Intelligent Automation & Soft Computing
DOI:10.32604/iasc.2021.015034
Article

Tech Science Press

Evolution of Influential Developer's Communities in OSS and its Impact on Quality

Beenish Khan[1], Muhammad Rafiq Mufti[2], Asad Habib[3], Humaira Afzal[4], Mohammad Abdul Moiz Zia[5], Afshan Almas[6], Shahid Hussain[7,*] and Bashir Ahmad[1]

[1]Department of Computer Science, COMSATS University, Islamabad, 45550, Pakistan
[2]Department of Computer Science, COMSATS University, Vehari, 61100, Pakistan
[3]Department of Computer Science, Kohat University of Science & Technology, Kohat, 26000, Pakistan
[4]Department of Computer Science, Bahaddin Zakeria University, Multan, 60800, Pakistan
[5]Department of Computer Science, University of Central Punjab, Lahore, 54590, Pakistan
[6]Institute of Computer Science & Information Technology, The Women University Multan, 60800, Pakistan
[7]Department of Computer and Information Science, University of Oregon, Eugene, 97401, Oregon, USA
*Corresponding Author: Shahid Hussain. Email: shussain@uoregon.edu
Received: 03 November 2020; Accepted: 04 February 2021

Abstract: The high turnover of developers in the Open-Source Software (OSS) systems is due to the lack of restriction on a developer's involvement and contributions. The primary developers start and administer an OSS project. However, they do not manage those who contribute. The literature shows that 80% of issues are resolved by 20% of developers when developing an OSS. Therefore, identifying influential developer communities is quite necessary for OSS stakeholders to reduce the efforts required to solve the issue through releases and predict quality. The purpose of this proposed empirical study is to explore influential communities by analyzing the relationship between their members as an OSS evolves and its impact on software quality. We performed several experiments with releases of three widely used OSS, namely "BIGDL," "INCUBATOR-MXNET" and "RECOMMENDERS." The major implications of the proposed study include; 1) The community development structure is not centralized and controlled, 2) Influential communities were observed in early releases of an OSS, 3) There is no guarantee of an influential community in the consecutive releases, 4) Notable developers are varied through the releases, and 5) The presence of influential communities in subsequent releases could lead to the maturity of an OSS.

Keywords: OSS; developers' community; influential community; influential members; quality

1 Introduction

Over the past two decades, many organizations have focused on Open-Source Software (OSS) systems due to their flexibility, agility, and cost-effectiveness. According to a recent report, more than 95% of IT organizations have adopted OSS development [1]. The development of OSS has resulted in many

well-known and high-quality projects such as the Mozilla browser, Linux operating system, Hadoop, and MySQL database system framework [2]. The process of developing an OSS differs from traditional software development in several ways. Although primary members manage the development of an OSS, they do not manage the contributor's involvement because of their primary focus on software artifacts. Moreover, contributing members of a project are not directly affected. As a result, no project manager can impose organizational constraints on the contributors [3]. In the open-source community, developers are privileged to participate in a project from any place in the world [4]. Members can help the project in a number of ways by submitting bug reports, lending technical knowledge, writing documentation, and improving the source code in various areas of the code base. In this way, the community of contributors is seen as a more critical aspect of OSS projects. In the OSS community, the turnover of developers is high as there are no restrictions on their contribution to a project. We investigated the fact that 80% of the work is done by 20% of OSS developers [5]. It leads to the notion of sub-communities within the community. Sub-communities include members who are actively involved in the project and who bring a rich information. These communities are seen as influential communities which could be involved in the effective development of each release [6].

The purpose of the proposed study is to empirically investigate the community development structure, the relationship between individuals in the community, the consistency of the community structure in all releases of an OSS, and the impact of influential community on software quality. The proposed study is carried out in a series of steps. First, we explore information on the structure of the developer community from Github repositories. Second, we leverage the capabilities of association rule mining [7] to determine the influential communities. Third, we used "Confidence" and Eigenvector [8] as proxies to identify the relationship between subgroup community members and the strength of community members across the releases. To achieve the research goal, we develop the following research questions.

Research Question-1 (RQ-1): Is there an influential community exists within the developer network. How do they remain engaged and consistent throughout the releases?

Research Question-2 (RQ-2): How likely is there a relationship among influential members of the community?

Research Question-3 (RQ-3): Which outstanding developers from influential communities could help improve the quality of OSS project code in terms issue solving ?

The remainder of the paper is organized as follows: Section 2 looks at the related work. Section 3 provides a description of the proposed methodology. Section 4 provides the findings and discussion. Section 5 talks about threats to validity. Section 6 concludes the paper and examines future directions.

2 Related Work

There are different roles in OSS software projects like primary developers, contributing developers, and bug reporters. However, developers and Bug reporters may be part of a primary development teams [9] depending on their participation. In addition, some developers may lose their interest in the project and let the project that could affect its quality [4,10]. Developers who actively participate in developing OSS software projects could create sub-communities or subgroups [6]. Dias et al. [11] have analysed the behavior of internal and external contributors of five well-known open-source projects. They noted that external developers have pulled 56.7% of requests. However, it was noted that contributions by external developers are shorter than those of internal developers. Crowston et al. [12] looked at the relationship between primary and peripheral members. They came to the conclusion that core developers contribute more than peripheral members.

Agrawal et al. [13] analysed hero projects very carefully by analysing the frequency of closed issues and bugs that are not significantly affected by project types (Public or Enterprise). However, for an important association between developers of hero projects, only project types and improving resolution rates are observed. Zhan [6] pointed to differences in developer community interests by analysing their collaboration across community networks. Community network's nodes could play a vital role by influencing neighboring nodes and effectively transmitting important information. To identify the upper k-nodes, Zhan [6] gathered datasets from the Stanford Large Network Dataset Collection (SNAP). He came up with a model that uses average centrality of Katz.

Xuguang [14] proposed a technique for assessing community performance using the complex network and analysed its similarity with an ideal solution (TOPSIS). They identified the influential nodes of the Facebook network as a function of their relative closeness. Chin and Chignell [15] pointed out how subgroups are formed in social networks. They extracted data from the Toronto's TorCamp Google group, where members actively responded to threats. They used hierarchical cluster analysis and k-plex analysis to determine sub-communities. The two techniques yielded the same results with 11 to 14 members in the sub-communities.

