Expert Systems With Applications 108 (2018) 108–118

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

# User influence analysis for Github developer social networks

Yan Hu[a,b], Shanshan Wang[a], Yizhi Ren[b], Kim-Kwang Raymond Choo[c,*]

[a] *Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, School of Software, Dalian University of Technology, Dalian, China*
[b] *School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China*
[c] *Department of Information Systems and Cyber Security, The University of Texas at San Antonio, San Antonio, TX 78249-0631, USA*

A R T I C L E   I N F O

*Article history:*
Received 22 November 2017
Revised 18 April 2018
Accepted 3 May 2018
Available online 4 May 2018

*Keywords:*
Distributed social coding
Developer social network
Github
Github social influence analysis
User influence
Following-Star-Fork-Activity

A B S T R A C T

Github, one of the largest social coding platforms, offers software developers the opportunity to engage in social activities relating to software development and to store or share their codes/projects with the wider community using the repositories. Analysis of data representing the social interactions of Github users can reveal a number of interesting features. In this paper, we analyze the data to understand user social influence on the platform. Specifically, we propose a Following-Star-Fork-Activity based approach to measure user influence in the Github developer social network. We first preprocess the Github data, and construct the social network. Then, we analyze user influence in the social network, in terms of popularity, centrality, content value, contribution and activity. Finally, we analyze the correlation of different user influence measures, and use Borda Count to comprehensively quantify user influence and verify the results.

## 1. Introduction

Distributed social coding is a recent trend in software development, which is partly made possible due to rapid technological advances that allow the hosting, maintaining, and accessing of (big) data in real-time. There are a number of developer social network platforms, and popular platforms include Github,[1] Stackoverflow,[2] and Altassian BitBucket,[3] where users (software developers) sign up for an account, interact and share information with other users (e.g. contributing to common software projects or repositories, and responding to questions posed on such platforms).

Github is one of the most popular social network platforms, perhaps due to the features and functionality available on its development management tool Git.[4] Using Git, users can easily search through the massive amounts of code, fork code from other users, and create branches for projects. Due to its popularity, Github has reportedly attracted tens of millions of developers and projects (Riihonen, 2017). The amount of social development activities results in large amount of data that can be mined for social and collaborative features (Hebig, Ho-Quang, Chaudron, Robles, & Fernandez, 2016; Saxena & Pedanekar, 2017). Not surprisingly, the research community has started mining data from Github, focusing on various aspects such as the structure and collaboration of social coding, code quality, programming languages used, and the types of software development undertakings (Avelino, Valente, & Hora, 2015; Casalnuovo, Vasilescu, Devanbu, & Filkov, 2015; Dabbish, Stuart, Tsay, & Herbsleb, 2012; Hebig et al., 2016; Kalliamvakou et al., 2014; Lima, Rossi, & Musolesi, 2014; Ray, Posnett, Filkov, & Devanbu, 2014; Saxena & Pedanekar, 2017; Thung, Bissyande, Lo, & Jiang, 2013).

Understanding the scope and magnitude of the influence of individuals in such social platforms is important for value-added services (Badashian & Stroulia, 2016; Cha, Haddadi, Benevenuto, & Gummadi, 2010a). For example, in marketing, researchers are interested in identifying the individual who is most capable of disseminating an idea widely in the network (i.e. identify the social network influencer) (Dubois & Gaffney, 2014). This also has applications in the dissemination of information such as opinion propagation and viral marketing (Katsimpras, Vogiatzis, & Paliouras, 2015; Peng, Wang, & Xie, 2017; Zhang, Zhao, & Xu, 2016), as influential individuals, particularly those not connected to the particular business, will be more convincing with their peers, target population and/or the general public (Aziz & Rafi, 2010). However, a comprehensive analysis of user influence on Github appears to be an understudied topic, and this is the gap we seek to address and fill in this paper.

* Corresponding author.
*E-mail addresses:* huyan@dlut.edu.cn (Y. Hu), miss_shans_w@163.com (S. Wang), renyz@hdu.edu.cn (Y. Ren), raymond.choo@fulbrightmail.org (K.-K.R. Choo).

[1] http://github.com.
[2] https://stackoverflow.com/.
[3] http://bitbucket.org.
[4] http://git-scm.com/.

https://doi.org/10.1016/j.eswa.2018.05.002
0957-4174/© 2018 Elsevier Ltd. All rights reserved.

As there are many potential avenues for work and communication in software developer social networks, identifying the right practices, strategies and structures that are unique to a specific aspect of influence is challenging (Badashian & Stroulia, 2016). Badashian and Stroulia (2016), for example, demonstrated that in Github, users with many followers are not necessarily influential. Thus, in this paper, we analyze the influence of Github users by considering the social relationship between users, relations between users and projects, and the activities. Star and fork between users and projects are taken into account. We analyze user influence from four perspectives, and the analysis are then merged for further exploration.

Specifically, at the time of this research, this is the first work to comprehensively analyze user influence in Github to our knowledge. We propose a framework based on follow relation, star relation, fork relation and user activity, in order to measure user influence. We then collect all Github users between April 1, 2008 and January 19, 2017 from Ghtorrent API.[5] The dataset collected (from 16 million Github users, involved in up to 40 million projects) is then evaluated using our proposed framework to determine individual influence scores.

Besides Github, there are many other complex social networks, such as Stack Overflow. These networks are different from simple weblink networks, and have more varied structures and entities. Measuring the influence of users in these networks is of great significance for information retrieval, information recommendation and so on. However, in complex networks, it is one-sided and it will not be convincing to measure user influence using a single metric. Thus, our model can be applied in such context.

In the next two sections, we will present background and related work. In Section 4, we present our proposed framework and describe the dataset. Section 5 presents our experiments and the findings. Finally, we conclude this paper in Section 6.

The dataset and the results from using our proposed method in this paper are available on the project's Github page (https://github.com/wangshans/GithubUserInfluenceDataset).

## 2. Background

Apart from code hosting and maintenance functionalities, Github also provides social network features, such as following users, starring or forking projects. In other words, Github users can follow other users, star projects of interest, fork projects and send pull requests to the owners. The basic social relations amongst Github users are presented in Fig. 1 in the form of edges in the graph, and are described below:

- The *follow* edge is directed, and denotes the following relationship between users – e.g. *follow* edge $A \rightarrow B$ means that user A is following user B.
- The *own* edge is a link from one user to some other user's project.
- The *star* edge from a user to a project indicates that the user finds the project useful.
- The *fork* edge is created when a user wishes to make a clone of other user's project, may thereafter contributes to the project and asks the original owner to merge the forked branch if possible.
- The *belong to* edge indicates whether a Github user is affiliated to an organization.

Github provides the functionality to search for users and projects. As the number of users and projects on Github is increasing rapidly, the ability to filter users and projects based on the influence of Github users will be important for both individual users

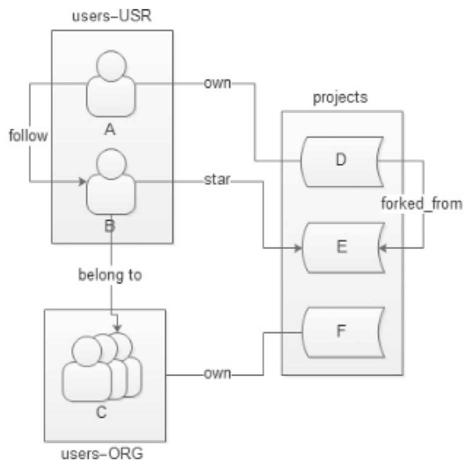


**Fig. 1.** A simplified Github network structure.

and organizations. For example, organizations may wish to target their advertising to users with a certain extent of influence to serve as "evangelists" or "champions" for their product(s) or service(s) to the community at large (Keller & Berry, 2003). "Influencer marketing" as a means of advertising and marketing facilitates the transfer of information about products and services from an organization to its customers (Brown & Hayes, 2008). For individual users, they may wish to only view the most influential users or projects in their search results on Github (analogous to users performing a Google search).

