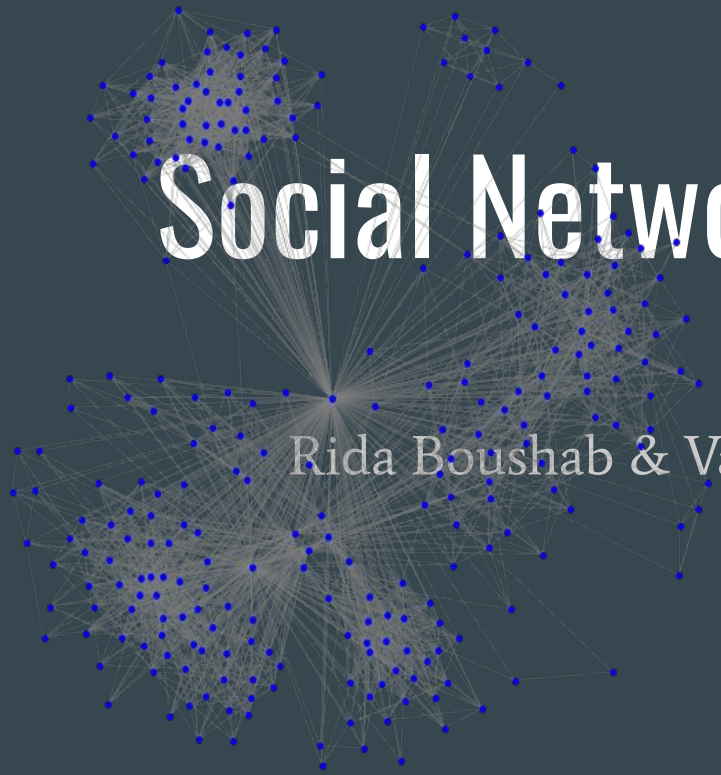
National School of Computer Science and System Analysis
Master Big Data & Data Science