In other research, Chin et al. [16] identified subgroups consistent with the development of time in social networks. They used the Social Cohesion Analysis of Networks (SCAN) and Data-Intensive Socially Similar Evolving Community Tracker (DISSECT) for the detection of cohesive subgroups. Besides, Przybilla et al. [17] empirically examined the relationship between community size and the core subgroup members.

3 Proposed Methodology

In this article, we looked closely at the involvement of developer communities and the impact of such participation on the strength of a project as it evolves. We did several experiments by selecting some open-source projects from Github. We have included the projects of Apache, Microsoft, Tensorflow, Google, Intel-Analytics, and low RISC. We excluded personal use and inactive projects because inactive repositories are considered the major perils of GitHub data [18]. Our main goal is to mine the influential developer communities and their power in OSS projects. To accomplish this, we extracted the data and metadata from developers. We did this with the GitHub web API. The proposed methodology workflow is illustrated in Fig. 1.

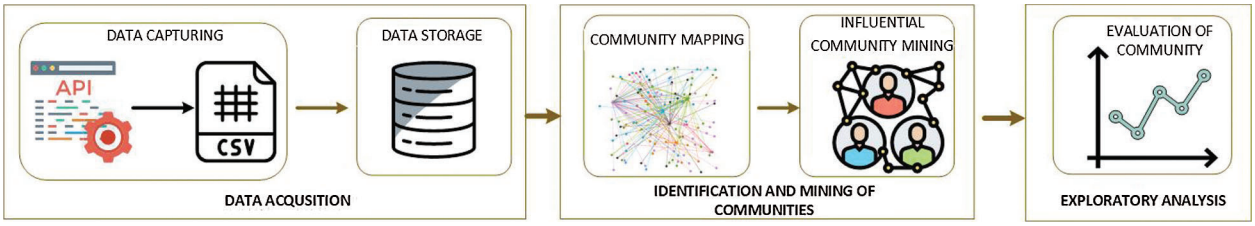

Figure 1: Layout of the proposed method

3.1 Data Acquisition

3.1.1 Data Capturing

To implement our methodology, we began by extracting project data from GitHub repositories. According to Octoverse, there are more than 44 million repositories and 20 million issues on Github (September 2019) [19]. To extract data from repositories, we used the GitHub Wrapper tool that is based on the GitHub REST API V3. The API is accessible using JSON results of HTTP Requests that may be displayed on the web browser and stored in CSV format. From the literature supporting the empirical

studies, we used the criteria presented in Tab. 1 to determine if a project is suitable for data collection or not. After reviewing thousands of GitHub projects, we chose 15 open-source projects with nearly 30,000 issues and pull requests.

Table 1: Descriptive statistics of control variables

Control Variables	Value
Duration >	6 months
No. of Issues (closed)>	200
No. of Releases >	3
Pull requests >	10
Collaborators >	20

3.1.2 Data Storage

After capturing the data, it was crucial to convert it into an appropriate form to generate a community network. All JSON files have been converted into CSV files through the python script and have been stored in a MySQL database. The layout of the data set structure is shown in the Tab. 2. Initially, we reviewed 14 attributes to capture this information. However, we preprocessed the trash fields. Our main goal is to showcase influential developer communities in open-source projects and validate the importance of developers across the ecosystem in all releases. We extracted data about developer involvement in all releases of an OSS project. We then divided the data depending on the release of each project. For this purpose, we applied SELECT queries in HeidiSQL (https://www.heidisql.com/) that are based on the date when the issues were created. After the implementation of these queries, finally, we get data from release wise developers and stored in separate CSV files.

Table 2: Overview of dataset structure

Variable	Description
project_name	This variable describes the name of a project.
bug_id	This variable describes the ID of a reported bug.
bug_number	This variable describes the number assigned to the reported bug.
Title	This variable describes the title of a reported bug.
user_id	This variable describes the ID of a user who reports the bug.
user_login	This variable describes the login of a user who reports the bug.
State	This variable describes the current state of a reported bug.
Locked	This variable describes the status (open/closed) of the reported bug.
assignee_id	This variable describes the ID of a developer who is assigned to resolve the bug.
assignee_login	This variable describes the login of a developer who is assigned to resolve the bug.
comment_count	This variable describes the total number of comments associated with a bug.
created_at	This variable describes the date when a bug is reported.
updated_at	This variable describes the date when a bug status is changed.
closed_at	This variable describes the date when a bug has been resolved.

3.2 Identification and Mining of Communities

Once we mapped the data for each project, we came up with a model to generate the community. Subsequently, we converted these releases into CSV files along the edges and nodes. To model and identify communities, the capabilities of the Gephi tool (https://gephi.org/) were leveraged. Gephi is one of the best known tools for doing network analysis. Using this tool, we imported the list edges and nodes for each release.

3.2.1 Community Mapping

Based on a community graph G = (V, E), the terms V and E represents contributors and their relationship respectively. Apart from this, E is all the edges where each edge assigns a weight to show the link between two nodes. Similarly, the set of V nodes refers to the list of contributors that make any sort of contribution, like opening an issue or making a new commit. Subsequently, we leveraged Gephi capabilities to extract the community graph. Consequently, our G graph consists of developers on nodes and bug numbers on edges for a particular release. We used the "modularity" measure to identify the communities within the G of each OSS project [20].

3.2.2 Influential Communities Mining

After identifying OSS project communities using the "modularity" measure, the next step is to identify and extract influential communities. To achieve this, we exploited information from developer's communities. Subsequently, we leveraged the capability of Association Rule Mining (ARM) to find data patterns to explore influential communities. ARM identifies the features/dimensions that have occurred together or in correlation with each other [7,21]. We used a well-known variant of ARM, namely the Apriori algorithm, to find common patterns in data [22]. We used the Support and Confidence measures to rank and assign an influential label to the developer's community. The Support measure (Shown in Eq. 1) is the fraction of transactions that contain an itemset, i.e., the ratio of the number of times a community occurs by the total number of items/records [23].

$$Support = \frac{Number\ of\ times\ a\ commnuity\ occur}{Total\ number\ of\ records} \tag{1}$$

The Confidence measure points to the relationship between members of an influential community [24]. Influential communities are extracted from those releases using a Support measure of the Apriori algorithm, i.e., We have defined the minimum support of two, which means that every itemset (i.e., community in our case) should be taken into account at least twice in the overall project. When a community meet this criterion, it refers to an influential community.