To measure user influence, we analyze most of the attributes of Github users. Specifically, based on the network structure of Github, we analyze user influence based on their *follow* relation, *star* relation, *fork* relation, and user activities. In the *follow* relation, we use PageRank (Langville & Meyer, 2011) and HITS (Hyperlink-Induced Topic Search) (Yan, Wei, Gui, & Chen, 2011) methods to quantify the status of users in social network of Github. Both PageRank and HITS are link analysis algorithms originally used to rank web pages. For example, Google defines PageRank as *"PageRank works by counting the number and quality of links to determine a rough estimate of how important the web site is. The underlying assumption is that more important web sites are likely to receive more link from other web sites."*. HITS algorithm divides web pages into two categories, namely: *hub* and *authority* pages. Authority pages represent the important pages on a certain topic, while hub pages represent the links to authority pages on the particular topic. Both algorithms have been widely used in network studies, and the effectiveness of the ranking has also been demonstrated in the literature.

In *star* and *fork* relations, we introduce H-index method to measure the capability of users. H-index, also known as H-factor, is an author-level metric that attempts to measure both the productivity and the citation impact of the publications of a scientist or scholar (Bornmann & Daniel, 2005). The index is based on the set of the scientist's most cited papers and the number of citations that they received in other publications (i.e. an individual scholar with an index of h has published at least h papers, and at least h papers has been cited h or more times). Generally, one considers a scholar with a high H-index to be the one with a high impact in the field, although there has been criticism of such a metric (Ayaz & Afzal, 2016). Despite the criticism associated with the use of H-index, it is generally an accepted measure within the research community.

In this paper, we use Spearman ranking correlation coefficient to analyze and compare results of our multi-facet analysis over the Github user influence. Finally, we use the Borda Count method

[5] http://www.ghtorrent.org/downloads.html.

to measure user influence comprehensively and verify the results based on the change ratio of users' followers.

## 3. Related work

Github has been the platform of focus by both code developers and researchers, for different reasons. From the researchers' perspective, Avelino et al. (2015) investigated the truck factor of popular Github applications. The authors found that most Github projects have a small truck factor (typically 1 or 2), meaning that those projects does not require many knowledgeable developers. Jurado and Rodriguez (2015) introduced sentiment analysis on the text in the issues raised on Github, to identify and monitor the underlying sentiments of those developers who wrote those text. Lima et al. (2014) characterized Github as both a social network and a collaborative platform. In the study of Casalnuovo et al. (2015), it was found that prior social connections combined with prior experience in the dominant languages of a project make the project development more productive, both initially and cumulatively. The study in Ray et al. (2014) examined the type and use of programming languages, in relation to software quality in Github. The authors in Hu, Zhang, Bai, Yu, and Yang (2016) analyzed the influence of Github repositories based on HITS and Star relation between users and repositories.

Another research focus has been on identifying influential users in social networks (Patil, Ghasemiesfeh, Ebrahimi, & Gao, 2013; Sun & Ng, 2013), such as Twitter and MicroBlog (Cha et al., 2010a; Dubois & Gaffney, 2014; Zhou, Zhang, Guo, & Guo, 2014). Sun and Ng (2013), for example, proposed the use of a graph model and transformed it to a user graph, and to identify the most influential users, they considered the properties and measures from both graphs. Cha, Haddadi, Benevenuto, and Gummadi (2010b) performed an in-depth comparison of three measures of influence, namely: indegree, retweets, and mentions, and found that popular users with high indegree are not necessarily influential in terms of spawning retweets or mentions. Another study performed on Github reported similar findings. Specifically, the findings from Badashian and Stroulia (2016) showed that the number of followers is not the best way to identify actively engaging influencers. In other words, there are other measures that quantify real influence in terms of impact and influence. The authors in Chai, Xu, Zuo, and Wen (2013) introduced a framework to identify and predict influential users in MicroBlog.

## 4. Our proposed approach

In this section, we will describe our approach to analyze Github user influence factors.

Our workflow is presented in Fig. 2, which consists of the following steps:

1. We collect the Github dataset and store it in a local database.
2. We preprocess the Github data, and study the attributes of users and repositories. A set of user and repository attributes are selected to measure the user influence.
3. We then construct the graph based developer social network, and apply the analysis from multiple perspectives.
4. We conduct experiments, and summarize the experimental results.

As previously discussed, our dataset was collected from Github API[6] between April 1, 2008 and January 19, 2017. This dataset comprises data of more than 16 million users, with more than 40 million repositories.

[6] http://www.ghtorrent.org/downloads.html.

**Table 1**
Information of users' basic indicators.

| Feature name | Maximum | Average | Median | Standard deviations |
|---|---|---|---|---|
| userFollowerNum | 45,112 | 6.89 | 2 | 1,167.42 |
| userFolloweeNum | 196,098 | 6.89 | 1 | 3,919.87 |
| userRepositoryNum | 24,178 | 11.34 | 5 | 370.92 |
| userStarNum | 369,085 | 20.08 | 0 | 2,764.62 |
| usersForkedNum | 33,342 | 3.40 | 0 | 2,306.24 |

### 4.1. Data analysis

#### 4.1.1. Basic user attribute analysis

Each Github user has a set of attributes, and we extract the following user attributes: User Follower Number (UFerN), User Followee Number (UFeeN), User Repository Number (URepoN), User Star Number (USN), and User Fork Number (UFkN).

Table 1 displays some basic facts about the selected Github user attributes. Take UFerN as an example, the average is six to seven, while the median is 733. The standard deviation of UFerN is large, which suggests that UFerN is polarized. The other attributes show a similar pattern, indicating that the distribution of user's attribute values is dispersed and the actual value of most users differs from the average. To verify this observation, we further plot the distribution of the selected user attributes, as explained in Section 4.1.2.

#### 4.1.2. Distribution of user attributes

We count the possible values of each user's influence indicators, and the number of users that correspond to them. Then, we draw the scatter diagrams, as shown in Fig. 3.

In Fig. 3, the horizontal axis represents the logarithm of the different values of user attributes, and the vertical axis is the logarithm of the number of users corresponding to an indicator. In the subgraphs, we have the overall distribution of the various metrics. Take the case of Fig. 3a, there is a point in the top left corner indicating around $10^6$, namely 1 million users do not have followers. Based on Fig. 3, and the existing studies on feature analysis of social networks (Feifei, Yabin, Zhuo, & Zhuang, 2013), we speculate that the user features conform to the Power Law Distribution (Clauset, Shalizi, & Newman, 2012). We use the Least Square method to fit the distribution of user features and obtain straight lines (shown as red dotted lines in the figure). The results demonstrate that these user features distributions are almost consistent with the Power Law Distribution, which can be formulated as $f(x) = Cx^{-\alpha}$.

Comparing the five subfigures, we observe that the lines in Figs. 3d and e deviate more from the straight line, and are slightly up-curved. It just happens that Figs. 3d and e are distribution of UFeeN and URepoN, which are decided by the users. The other features are decided by the network or other users. Thus, we can see that users who are proactive in Github and enhance their influence by following others and creating new projects. Such a phenomenon is also related to the formation of Power Law Distribution in social networks.

#### 4.1.3. Relation between UFerN and URepoN

User's influence is reflected in the number of followers and repositories. Therefore, we analyzed the relationship between UFerN and URepoN. From Fig. 4, it is clear that UFerN and URepoN have a positive correlation, but there are two special cases.

