GitHubNet: Understanding the Characteristics of GitHub Network

Abdullah Talha Kabakus

Department of Computer Engineering, Faculty of Engineering, Duzce University, Turkey
E-mail: talhakabakus@duzce.edu.tr

Received 12 May 2020; Accepted 22 June 2020;
Publication 15 September 2020

Abstract

Web 2.0 technologies have not only raised microblogs, but also social software development and collaboration platforms. *GitHub* is the most popular software development platform that provides social collaboration. Within the scope of this study, a novel graph-based analysis model is proposed which targets to reveal (1) the characteristics of the *GitHub* in order to shed light on social software development in general, and (2) the most popular programming languages, repositories, and developers in order to shed light on the trending software development technologies. To this end, a subset of the *GitHub* network, which contains $84,737$ developers and $209,100$ repositories, was collected through the *GitHub API* and stored on a graph database namely *neo4j* to be later analyzed. The result of the analysis shows that (1) the connections in *GitHub* are not mutually linked, (2) *JavaScript*, *Python*, and *Java* are currently the most popular three programming languages, (3) *You-Dont-Know-JS*, *oh-my-zsh*, and *public-apis* are the most popular three repositories, and (4) *TarrySingh (Tarry Singh)*, *indrajithbandara (Indrajith Bandara)*, and *rootsongjc (Jimmy Song)* are the most popular three developers. Furthermore, the proposed novel analysis model can be easily applied to other social networks.

Keywords: GitHub, social coding, social network analysis, link analysis, graph database.

Journal of Web Engineering, Vol. 19_5–6, 557–574.
doi: 10.13052/jwe1540-9589.19561

1 Introduction

The effects of Web 2.0 were not only emerged in social media, but also in the way the software projects are/were developed. Web 2.0, *a.k.a. participatory web*, lets users of web send data to the websites which allow users to participate. As a natural consequence of this investment, social networking platforms such as *Facebook*, *Twitter*, etc. have been born, where the contents of these platforms are completely generated by the users. *GitHub*[1] is an online platform that lets users share their software projects, which are also called repositories, thanks to the revision control and source code management system, namely *Git*,[2] that it is built on. Alongside publicly sharing the sources of the repositories, *GitHub* lets users (who are also called *developers*) collaborate while developing software projects which allow developers from all around the world to contribute to each other's repositories even they do not know each other in real life. This collaboration is nowadays a necessity for developing software projects that are built on many different technologies (i.e. front-end, backend, etc.) from various parts of software development disciplines (i.e. designing, coding, testing) [1]. *GitHub* is commonly called a *social* coding platform as it provides interaction between developers and repositories such as (1) favoring someone's repository (similar to the *'like'* action on some several social networks), which is called *stargazer* in order to show their appreciation, and bookmarking the repository [2], (2) *watching* repositories to be notified when some changes are applied to them, (3) creating an own exact copy of an existing repository to be able to make changes on it, which is called *forking*, (4) *creating issues* regarding the repository, which may led the issue to be resolved since *GitHub* notifies the repository owner, and (5) the requests by the *forkers* when they want to *commit* the changes on their own copies to the original repository, which are called *pull requests*. These interactions are presented in Figure 1.

Developers can directly communicate with each other on *GitHub* by adding comments while making *commits*, *creating issues*, or making *pull requests*. In addition to the interactions between developers and repositories, a developer can *follow* another one in order to see his/her recent public activities (i.e. his/her new *commits*, recently created *issues*, and *pull requests*) on his/her news feed as well as showing respect or support to the developer being followed [3]. Following someone in *GitHub* is a one-way relationship, which means the developer being followed may not follow his/her follower. There

[1] https://github.com
[2] https://git-scm.com

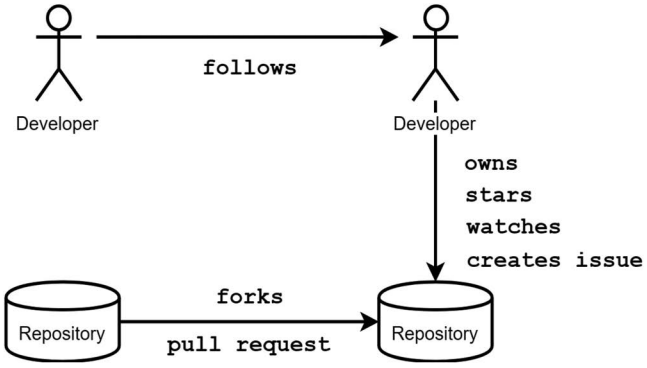

Figure 1 The interactions available on *GitHub*.

