**Community Detection with Multi Layered  
Graphs using Social Network Analysis**

Submitted by

**DIBYA KUNDU 13000119063**

**SAYANTA KUNDU 13000119095**

**ABHIRUP GHOSH 13000119097**

**ANUSHKA DAS 13001619098**

Submitted for the partial fulfillment for the degree of Bachelor of Technology in Computer Science and Engineering



Techno Main Salt Lake,

EM 4/1, SaltLake, Sector V, Kolkata – 700 091

**CERTIFICATE**

This is to certify that the project entitled “**Community detection with Multi Layered  
graphs using Social Network Analysis**” prepared by Dibya Kundu (13000119063), Sayanta Kundu (13000119095), Abhirup Ghosh (13000119097) and Anushka Das (13001619098) of B. Tech (Computer Science & Engineering), Final Year, has been done according to the regulations of the Degree of Bachelor of Technology in Computer Science & Engineering. The candidates have fulfilled the requirements for the submission of the project report.

It is to be understood that, the undersigned does not necessarily endorse any statement made, opinion expressed or conclusion drawn thereof, but approves the report only for the purpose for which it has been submitted.

-------------------------------------------- --------------------------------------------

(Signature of the Internal Guide) (Signature of the HOD)

-------------------------------------------- --------------------------------------------

(Signature of External Guide, if applicable) (Signature of the External Examiner)

--------------------------------------------------------------------------------------------

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

--------------------------------------------------------------------------------------------

**ACKNOWLEDGEMENT**

We would like to express our sincere gratitude to our mentor of the department of Computer Science and Engineering, whose role as project guide was invaluable for the project. We are extremely thankful for the keen interest he / she took in advising us, for the books and reference materials provided for the moral support extended to us.

Last but not the least we convey our gratitude to all the teachers for providing us the technical skill that will always remain as our asset and to all non-teaching staffs for the gracious hospitality they offered us.

Place: Techno India, Salt Lake

Date:

………………………………

………………………………

………………………………

………………………………

Contents

[1 Introduction 5](#_Toc5900)

[1.1 Abstract 5](#_Toc24468)

[1.2 Problem Domain 5](#_Toc6402)

[1.3 Related Studies 5](#_Toc603)

[1.4 Glossary 6](#_Toc22866)

[2 Problem Definition 7](#_Toc24467)

[2.1 Scope 7](#_Toc7886)

[2.2 Exclusions 7](#_Toc20694)

[2.3 Assumptions 7](#_Toc31368)

[3 Project Planning 7](#_Toc20802)

[3.1 Software Life Cycle Model 7](#_Toc22551)

[3.2 Scheduling 8](#_Toc20286)

[3.3 Cost Analysis 9](#_Toc199)

[4 Requirement Analysis 11](#_Toc14687)

[4.1 Requirement Matrix 11](#_Toc12652)

[4.2 Requirement Elaboration 11](#_Toc1911)

[5 Design 11](#_Toc8116)

[5.1 Technical Environment 11](#_Toc18704)

[5.2 Detailed Design 11](#_Toc29487)

[6 Implementation 12](#_Toc19691)

[6.1 Flow Chart 12](#_Toc5984)

[6.2 Dataset Preparation 13](#_Toc20786)

[6.3 Plotting 2D graph from dataset 15](#_Toc14233)

[6.4 Algorithms for Community Detection in Multi-layered Network 16](#_Toc27263)

[6.5 System Installation Steps 19](#_Toc31341)

[7 Test Results and Analysis 21](#_Toc32750)

[8 Conclusion 31](#_Toc510)

[8.1 8.1 Project Benefits 31](#_Toc27847)

[8.2 Future Scope for improvements 32](#_Toc8051)

[9 References / Bibliography 33](#_Toc7309)

# Introduction

## Abstract

In recent years, there’s an increasing focus on the rapid development of more complicated networks, namely multilayer networks. Community detection is a popular research topic in a broad range of complicated systems, from biology to sociology. Communities in a single-layer network are groups of nodes that are more strongly connected among themselves than the others, while in multilayer networks, a group of well-connected nodes are shared in multiple layers.

Many traditional algorithms can rarely perform well on a multilayer network without modifications. Thus, in this project, we are providing a comprehensive understanding of community detection methods in multilayer networks. Assuming that all the graph layers are informative, they are likely to provide complementary information and thus to offer richer information than any single layer taken in isolation.

## Problem Domain

Our project is aimed at analyzing multi-layered network virtual communities created within online social networks for the purpose of grouping users, in order to study their internal organization and to find whether a common structure exists.