1. A large UFerN but a small URepoN: This case is usually due to the celebrity effect, that is, some users have great influence in practice. Thus, when they join Github, they naturally attract many followers easily. For example, the founder of RubyOnRails, David Heinemeier Hansson, as of January 19, 2017, has 10,425

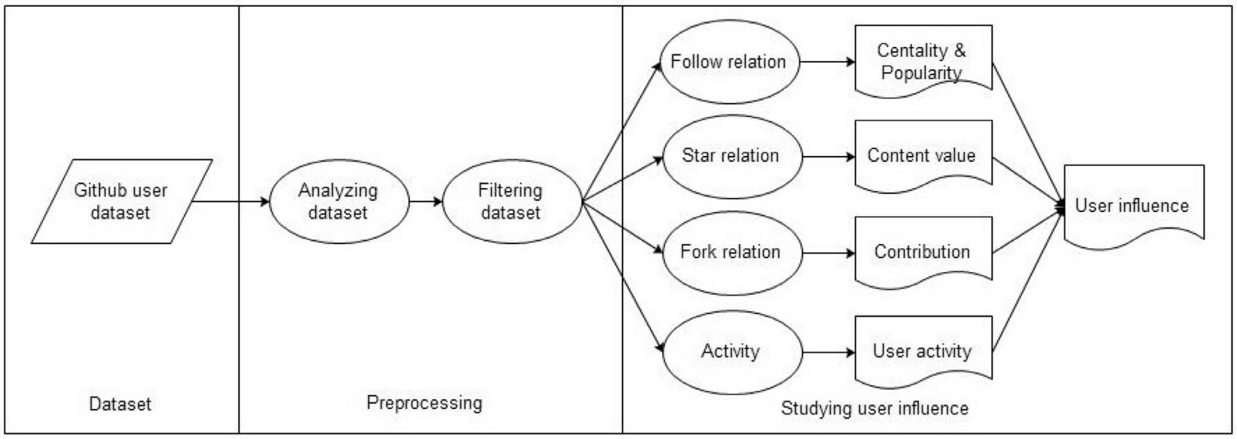
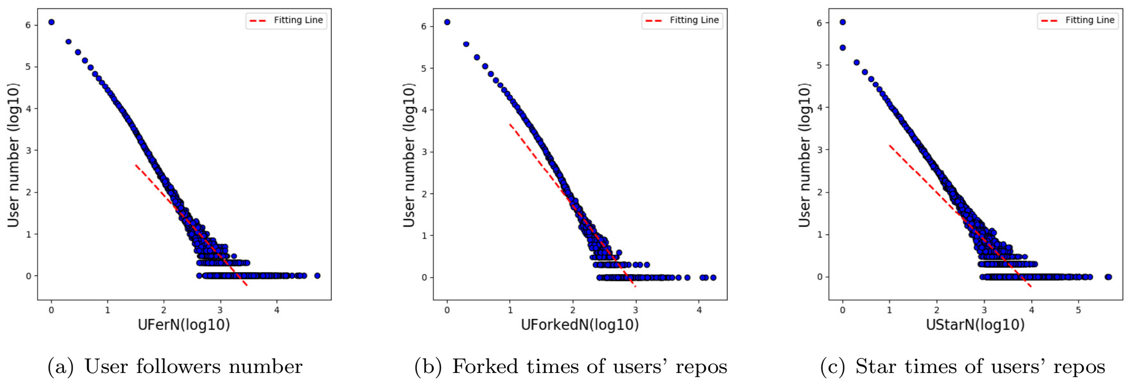
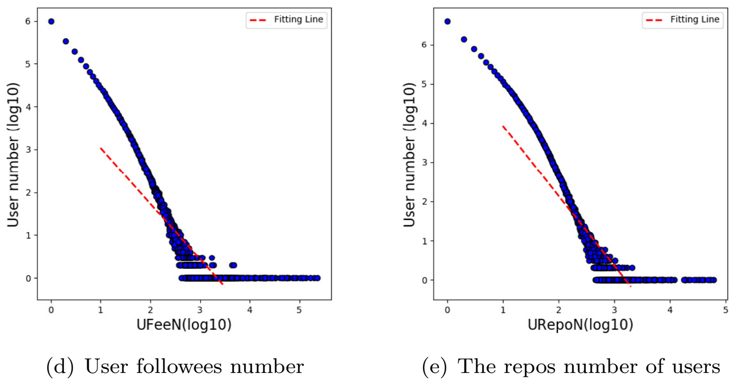


**Fig. 2.** Our workflow.



(a) User followers number (b) Forked times of users' repos (c) Star times of users' repos



(d) User followees number (e) The repos number of users

**Fig. 3.** Distribution of selected user attributes.

followers. However, he only has 4 repos. As of April 18, 2018, the same user has 11.7k followers and 21 repos.

2. A large URepoN but a small UFerN: This happens mainly in the case of organizational users. Many organizational users have their own product projects. For Github itself, organizational users cannot be followed until 2014; therefore, the organizational user followers number is smaller in comparison.

### 4.2. Filtering dataset

In order to eliminate the possibility that these correlations are due to the large numbers of Github members with one or more zero values, we undertook a more focused analysis. Specifically, we examined only relatively influential users. Therefore, we first filtered the dataset.

Users can be ‘real‘ or ‘fake‘. Real users can own projects and perform actions such as open issues, create pull requests and push commits. Fake users only appear as authors or committers of commits. Fake users are marked by the *fake* field[7] with the value of 1. For the users, we filtered and omitted the deleted or fake users, where the value of *deleted* and *fake* filed in the SQL table of users is 1. As the user's influence is mainly dependent on the followers and projects, users with no follower and no project has a minimal influence, if any. Therefore, in our study, we filter and remove such

[7] http://ghtorrent.org/relational.html.

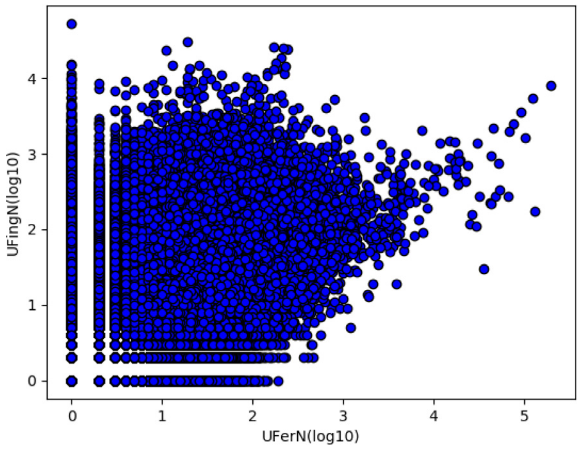


**Fig. 4.** The relation between UFerN and URepoN.

**Table 2**
Dataset before and after preprocessing.

| Data | | Initial data | filtered data |
|---|---|---|---|
| Users | USER | 15,551,557 | 1,826,170 |
| | ORG | 636,504 | 494,986 |
| | Total | 16,188,061 | 2,321,156 |
| Projects | | 46,772,903 | 20,798,840 |

users. For the organization users, we will use the sum of follower numbers of their members as the number for their follower numbers. For example, for organization users A and B, A has members $a_1$, $a_2$, and B has members $b_1$, $b_2$, $b_3$. The number of followers of A is the sum of the numbers of followers of $a_1$ and $a_2$, and the number of followers of B is the sum of the numbers of followers of $b_1$, $b_2$, and $b_3$.

For projects, we only consider the projects owned by candidate users (i.e. users that have not been filtered or removed).

Details of the dataset before and after preprocessing are shown in Table 2.

## 4.3. Studying user influence

We will now present the analysis methods used in our approach, namely: UserRank, HITS, H-index, Spearman rank correlation algorithm, and Borda Count method.

### 4.3.1. UserRank

As the Github social network is similar to the webpage network, we propose the UserRank algorithm, which is a version of PageRank adapted for Github. UserRank attempts to evaluate the centrality of users based on two assumptions, namely:

- **Quantity assumption:** Users with high influence usually have more followers; and
- **Quality assumption:** Users followed by influential ones usually have high influence.