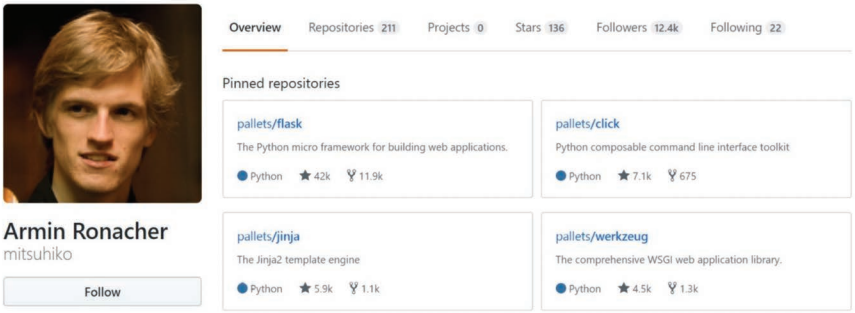

Figure 2 A sample profile page of a *GitHub* developer.

is no limit for a developer to create repositories [4]. *GitHub* provides a profile page for each developer as it is presented in Figure 2 that hosts an overview of the developer including (1) his/her location, (2) email, (3) website, (4) a brief biography of the developer, (5) the organizations of the developer, (6) repositories owned by him/her, (7) his/her starred repositories, (8) the project boards of the developer, which provide developers to create customized workflows such as comprehensive roadmaps, and release checklists, (9) his/her followers, and (10) the developers s/he follows, described as *followings*.

Similar to developers, *GitHub* provides a profile page for the repositories as it is presented in Figure 3 that hosts (1) the source code of the repository including its commits, branches, releases and contributors, (2) the created issues of the repository, (3) the pull requests of the repository, (4) the project boards of the repository, (5) the insights of the repository, and (6) a wiki for

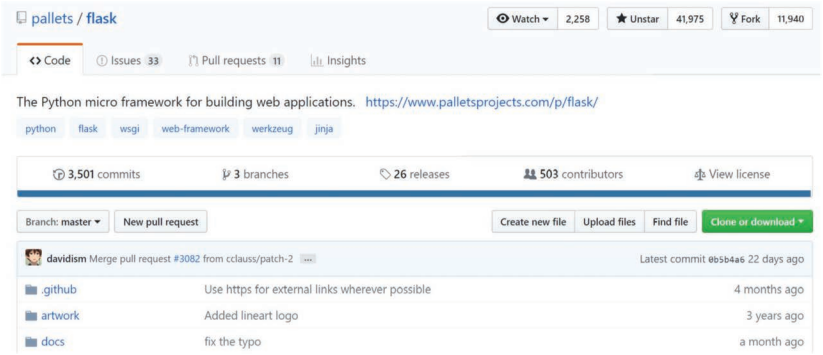

Figure 3 A sample profile page of a *GitHub* repository.

the repository that provides a place in the repository to lay out the roadmap of the project and show the current status of the repository.

GitHub allows two types of project owners: (1) personal accounts, and (2) organizational accounts. While the former is intended for individuals, the latter is intended for companies or non-profit organizations. The main difference between the personal and organizational accounts is that the organizational accounts cannot have followers [5] as, according to the *GitHub,* it is currently not supported by design [6]. A personal account can be easily converted into an organizational account but that would make the account lose all its followers.

GitHub is currently the most popular social coding site that hosts more than 96 million repositories from 31+ million developers [7]. The main motivation of this study is the idea of having a good understanding that these huge populations can indeed help to reveal some useful characteristics from *GitHub,* as a social coding platform. These characteristics include but not limited to both of the software projects and how *GitHub* is used by the developers. From this point of view, a database was constructed by crawling the *GitHub API* [8] in order to analyze *GitHub* through a subset of the entire *GitHub* network instead of the whole *GitHub* network as it would be difficult to perform the analysis on the latter due to its size and limitation on human and computation resources. The main contributions of this study are listed as follows:

- An up-to-date subset of *GitHub* network, which is publicly available[3], was constructed by preserving the connections, which model the interactions available in *GitHub*.

[3]https://www.kaggle.com/talhakabakus/githubnet

- The trending programming languages and software technologies were revealed by analyzing the constructed dataset.
- The nature of the connections between the developers in *GitHub* was revealed through the experiments conducted.
- The most influential developers and repositories were revealed by utilizing centrality measurements on the graph model constructed.
- A novel graph-based analysis model that can be easily adapted to any social networks was proposed.

The rest of the paper is structured as follows: Section 2 briefly describes the related work. Section 3 presents the material and method used by the proposed study. Section 4 presents the experimental results and discussion. Finally, Section 5 concludes the paper with future directions.

2 Related Work

Hu et al. [9] proposed an analysis of the influences of *GitHub* repositories on the basis of the *star* connection between developers and repositories. As a result of the analysis, they revealed the most influential repositories on *GitHub*. Thung et al. [10] collected $100,000$ repositories and $30,000$ developers in order to compute various characteristics of the *developer-developer* and *repository-repository* graphs constructed. In the wake of their experiments, they deduced that project networks were more interconnected than the developer networks.