## Related Studies

The traditional single-layer network assumes that all the links (edges) between vertices take place at the same level, which is too big a constraint that may occasionally result in the failure to fully capture the details of complex systems.

For example, in web image retrieval, the visual information of images needs to be separated from their textual tags.Therefore, the multi-layer networks have received an

increasing amount of attention because they can precisely characterize and model systems in real world

Community detection in networks has become a hot topic since communities shed light on structure-function relations, which has been extensively studied in the single-layer networks. The most straightforward strategy is to extend the single-layer community detection algorithms to the multi-layer networks to develop network-compression- and consensus-based approaches. Network-compression-based approaches compress multi-layer networks into a single-layer network in which the single-layer community detection algorithms are adopted.

Thus, there is a critical need to develop effective algorithms for community detection in multi-layer networks,rather than by simply extending the available single-layer network algorithms. To identify communities in multi-layer networks, we must simultaneously take into account multiple layers during the community search procedure.The first step in multi-layer community detection is to quantify the community in multi-layer networks.

According to the authors of M. E. J. Newman et al. 2002,a vertex v in a random graph is equally probable to be linked to any of the N-1 number of nodes(Graph has total N number of nodes). They suggest edge distribution of a random graph follows Poisson distribution.

Aaron Clauset et al. 2004, proposed a hierarchical agglomeration community algorithm for detecting community structure with running time of O(nmd\*log n) for network of n nodes, m edges and d depth.

Blondel et al. 2008 devise a community detection algorithm based on the optimization of modularity score.

Greene et al. 2010 prescribe a model for tracking the progress of dynamic communities in various states such as, merge, expansion, contraction.

T Yang et al. 2011 proposed a model whichcaptures the evolution of dynamic communities by explicitly assigning the transition values of community memberships for each node in the network.

Seshadhri et al. 2012, based on mathematical arguments, the authors hypothesize that any graph with a heavy-tailed degree distribution and community structure must contain a scalefree collection of dense ER subgraphs.

Gopalan et al. 2013 proposed an community detection algorithm which allows nodes to be part of multiple communities based on Bayesian model.

The authors of Yang et al. 2015[42] have distinguished between structural and functional definitions of network communities. Using ground-truth data they

propose a newer approach to evaluate community detection algorithms.

## Glossary

|  |  |
| --- | --- |
| Acronyms | Expansion |
| OSN | Online Social Network |
| SNA | Social Network Analysis |
| HTTP | Hypertext Transfer Protocol |
| NMI | Normalized mutual information |

# Problem Definition

## Scope

Our project will analyze the connection patterns among their users on social media.

SNA can be used to explore how a product or service becomes popular within a given community. Advertisers are particularly keen to understand how something becomes popular and how individuals influence one another in the adoption of products, services, and ideas. Our project will analyze the flow of visible interactions between the followers

This project can be useful to turn viral marketing into something that can be understood and can be reproduced, for obvious reasons.

## Exclusions

In our project, we are planning to do community detection using multi layer network approach of various social networking sites.

The NMI and density purity index that’ll be calculated from our project for all single layers as well as the whole multilayer network is extremely useful to analyze various such inter-layer clusters when extended to other such real-life networks where the data extracted is huge and many inter-layer clusters can be formed.

## Assumptions

* We detect the various clusters formed in the Multi layered network.
* We assume each group to be a layer of the multi-layer network.

# Project Planning

## Software Life Cycle Model

We have chosen Agile Model as Software Development Life Cycle model.

Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning. The project scope and requirements are laid down at the beginning of the development process. Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance.