First, we create a directed graph based on the *follow* relation. Each vertice in the graph stands for a user, and an edge from user A to user B means that A is a follower of B. The equation is as follows:

$$UserRank(u) = \frac{1-q}{N} + q \sum_{v follows u} \frac{UserRank(v)}{Followees(v)} \quad (1)$$

In the above equation, $u$ denotes the influencing user, $v$ is one follower of $u$, $Followees(v)$ is the number of users that $v$ follows, $N$ is the total number of users, $q$ is a damping factor (i.e. the probability that a user will continue to browse backward at any time after arriving at a page). It is generally assumed that the damping factor will be set around 0.85, namely $q = 0.85$ (Sehgal, Kaur, & Kumar, 2009).

### 4.3.2. HITS

Similar to webpages, we use the following assumptions:

- A good hub user usually follows many high influential (authoritative) users;
- An authoritative user is usually followed by many good hub users.

Thus, the users of Github can also be divided into authoritative users and hub users. An *authoritative* user is a user perceived to have a high quality, and this user is also generally considered an important user. A *hub* user refers to a user who follows many other users with high authority score. Thus, we propose user-HITS based on HITS to measure the authority and hub score of users, using the following iterative computations:

$$\begin{cases} a(u) = \sum\limits_{v follows u} h(v) \\ h(u) = \sum\limits_{u follows w} a(w) \end{cases} \quad (2)$$

In the above equation, $a(u)$ and $h(u)$ respectively represent the authority value and the hub value of user $u$. When the results converge, each user obtains an authority value and a hub value.

### 4.3.3. H-index

To measure users' impact on projects, we adapt the H-index algorithm, which is also called the H-factor. We apply H-index to measure both the productivity and the star and fork impact of projects from a user. The index is based on the set of user's most stared or forked projects and the number of stars and forks that they receive from other users. A user with an index of $h$ means (s)he has created $h$ projects, and each of these projects has been stared or forked by others at least $h$ times. A user with a high H-index has a high impact in Github.

### 4.3.4. Betweenness centrality

Betweenness centrality is a measure of centrality in a graph. It represents the degree of the nodes that stand between each other. The more times a node acts as a mediator, the greater the center of his/her betweenness centrality. Betweenness centrality has widespread applications in network theory, and the betweenness centrality of a node $t$ can be expressed mathematically as follows:

$$g(t) = \sum_{v \neq u \neq t} \frac{\sigma_{vu}(t)}{\sigma_{vu}}, \quad (3)$$

where $\sigma_{vu}$ is the total number of shortest paths from node $v$ to $u$, and $\sigma_{vu}(t)$ is the number of those paths that pass through $t$.

### 4.3.5. Borda count

The Borda count method is usually used for sample evaluation and calculation. It is easy to calculate, and has been widely adopted in many fields, such as group decision making, project demonstration, artificial economic evaluation, and quality assessment (Xiaoqing & Xueliang, 2006). It can be expressed mathematically as follows:

$$B(u) = \sum_{i \epsilon I} r(i), \quad (4)$$

where $r(i)$ is the score based on evaluation factors $i$. $B(u)$ is the value of object $u$, which is the measurement.

**Table 3**
*follow* based users analysis.

| Rank | UserFollowersNum | UserPageRankScore | UserAuthority | UserHub |
|---|---|---|---|---|
| 1 | torvalds | torvalds | torvalds | angusshire |
| 2 | JakeWharton | JakeWharton | Tj | akmoose |
| 3 | Tj | michaelliao | addyosmani | cusspvz |
| 4 | addyosmani | Tj | JakeWharton | ternsip |
| 5 | paulirish | mojombo | visionmedia | MichalPaszkiewicz |
| 6 | mojombo | paulirish | paulirish | popsiclesarecool |
| 7 | defunkt | githubpy | douglascrockford | alex-cory |
| 8 | sindresorhus | defunkt | ruanyf | zhiaixuexi |
| 9 | douglascrockford | addyosmani | sindresorhus | woxly |
| 10 | mbostock | jeresig | yinwang0 | Mrzeron |
| 11 | jeresig | douglascrockford | mojombo | Brunocasanova |
| 12 | ruanyf | visionmedia | jeresig | rootux |
| 13 | daimajia | mattt | mbostock | haoranw |
| 14 | mattt | kennethreitz | defunkt | threejs-cn |
| 15 | mdo | tpope | chrisbanes | free1 |
| 16 | kennethreitz | mbostock | substack | chenyoufu |
| 17 | schacon | gaearon | mdo | VonixPro |
| 18 | gaearon | substack | daimajia | Linux-Player |
| 19 | jlord | sindresorhus | jashkenas | WilberTian |
| 20 | visionmedia | wycats | gaearon | ahmetabdi |

**Table 4**
Betweenness centrality and UserHub ranks of sub-graph.

| Rank | Betweenness centrality | UserHub |
|---|---|---|
| 1 | Marak | Marak |
| 2 | maxogden | maxogden |
| 3 | andrew | andrew |
| 4 | chovy | springmeyer |
| 5 | tmcw | tmcw |
| 6 | mikolalysenko | chovy |
| 7 | dshaw | mikolalysenko |
| 8 | springmeyer | stagas |
| 9 | substack | jmettraux |
| 10 | brianleroux | hij1nx |
| 11 | mafintosh | defunkt |
| 12 | timoxley | substack |
| 13 | hughsk | mafintosh |
| 14 | elliottcable | 0x00A |
| 15 | pengwynn | h4ck3rm1k3 |
| 16 | darwin | dshaw |
| 17 | h4ck3rm1k3 | kotp |
| 18 | jmettraux | joshbuddy |
| 19 | tbranyen | ericflo |
| 20 | ralphtheninja | ralphtheninja |

*4.3.6. Spearman's rank correlation coefficient*

Spearman's rank correlation coefficient is a measure of the strength of association between two rank sets (Pirie, 1988), as described below:

$$\rho = 1 - \frac{6\sum(x_i - y_i)^2}{N^3 - N} \tag{5}$$

In the above equation, $x_i$ and $y_i$ denote the ranks of users based on two different ranking measures in a data set with $N$ users. As a non-parametric test, the Spearman's rank correlation coefficient assesses how well an arbitrary monotonic function can describe the correlation between two variables, without making any other assumption over the relation between the variables. Our inclusive and complete dataset guarantees the reliability of the correlation estimates. The value p ranges from $+1$ to $-1$, where a perfectly positive correlation is $+1$, a perfectly negative correlation is $-1$, and no correlation is 0.

## 5. Findings

In this section, we will analyze the users' influence using the approach outlined in Section 4.3.

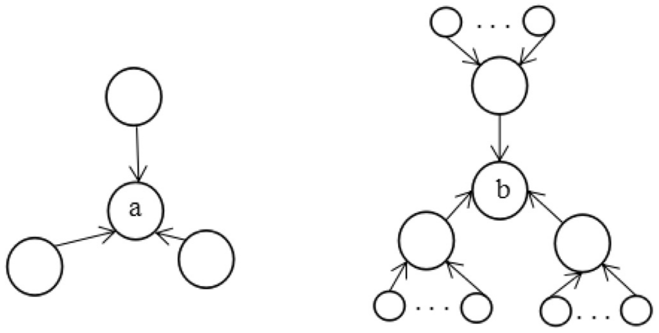


**Fig. 5.** An example of users with followers.

### 5.1. Follow relation based analysis

We build Github social network $G$ based on the *follow* relationship between users, in order to analyze user influence.

*5.1.1. User followers' number*

The most direct way to measure user influence is to count the number of followers, namely: the in-degree of user nodes in network $G$. Intuitively, users with more followers are more popular.

So we introduce assumption one: *"Users with more followers have greater influence."* We also propose the UFerN based user influence analysis algorithm.

*5.1.2. UserRank*

Although the UFerN based influence analysis algorithm is simple and direct, it does not consider the different qualities of each link. As shown in Fig. 5, users $a$ and $b$ each have three followers, but $a$'s followers have no follower and $b$'s followers have other followers. Clearly, links to B should have a higher value, rather than considering both $a$ and $b$ to have the same level of influence simply because they have the same number of followers. Therefore, UserRank is used to measure user influence.