Jiang et al. [11] analyzed the dissemination of the repositories in *GitHub* in terms of the speed and range on the dataset collected, which consisted of $2,665$ repositories and $747,107$ developers. According to the experimental results, they highlighted that social connections were not reciprocal, and the social links played a notable role in repository dissemination.

Ray et al. [12] carried out a study to shed light on the question of "*What is the effect of the programming language on software quality?"* through *GitHub.* To this end, they collected 729 repositories in 17 different languages from *GitHub*. According to the experimental results, their observations with regard to the effect of the language design are such that: (1) Strong typing is modestly better than weak typing, (2) static typing is better than dynamic typing, and (3) functional programming languages are better than procedural languages. They clearly noted that the effects arising from the design of the language were dominated by the process factors such as the size of the project, the size of the project team, and the commit size. Borges et al.

[2] analyzed the insights of the starring practices in *GitHub* through the $5,000$ repositories collected and conducted surveys on the developers. They concluded their study with notable conclusions as such that stars were viewed as the most useful measure of popularity in *GitHub*, and developers in *GitHub* considered the number of stars of the repository before using or contributing to the repository.

Dabbish et al. [3] studied how the developers in *GitHub* inferred other developers' technical goals and vision, which eventually advanced their technical skills and knowledge. The methodology they used was conducting a series of semi-structured interviews with 24 *GitHub* developers, which were documented in order to be analyzed in detail. *Rusk and Coady* [13] proposed an approach that characterized the developer activity by the geographic location with an aim to identify (1) the regional skilled professionals for employers, and (2) which local skills would be in greatest demand for developers. Jiang et al. [6] investigated the unfollowing behavior on *GitHub*. To this end, they constructed a dataset that contained $701,364$ developers. According to their experiments, they concluded the study that developers were more likely to unfollow those who had had fewer activities, lower programming language similarity, and asymmetric connections.

Vasilescu et al. [14] analyzed the impact of social diversity within the project teams on the software development process. To this end, they collected a team social diversity data set of $23,493$ repositories from *GitHub*. They used some measurements such as the genders and the countries of team members, the number of contributors leaving, joining, and staying in the team as well as the turnover ratio in order to analyze the impact of the social diversity. Yu et al. [15] analyzed the growth curves of *GitHub* compared with the traditional open source communities in order to reveal why *GitHub* grew in an explosive way. They also studied the *follow-network* of the developers and presented various patterns of social behavior.

Apart from *GitHub*, some researchers [16, 17] investigated the use of microblogs (i.e. *Twitter*[4]) in software development. Treude et al. [18] investigated the way *Stack Overflow*[5], which is a quite popular question and answer platform for developers, is used by developers when developing software projects. Surian et al. [19] studied the network structure of *SourceForge*[6], another popular online software development and collaboration platform.

[4] https://twitter.com
[5] https://stackoverflow.com
[6] https://sourceforge.net

3 Material and Method

GitHub provides a REST API[7] to query the data stored on its own database. Within the scope of this study, an up-to-date subset of the *GitHub* network, namely, *GitHubNet*, was constructed using this REST API of *GitHub*. Since social networks (i.e. *Twitter*, *Facebook*, *Instagram*, etc.) can be effectively modeled as graphs, the constructed dataset was stored on a graph database, namely *neo4j*[8], where the entities were modeled as the nodes, and the connections between entities were modeled as the edges between nodes. The following graphs were constructed in order to investigate the insights of the *GitHub* network.

3.1 Developer-Developer Network

Developer-Developer (D-D) network is a directed graph of developers. The direct interaction between developers that *GitHub* provides is the "*follow*" action, which lets developers follow other developers. This "*follow*" connection does not have to be mutual. Therefore, the following action was defined by the $FOLLOWS$ connection from the developer to the developer, who was being followed. Developers who contribute in at least one same project were identified as "*collaborated developers*" and a connection named $COLLABORATES$ that indicates the collaboration between two developers was established. This connection was established for each developer. The developers, whose descriptions on *GitHub* contained the "$developer$" keyword, were retrieved from *GitHub API* and the $D - D$ network was constructed. The pseudo-code of the algorithm that is used to construct the $D - D$ network is presented in Algorithm 1.

The constructed connections of the $D - D$ network, which are $FOLLOWS$, and $COLLABORATES$, are illustrated in Figure 4.