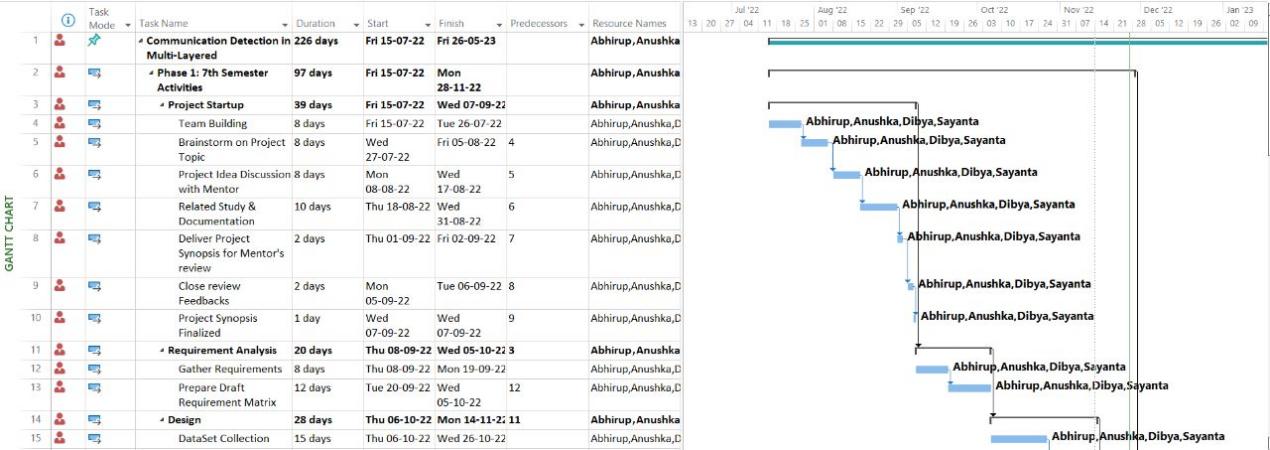
Figure 1: Agile Model for Software Development

The agile software development emphasizes on four core values.

1. Individual and team interactions over processes and tools
2. Working software over comprehensive documentation
3. Collaboration over contract negotiation
4. Responding to change over following a plan

## Scheduling

Gantt chart of project planning is shown below:



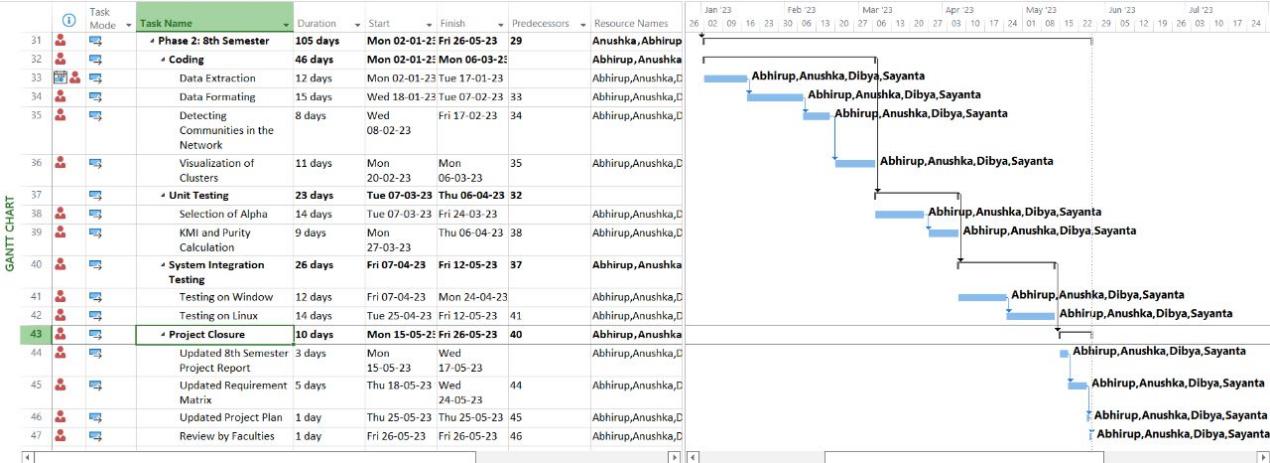
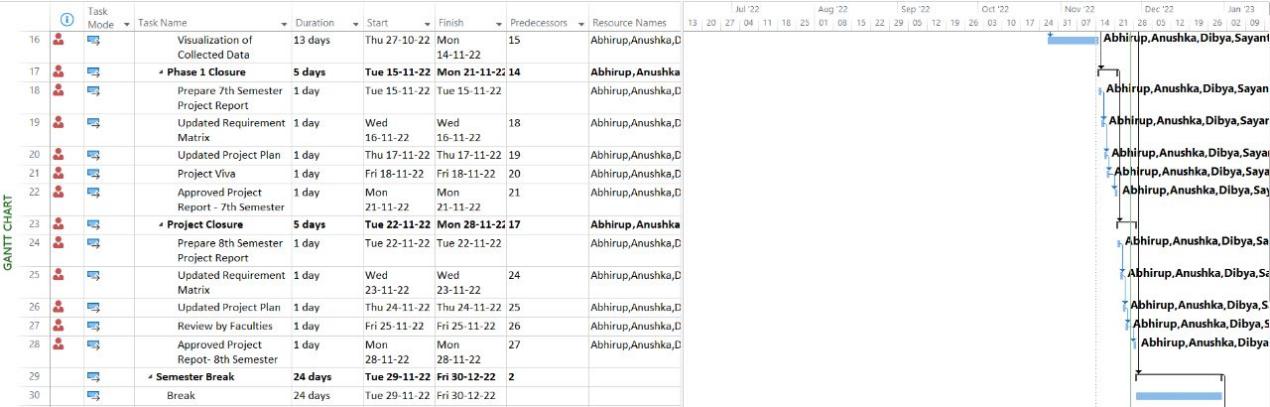