*5.1.3. HITS*

The UserRank algorithm finds users at the center of the following relationship. However, users are less likely to receive much attention when they first join, despite their potential. Thus, we may be able to identify these potential influential users by examining

**Table 5**
Top 20 users based on *follow* relation analysis.

| Rank | USER | | | ORG | | |
|---|---|---|---|---|---|---|
| | UFerN | UserRank | UserAuth | UFerNSum | UserRankSum | UserAuthSum |
| 1 | torvalds | torvalds | torvalds | github | github | google |
| 2 | JakeWharton | JakeWharton | Tj | google | google | Microsoft |
| 3 | Tj | michaelliao | addyosman | component | component | github |
| 4 | addyosmani | Tj | JakeWharto | yeoman | jquery | facebook |
| 5 | paulirish | mojombo | visionmedi | jquery | h5bp | EpicGames |
| 6 | mojombo | paulirish | pauliris | h5bp | facebook | mozilla |
| 7 | defunkt | githubpy | douglascrockfor | facebook | twitter | Azure |
| 8 | sindresorhus | defunkt | ruany | components | yeoman | jenkinsci |
| 9 | douglascrockford | addyosmani | sindresorhu | twitter | components | coderwall-charity |
| 10 | mbostock | jeresig | yinwang | nodeschool | rubinius | twitter |
| 11 | jeresig | douglascrockford | mojomb | GoogleChrome | rails | nodeschool |
| 12 | ruanyf | visionmedia | jeresi | rails | nodeschool | apache |
| 13 | daimajia | mattt | mbostoc | yarnpkg | GoogleChrome | coderwall-polygamous |
| 14 | mattt | kennethreitz | defunk | html5rocks | square | jquery |
| 15 | mdo | tpope | chrisbane | expressjs | mozilla | coderwall-forked |
| 16 | kennethreitz | mbostock | substac | Microsoft | LearnBoost | GoogleCloudPlatform |
| 17 | schacon | gaearon | md | reworkcss | heroku | railsrumble |
| 18 | gaearon | substack | daimaji | LearnBoost | yarnpkg | recursecenter |
| 19 | jlord | sindresorhus | jashkena | WebComponents | Microsoft | component |
| 20 | visionmedia | wycats | gaearo | koajs | html5rocks | yeoman |

better hub users (and their followers). So we introduce the HITS algorithm to the following relation networks.

The top 20 users ranked by user followers number, UserRank, HITS's authority users, HITS's hub users are shown in Table 3.

### 5.1.4. Betweenness centrality

Betweenness centrality is based on shortest paths. So it needs to compute the shortest path between any two nodes. Limited by the time and complexity of the calculation, we only verify the betweenness centrality on a subgraph.

First, we obtain a graph of users who follow each other based on *followers*. Then, we use a classical community discovery algorithm LOUVAIN to calculate the above graph. LOUVAIN is a community detection algorithm based on module degree. The algorithm performs better in terms of efficiency and effectiveness, and it can discover hierarchical community structure. From the communities we obtained, we then select the largest one to study, which comprises 46,393 nodes. The top 20 users of betweenness centrality and their hub value of HITS is shown in Table 4.

The spearman correlation between betweenness centrality and userHub value is 0.734.

## 5.2. Organizational user (ORG) follow relation analysis

As users cannot directly follow an organization, the organization's followers in the dataset is not an accurate reflection of the organizational users' status in the social network. However, there is a direct and obvious link from the organization members to organization. So the influence status of organization members in *follow* relation represents the organization's influence status in the social network. We sum the various indicators of members and take the results as influence indicators of the organization.

The top 20 users ranked by UFerN, UserRank, and user-HITS authority users are shown in Table 5.

## 5.3. Star relation based analysis

As previously discussed, users and projects are core elements of the Github network. The project achievement can have a great effect on user influence, in addition to the number of followers in the social networks. Typically a user with high achievement also has a higher influence. We measure user influence by measuring

**Table 6**
*star* based analysis.

| Rank | Hindex | AveNum | MaxNum | SumNum |
|---|---|---|---|---|
| 1 | sindresorhus | nnnick | FreeCodeCamp | sindresorhus |
| 2 | substack | zxing | vhf | FreeCodeCamp |
| 3 | Tj | arasatasaygin | robbyrussell | Tj |
| 4 | visionmedia | allmobilize | sindresorhus | docker |
| 5 | maxogden | torvalds | getify | substack |
| 6 | mafintosh | nickbutcher | docker | JakeWharton |
| 7 | Firebase | jmechner | daneden | vhf |
| 8 | tpope | ViccAlexander | mbostock | visionmedia |
| 9 | Spatie | openssl | Apple | Apple |
| 10 | JakeWharton | valums | torvalds | kennethreitz |
| 11 | addyosmani | DrKLO | jwasham | getify |
| 12 | nicklockwood | iluwatar | nodejs | robbyrussell |
| 13 | docker | fancyapps | mrdoob | gaearon |
| 14 | feross | mleibman | jlevy | nodejs |
| 15 | ccoenraets | venomous0x | vinta | mbostock |
| 16 | Esri | CoderMJLee | blueimp | daneden |
| 17 | mattn | FreeCodeCamp | necolas | addyosmani |
| 18 | keijiro | MengTo | Dogfalo | antirez |
| 19 | hadley | masonry | nvbn | wasabeef |
| 20 | soffes | nostra13 | rg3 | bevacqua |

**Table 7**
*fork* based analysis.

| Rank | Average Num | Hindex | MaxNum | SumNum |
|---|---|---|---|---|
| 1 | MyCoolTest | Esri | rdpeng | rdpeng |
| 2 | LarryMad | sindresorhus | jtleek | jtleek |
| 3 | octocat | ccoenraets | octocat | octocat |
| 4 | rdpeng | android | MyCoolTest | MyCoolTest |
| 5 | nnnick | Tj | LarryMad | nodejs |
| 6 | zxing | chef | barryclark | LarryMad |
| 7 | venomous0x | substack | jlord | elastic |
| 8 | learnstreet-dev | elastic | gabrielecirulli | docker |
| 9 | Tower-KevinLi | nicklockwood | vhf | jlord |
| 10 | torvalds | PayPal | mbostock | barryclark |
| 11 | jtleek | Firebase | robbyrussell | JakeWharton |
| 12 | XcodeGhostSource | SlimRoms | nodejs | jashkenas |
| 13 | opencart | hadley | torvalds | hakimel |
| 14 | aporter | docker | hakimel | mbostock |
| 15 | openssl | JakeWharton | wakaleo | gabrielecirulli |
| 16 | julycoding | peachananr | docker | vhf |
| 17 | fancyapps | codefellows | mrdoob | robbyrussell |
| 18 | barryclark | kennethreitz | blueimp | cocos2d |
| 19 | aren19 | jaredhanson | jashkenas | torvalds |
| 20 | mleibman | maxogden | daneden | blueimp |

**Table 8**
*fork* with pull request based analysis.

| Rank | ForkedPRedAveNum | ForkedPRedHindex | ForkedPRedMaxNum | ForkedPRedSumNum |
|---|---|---|---|---|
| 1 | LarryMad | elastic | LarryMad | LarryMad |
| 2 | octocat | chef | jlord | jlord |
| 3 | Tower-KevinLi | sindresorhus | octocat | octocat |
| 4 | thank-you-github | docker | rdpeng | elastic |
| 5 | CUcsci2270 | substack | wbond | docker |
| 6 | nnnick | Tj | docker | rdpeng |
| 7 | openssl | codefellows | robbyrussell | makersacademy |
| 8 | opencart | hadley | elastic | wbond |
| 9 | codeschool-kiddo-level-1-complete | servo | mitchellh | sindresorhus |
| 10 | qlcoder-oreo | nodejs | adambard | chef |
| 11 | brianfrankcooper | PayPal | michaelliao | nodejs |
| 12 | perma-id | makersacademy | tomchristie | substack |
| 13 | mleibman | kennethreitz | mrdoob | mitchellh |
| 14 | zxing | JakeWharton | cocos2d | robbyrussell |
| 15 | hector-client | mitsuhiko | nodejs | servo |
| 16 | doxygen | tpope | vhf | cocos2d |
| 17 | tg-msft | maxogden | Tower-KevinLi | Tj |
| 18 | jashkenas | schmittjoh | discourse | jashkenas |
| 19 | DrKLO | kartik-v | soulteary | kennethreitz |
| 20 | s3tools | ekmett | leereilly | mitsuhiko |

his/her project achievement, which directly reflects on the number of received star.