3.2 Developer-Repository Network

Developer-Repository (D-R) network is a bipartite type of network where the nodes represent the developers and the repositories. The direct interactions between a developer and a repository are (1) developers may create repositories, which are named "*owned repositories*", (2) developers may mark repositories as favorite, which is also known as *stargazer*, (3) developers may

[7] https://developer.github.com/v3/

[8] https://neo4j.com

1: fetch developers from GitHub and store as $\boldsymbol{developer\_list}$
2: for each $\boldsymbol{developer}$ *in* $\boldsymbol{developer\_list}$:
3: find the developer's followers and store as $\boldsymbol{followers}$
4: for each $\boldsymbol{follower}$ *in* $\boldsymbol{followers}$
5: establish a $\boldsymbol{FOLLOWS}$ *connection from the* $\boldsymbol{follower}$ *to the* $\boldsymbol{developer}$
6: find the developers that the $\boldsymbol{developer}$ *follows and store as* $\boldsymbol{followed\_devs}$
7: for each $\boldsymbol{followed\_dev}$ *in* $\boldsymbol{followed\_devs}$
8: establish a $\boldsymbol{FOLLOWS}$ *connection from the* $\boldsymbol{developer}$ *to the* $\boldsymbol{followed\_dev}$
9: retrieve the repositories of the $\boldsymbol{developer}$ *and store as* $\boldsymbol{repos}$
10: for each $\boldsymbol{repo}$ *in* $\boldsymbol{repos}$:
11: find the contributed developers of the repo and store as $\boldsymbol{contributed\_devs}$
12: for each $\boldsymbol{contributed\_dev}$ *in* $\boldsymbol{contributed\_devs}$:
13: establish a $\boldsymbol{COLLABORATES}$ *connection from the* $\boldsymbol{developer}$ *to the* $\boldsymbol{contributed\_dev}$
14: establish a ***COLLABORATES*** *connection from the* $\boldsymbol{contributed\_dev}$ *to the* $\boldsymbol{developer}$

Algorithm 1 The pseudo-code of the algorithm that is used to construct the $D-D$ network.

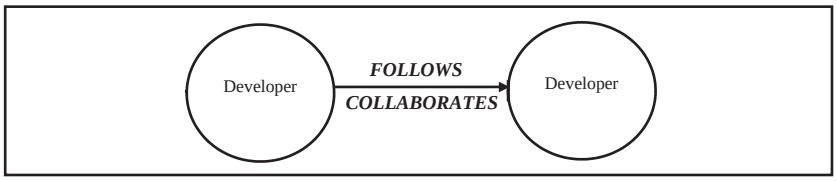

Figure 4 The constructed connections of the $D-D$ network.

watch the repositories in order to be informed when they are updated, and (4) developers may fork others' repositories in order to create the same copy of the repository that is open for making changes on it. Therefore, the $D-R$ network included a subnetwork that contains the $FORK$ connections from the original repository to the forked one. Alongside owning a repository, a developer may contribute to others' repositories by submitting changes in the code (*a.k.a. commits*), *creating issues* regarding the project, making feature requests, and proposing *pull requests* in order to submit the changes on the owned repository to the original one. The repositories that each developer contributed to were retrieved from the *GitHub API* and $CONTRIBUTES$ connections were established between the developers and the repositories they contributed to. The pseudo-code of the algorithm that was used to construct the $D-R$ network is presented in Algorithm 2.

The constructed connections of the $D-R$ network, which are $OWNS$, $STARS$, $WATCHES$, $CONTRIBUTES$, and $FORKS$, are illustrated in Figure 5.

1: fetch developers from GitHub and store as $\boldsymbol{developer\_list}$
2: for each $\boldsymbol{developer}$ *in* $\boldsymbol{developer\_list}$:
3: find the developer's $\boldsymbol{ownrepositories}$ *and store as* $\boldsymbol{owned\_repositories}$
4: for each $\boldsymbol{repo}$ *in* $\boldsymbol{owned\_repositories}$:
5: establish an $\boldsymbol{OWNS}$ *connection from the* $\boldsymbol{developer}$ *to the* $\boldsymbol{owned\_repo}$
6: retrieve the forked repositories of this repo and store as $\boldsymbol{forked\_repositories}$
7: for each $\boldsymbol{forked\_repo}$ *in* $\boldsymbol{forked\_repositories}$
8: establish a $\boldsymbol{FORKS}$ *connection from the* $\boldsymbol{forked\_repo}$ *to the* $\boldsymbol{repo}$
6: find the repositories that the $\boldsymbol{developer}$ *has starred and store as* $\boldsymbol{starred\_repositories}$
7: for each $\boldsymbol{repo}$ *in* $\boldsymbol{starred\_repositories}$:
8: establish a $\boldsymbol{STARS}$ *connection from the* $\boldsymbol{developer}$ *to the* $\boldsymbol{repo}$
9: find the repositories that the $\boldsymbol{developer}$ *has watched and store as* $\boldsymbol{watched\_repositories}$
10: for each $\boldsymbol{repo}$ *in* $\boldsymbol{watched\_repositories}$:
11: establish a $\boldsymbol{WATCHES}$ *connection from the* $\boldsymbol{developer}$ *to the* $\boldsymbol{repo}$
12: find the repositories that the $\boldsymbol{developer}$ *has forked and store as* $\boldsymbol{forked\_repositories}$:
13: for each $\boldsymbol{repo}$ *in* $\boldsymbol{forked\_repositories}$:
14: establish a $\boldsymbol{FORKS}$ *connection from the* $\boldsymbol{developer}$ *to the* $\boldsymbol{repo}$
15: find the repositories that the $\boldsymbol{developer}$ *has contributed to and store as* $\boldsymbol{contributed\_repositories}$:
16: for each $\boldsymbol{repo}$ *in* $\boldsymbol{contributed\_repositories}$:
17: establish a $\boldsymbol{CONTRIBUTES}$ *connection from the* $\boldsymbol{developer}$ *to the* $\boldsymbol{repo}$