Figure 2: Project Plan

## Cost Analysis

To apply the COCOMO model for cost estimation, the project must be classified under one of the three Basic models as postulated by Boehm: Organic, Semi-Detached or Embedded. For our project, the Basic Organic model is suitable for the following reasons:

* Size of the development team is reasonably small.
* Development of a well understood application program.
* Size of the program is in between (2-50) KLOC
* Our project has a flexible deadline as we start our final year project from the 7th semester.

Therefore, the formula to be applied is, Effort = 2.4(KLOC)1.05 PM …. (i)

Tdev = 2.5(Effort)0.38Months …. (ii)

Where, (i) is the estimation of development effort

(ii) is the estimation of development time. KLOC is ‘Kilo Lines of Code’ and PM is Person Months.

Since our project is purely developed upon open-source platforms and extracted the datasets using free modules in python so we have estimated the cost per month solely based on the effort, as made by the team members.

We implemented 787 lines of code = 0.787 KLOC Effort = 2.4(0.787)1.05= 1.86631 PM

Tdev= 2.5(1.86631)0.38= 3.1689 Months

Therefore, the estimated time for development is 3.1689 months Now, calculating estimated cost of project as,

Total cost=Cost per month × Total no. of months

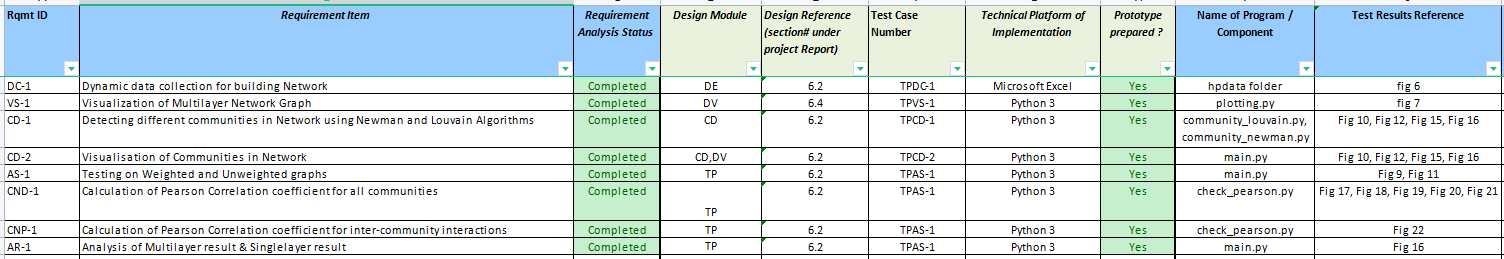
Total cost for developing =1000 × 3.1689=3168.9 Rupees

We have collected different information about how to build this and gave effort for collecting information and reading different articles so if we consider cost estimated based on this effort let for approx. 12 months = 500(per month) \* 12 = 6000

As hardware we used two computers so as cost, we can say approx. 60,000 Rupees. So, approximately total cost is: 60,000+6000+3168.9= 69168.9 Rupees.

# Requirement Analysis

## Requirement Matrix



## Requirement Elaboration

### Data Collection to build network

For We are going to apply the community detection algorithms in undirected weighted and unweighted graph structures. We will use .csv files as a dataset. We can collect different social media data from publicly available datasets in different online platform and put that in a csv file where 1st and 2nd columns will be the nodes that are having an edge in between them and 3rd column as edge weights if applicable.

### Visualization of multi-layer network graph

For visualizations we are going to use different python libraries like pandas, matplotlib and plotly. After detecting communities using different algorithm we are going to visualize the communities using the same libraries.

# Design

## Technical Environment

The minimum hardware requirements for our project are mentioned below:

* + - * Microsoft Windows 8/10(64 bit)
      * 4 GB RAM minimum,8 GB RAM recommended

The software requirements for our project has been mentioned below:

* + - * OS – Windows 10/ Ubuntu 18.04
      * Programming Language: Python 3
      * Tools: Pycharm Notebook OR Notepad++
      * Dataset: Microsoft Excel(.csv)

## Detailed Design

Provide hierarchy of modules or overall system diagram.