3.3 Exploratory Analysis

After extracting the influential communities, the next step is to analyse the relationship among their developers. In this regard, we used the confidence measure of the Apriori algorithm to analyze the association of members. We have defined the minimum confidence to be 0.5. Subsequently, we investigate the strength the developers of an influential community'. We used a centrality measure,namely Eigenvector centrality (Shown in Eq. 2). The Eigenvector centrality measure is used to determine the significance of a node according to the importance on its neighbors.. The magnitudes of the Eigenvector elements include useful information concerning a network. Consequently, it is the "strength of communities" where the vertices refer to the developer's position in the community network [8].

$$\lambda C_i^{(eig)} = \sum_{j=1}^{N} a_{ji} C_j^{(eig)} \tag{2}$$

Where λ is proportionality constant. The a_{ji} term emphasizes that node i receives the contribution to centrality from its neighbors through the incoming links. $C_j^{(eig)}$ is a vector of centralities [25].

4 Results and Discussions

In this section, we examined the findings of the proposed methodology by conducting several experiments and organizing the findings to answer our research questions. We analyzed the formation of sub-communities or subgroups and their development.

4.1 Response to RQ-1

First, we assumed several communities of the developer network [5,6] of a release. In this respect, we leverage ARM's capabilities to extract a number of possible patterns from the data. Each pattern relates to a community within the developer's network. Second, we identified the communities with the highest Support relative to the threshold value and designated them as "Influential communities" of a developer network. We included 15 projects for the empirical investigation. However, in this analysis, we only considered projects that meet the criteria given in the Tab. 1. In this respect, the results of each release of "BIGDL" "INCUBATOR-MXNET" and "RECOMMENDERS" are shown in the Tabs. 3–8 respectively. In the case of the "BIGDL" project, we first identify the influential and non-influential communities of each release as shown in Tab. 3.

Table 3: Descriptive statistics of the total number of communities and influential communities of "BIGDL" releases-wise

Release Number	Release URL	Release Duration		Total Number of Communities	Total Number of Influential Communities
		Start Date	End Date		
1	https://github.com/intel-analytics/BigDL/releases/tag/v0.1.0	8-Aug-16	7-Apr-17	2	6
2	https://github.com/intel-analytics/BigDL/releases/tag/v0.1.1	7-Apr-17	16-Jun-17	4	7
3	https://github.com/intel-analytics/BigDL/releases/tag/v0.2.0	16-Jun-17	24-Jul-17	4	5
4	https://github.com/intel-analytics/BigDL/releases/tag/v0.3.0	24-Jul-17	8-Nov-17	4	7
5	https://github.com/intel-analytics/BigDL/releases/tag/v0.4.0	8-Nov-17	4-Jan-18	5	5
6	https://github.com/intel-analytics/BigDL/releases/tag/v0.5.0	4-Jan-18	30-Mar-18	4	5
7	https://github.com/intel-analytics/BigDL/releases/tag/v0.6.0	30-Mar-18	29-Jun-18	2	1
8	https://github.com/intel-analytics/BigDL/releases/tag/v0.7.0	29-Jun-18	13-Oct-18	4	0
9	https://github.com/intel-analytics/BigDL/releases/tag/v0.8.0	13-Oct-18	28-Mar-19	2	0
10	https://github.com/intel-analytics/BigDL/releases/tag/v0.9.0	28-Mar-19	22-Jul-19	2	0
11	https://github.com/intel-analytics/BigDL/releases/tag/v0.10.0	22-Jul-19	5-Nov-19	3	0

The major implications of the Tab. 3 results are as follows;

- The number of the developer communities (i.e., The total number of influential and non-influential communities) varies according to the releases of the "BIGDL" project, which describes the incoherence in the participation and the interest of the developers for each release. It also describes the irregularity in the forming a team.
- We have looked at the limited number of developer communities in the later releases. It indicates an increase in the quality and stability of the project in terms of decreasing the number of issues during the evolution of "BIGDL."
- We observed less variation in the number of influential communities during the initial releases of "BIGDL." This indicates that many proponent communities are interested in addressing early development issues and stabilizing the project.
- The number of non-influential communities present in future releases of "BIGDL" varies. Consequently, we do not observe any influential community that indicates random developer participation in issue resolving.

Subsequently, we extracted the eight developers' communities and presented their involvement (as an influential community) through the releases of "BIGDL" as shown in the Tab. 4. Although, we have observed the involvement of the developers' community (members are yangw1234 and yiheng) for the development of "BIGDL" in the first seven releases. However, this community becomes influential on the second release. Similarly, we have observed that few developer communities seek to influence in different releases. For example, the developer's community (i.e., SeaOfOcean and jenniew) is looking for an influential community of Release-1, Release-4, Release-5, and Release-6, which indicate a change in community interest through the releases.

Table 4: Influential communities of "BigDL" release-wise

Releases of BIGDL	Influential Communities							
	SeaOfOcean, jenniew	qiuxin2012, dding3	zhangxiaoli73, dding3	jason-dai, helenlly	i8run, yiheng	shane-huang, zhichao-li	yangw1234, yiheng	qiuxin2012, zhangxiaoli73, dding3
Release-1	✓	✓	✓		✓	✓		✓
Release-2		✓	✓	✓	✓	✓	✓	✓
Release-3		✓	✓		✓		✓	✓
Release-4	✓	✓	✓	✓		✓	✓	✓
Release-5	✓	✓		✓		✓	✓	
Release-6	✓			✓	✓	✓	✓	
Release-7							✓	
Release-8								
Release-9								
Release-10								
Release-11								

In the case of the "INCUBATOR-MXNET" project, we first identified the influential and non-influential communities of each release, as shown in Tab. 5. The major consequences of the Tab. 5 results are as follows;

Table 5: Descriptive statistics of the total number of communities and influential communities of "INCUBATOR-MXNET" releases-wise