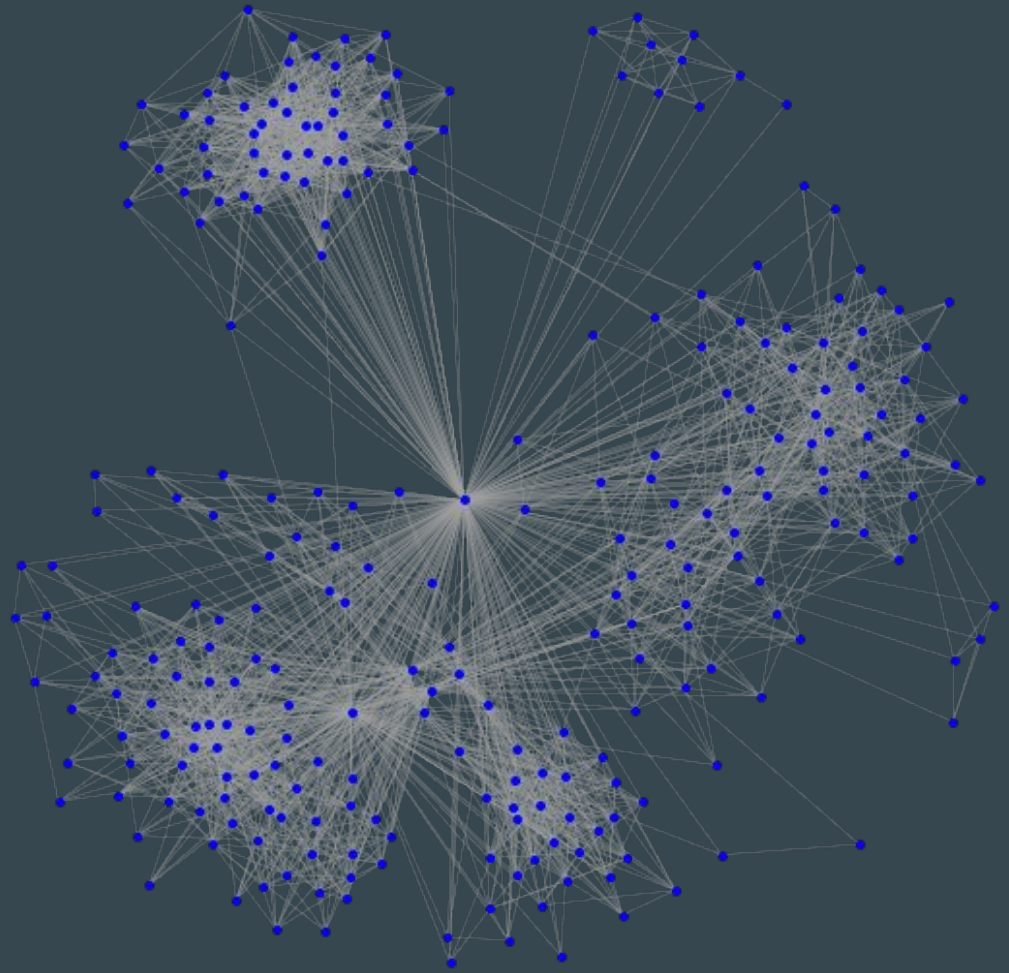
Social Network Analysis in GitHub

...

Rida Boushab & Vasconselos Fernando Victor Oliveira

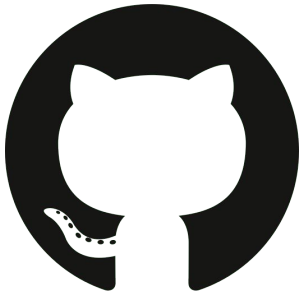
Plan

- Introduction
- Developing Hypothesis
- Data Acquisition
- Data Preprocessing
- Network visualisation
- Similarity analysis
- Conclusion



Introduction

GitHub is an open source software forge founded in 2008 with the purpose to simplify code sharing. GitHub is growing fastly month by month and nowadays is probably the largest repository for open source software projects.



Terminology

Repository: A repository is the most basic element of GitHub. They're easiest to imagine as a project's folder. A repository contains all of the project files.(including documentation), and stores each file's revision history.

Committer: is the person who has right access to a repository.

Star: users on the GitHub website are able to "star" other people's repositories, thereby saving them in their list of Starred Repos.