* Import Modules
* Collection of Data
* Visualization of Weighted and Unweighted Graphs
* Algorithm Implementation
* Visualization of Communities
* Analyzing the results

# Implementation

## Flow Chart

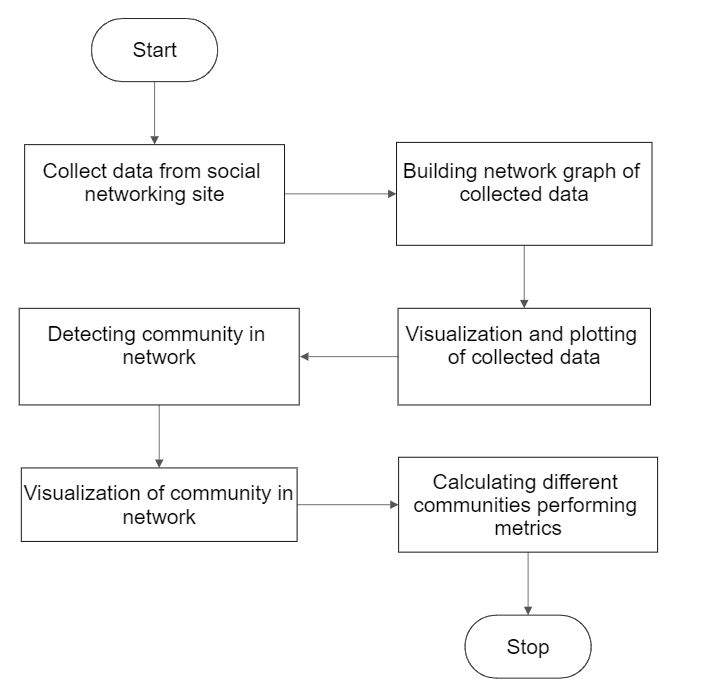


Figure 4:Flow Chart

## Dataset Preparation

We have used .csv files as our dataset. To create manual dataset to test the algorithm:

For Unweighted graphs:

Use two columns as to denote two nodes on the side of an edge

For Weighted graphs:

Use the third column to denote the edge weight.

We have used a movie data to test our algorithm. We have divided the movie “Harry Potter and the Sorcerer's stone” into 17 different scenes and created nodes as per the characters. The edge weight is the value of interaction in between the characters.

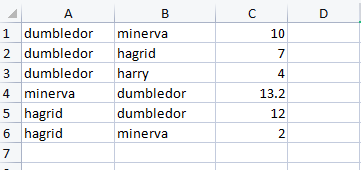


Figure 5:Dataset file scene1.csv

Then we merged all the scenes in a single csv file to get the total interaction values between characters in total movie.



Figure 6:Dataset file for the whole movie

## Plotting 2D graph from dataset

To plot a 3D graph from a dataset we have used four libraries:

* **NetworkX:** It is a Python language software package for the creation, manipulation, and study of the structure, dynamics, and function of complex networks. It is used to study large complex networks represented in form of graphs with nodes and edges. Using Networkx we can load and store complex networks. We can generate many types of random and classic networks, analyze network structure, build network models, design new network algorithms and draw networks.
* **Pandas:** Pandas is an open-source library that is made mainly for working with relational or labeled data both easily and intuitively. It provides various data structures and operations for manipulating numerical data and time series. Using Pandas data from different file objects can be loaded.
* **Matplotlib:** Matplotlib is a visualization library in Python for 2D plots of arrays.
* **Plotly:** The **Plotly Python**library is an interactive open-source library. This can be a very helpful tool for data visualization and understanding the data simply and easily. plotly graph objects are a high-level interface to plotly which are easy to use. It can plot various types of graphs and charts like scatter plots, line charts, bar charts, box plots, histograms, pie charts, etc.

**Extracting Data:** We use pandas to extract data from a given csv file. After extracting the data we can create a network using Networkx.

**Plotting Data:** After creating the network we can plot using both matplotlib and plotly libraries. For 2D plotting we use matplotlib. For 3d plotting we use Plotly library.

We’ve used a data set of Facebook publicly available on Stanford University datasets collection.