Algorithm 2 The pseudo-code of the algorithm that is used to construct the $D - R$ network.

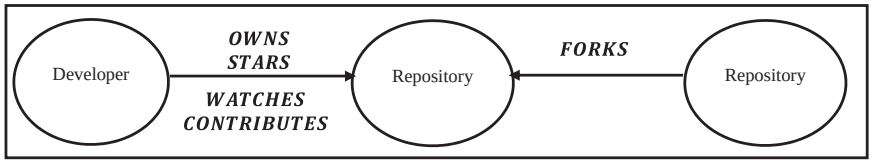

Figure 5 The constructed connections of the $D - R$ network.

3.3 Repository-Language Network

Repository-Language (R-L) network is a bipartite type of network where the nodes represent the repositories and the programming languages. *GitHub* defines a primary language for each repository. Even though a repository contains source code from different programming languages (i.e. dynamic web applications, native mobile applications, etc.), the primary programming language is set as the language of the repository. The pseudo-code of the algorithm that is used to construct the $R - L$ network is presented in Algorithm 3.

1: fetch developers from GitHub and store as $\boldsymbol{developer\_list}$
2: for each $\boldsymbol{developer}$ *in* $\boldsymbol{developer\_list}$:
3: find the developer's $\boldsymbol{ownrepositories}$ *and store as* $\boldsymbol{owned\_repositories}$
4: for each $\boldsymbol{repo}$ *in* $\boldsymbol{owned\_repositories}$
5: retrieve the language of the $\boldsymbol{repo}$ *and store as the* $\boldsymbol{lang}$
6: establish a $\boldsymbol{LANG}$ *connection from the* $\boldsymbol{repo}$ *to the* $\boldsymbol{lang}$

Algorithm 3 The pseudo-code of the algorithm that is used to construct the $R-L$ network.

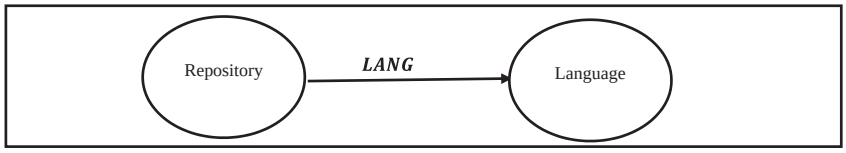

Figure 6 The constructed connection of the $R-L$ network.

The constructed connection of the $R-L$ network, which is $LANG$, is illustrated in Figure 6.

4 Experimental Results and Discussion

In order to identify the insights of the *GitHub* network, a large subset of the *GitHub* network was collected from the *GitHub API* thanks to the implemented *Python* scripts. The developers were queried through their descriptions as long as their descriptions contain the keyword "$developer$", they were collected. A generic keyword such as "$developer$" was intentionally selected to construct a comprehensive network, that covers various developers. Based on this set, the repositories of these developers were formed the repository set. Consequently, $84,737$ developers and $205,229$ repositories were collected. Since different connections establish different types of entities, which are available in *GitHub*, three different graph databases were constructed. Each graph database consisted of its own entities (nodes) and the connections (edges) between these entities. In order to understand the characteristics of each network, several measurements are needed to be calculated as follows: (1) $closeness\ centrality$, which defines the sum of the length of the shortest path between the node and all other nodes, (2) $harmonic\ centrality$, which is a variant of $closeness\ centrality$, and (3) $average\ clustering\ coefficient$, which is the mean of local clustering. In order to weigh the importance of webpages based on their links as a part of the *Google*[9] search engine, the