Firstly, we count the stars from each user for all his/her projects, including projects with 0 star. Secondly, we calculate the sum number (UStarSumNum), the average number (UStarAveNum), and the maximum number (UStarMaxNum) of stars received by the user for all projects. Then, we propose the H-star method based on H-index to comprehensively measure the user influence inflected in projects. A user's H-star score of h indicates that this user has at least h projects receiving a star of at least h times. H-star can accurately reflect the project achievement of a user by quality and quantity. A higher h score indicates a higher project achievement, so the user has a higher influence.

## 5.4. Fork relation based analysis

Forking projects and pull requests are unique features in Github. Users can easily copy projects to their own accounts by "fork", use them independently and contribute code for the forked project. First, the user finds the project that he/she wants to participate in and then the user has full access to the project as if the project was originally created by himself/herself. When the user completes the development and pushes to his/her own library, then he/she can send a "pull request" to the original owner of the project. The owner receives the pull request and checks the code, before deciding to accept (merge) or reject (close) the pull request (Tables 6 and 11).

Through the branching relation between projects, a fork network is established between users. We then analyze the fork relation. Similarly with star relation, we count the sum number, average number and max number of all projects of a user being forked. And we introduce H-index as the H-fork algorithm. The top-20 users of each results are shown in Table 7.

The flow of the forking is shown in Fig. 6. Considering this scenario: a user has a project which has been forked many times, but the users who forked it do not submit any changes back. Hence, no other observable Github activity relating to these forks exists. Another user has a project at the same time "forked", but the forks have both commits and pull requests. Thus, we should not consider them to have the same influence. Therefore, we filter forks with pull requests as more effective forks. According to our analysis, 36.67% of the Github projects were forked from others. Of these projects, 36.54% had sent pull requests to the project it forked from, and 34.08% had been merged. For the users, 23% of all Github users had projects which were forked from others, and 57.63% of them had send pull requests to the owner of projects they forked from. However, only 17% had pull requests merged – see Table 8.

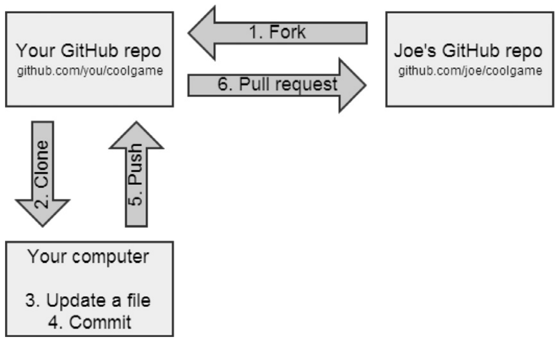


**Fig. 6.** Flow of fork.

## 5.5. Activity

In Github, users can commit, create projects, open issues, send pull requests and so on. Activities convert the user's potential influence into actual impact. More active participation means the user has a higher level of preference and contribution, and frequent interaction can enhance the trust between each other. These are decisive characteristics of social influence. Furthermore, the probability of user known by others is related to the level of his/her activeness (e.g. an active social media user is likely to be known to more users, or easier acquainted by others). Therefore, activities will enhance one's online influence.

Github counts the number of users to commit, create projects, open issues and send pull requests as the amount of users' contribution. We use the same measure to quantify the user's activity. The results of the top 20 based on extracted activities over three months are presented in Table 9.

## 5.6. Correlation

In the preceding sections, we analyzed Github user influence from different perspectives. In this section, we compared user ranks obtained by different influence metrics with Spearman rank correlation coefficient.

**Table 9**
Activity based analysis.

| Rank | USER | ORG |
|---|---|---|
| 1 | greenkeeperio-bot | the-domains |
| 2 | koorellasuresh | docker-rubygem |
| 3 | everypoliticianbot | latin-ocr |
| 4 | openpublishtest | learn-co-students |
| 5 | gitter-badger | GITenberg |
| 6 | npmcdn-to-unpkg-bot | circle-test-organization |
| 7 | shuhongwu | tizenorg |
| 8 | digideskio | htmlacademy-htmlcss |
| 9 | ahgood | Trietptm-on-Security |
| 10 | atomist-test | DevFactory |
| 11 | maxtortime | InsightsDev |
| 12 | pazjacket | staticmanapp |
| 13 | bestwpw | Sym123Blue |
| 14 | din982 | cygwinports-extras |
| 15 | xws-bench | DEV-learn-co-students |
| 16 | pombredanne | techtronics |
| 17 | rawlingsj | mixpanel-platform |
| 18 | jotegui | CodecoolBP20161 |
| 19 | test-driver | LineageOS |
| 20 | waffle-iron | conda-forge |

**Table 10**
Correlation based analysis.

| Spearman | | UAct | UAuth | UFerN | URFH | URSH | URank |
|---|---|---|---|---|---|---|---|
| 10 | UAct | 1 | 0.2 | 0.115 | 0.411 | 0.552 | −0.115 |
| | UAS | 0.2 | 1 | .818** | −0.006 | 0.055 | 0.6 |
| | UFN | 0.115 | .818** | 1 | 0.117 | 0.139 | .879** |
| | UFH | 0.411 | −0.006 | 0.117 | 1 | .951** | −0.067 |
| | USH | 0.552 | 0.055 | 0.139 | .951** | 1 | −0.127 |
| | UPRS | −0.115 | 0.6 | .879** | −0.067 | −0.127 | 1 |
| 100 | UAct | 1 | −0.092 | −0.021 | .293** | .432** | 0.035 |
| | UAS | −0.092 | 1 | .834** | 0.16 | 0.081 | .689** |
| | UFN | −0.021 | .834** | 1 | .202* | 0.104 | .809** |
| | UFH | .293** | 0.16 | .202* | 1 | .880** | .305** |
| | USH | .432** | 0.081 | 0.104 | .880** | 1 | .212* |
| | UPRS | 0.035 | .689** | .809** | .305** | .212* | 1 |
| 1000 | UAct | 1 | −.063* | 0.037 | .181** | .360** | .086** |
| | UAS | −.063* | 1 | .790** | .240** | .188** | .578** |
| | UFN | 0.037 | .790** | 1 | .316** | .253** | .714** |
| | UFH | .181** | .240** | .316** | 1 | .843** | .359** |
| | USH | .360** | .188** | .253** | .843** | 1 | .324** |
| | UPRS | .086** | .578** | .714** | .359** | .324** | 1 |
| 10000 | UAct | 1 | .023* | .084** | .148** | .313** | .068** |
| | UAS | .023* | 1 | .715** | .247** | .271** | .499** |
| | UFN | .084** | .715** | 1 | .349** | .369** | .694** |
| | UFH | .148** | .247** | .349** | 1 | .779** | .348** |
| | USH | .313** | .271** | .369** | .779** | 1 | .323** |
| | UPRS | .068** | .499** | .694** | .348** | .323** | 1 |

For each user, we computed the value of each influence measurement. Rather than comparing the values directly, we used the relative order of users' rank. In order to do this, we sorted users by each measurement, so that the rank of 1 indicates the most influential user and a reduced rank score indicates a less influential user. Intuitively, users with the same influence value receive the highest of the rank among them. Once every user is assigned a rank for each measure, we quantify how a user's rank varies across different measures and examine what kinds of users are ranked high for a given measure.