After plotting the network in 2D using matplotlib:

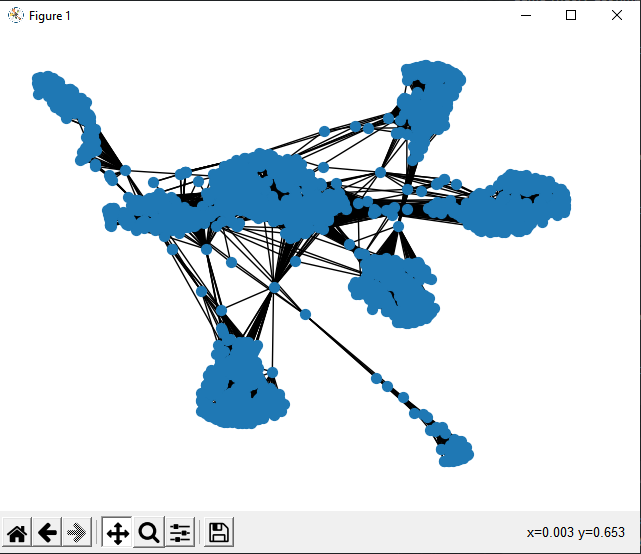


Figure 7:Visualization of Multinetwork Graph

## Algorithms for Community Detection in Multi-layered Network

**The Girvan-Newman algorithm**

1. **Initialization**

The algorithm starts by initializing the graph and calculating the betweenness centrality of all edges. Betweenness centrality is a measure of how often an edge is used in the shortest paths between pairs of nodes. The edge with the highest betweenness centrality is considered to be the most important edge in the graph.

2. **Iterative removal of edges**

The algorithm then iteratively removes edges from the graph, starting with the edge with the highest betweenness centrality. After each edge is removed, the betweenness centrality of all remaining edges is recalculated. This process continues until there are no edges left in the graph.

3. **Identification of communities**

The communities in the graph can be identified by looking at the connected components of the graph. A connected component is a subgraph in which all nodes are connected to each other. The communities in the graph are the connected components that are not connected to each other.

Example

Consider the following graph:

Code snippet

A---B---C

|    |

D---E

The betweenness centrality of all edges in this graph is as follows:

|  |  |
| --- | --- |
| Edge | Betweenness centrality |
| AB | 2 |
| BC | 1 |
| CD | 1 |
| DE | 1 |

The edge with the highest betweenness centrality is AB, so this edge is removed from the graph. The resulting graph is as follows:

Code snippet

B---C

|    |

D---E

The betweenness centrality of all edges in this graph is as follows:

|  |  |
| --- | --- |
| Edge | Betweenness centrality |
| BC | 1 |
| CD | 1 |
| DE | 1 |

The edge with the highest betweenness centrality is BC, so this edge is removed from the graph. The resulting graph is as follows:

Code snippet

C

|

D---E

The graph now has two connected components, so there are two communities in the graph. The communities are {A, B} and {C, D, E}.

Advantages and disadvantages

The Girvan-Newman algorithm is a simple and efficient algorithm for detecting communities in graphs. It is also relatively robust to noise and outliers. However, the algorithm can be computationally expensive for large graphs.

Here are some additional advantages and disadvantages of the Girvan-Newman algorithm:

**Advantages**

* The algorithm is simple and easy to implement.
* The algorithm is efficient and can be applied to large graphs.
* The algorithm is robust to noise and outliers.

**Disadvantages**

* The algorithm can be computationally expensive for large graphs.
* The algorithm can be sensitive to the choice of parameters.
* The algorithm can produce overlapping communities.

**Modularity Maximization Algorithm**

The modularity maximization algorithm is a greedy algorithm that works by iteratively merging communities that have high modularity variation. The modularity of a partition is a measure of how well the nodes in the network are grouped into communities. A higher modularity indicates a better community structure.

The algorithm works as follows:

1. Start with each node in its own community. This means that there will be no edges between nodes in different communities.
2. Inspect each pair of communities connected by at least one link and compute the modularity variation obtained if we merge these two communities. Modularity variation is calculated as follows:

Code snippet

ΔM = M(C1 ∪ C2) - M(C1) - M(C2)

where:

* M(C) is the modularity of community C
* C1 and C2 are the two communities being merged

1. Identify the community pairs for which ΔM is the largest and merge them. Note that modularity of a particular partition is always calculated from the full topology of the network.
2. Repeat step 2 until all nodes are merged into a single community.
3. Record for each step and select the partition for which the modularity is maximal.