Release Number	Release URL	Release Duration		Total Number of Communities	Total Number of Influential Communities
		Start Date	End Date		
1	https://github.com/apache/incubator-mxnet/tree/1.0.0	16-Nov-17	4-Dec-17	8	2
2	https://github.com/apache/incubator-mxnet/releases/tag/1.0.0	4-Dec-17	20-Feb-18	9	5
3	https://github.com/apache/incubator-mxnet/releases/tag/utils	20-Feb-18	23-Apr-18	6	9
4	https://github.com/apache/incubator-mxnet/releases/tag/1.2.0	23-Apr-18	22-May-18	6	9
5	https://github.com/apache/incubator-mxnet/releases/tag/1.2.1	22-May-18	17-Jul-18	4	5
6	https://github.com/apache/incubator-mxnet/releases/tag/1.3.0	17-Jul-18	12-Sep-18	9	0
7	https://github.com/apache/incubator-mxnet/releases/tag/1.3.1	12-Sep-18	13-Nov-18	4	0
8	https://github.com/apache/incubator-mxnet/releases/tag/1.4.0	13-Nov-18	16-Feb-19	5	0
9	https://github.com/apache/incubator-mxnet/releases/tag/1.4.1	16-Feb-19	30-Apr-19	5	0
10	https://github.com/apache/incubator-mxnet/releases/tag/1.5.0	30-Apr-19	8-Jun-19	6	0
11	https://github.com/apache/incubator-mxnet/releases/tag/1.5.1	8-Jun-19	5-Sep-19	9	0

- Like the "BIGDL" project, the number of developers' communities (total number of influential and non-influential communities) varies depending on the releases of the "INCUBATOR-MXNET" project, which describes the inconsistency in the developer engagement and interest in each release. In addition, as "BIGDL," we observed irregularity in the formation of the "INCUBATOR-MXNET" team structure.
- Like the "BIGDL" project, we observed limited number of developers' communities in the later releases. This indicates an increase in the quality and stability of the project in terms of reducing the number of issues during the evolution of "INCUBATOR-MXNET."
- We observed less variation in the number of influential communities during the initial releases of "INCUBATOR-MXNET" which describes the involvement of several developer communities in resolving development issues and evolution of the project.
- Like the "BIGDL" project, we do not see any influential community in subsequent releases of "INCUBATOR-MXNET" that indicate the random community participation in the resolution of reported issues.

Subsequently, we extracted the nine developer' communities and presented their involvement (as an influential community) through the releases of "INCUBATOR-MXNET." The results are shown in Tab. 6. We observed the involvement of the developer community (members are ZiyueHuang and eric-haibin-lin) as influential in the development of "INCUBATOR-MXNET" in the first five releases.

Table 6: Influential communities of "INCUBATOR-MXNET" release-wise

Influential Communities	ThomasDelteil, piiswrong	anirudh2290, ZiyueHuang	ZiyueHuang, eric-haibin-lin	haojin2, ZiyueHuang	anirudh2290, eric-haibin-lin	zhreshold, dwSun	haojin2, eric-haibin-lin	ZiyueHuang, eric-haibin-lin, anirudh2290	haojin2, eric-haibin-lin, ZiyueHuang
Release-1			✓			✓			
Release-2		✓	✓		✓	✓		✓	
Release-3	✓	✓	✓	✓	✓	✓	✓	✓	✓
Release-4	✓	✓	✓	✓	✓	✓	✓	✓	✓
Release-5	✓		✓	✓			✓		✓
Release-6									
Release-7									
Release-8									
Release-9									
Release-10									
Release-11									

Similarly, we observe that there are several developers' communities that are looking to influence the limited number of releases. This indicates either the lack of consistency in the formation of team structure or the rise of a limited number of issues. In the case of the "RECOMMENDERS" project, we first identify the influential and non-influential communities of each release as shown in the Tab. 7.

Table 7: Descriptive statistics of the number of communities and influential communities of "RECOMMENDERS" release-wise

Release Number	Release URL	Release Duration		Total Number of Communities	Total Number of Influential Communities
		Start Date	End Date		
1	https://github.com/microsoft/recommenders/releases/tag/0.1.0	19-Sep-18	12-Nov-18	2	3
2	https://github.com/microsoft/recommenders/releases/tag/0.1.1	12-Nov-18	12-Dec-18	3	3
3	https://github.com/microsoft/recommenders/releases/tag/2019.02	12-Dec-18	20-Feb-19	2	4
4	https://github.com/microsoft/recommenders/releases/tag/2019.06	20-Feb-19	3-Jun-19	5	3
5	https://github.com/microsoft/recommenders/releases/tag/2019.09	3-Jun-19	18-Sep-19	4	1

The major consequences of the Tab. 7 results are as follows:

- The number of developers' communities (total number of influential and non-influential communities) varies depending on the releases of the "RECOMMENDERS" projectwhich describes the inconsistency in developers' engagement and interest in each release as "BIGDL" and "INCUBATOR-MXNET" projects.
- We have observed the involvement of influential communities throughout in all releases of "RECOMMENDERS." That could be considered to signify the stability of the community in terms of issues solving.

- We observed less variation in the number of influential communities in the initial releases of "RECOMMENDERS" which describes the interest of several developer communities to solve early development issues and stabilize the project.

Subsequently, we extracted the four developer communities and presented their involvement (as an influential community) through the releases of "RECOMMENDERS" as shown in Tab. 8. We observed the involvement of developer communities (such as (yueguoguo and anargyri) and (miguelgfierro and loomlike)) as influential communities for the development of "RECOMMENDERS" in all releases.

Table 8: Influential communities release-wise of "RECOMMENDERS"

Influential Communities	yueguoguo, anargyri	eisber, dciborow	gramhagen, yueguoguo	miguelgfierro, loomlike
Release-1	✓	✓		✓
Release-2	✓	✓		✓
Release-3	✓	✓	✓	✓
Release-4	✓		✓	✓
Release-5			✓	

Similarly, we observed the existence of some developer communities that seemed influential in various releases. As in, the developer community (i.e. gramhagen and yueguoguo) is looking for influential ones in the first three releases. This indicates the variability and inconsistency in the interest of the community through the releases.