We identified different sets of top users based on each measure: the top 10, top 100,top 1000, top 10,000 users. Then, we calculated the pairwise Spearman correlation between different measures, which are shown in Table 10. The correlations between URFH and URSH are mostly strong, and the correlations between UAct and others are generally weak or non-significant.

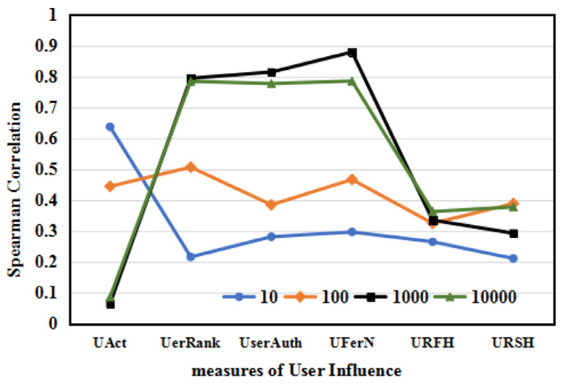


**Fig. 7.** Correlation analysis of Borda Count results with other measures.

## 5.7. Borda count

The above scores measure the user's influence from different angles. We regard each measure as a vote on the overall influence of one user, and use Borda Count to measure user influence comprehensively, which is often used in real-world general elections.

Suppose the total number of users is 100, with two sorted lists. User $u$ ranks 3 and 5 respectively in the two lists, and user $v$ ranks 2 and 20 respectively (the highest ranking is 1, and the smaller the rank number is, the higher the ranking is). By using the Borda Count method, user $u$ obtains a score of 192 $(97 + 95)$, user $v$ has a score of 178 $(98 + 80)$. Assuming the minimum score to be 0 and the maximum score to be 198, we normalize the scores to $0 - 1$, with $finalSocre = (score - minimum(0))/(maximum(198) - minimum(0))$. Thus, the final score of user $u$ is 0.9697, and the final score of user $v$ is 0.8990. In our final user influence metric, we also calculate their percentiles in the rank for each user. Specifically, the top 1% of users are considered highly influential users in Github.

We analyze the correlation between the Borda Count results and the other measures, with users of top 10, 100, 1000, 10000. It can be seen from Fig. 7 that the comprehensive user influence measured by Borda Count differs from the other individual measures. With an increase in the number of users selected to measure correlation, from top 10 to 10000, the measure with the highest correlation coefficient with the Borda Count results switch from UAct to URank and UAuth, and finally UFerN outperforms all the other approaches. For top 10 and 100 users, the user's activity has a higher correlation with the Borda Count result, indicating that for some of the most influential users, their activities have a great impact on the overall influence. As the range extends, the measures computed by follow relation have an advantage over the others, especially UFerN. It shows that the UFerN has a more important impact on the influence of most users. By contrast, the URFH (H-index of user repos fork relation) and URSH (H-index of user repos star relation) has lower correlation with the Borda Count results in the range of top 10 to top 10000. That is partly because there is little difference in the H-index between users. In particular, the URSH values of more than 1.8 million users are distributed between 0 and 128, so the distribution is concentrated. In addition, the users' repos URFH and URSH present similar correlation results, because they are both computed from users' repos and there is high consistency between them.

Users with high influence have more important impact in the network, and can deliver information to more users and receive more attention. So the change of the followers number can reflect the influence of a user in a period of time. In this section, we will analyze and verify the measures of comprehensive influ-

**Table 11**
Correlation analysis.

| | Borad | 12MonthAct | Auth | FollowersNum | UFH | USH | UPRS | Average |
|---|---|---|---|---|---|---|---|---|
| Borad | 1 | .519** | .583** | .668** | .460** | .478** | .736** | **0.6348** |
| 12MonthAct | .519** | 1 | −0.092 | −0.021 | .293** | .432** | 0.035 | 0.3094 |
| Auth | .583** | −0.092 | 1 | .834** | 0.16 | 0.081 | .689** | 0.465 |
| UFN | .668** | −0.021 | .834** | 1 | .202* | 0.104 | .809** | 0.5137 |
| UFH | .460** | .293** | 0.16 | .202* | 1 | .880** | .305** | 0.4714 |
| USH | .478** | .432** | 0.081 | 0.104 | .880** | 1 | .212* | 0.4552 |
| UPRS | .736** | 0.035 | .689** | .809** | .305** | .212* | 1 | 0.5408 |

**Table 12**
Top 10 users of Borda Count and their followers.

| Rank | USER | Original UFerN | Current UFerN | Variation ratio |
|---|---|---|---|---|
| 1 | sindresorhus | 15.7 | 23.8 | 0.2033 |
| 2 | gaearon | 13.0 | 27.4 | 0.3562 |
| 3 | JakeWharton | 30.1 | 45.2 | 0.1995 |
| 4 | paulirish | 24.5 | 25.7 | 0.0237 |
| 5 | feross | 5.6 | 7.2 | 0.1270 |
| 6 | hadley | 8.3 | 12.7 | 0.2084 |
| 7 | mbostock | 14.9 | 17.5 | 0.0793 |
| 8 | domenic | 2.4 | 3.1 | 0.1266 |
| 9 | Tj | 25.8 | 34.5 | 0.1438 |
| 10 | addyosmani | 24.6 | 29.1 | 0.0837 |
| | Overall | 164.9 | 226.2 | **0.1567** |

**Table 13**
Top 10 users of UFerN and their followers.

| Rank | USER | Original UFerN | Current UFerN | Variation ratio |
|---|---|---|---|---|
| 1 | torvalds | 52.7 | 70.7 | 0.1457 |
| 2 | JakeWharton | 30.2 | 45.2 | 0.1996 |
| 3 | Tj | 25.8 | 34.5 | 0.1438 |
| 4 | addyosmani | 24.6 | 29.1 | 0.0837 |
| 5 | paulirish | 24.5 | 25.7 | 0.0237 |
| 6 | mojombo | 23.1 | 20.9 | −0.0494 |
| 7 | defunkt | 18.5 | 16.8 | −0.0487 |
| 8 | sindresorhus | 15.8 | 23.8 | 0.2033 |
| 9 | douglascrockford | 15.4 | 16.5 | 0.0346 |
| 10 | mbostock | 14.9 | 17.5 | 0.0793 |
| | Overall | 245.5 | 300.7 | **0.1011** |

ence by comparing the follower number at current and at the time when the experiment data set was collected.

Tables 12 and 13 show the top 10 users measured by Borda Count and UFerN, respectively. The Original UFerN refers to the number of followers during data collection (i.e. January 19, 2017), and the Current UFerN is the number of followers as of April 03, 2018. We then compute $VariationRatio = (OriUFerN - CurUFerN)/(OriUFerN + CurUFerN)$.

From Tables 12 and 13, we observe that the overall UFerN's variation ratio of the top 10 users measured by Borda Count is 15.67%. While the variation ration of the top 10 users measured by UFerN is 10.11%. In other words, after a period of time, the increase in the follower number of highly influential users ranked by Borda Count is higher than the number of top users ranked by UFerN, which verify the comprehensiveness of Borda Count user influence results to a certain extent.

## 6. Conclusion

In this paper, we proposed a framework to more comprehensively determine the influence of users in Github. Unlike existing approaches that adopt either a single dimension or a few indicators to quantify user influence, we extracted features from social links, projects, and activity data to obtain user influence indicators, including users *follow* relation, *star* and *fork* relation, and user activity. We then adopted HITS, PageRank, and H-index to measure the influence of users from these different perspectives, and used a classical voting algorithm, Borda count, to synthesize different results and obtain a comprehensive measure of the influence of each user. We validated our approach using the change ratio of users' followers crawled from Github.