The following example illustrates how the modularity maximization algorithm works. Consider the following graph:

Code snippet

A---B---C

     \ /

      D

The modularity of this graph is 0.5. This is because there are 2 edges between nodes in the same community (A-B and B-C) and 1 edge between nodes in different communities (A-D).

The algorithm would first merge communities A and B. This would increase the modularity of the graph to 0.6. This is because there are now 3 edges between nodes in the same community (A-B, B-C, and A-B) and 1 edge between nodes in different communities (A-D).

The algorithm would then merge communities B and C. This would increase the modularity of the graph to 0.7. This is because there are now 4 edges between nodes in the same community (A-B, B-C, B-C, and A-B) and 1 edge between nodes in different communities (A-D).

The algorithm would then merge communities B and D. This would increase the modularity of the graph to 0.8. This is because there are now 5 edges between nodes in the same community (A-B, B-C, B-C, A-B, and B-D) and 0 edges between nodes in different communities.

The algorithm would then merge communities A and B-D. This would not increase the modularity of the graph. Therefore, the algorithm would stop.

The final community structure is shown below:

Code snippet

A---B---C---D

The modularity of this community structure is 0.8. This is the highest modularity that can be achieved for this graph.

The modularity maximization algorithm is a popular method for detecting communities in networks. It is relatively simple to implement and can be used to detect communities in large networks. However, the algorithm can be sensitive to noise and may not always produce accurate results.

**Louvain Algorithm for weighted graphs**

The algorithm starts by assigning each node to its own community. Then, it repeatedly merges communities that can increase the modularity of the graph. The algorithm terminates when no more communities can be merged without decreasing the modularity.

The Louvain algorithm is a very efficient algorithm for community detection in weighted graphs. It has been shown to be able to find communities in graphs with millions of nodes and edges.

Here is a more detailed explanation of the algorithm:

1. Initialize each node to be in its own community.
2. Repeat the following steps until no more communities can be merged:
   * For each node u, find the community c that u belongs to that has the highest modularity gain when u is moved to c.
   * Move u to c.
3. The communities found at the end of the algorithm are the final communities.

The modularity gain of moving a node u to community c is given by the following formula:

ΔQ(u → c) = Q(u ∈ c) - Q(u ∈ c')

\

where Q(u ∈ c) is the modularity of the graph when u is in community c and Q(u ∈ c') is the modularity of the graph when u is in its current community c'.

The modularity of a graph is given by the following formula:

Q = ∑\_c m\_c (A\_c - k\_c^2 / 2m)

where m\_c is the number of edges within community c, A\_c is the total weight of the edges within community c, and k\_c is the number of nodes in community c.

The Louvain algorithm is a very effective algorithm for community detection in weighted graphs. It has been shown to be able to find communities in graphs with millions of nodes and edges. However, it is important to note that the Louvain algorithm is a heuristic algorithm, which means that it does not always find the optimal solution.

## System Installation Steps

**Installation Required**

Install Python in your windows and set path variable so that the programs can be runned through command prompt

**Python Packages Required**

1)Install the current release of networkx with pip: pip install networkx

2)Install the current release of matplotlib with pip: pip install matplotlib

1. Install the current release of Pandas with pip: pip install pandas
2. Install the current release of Plotly with pip: pip install plotly

**Execution Instruction**

Run the main file “main.py” with the dataset filename and output filename as argument variable. If the dataset is stored as data.csv and the output needs to be stored at “out.txt” one should run the command below-

*“python main.py data.csv out.txt”*

# Test Results and Analysis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sl. no | Graph Type | File Name | Detected Communities | Modularity |
| 1. | Unweighted | demo\_unweighted.csv | Community 1  1, 2, 3, 4  Community 2  9, 10, 11, 12  Community 3  5, 6, 7, 8 | 0.56624999999 |
| 2. | Weighted | demo\_weighted.csv | Community 0  9, 10, 11, 12  Community 1  6, 7, 8  Community 2  1, 2, 3  Community 3  4, 5 | 0.39125 |
| 3. | Weighted | hptotal.csv | **Community 1**  dumbledor, percy, spells, students, nearly\_headless\_nick, portraits, seamus, hooch, dean, ghosts, flitwick, commentator  **Community 2**  minerva, ron, hermione, neville, trevor, chess  **Community 3**  draco, filch, fang  **Community 4**  hagrid, harry, vernon, dudley, snakes, petunia, quirrel, ollivander, pub, gringotts, sorting\_hat, snape, wood, lily, james, norbert, firenze, voldemort  **Community 5**  mrs\_weasley, ginny, fred\_george | 0.21707124389543234 |

.