4.2 Response to RQ-2

In response to our second research question, we examined the relationship between members of influential communities, whether or not it was a coincidence. In this respect, we relied on the "Confidence" measure of the Apriori algorithm to assess the likelihood between entities (i.e., Members in our case). Consequently, we evaluated the association of influential community members using the "Confidence" measure. We assumed that the involvement members in a community is coincidental if the value of "Confidence" measure is less than 0.50. This means that if a member of an influential community will be present with 0.5 confidence in a release, then other members will also be present in that release. In the case of the "BIGDL" project, the value of the "Confidence" measure of each influential community is shown in Fig. 2a. The major consequences are;

- We observed a strong co-occurrence of "shane-huang" and "zhichao-li" members in all releases. This means that if "shane-huang" were present in a single release, there would be an 83% chance of being involved in "zhichao-li" and vice versa. We also observed the joint presence of "yangw1234" and "yiheng" with a confidence value of 0.75 (i.e., 75%).
- We observed the random co-occurrence of "i8run" and "yiheng" with a confidence value of 0.50 (i.e., 50%).
- We might observe the developer co-occurrence regarding member's perspective, such as in the case of an influential community (i.e., A community of "qiuxin2012" and "dding3" members), the existence of "qiuxin2012" in a team indicates an 83% chance of "dding3" in the same team. Similarly, for the existence of "dding3" in the team, there is a 55% chance of "qiuxin2012" being part of the same team.

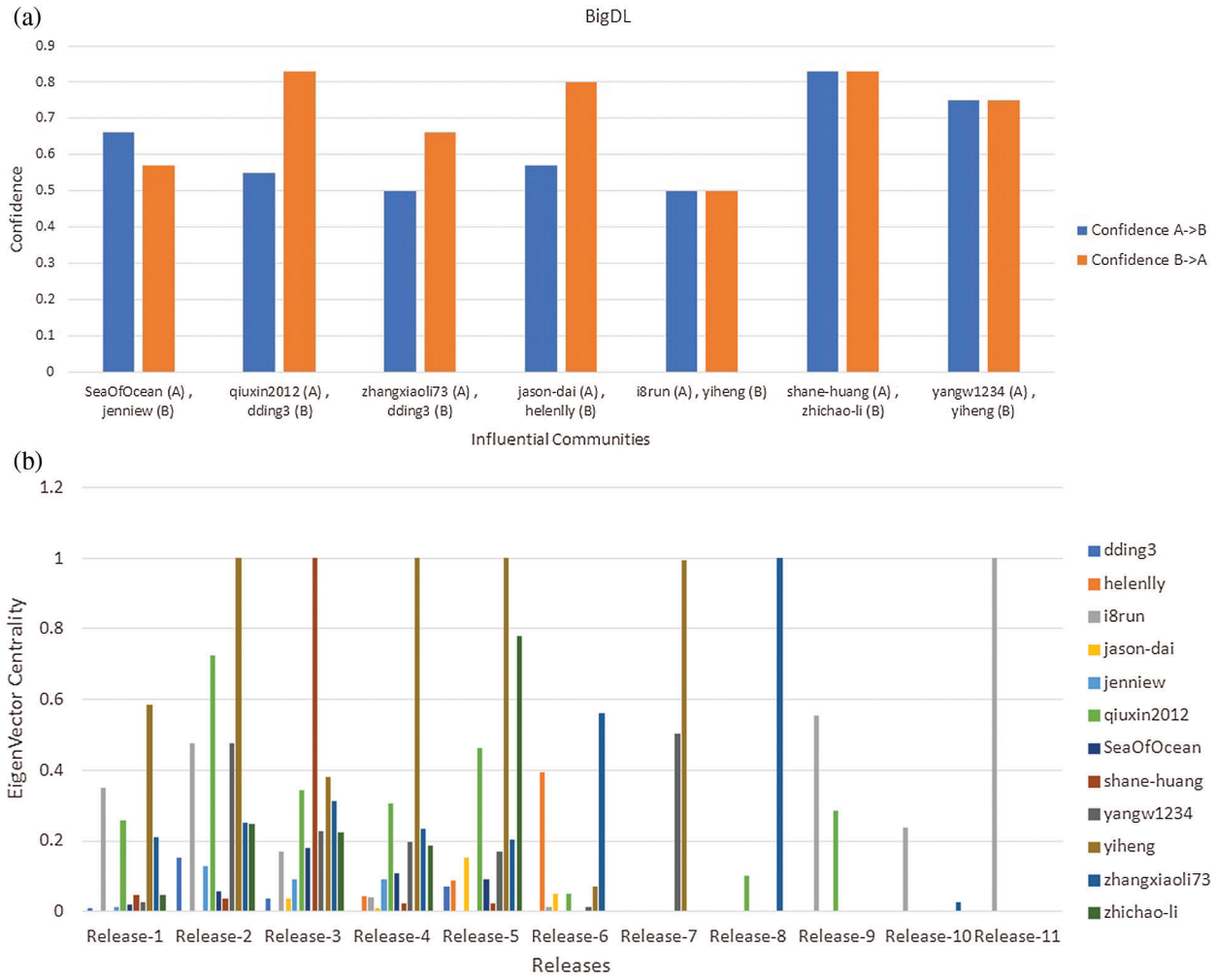

Figure 2: a) Relationship between the members of the influential community of "BIGDL," b) Developers of influential communities with Eigenvalue in each release of "BIGDL."

In the case of the "INCUBATOR-MXNET" project, the value of "Confidence" measure of each influential community is illustrated in the Fig. 3a. The major consequences are:

- We observed a joint presence of "anirudh2290" and "eric-haibin-lin" members in all releases. This means that if "shane-huang" was present in one release, there would be a 50% chance of "zhichao-li" involvement and vice versa. In addition, we found the co-occurrence of "haojin2" and "eric-haibin-lin" with a confidence value of 0.5 (i.e., 50%).
- We found a strong co-occurrence of "ZiyueHuang" and "eric-haibin-lin" with a confidence value of 1.0 (i.e., 100% with "ZiyueHuang" perspective while 71% with "eric-haibin-lin" perspective).
- We could observe developer co-occurrence on the member's point of view in the case of an influential community (i.e., A community of "haojin2" and "ZiyueHuang" members). Given the existence of "haojin2" in a team, there is a 50% chance of involvement of "ZiyueHuang" in the same team. Similarly, given the existence of "ZiyueHuang" in a team, there is a 60% chance for the involvement of "haojin2" in the same team.