[9] https://www.google.com

Table 1 The analysis result of the $D - D$ network

#	PageRank	Closeness Centrality	Harmonic Centrality
1	*TarrySingh*	*gitter-badger*	*gitter-badger*
2	*indrajithbandara*	*pra85*	*TarrySingh*
3	*rootsongjc*	*TarrySingh*	*indrajithbandara*
4	*mitsuhiko*	*mitsuhiko*	*pra85*
5	*amueller*	*benbalter*	*invalid-email-address*
6	*onevcat*	*invalid-email-address*	*mitsuhiko*
7	*alvarotrigo*	*indrajithbandara*	*ReadmeCritic*
8	*ryanflorence*	*amueller*	*amueller*
9	*fengmk2*	*toddmotto*	*pborreli*
10	*jamesmontemagno*	*ReadmeCritic*	*benbalter*

Average Clustering Coefficient: 0.04967589042886072

$PageRank$ algorithm was proposed by ***Brin*** and ***Page*** [20]. The most influential and central nodes for each network were identified using $PageRank$, $closeness\ centrality$, and $harmonic\ centrality$ measurements. In addition to that, the $average\ clustering\ coefficient$ was also calculated for each network in order to reveal the mean of the local clustering. For the $D - D$ network, the $PageRank$ algorithm reveals the most influential developers since all connections in this network were established with developer nodes. Table 1 lists the analysis result of the $D - D$ network. According to the analysis result, ***TarrySingh (Tarry Singh)***,[10] ***indrajithbandara (Indrajith Bandara)***,[11] and ***rootsongjc (Jimmy Song)***[12] were the three most popular developers.

The $PageRank$ algorithm ranks the nodes (webpages) according to incoming links to them [21]. The higher the indegree of a node is, the higher the rank the node would have. Since all connections were established with repository nodes in the $D - R$ network, the $PageRank$ algorithm revealed the most influential repositories in this network as they are listed in Table 2. In addition to that, most central developers and repositories were revealed. According to the analysis result, ***You-Dont-Know-JS***[13], ***oh-my-zsh***[14], and

[10]https://github.com/TarrySingh
[11]https://github.com/indrajithbandara
[12]https://github.com/rootsongjc
[13]https://github.com/getify/You-Dont-Know-JS
[14]https://github.com/robbyrussell/oh-my-zsh

Table 2 The analysis result of the $D - R$ network

#	PageRank	Closeness Centrality	Harmonic Centrality
1	*getify/You-Dont-Know-JS*	*xujihui1985*	*getify/You-Dont-Know-JS*
2	*robbyrussell/oh-my-zsh*	*getify/You-Dont-Know-JS*	*robbyrussell/oh-my-zsh*
3	*toddmotto/public-apis*	*toddmotto/public-apis*	*xujihui1985*
4	*erikras/react-redux-universal-hot-example*	*trimstray/nginx-quick-reference*	*toddmotto/public-apis*
5	*onevcat/VVDocumenter-Xcode*	*imthenachoman/How-To-Secure-A-Linux-Server*	*getify*
6	*erikras/redux-form*	*sveinbjornt/Sloth*	*jefflau*
7	*onevcat/Kingfisher*	*mitsuhiko/pipsi*	*nikolay*
8	*tangqiaoboy/iOSBlogCN*	*eranyanay/1m-go-websockets*	*imthenachoman/How-To-Secure-A-Linux-Server*
9	*getify/Functional-Light-JS*	*leon-ai/leon*	*trimstray/nginx-quick-reference*
10	*kymjs/KJFrameForAndroid*	*denoland/deno*	*mitsuhiko/pipsi*

Average Clustering Coefficient: 0.0002476306626778342

public-apis[15] were the three most popular repositories. The neighborhood of the $D-R$ network was not complete as the $average\ clustering\ coefficient$ was calculated as low as 0.0002476306626778342 as it measures that how complete is a node's neighborhood, and the clustering coefficient of a network is the *average of clustering coefficient* of all the nodes in the network [22].

Since all connections are established through programming language nodes in the $R - L$ network, the $PageRank$ algorithm reveals the most influential programming languages in the network as they are listed in Table 3. In addition to that, the repositories and programming languages were revealed through their centralities. According to the analysis result,

[15]https://github.com/toddmotto/public-apis

Table 3 The analysis result of the R-L network

#	PageRank	Closeness Centrality	Harmonic Centrality
1	*JavaScript*	*JavaScript*	*JavaScript*
2	*Python*	*getify/A-Tale-Of-Three-Lists*	*getify/A-Tale-Of-Three-Lists*
3	*Java*	*getify/asyncGate.js*	*getify/asyncGate.js*
4	*PHP*	*getify/asynquence*	*getify/asynquence*
5	*HTML*	*getify/breakthewebforward.com*	*getify/breakthewebforward.com*
6	*Ruby*	*getify/CAF*	*getify/CAF*
7	*CSS*	*getify/cloud-sweeper*	*getify/cloud-sweeper*
8	*C#*	*getify/deePool*	*getify/deePool*
9	*Objective-C*	*getify/DOMEventBridge*	*getify/DOMEventBridge*
10	*TypeScript*	*getify/ES-Feature-Tests*	*getify/ES-Feature-Tests*

Average Clustering Coefficient: 0.0

JavaScript, *Python*, and *Java* were the three most popular programming languages when the *GitHub* knowledge-base was queried for the keyword "$developer$". Similar to the $D - R$ network, the neighborhood of the $R - L$ network was not complete as the *average clustering coefficient* was calculated as low as 0.