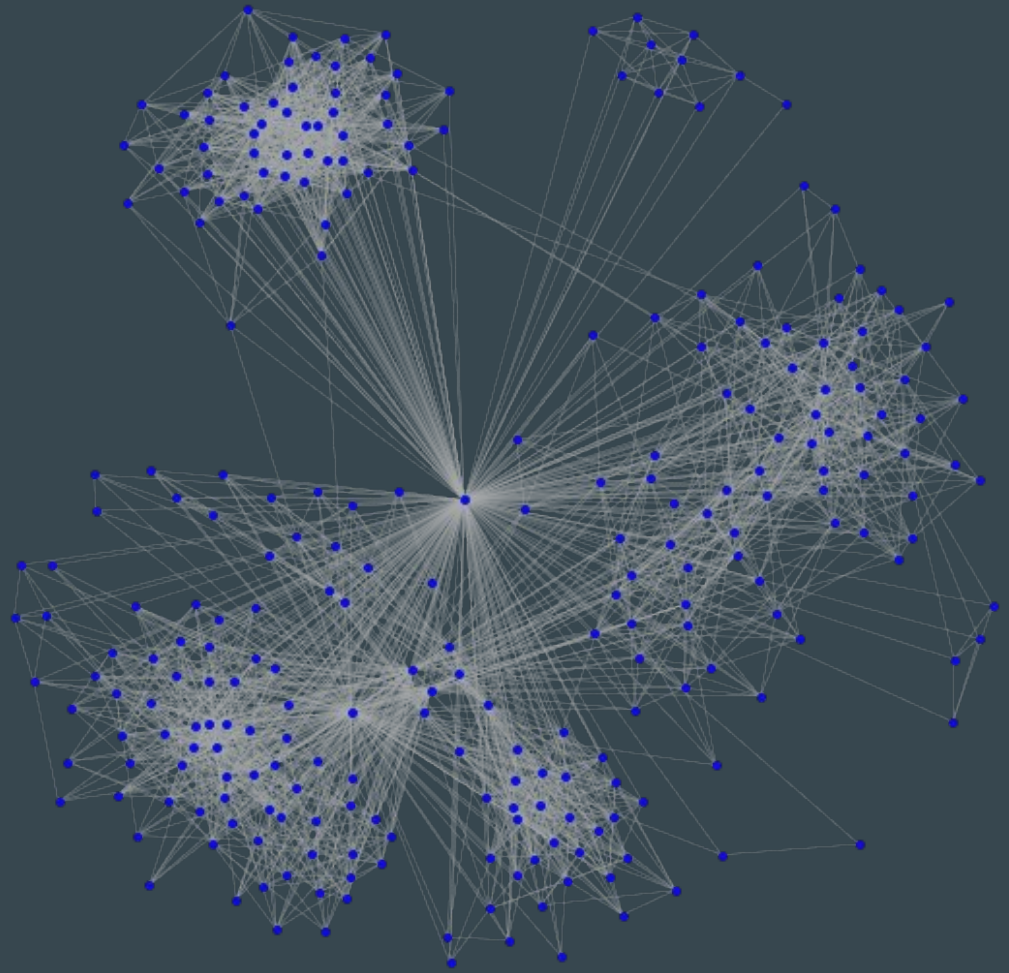
Developing Hypothesis

Studying the connection or the link between GitHub users based on repositories each user made.

Studying the similar repository between users

Library used

- github api v3
- networkx
- neo4j
- py2neo
- sklearn



Data Acquisition

Data acquisition is the most important phase of every research project, in our project for data collection we use a python library that based on Github API v3. This library designed to return all information about repositories, users and connection between them. To achieve our goal we follow this steps :

1. Get Repositories
2. Get the lists the users that have starred the repository
3. Get list of committers
4. Get list of assignees

Data Acquisition: exemple

```
Get Repositories
```

In [4]:
```
#for i in range(len(url)):
#r1.json()
r = requests.get('https://api.github.com/repositories', auth=(username,token))

for entry in r.json():
    url[entry['name']]= 'https://api.github.com/repos/' + entry['owner']['login'] + '/' + entry['name'] + '/readme'
    star_url[entry['name']]= 'https://api.github.com/repos/' + entry['owner']['login'] + '/' + entry['name'] + '/stargazers'
    languages_url[entry['name']]= 'https://api.github.com/repos/' + entry['owner']['login'] + '/' + entry['name'] + '/languages'
    commits_url[entry['name']]= 'https://api.github.com/repos/' + entry['owner']['login'] + '/' + entry['name'] + '/commits'
    issues_url[entry['name']]= 'https://api.github.com/repos/'+ entry['owner']['login']+'/'+ entry['name']+'/issues'
    labels[entry['name']]= 'https://api.github.com/repos/'+ entry['owner']['login']+'/'+ entry['name']+'/labels'
for x in url:
    repos_readme[x]= requests.get(url[x], auth=(username, token))

x2remove = []
for x in repos_readme:
    if ("download_url" in repos_readme[x].json()) == False:
        x2remove.append(x) #tout les fichier qui non pas de read me file (data aquestion)

for i in range(len(x2remove)):
    repos_readme.pop(x2remove[i])

for x in repos_readme:
    #requests.get(x.json()['download_url'], allow_redirects=True)
    name = '' + x +'.md'
    open(name, 'wb').write(requests.get(repos_readme[x].json()['download_url'], allow_redirects=True).content)
```

```
Get the lists the people that have starred the repository.
```

Data Preprocessing

After Data acquisition, data preprocessing is another important phase. to preprocess we:

1. Get rid of all the repositories which they haven't a "Readme file"
2. Using a technique of text mining which is "Term Frequency-Inverse Document Frequency (TF-IDF)" , to find a vector that best describes the document using the words that occur in the document.

Data Preprocessing: TF-IDF

$$TF - IDF = TF_t \times \log \frac{N}{DF_t} \tag{1}$$

TF_t: Term Frequency of the term t. How many times does the term occur in the document.