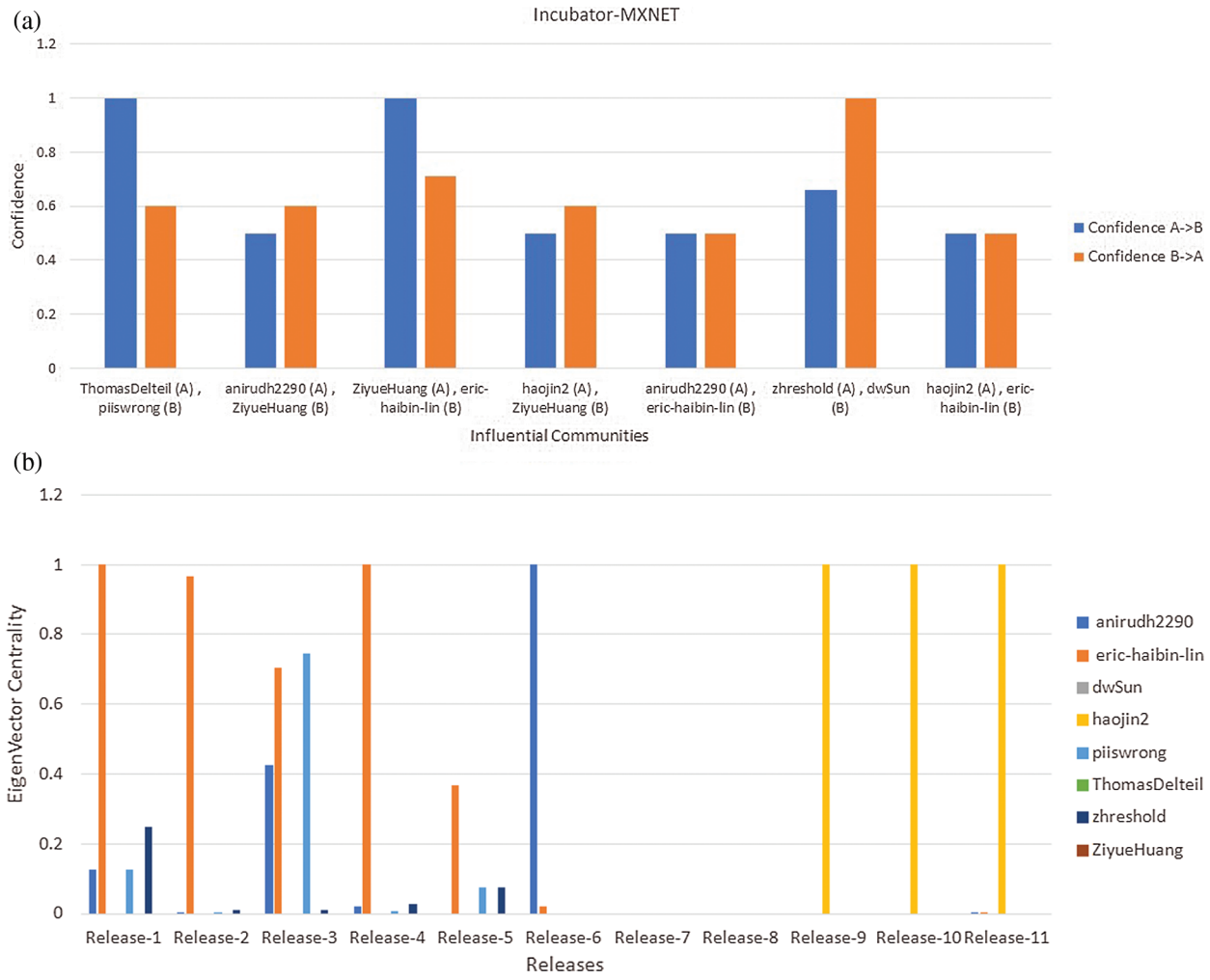

Figure 3: a) Relationship between the members of the influential community of "INCUBATOR-MXNET," b) Developers of influential communities with Eigenvalue in each release of "INCUBATOR-MXNET"

In the case of the "RECOMMENDERS" project, the value of "Confidence" value of each influential community is shown in Fig. 4a. The major consequences are;

- We observed a high co-occurrence of "eisber" and "dciborow" members across all releases. This means that if "eisber" is present in a release, then there is a 100% chance of "dciborow" involvement and vice versa. Moreover, we observed the co-occurrence of "miguelgfierro" and "loomlike" with a confidence value 0.5 (i.e., 80%).
- We could observe developer co-occurrence on the member's point of view as in the case of an influential community (i.e., A community of "gramhagen" and "yueguoguo" members), the existence of "gramhagen" in a team, there is a 75% chance of "yueguoguo" involvement in the same team. Similarly, given the existence of "yueguoguo" in a team, there is a 60% chance of "gramhagen" involvement in the same team.

