likelihood of unfollowing behavior. Results show that developers are more likely to unfollow those who have fewer activities, lower programming language similarity, and asymmetric relationships.

Acknowledgment

This work is supported by the National Key Research and Development Program of China No. 2018YFB1004202, National Natural Science Foundation of China under Grant No. 61672078 and the State Key Laboratory of Software Development Environment under Grant No.SKLSDE-2017ZX-20.

References

[1] H. Kwak, H. Chun, S. Moon, Fragile online relationship: a first look at unfollow dynamics in twitter, in: Proc. of CHI, Vancouver, Canada, 2011.
[2] J.A. Feeney, P. Noller, Attachment style and romantic love: relationship dissolution, Aust. J. Psychol. 44 (2) (1992) 69C74.
[3] R.A. Owens, Friendship Features Associated wtih College Students Friendship Maintenance and Dissolution Following Problems, Ph.D. thesis, West Virginia University, 2003.
[4] H. Kwak, S. Moon, W. Lee, More of a receiver than a giver: why do people unfollow in twitter? in: Proc. of ICWSM, Dublin, Ireland, 2012.
[5] B. Xu, Y. Huang, H. Kwak, N.S. Contractor, Structures of broken ties: exploring unfollow behavior on twitter, in: Proc. of CSCW, San Antonio, USA, 2013.
[6] F. Kivran-Swaine, P. Govindan, M. Naaman, The impact of network structure on breaking ties in online social networks: unfollowing on twitter, in: Proc. of CHI, Vancouver, Canada, 2011.
[7] D. Quercia, M. Bodaghi, J. Crowcroft, Loosing friends on facebook, in: Proc. of WebSci, Illinois, USA, 2012.
[8] C. Sibona, S. Walczak, Unfriending on facebook: friend request and online/offline behavior analysis, in: Proc. of HICSS, Hawaii, USA, 2011.
[9] C. Sibona, Facebook fallout: the emotional response to being unfriended on facebook, in: Proc. of HICSS, Hawaii, USA, 2014.
[10] W. Xie, C. Li, F. Zhu, E.-P. Lim, X. Gong, When a friend in twitter is a friend in life, in: Proc. of WebSci, Illinois, USA, 2012, pp. 344–347.
[11] C. Wilson, B. Boe, A. Sala, K.P.N. Puttaswamy, B.Y. Zhao, User interactions in social networks and their implications, in: Proc. of EuroSys, Nuremberg, Germany, 2009, pp. 205–218.
[12] J. Jiang, C. Wilson, X. Wang, P. Huang, W. Sha, Y. Dai, B.Y. Zhao, Understanding latent interactions in online social networks, in: Proc. of IMC, Melbourne, Australia, 2010, pp. 369–382.
[13] S. Huang, J. Zhang, L. Wang, X.-S. Hua, Social friend recommendation based on multiple network correlation, IEEE Trans. Multimedia 18 (2) (2016) 287–299.
[14] Z. Yu, C. Wang, J. Bu, X. Wang, Y. Wu, C. Chen, Friend recommendation with content spread enhancement in social networks, Inf. Sci. 309 (2015) 102C118.
[15] Y. Sasaki, D. Kawai, S. Kitamura, The anatomy of tweet overload: how number of tweets received, number of friends, and egocentric network density affect perceived information overload, Telematics Inf. 32 (2015) 853C861.
[16] L. Dabbish, C. Stuart, J. Herbsleb, Social coding in github: transparency and collaboration in an open software repository, in: Proc. of CSCW, Washington, USA, 2012.
[17] K. Blincoe, J. Sheoran, S. Goggins, E. Petakovic, D. Damian, Understanding the popular users following, affiliation influence and leadership on github, Inf. Softw. Technol. 70 (2016) 30–39.
[18] J. Jiang, L. Zhang, L. Li, Understanding project dissemination on a social coding site, in: Proc. of WCRE, Koblenz, Germany, 2013.
[19] A. Lima, L. Rossi, M. Musolesi, Coding together at scale: github as a collaborative social network, in: Proc. of AAAI, Qubec, Canada, 2014.
[20] Y. Yu, G. Yin, H. Wang, T. Wang, Exploring the patterns of social behavior in github, in: Proc. of CrowdSoft, Hong Kong, China, 2014.
[21] K. Blincoe, F. Harrison, D. Damian, Ecosystems in github and a method for ecosystem identification using reference coupling, in: Proc. of MSR, Florence, Italy, 2015.
[22] A. Sharma, Y. Tian, A. Sulistya, D. Lo, Recommending who to follow in the software engineering twitter space, ACM Trans. Softw. Eng.Methodol., Accepted.
[23] G. Bougie, J. Starke, M.-A. Storey, D.M. German, Towards understanding twitter use in software engineering: preliminary findings, ongoing challenges and future questions, in: Proc. of Web2SE, Honolulu, USA, 2011, pp. 31–36.
[24] A. Sharma, Y. Tian, A. Sulistya, D. Lo, A.F. Yamashita, Harnessing twitter to support serendipitous learning of developers, in: Proc. of Saner, Campobasso, Italy, 2017, pp. 387–391.
[25] V. Cosentino, J. Luis, J. Cabot, Findings from github: methods, datasets and limitations, in: Proc. of MSR, Austin, USA, 2016.
[26] T.F. Bissyande, D. Lo, L. Jiang, L. Reveillere, J. Klein, Y.L. Traon, Got issues? Who cares about it? A large scale investigation of issue trackers from github, in: Proc. of ISSRE, Washington DC, USA, 2013.
[27] K.H. Fung, A. Aurum, D. Tang, Social forking in open source software: an empirical study, CAiSE Forum, Poland, 2012.
[28] G. Gousios, M. Pinzger, A. van Deursen, An exploratory study of the pull-based software development model, in: Proc. of ICSE, Hyderabad, India, 2014.
[29] B. Vasilescu, Y. Yu, H. Wang, P. Devanbu, V. Filkov, Quality and productivity outcomes relating to continuous integration in github, in: Proc. of FSE, Bergamo, Italy, 2015.
[30] G. Gousios, D. Spinellis, Ghtorrent: githubs data from a firehose, in: Proc. of MSR, Zurich, Switzerland, 2012.
[31] G. Gousios, A. Zaidman, M.-A. Storey, A. van Deursen, Work practices and challenges in pull-based development: the integrator perspective, in: Proc. of ICSE, Florence, Italy, 2015.