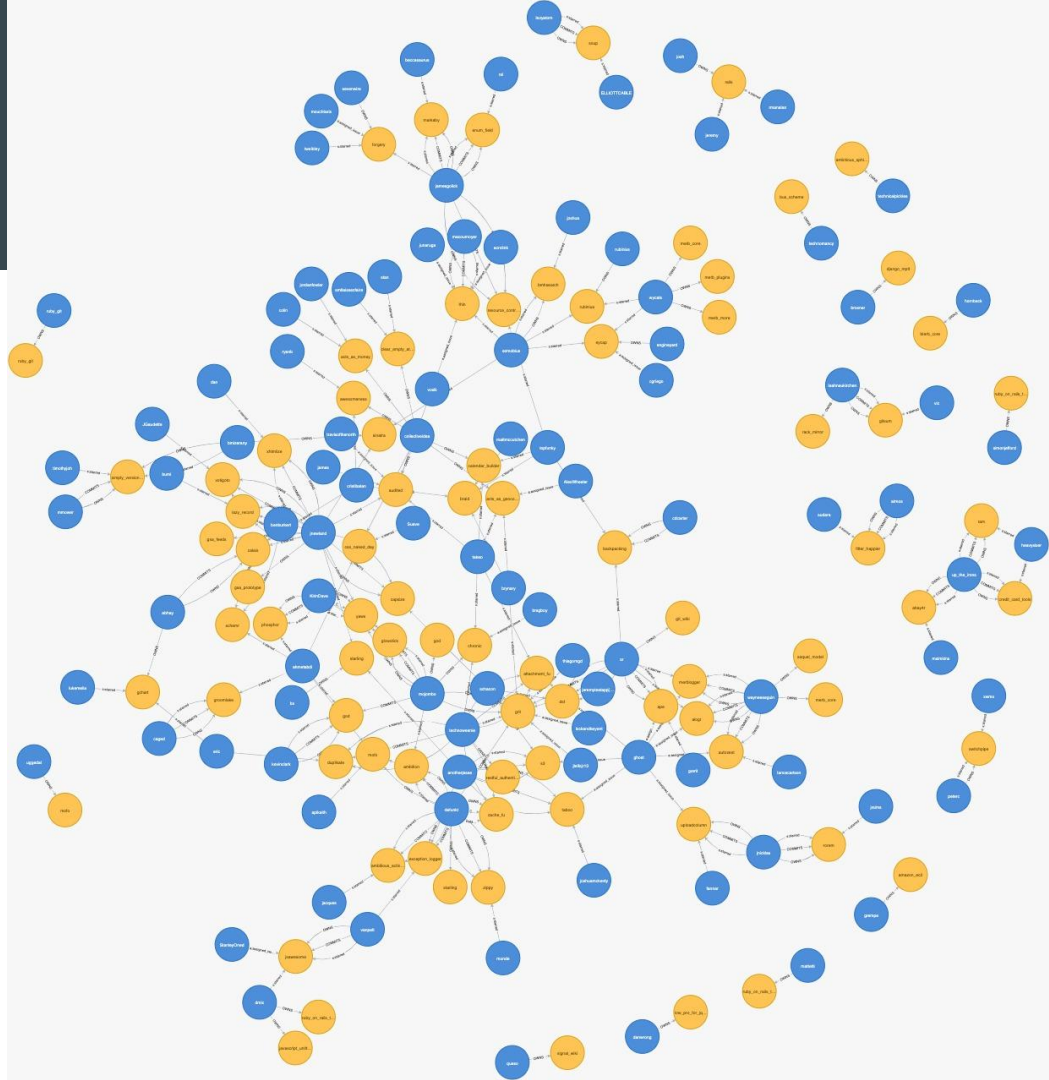
N- Total number of documents in the corpus.

DF_t: Document Frequency of t. How many documents have the term t.

A number of approaches exist for computing TF-IDF including normalizing the term frequencies to account for the variation in length of the documents in the corpus.