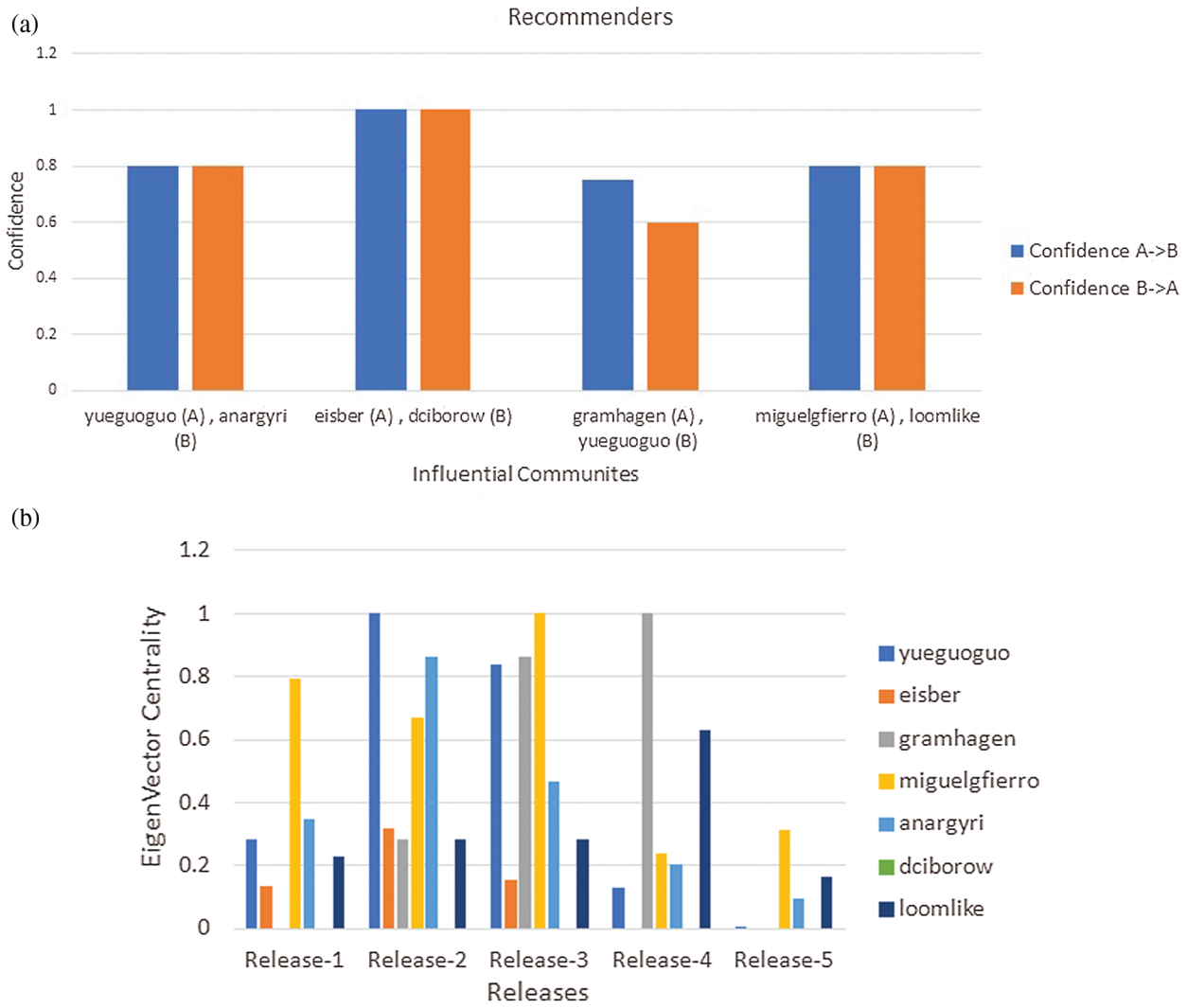

Figure 4: a) Relationship between the members of the influential community of "RECOMMENDERS", b) Developers of influential communities with Eigenvalue in each release of "RECOMMENDERS"

4.3 Response to RQ-3

In RQ-1 and RQ-2's response, we empirically investigated and reported the existence of influential communities and the likelihood of their involvement. However, the recommendation of noteable developers of their network is quite necessary to solve issues more appropriately. For example, assigning issues to appropriate and dedicated developers could reduce their time and effort. In this regard, we have built networks of developers that are the members of either influential or non-influential communities. Subsequently, we used the value of "Eigenvector centrality" to identify each developer (i.e., node) of the network and classify them according to their influence. We observed the influence of nodes in a network that is assessed by their neighboring nodes. A node with the highest value is regarded as having as much influence as its neighbor nodes [25,26].

Consequently, we examined the importance of developer nodes in each project release. The results of noteworthy developers of "BIGDL" "INCUBATOR-MXNET" and "RECOMMENDERS" are shown in Figs. 2b, 3b, and 4b respectively. These figures suggest that a few developers are more important

(developers with the highest Eigenvalue) in all project releases. Moreover, we observed that the developers with the highest Eigenvalue appear to be active participants in issues solving, reviews, and feedback.

The result of Fig. 2b indicates that "yiheng" is looking for a noteworthy developer in Release-2, Release-4, Release-5, and Release-7 respectively. Moreover, we could not recommend any noteworthy developer in all releases of BIGDL.

The result of Fig. 3b indicates that "eric-haibin-lin" is considered a noteworthy developer in the first five releases. In comparison, "haojin2" appears noteworthy in the last three releases of "INCUBATOR-MXNET". Subsequently, we cannot find a developer worth mentioning in Release-7 and Release-8. Moreover, we could not recommend any noteworthy developer in all releases of "INCUBATOR-MXNET".

The result of Fig. 4b indicates that "miguelgfierro" appears to be a noteworthy developer in all releases of "RECOMMENDERS". Subsequently, we were able to observe that in Release-2 and Release-3 releases, most developers of influential communities are looking influential over other releases. Moreover, we could not recommend any noteworthy developer in all releases of "RECOMMENDERS."

5 Implication for Research Communities

We presented the experimental findings in detail. However, the main effect of this study may help the research community in several areas such as:

- There is no centrally controlled team structure in the context of open-source software development. However, in each release, developers could work in several communities based on their collaboration and involvement to resolve issues.
- In early releases of an OSS, we observed several influential communities that could indicate the immaturity of the software in terms of the developer's dedication in solving issues.
- In subsequent releases of an OSS, the ratio of influential communities decreases from initial releases.
- We could not recommend an influential community of one release as an influential community in subsequent releases.
- By measuring the community's "Confidence" value, we could assess the likelihood of proponent's partner.
- We cannot provide a guarantee to recommend a developer as a noteworthy developer of all releases. Their involvement would allow us to identify them.

6 Threats to Validity

6.1 Internal Validity

To complete this study, we looked at more than 200 GitHub projects, but most of these projects were rejected based on their descriptive statistics of control variables. We developed criteria for the duration, releases, a number of issues, pull-requests, and project collaborators. We discarded those repositories that have fewer than three releases, one hundred issues, twenty collaborators, ten pull requests, and six months of development time.

6.2 External Validity

We have inspected over 200 GitHub projects with various contributors. Since GitHub is made up of software development data, and not all steps are recorded in the project. So, we had to get rid of projects that have no issue data. Hence, GitHub issues have been used to build the graphs. We extracted those issues which could be labelled as a "bug" or "enhancement". However, including these issues that are labeled through different tags could affect the results.

7 Conclusion and Future Work

In this article, we extracted the community structure of the network of OSS developers and examine their changes in the evolution of the project. We also looked at the relationship among community members and the impact of noteworthy developers on software quality. We conducted several experiments using the capabilities of widely used Apriori algorithms and used "Support" and "Confidence" measures to address our research questions. The major consequences of this study are: 1) the community structure of the developer network is decentralized, 2) we could not guarantee an influential community (in terms of their participation) and the noteworthy developers throughout all releases of an OSS, 3) the noteworthy developer of a release could be influential in subsequent releases, 4) the developers mostly take interest and remain involved in the early releases of an OSS, and 5) the developer's commitment is reduced as the OSS evolves. In the future, we will explore other projects to compare our findings in terms of determining influential communities, noteworthy developers, and the relationship between developers of an influential community.