As described in the related work in Section 2, there are some studies that provide analysis regarding software development over social networks including *GitHub*, *Stack Overflow*, *SourceForge*, and even *Twitter,* which does not have a purpose to aid the software development process. A comparison of the related work to the proposed one, namely *GitHubNet*, in terms of (1) the number of developers each dataset contains, (2) the number of repositories each dataset contains, and (3) the year of each dataset constructed is listed in Table 4.

Each *GitHub* repository may contain a *README* file (commonly named as $README.md$) that describes the project proposed in the repository. This file commonly contains some technical background regarding repositories. The descriptions of the collected repositories were fetched from the *GitHub API* in order to reveal the trending technologies and topics from the insights. Since these *README* files contain markup symbols that define the styles of the text, the fetched text was cleared from these symbols as well as punctuation marks, digits and the stop words in English (e.g. "the", "a", "an", "is", etc.). When the descriptions of the $85,734$ original repositories (the repositories which are not forked from another repository)

Table 4 A comparison of the proposed work with the related work

Related Work	Number of Developers	Number of Repositories	Year of the Dataset Constructed
Jiang et al. [11]	747,107	2,665	2013
Luo et al. [1]	499,485	108,718	2014
Surian et al. [19]	55,694	28,087	2009
Bana and Arora [22]	8,000	NA	2018
Thung et al. [10]	30,000	100,000	2013
Leibzon [23]	NA	NA	2016
Vasilescu et al. [14]	NA	23,493	2014
Jiang et al. [6]	701,364	NA	2016
GitHubNet	**84,737**	**209,100**	**2019**

Figure 7 The word cloud of the descriptions of the 85,734 original repositories.

were analyzed, the notable words, that give clue about the most demanding software development technologies, were found as *JavaScript,* which is the most popular programming language according to the *Stack Overflow Developer Survey 2019* [24], *Node* which is the most popular web framework based on JavaScript [24], and *React,* which is the $4th$ most popular repository[16] on *GitHub* alongside the 96+ million repositories. The word cloud of the analysis is presented in Figure 7.

[16] According to the http://gitmostwanted.com/top/stars/ and based on the number of stars.

5 Conclusion

Web 2.0 technologies have not only raised microblogs and social networking sites, but also social software development and collaboration platforms. *GitHub* is the most popular software development platform that provides social collaboration. Within the scope of this study, a novel graph-based analysis model is proposed which targets to reveal (1) the insights of the *GitHub* in order to shed light on social software development in general, and (2) the most popular programming languages, repositories, and developers in order to shed light on the trending software development technologies. For this reason, a subset of the *GitHub* network, which contained $84,737$ developers and $209,100$ repositories, was collected through the *GitHub API* by querying for the keyword "$developer$" and stored on a graph database namely *neo4j* to perform analysis on it. The result of the analysis shows that (1) the connections in *GitHub* are not mutually linked, (2) *JavaScript*, *Python*, and *Java* are currently the most popular three programming languages, (3) *You-Dont-Know-JS*, *oh-my-zsh*, and *public-apis* are the most popular three repositories, and (4) *TarrySingh (Tarry Singh)*, *indrajithbandara (Indrajith Bandara)*, and *rootsongjc (Jimmy Song)* are the most popular three developers.

As future work, longitudinal research to investigate how the *GitHub* network structure evolves over time can be studied. To this end, various snapshots of the *GitHub* network are needed to be collected and analyzed over time.

References

[1] Z. Luo, X. Mao, and A. Li, "An Exploratory Research of GitHub Based on Graph Model," in *Proceedings – 2015 9th International Conference on Frontier of Computer Science and Technology*, FCST 2015, 2015, pp. 96–103.

[2] H. Borges and M. Tulio Valente, "What's in a GitHub Star? Understanding Repository Starring Practices in a Social Coding Platform," *J. Syst. Softw.*, vol. 146, pp. 112–129, 2018.

[3] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb, "Social Coding in GitHub: Transparency and Collaboration in an Open Software Repository," in *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work (CSCW '12)*, 2012, pp. 1277–1286.

[4] "Pricing – Plans for every developer," GitHub, 2019. [Online]. Available: https://github.com/pricing. [Accessed: 06-Dec-2019].