In future, we plan to develop a full-fledged tool to automate both data collection and data analysis, with the aim of informing decision making by users and developers. We will apply our analysis model of user influence to other similar complex networks (such as stack overflow) to identify influential users, discovery leader users and so on.

## Appendix A. Abbreviations

| | |
|---|---|
| CurUFerN | Current(April 03, 2018) UFerN |
| ForkedPRed | ForkED and Pull Request |
| HITS | Hyperlink-Induced Topic Search |
| ORG | ORGanization user |
| OriUFerN | Original (January 19, 2017) UFerN |
| UAct | User Activity |
| UAuth | The Authority value of User correlated by HITS |
| UFerN | User Follower Number |
| UFerNSum | The sum of the members' UFerN |
| UFeeN | User Followee Number |
| URFN | User Repository Forked Number |
| URFH | User Repository Forked H-index |
| URepoN | User Repository Number |
| URSN | User Repository Star Number |
| URSH | User Repository Star H-index |
| URank | The score of user computed by UserRank |
| URankSum | The sum of the members' Rank |
| USER | individual USER (not organizaiton) |

## References

Avelino, G., Valente, M. T., & Hora, A. (2015). What is the truck factor of popular GitHub applications? A first assessment. *Technical Report*. PeerJ PrePrints.
Ayaz, S., & Afzal, M. T. (2016). Identification of conversion factor for completing-h index for the field of mathematics. *Scientometrics, 109*(3), 1511–1524.
Aziz, M., & Rafi, M. (2010). Identifying influential bloggers using blogs semantics. In *Proceedings of the 8th international conference on frontiers of information technology* (p. 7). ACM.
Badashian, A. S., & Stroulia, E. (2016). Measuring user influence in Github: The million follower fallacy. In *Crowdsourcing in software engineering (CSI-SE), 2016 IEEE/ACM 3rd international workshop on* (pp. 15–21). IEEE.
Bornmann, L., & Daniel, H.-D. (2005). Does the h-index for ranking of scientists really work? *Scientometrics, 65*(3), 391–392.
Brown, D., & Hayes, N. (2008). *Influencer marketing: Who really influences your customers?*. Routledge.
Casalnuovo, C., Vasilescu, B., Devanbu, P., & Filkov, V. (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* (pp. 817–828). ACM.

Cha, M., Haddadi, H., Benevenuto, F., & Gummadi, P. K. (2010a). Measuring user influence in twitter: The million follower fallacy. *Icwsm, 10*(10–17), 30.

Cha, M., Haddadi, H., Benevenuto, F., & Gummadi, P. K. (2010b). Measuring user influence in twitter: The million follower fallacy. In *Proceedings of the fourth international conference on weblogs and social media*. ACM.

Chai, W., Xu, W., Zuo, M., & Wen, X. (2013). Acqr: A novel framework to identify and predict influential users in micro-blogging. In *Pacis* (p. 20).

Clauset, A., Shalizi, C. R., & Newman, M. E. J. (2012). Power-law distributions in empirical data. *Siam Review, 51*(4), 661–703.

Dabbish, L., Stuart, C., Tsay, J., & Herbsleb, J. (2012). Social coding in github: transparency and collaboration in an open software repository. In *Proceedings of the ACM 2012 conference on computer supported cooperative work* (pp. 1277–1286). ACM.

Dubois, E., & Gaffney, D. (2014). The multiple facets of influence: Identifying political influentials and opinion leaders on twitter. *American Behavioral Scientist, 58*(10), 1260–1277.

Feifei, L., Yabin, X., Zhuo, L., & Zhuang, W. (2013). Analysis of characteristics of social networks in terms of microblog impact. *Journal of Computer Applications, 33*(12), 3359–3362.

Hebig, R., Ho-Quang, T., Chaudron, M., Robles, G., & Fernandez, M. A. (2016). The quest for open source projects that use uml: Mining github. In *Proceedings of the ACM/IEEE 19th international conference on model driven engineering languages and systems* (pp. 173–183). ACM.

Hu, Y., Zhang, J., Bai, X., Yu, S., & Yang, Z. (2016). Influence analysis of github repositories. *SpringerPlus, 5*(1), 1268.

Jurado, F., & Rodriguez, P. (2015). Sentiment analysis in monitoring software development processes: An exploratory case study on github's project issues. *Journal of Systems and Software, 104*, 82–89.

Kalliamvakou, E., Gousios, G., Blincoe, K., Singer, L., German, D. M., & Damian, D. (2014). The promises and perils of mining github. In *Proceedings of the 11th working conference on mining software repositories* (pp. 92–101). ACM.

Katsimpras, G., Vogiatzis, D., & Paliouras, G. (2015). Determining influential users with supervised random walks. In *Proceedings of the 24th international conference on world wide web* (pp. 787–792). ACM.

Keller, E., & Berry, J. (2003). *The influentials: One American in ten tells the other nine how to vote, where to eat, and what to buy*. Simon and Schuster.

Langville, A. N., & Meyer, C. D. (2011). *Google's PageRank and beyond: The science of search engine rankings*. Princeton University Press.

Lima, A., Rossi, L., & Musolesi, M. (2014). Coding together at scale: Github as a collaborative social network. *Icwsm*.

Patil, A., Ghasemiesfeh, G., Ebrahimi, R., & Gao, J. (2013). Quantifying social influence in epinions. In *Social computing (SocialCom), 2013 international conference on* (pp. 87–92). IEEE.

Peng, S., Wang, G., & Xie, D. (2017). Social influence analysis in social networking big data: Opportunities and challenges. *IEEE Network, 31*(1), 11–17.

Pirie, W. (1988). Spearman rank correlation coefficient. *Encyclopedia of Statistical Sciences*.

Ray, B., Posnett, D., Filkov, V., & Devanbu, P. (2014). A large scale study of programming languages and code quality in github. In *Proceedings of the 22nd ACM SIGSOFT international symposium on foundations of software engineering* (pp. 155–165). ACM.

Riihonen, M. (2017). Its hard for software developers to imagine life without github. http://www.newelectronics.co.uk/electronics-blogs/its-hard-for-software-developers-to-imagine-life-without-github/159931/ Online; accessed 20 October 2017.

Saxena, R., & Pedanekar, N. (2017). I know what you coded last summer: Mining candidate expertise from github repositories. In *Proceedings of the 2017 ACM conference on computer supported cooperative work and social computing* (pp. 299–302). ACM.

Sehgal, U., Kaur, K., & Kumar, P. (2009). The anatomy of a large-scale hyper textual web search engine. In *Second international conference on computer and electrical engineering* (pp. 491–495).

Sun, B., & Ng, V. T. (2013). Identifying influential users by their postings in social networks. In *Ubiquitous social media analysis* (pp. 128–151). Springer.

Thung, F., Bissyande, T. F., Lo, D., & Jiang, L. (2013). Network structure of social coding in github. In *Software maintenance and reengineering (csmr), 2013 17th European conference on* (pp. 323–326). IEEE.

Xiaoqing, Z., & Xueliang, L. (2006). Analysis and evaluation of academic influence for journals on nuclear science by borda method. *Acta Editologica*, (s1), 192–194.

Yan, L., Wei, Y., Gui, Z., & Chen, Y. (2011). Research on pagerank and hyperlink-induced topic search in web structure mining. In *Internet technology and applications (iTAP), 2011 international conference on* (pp. 1–4). IEEE.

Zhang, L., Zhao, J., & Xu, K. (2016). Who creates trends in online social media: The crowd or opinion leaders? *Journal of Computer-Mediated Communication, 21*(1), 1–16.

Zhou, C., Zhang, P., Guo, J., & Guo, L. (2014). An upper bound based greedy algorithm for mining top-k influential nodes in social networks. In *Proceedings of the 23rd international conference on world wide web* (pp. 421–422). ACM.