Funding Statement: The author(s) received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] K. McCaskey, Gartner, "The crucial role of oss license compliance," 2019. [Online]. Available at: https://blog.sonatype.com/gartner-the-crucial-role-of-oss-license-compliance.

[2] P. S. Kochhar, E. Kalliamvakou, N. Nagappan, T. Zimmermann and C. Bird, "Moving from closed to open source: Observations from six transitioned projects to github," *IEEE Trans. on Software Engineering (Early Access)*, pp. 1, 2019.

[3] C. Bird, "Community structure in OSS projects," Technical Report, Davis: University of California, 2006. [Online]. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?rep=rep1&type=pdf&doi=10.1.1.122.4758.

[4] M. Rashid, P. M. Clarke and R. V. O'Connor, "Exploring knowledge loss in open source software (oss) projects, " in *Int. Conf. on Software Process Improvement and Capability Determination*. Cham, Palma de Mallorca, Spain: Springer, pp. 481–495, 2017.

[5] S. Majumder, J. Chakraborty, A. Agrawal and T. Menzies, "Why software projects need heroes (lessons learned from 1000+ projects)," 2019.

[6] J. Zhan, V. Guidibande and S. P. K. Parsa, "Identification of top-K influential communities in big networks," *Journal of Big Data*, vol. 3, no. 1, pp. 7821, 2016.

[7] R. D. Padua, E. L. S. Junior, L. P. Carmo, VOde Carvalho and S. Rezende, "Preprocessing data sets for association rules using community detection and clustering: A comparative study," in *XIII Encontro Nacional de Intelig^encia Artificial e Computational*. Recife, Pernambuco, Brazil, 553–564, 2016.

[8] M. E. J. Newman, "Finding community structure in networks using the Eigenvectors of matrices," *Physical Review E*, vol. 74, no. 3, pp. 66, 2006.

[9] M. Aberdour, "Achieving quality in open-source software," *IEEE Software*, vol. 24, no. 1, pp. 58–64, 2007.

[10] G. Iaffaldano, I. Steinmacher, F. Calefato, M. Gerosa and F. Lanubile, "Why do developers take breaks from contributing to oss projects?: A preliminary analysis," in *ICSE '19: 41st Int. Conf. on Software Engineering*. Montreal, Quebec, Canada, 9–16, 2019.

[11] L. F. Dias, I. Steinmacher and G. Pinto, "Who drives company-owned OSS projects: Internal or external members?" *Journal of the Brazilian Computer Society*, vol. 24, no. 1, pp. 16, 2018.

[12] K. Crowston and I. Shamshurin, "Core-periphery communication and the success of free/libre open source software projects," *Journal of Internet Services and Applications*, vol. 8, no. 1, pp. 564, 2017.

[13] J. L. Badaracco Jr, "We don't need another hero," *Harvard Business Review*, vol. 79, no. 8, pp. 120, 2001.

[14] W. Xuguang, "Identify influential nodes in complex networks based on modified topsis," in *2017 36th Chinese Control Conf. (CCC)*. Dalian, China, 1474–1479, 2017.

[15] A. Chin and M. Chignell, "Identifying active subgroups in online communities," in *Proc. of the 2007 Conf. of the Center for Advanced Studies on Collaborative Research*, Richmond Hill, Ontario, Canada, pp. 280–283, 2007.

[16] A. Chin, M. Chignell and H. Wang, "Tracking cohesive subgroups over time in inferred social networks," *New Review of Hypermedia and Multimedia*, vol. 16, no. 1–2, pp. 113–139, 2010.

[17] L. Przybilla, M. Rahn, M. Wiesche and H. Krcmar, "The more the merrier? The effect of size of core team subgroups on success of open source projects," in *14th Int. Conf. on Wirtschaftsinformatik*. Siegen, Germany, 2019.

[18] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D. M. German *et al.,* "The promises and perils of mining github," in *11th Working Conf. on Mining Software Repositories Hyderabad*. India, 92–1012014.

[19] GitHub Inc, "*The State of the Octoverse*," 2019. [Online]. Available at: https://octoverse.github.com/.

[20] M. E. J. Newman, "Modularity and community structure in networks," *Proc. of the National Academy of Sciences of The United States of America*, vol. 103, no. 23, pp. 8577–8582, 2006.

[21] B. Liu, W. Hsu and Y. Ma, "Integrating classification and association rule mining," in *ACM Special Interest Group on Knowledge Discovery and Data Mining (KDD)*. New York, NY, 80–86, 1998.

[22] S. A. Abaya, "Association rule mining based on apriori algorithm in minimizing candidate generation," *Int. Journal of Scientific & Engineering Research*, vol. 3, no. 7, pp. 1–4, 2012.

[23] C. C. Aggarwal, C. Procopiuc and P. S. Yu, "Finding localized associations in market basket data," *IEEE Trans. on Knowledge and Data Engineering*, vol. 14, no. 1, pp. 51–62, 2002.

[24] B. Wang and I. Rahal, "Wc-clustering: Hierarchical clustering using the weighted confidence affinity measure, " in *Seventh IEEE Int. Conf. on Data Mining Workshops*. Omaha, NE, 355–360, 2007.

[25] S. Gómez, "Centrality in networks: Finding the most important nodes, in *Business and Consumer Analytics: New Idea*. Cham, Switzerland: Springer, Chapter No. 8, pp. 401–433, 2019. [Online]. Available at: https://link.springer.com/chapter/10.1007/978-3-030-06222-4_8.

[26] D. L. Hansen, B. Shneiderman and M. A. Smith, Social network analysis: measuring, mapping, and modeling collections of connections. In: *Analyzing Social Media Networks with NodeXL: Insights from a Connected World*. *Chapter No. 3*. 2nd ed., vol. Section I. Burlingto, U.S.: Elsevier Inc, 31–52, 2019. [Online]. Available at: https://booksite.elsevier.com/samplechapters/9780123822291/Chapter_3.pdf.