Network visualisation

Network between user and repository

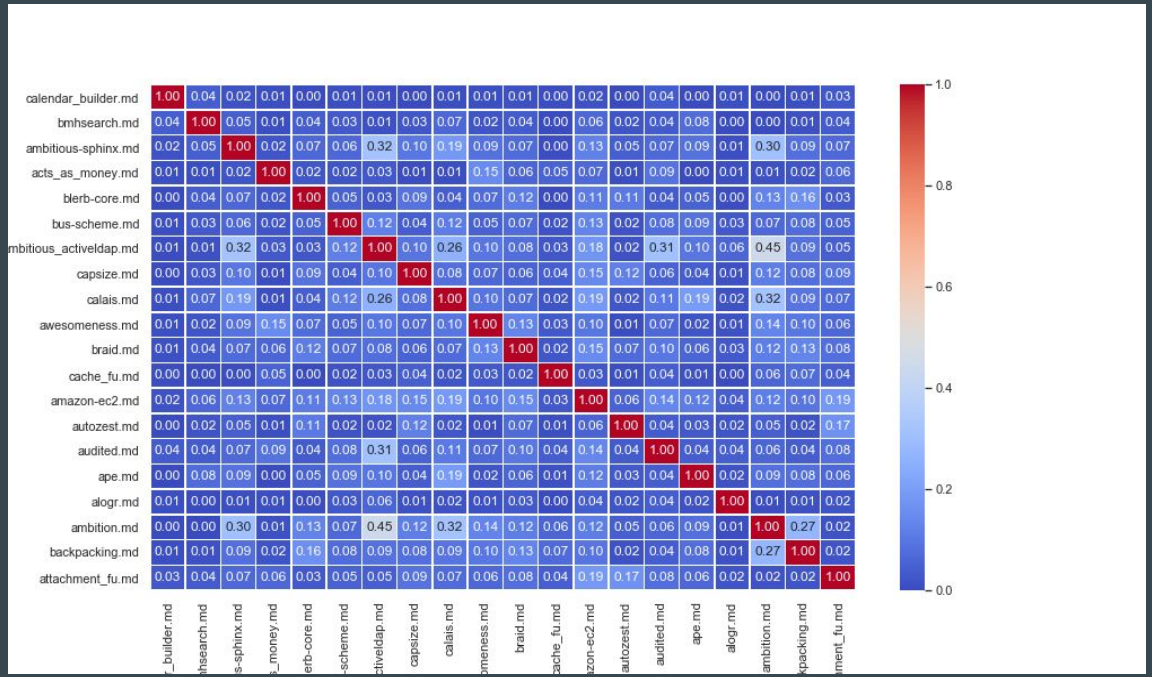


Metric analysis

In order to measure the similarity of repositories we worked with three metrics:

Metric analysis: Readme concept

In order to compute the similarity of README files among repositories, we used the “Cosine Similarity” to measure how similar the documents they are.



In Figure we present a heatmap table of the similarity among 20 Readme files of repositories.

Metric analysis: Star concept

two repositories that are starred at a similar period of time by user are likely to be similar.

$$t = |T(U, R1) - T(U, R2)|$$

$$\text{StarScore} = \begin{cases} 1 & if \quad t \in [0, 10] \\ 1 - \frac{t-10}{100} & if \quad t > 10 \end{cases} \qquad *$$

such that :

t is the difrrence time of two repositories that are starred in minute

T is the time that the user star a repositorie

* if the user is stared the two repositories R1 and R2 between 0 to 10 minutes that's the top score of the star score.

Metric analysis: Issue concept

$$\text{IssueScore} = \begin{cases} 1 & if \quad Similar \\ 0 & if \quad NotSimilar \end{cases}$$

Metric analysis: Composer Algorithm

The composer calculate the total similarity of the repository:

INPUT = R2, R1
OUTPUT =

$$\left[composer = \frac{Readme(R1, R2) * 50 + star(R1, R2) * 30 + Issue(R1, R2) * 20}{100}\right]$$

if composer >= 0.5 then
return R1 AND R2 ARE SIMILAR
ELSE return Not SIMILAR
end

Conclusion and Perspectives

we have performed a study of similarities among public repositories at GitHub to gain an intuitive understanding of the way projects are similar over this social coding platform.

In the future, we wish to allocate more resources into extracting and computing statistics for GitHub network of millions of developers contributing to a more extensive set of projects in GitHub.

References

- ★ Shahzad Qaiser and Ramsha Ali. Text mining: Use of tf-idf to examine the relevance of words to documents. International Journal of Computer Applications, 181, 07 2018.
- ★ Faisal Rahutomo, Teruaki Kitasuka, and Masayoshi Aritsugi. Semantic cosine similarity. 10 2012.
- ★ Matthew A. Russell. Mining the Social Web 2014