[5] K. Blincoe, J. Sheoran, S. Goggins, E. Petakovic, and D. Damian, "Understanding the popular users: Following, affiliation influence and leadership on GitHub," *Inf. Softw. Technol.*, vol. 70, pp. 30–39, 2016.
[6] J. Jiang, D. Lo, Y. Yang, J. Li, and L. Zhang, "A first look at unfollowing behavior on GitHub," *Inf. Softw. Technol.*, vol. 105, pp. 150–160, 2019.
[7] "The State of the Octoverse," GitHub, 2019. [Online]. Available: https://octoverse.github.com. [Accessed: 06-Dec-2019].
[8] "GitHub API v3 | GitHub Developer Guide," GitHub, 2019. [Online]. Available: https://developer.github.com/v3/. [Accessed: 06-Dec-2019].
[9] Y. Hu, J. Zhang, X. Bai, S. Yu, and Z. Yang, "Influence analysis of Github repositories," *Springerplus*, vol. 5, no. 1, pp. 1–19, 2016.
[10] F. Thung, T. F. Bissyandé, D. Lo, and L. Jiang, "Network structure of social coding in GitHub," in *Proceedings of the 17th European Conference on Software Maintenance and Reengineering (CSMR 2013)*, 2013, pp. 323–326.
[11] J. Jiang, L. Zhang, and L. Li, "Understanding Project Dissemination on a Social Coding Site," in *Proceedings of 20th Working Conference on Reverse Engineering (WCRE 2013)*, 2013, pp. 132–141.
[12] B. Ray, D. Posnett, P. Devanbu, and V. Filkov, "A large-scale study of programming languages and code quality in GitHub," *Commun. ACM*, vol. 60, no. 10, pp. 91–100, 2017.
[13] D. Rusk and Y. Coady, "Location-based analysis of developers and technologies on GitHub," in *Proceedings of 2014 28th International Conference on Advanced Information Networking and Applications Workshops*, 2014, pp. 681–685.
[14] B. Vasilescu, A. Serebrenik, and V. Filkov, "A Data Set for Social Diversity Studies of GitHub Teams," in *Proceedings of 2015 IEEE/ACM 12th Working Conference on Mining Software Repositories, 2015*, pp. 514–517.
[15] Y. Yu, G. Yin, H. Wang, and T. Wang, "Exploring the patterns of social behavior in GitHub," in *Proceedings of the 1st International Workshop on Crowd-based Software Development Methods and Technologies (CrowdSoft 2014)*, 2014, pp. 31–36.
[16] G. Bougie, J. Starke, M.-A. Storey, and D. M. German, "Towards understanding twitter use in software engineering: Preliminary findings, ongoing challenges and future questions," in *Proceedings of the 2nd International Workshop on Web 2.0 for Software Engineering (Web2SE '11)*, 2011, pp. 31–36.

[17] Y. Tian, P. Achananuparp, I. N. Lubis, D. Lo, and E. P. Lim, "What does software engineering community microblog about?," in *IEEE International Working Conference on Mining Software Repositories*, 2012, pp. 247–250.

[18] C. Treude, O. Barzilay, and M.-A. Storey, "How do programmers ask and answer questions on the web?: NIER track," in *Proceeding of the 33rd international conference on Software engineering (ICSE '11)*, 2011, pp. 804–807.

[19] D. Surian, D. Lo, and E. P. Lim, "Mining Collaboration Patterns from a Large Developer Network," in *Proceedings of 2010 17th Working Conference on Reverse Engineering*, 2010, pp. 269–273.

[20] S. Brin and L. Page, "The anatomy of a large scale hypertextual Web search engine," *Comput. Networks ISDN Syst.*, vol. 30, pp. 107–117, 1998.

[21] E. Agirre and A. Soroa, "Personalizing PageRank for word sense disambiguation," in *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, 2009, pp. 33–41.

[22] R. Bana and A. Arora, "Influence Indexing of Developers, Repositories, Technologies and Programming languages on Social Coding Community GitHub," in *2018 Eleventh International Conference on Contemporary Computing (IC3)*, 2018, pp. 1–6.

[23] W. Leibzon, "Social network of software development at GitHub," in *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2016)*, 2016, pp. 1374–1376.

[24] "Stack Overflow Developer Survey 2019," Stack Overflow, 2019. [Online]. Available: https://insights.stackoverflow.com/survey/2019. [Accessed: 06-Dec-2019].

Biography

Abdullah Talha Kabakus received the bachelor's degree in computer engineering from Cankaya University in 2010, the master's degree in computer engineering from Gazi University in 2014, and the philosophy of doctorate degree in Electrical-Electronics & Computer Engineering from Duzce University in 2017, respectively. He is currently working as an Assistant Professor at the Department of Computer Engineering, Faculty of Engineering, Duzce University. His research areas include mobile security, deep learning, and social network analysis. He has been serving as a reviewer for many highly-respected journals.