[32] J. Tsay, L. Dabbish, J. Herbsleb, Influence of social and technical factors for evaluating contribution in github, in: Proc. of ICSE, Hyderabad, India, 2014.
[33] J.L. Martin, K.-T. Yeung, Persistence of close personal ties over a 12-year period, Social Netw. 28 (4) (2006) 331C362.
[34] T.F. Bissyande, F. Thung, D. Lo, L. Jiang, L. Reveillere, Popularity, interoperability, and impact of programming languages in 100,000 open source projects, in: Proc. of COMPSAC, Kyoto, Japan, 2013.
[35] J. Jiang, D. Lo, J. He, X. Xia, P.S. Kochhar, L. Zhang, Why and how developers fork what from whom in github, Empirical Softw. Eng. 22 (2017) 547–578.
[36] B. Ray, D. Posnett, V. Filkov, P.T. Devanbu, A large scale study of programming languages and code quality in github, in: Proc. of FSE, Hong Kong, China, 2014.
[37] M. Ohira, N. Ohsugi, T. Ohoka, K. ichi Matsumoto, Accelerating cross-project knowledge collaboration using collaborative filtering and social networks, ACM SIGSOFT Softw. Eng. Notes 30 (4) (2005) 1–5.
[38] J.-P. Onnela, J. Saramaki, J. Hyvonen, G. Szabo, D. Lazer, K. Kaski, J. Kertesz, A.-L. Barabasi, Structure and tie strengths in mobile communication networks, PNAS 104 (18) (2007) 7332–7336.
[39] M.J. Lee, B. Ferwerda, J. Choi, J. Hahn, J.Y. Moon, J. Kim, Github developers use rockstars to overcome overflow of news, in: Proc. of CHI EA, Paris, France, 2013.
[40] P. Sedgwick, Pearson's correlation coefficient, Br. Med. J. 345 (2012) 1–2.
[41] E.-P. Lim, D. Correa, D. Lo, M. Finegold, F. Zhu, Reviving dormant ties in an online social network experiment, in: Proc. of ICWSM, Boston, USA, 2013, pp. 361–369.
[42] M. Gao, E.-P. Lim, D. Lo, R-energy for evaluating robustness of dynamic networks, in: Proc. of WebSci, Paris, France, 2013.
[43] T. Wang, J. We, W. Zhang, H. Zhong, T. Huang, Workload-aware anomaly detection for web applications, J. Syst. Softw. 89 (2014) 19–32.
[44] T. Wang, W. Zhang, C. Ye, J. Wei, H. Zhong, T. Huang, Fd4c: automatic fault diagnosis framework for web applications in cloud computing, IEEE Trans. Syst. Man Cybern. 46 (1) (2016) 61–75.
[45] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D.M. German, D. Damian, The promises and perils of mining github, in: Proc. of MSR, Hyderabad, India, 2014.
[46] A.S. Badashian, A. Esteki, A. Gholipour, A. Hindle, E. Stroulia, Involvement, contribution and influence in github and stack overflow, in: Proc. of CASCON, Markham, Canada, 2014.