For test case 1 and 2 we have provided a dataset with and without weights to see how edge weights can affect a position of a node in a certain community.

The “demo\_weighted.csv” file is given below:

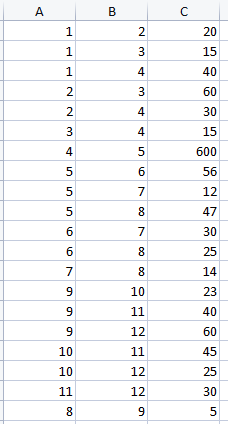


Figure 8:demo\_weighted.csv file

If the edge weights are not considered then the graph will look like this:

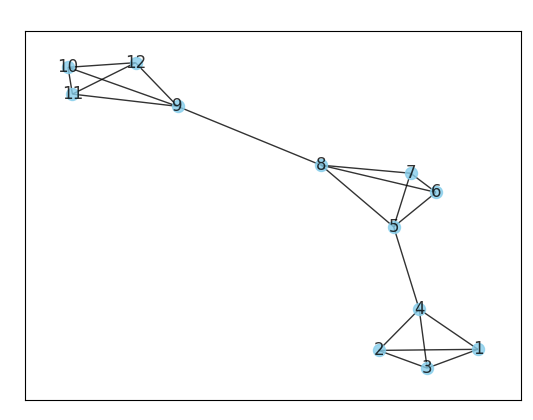


Figure 9:Visualization of Unweighted Demo data

We can clearly see 3 communities/clusters present with our eyes. The communities are

[ [1,2,3,4], [5,6,7,8], [9,10,11,12] ]

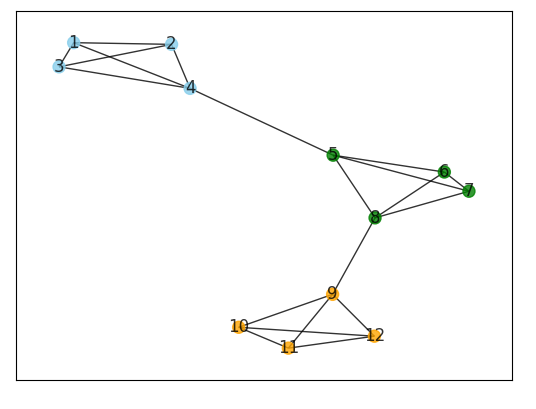


Figure 10:Visualization of communities in Unweighted Demo data

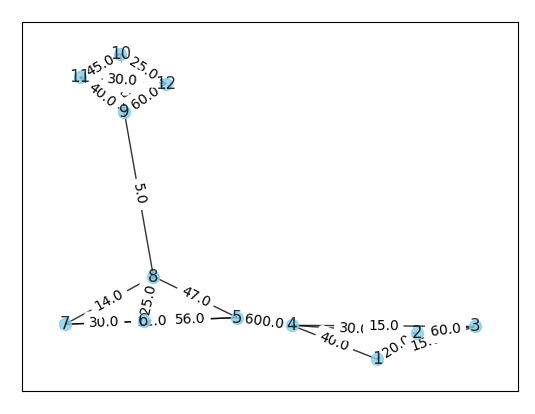
Now if we consider the edge weights as given above we can clearly see a large amount of edge weight is induced between nodes 4 and 5. 

Figure 11:Visualization of weighted Demo data

So after running the algorithm on the same dataset with weights we get the communities as

[ [1,2,3], [4,5], [6,7,8], [9,10,11,12] ]

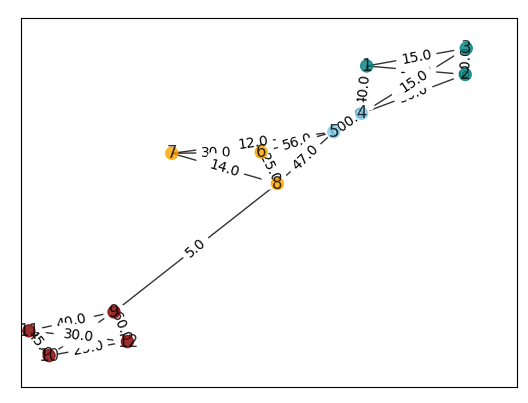


Figure 12:Visualization of communities in weighted Demo data

We can clearly see how edge weights have affected the result in detected communities.

Now we ran the same on our Harry Potter Movie data.

The data has 42 nodes as 42 characters and the edge\_weights between them as interactions.

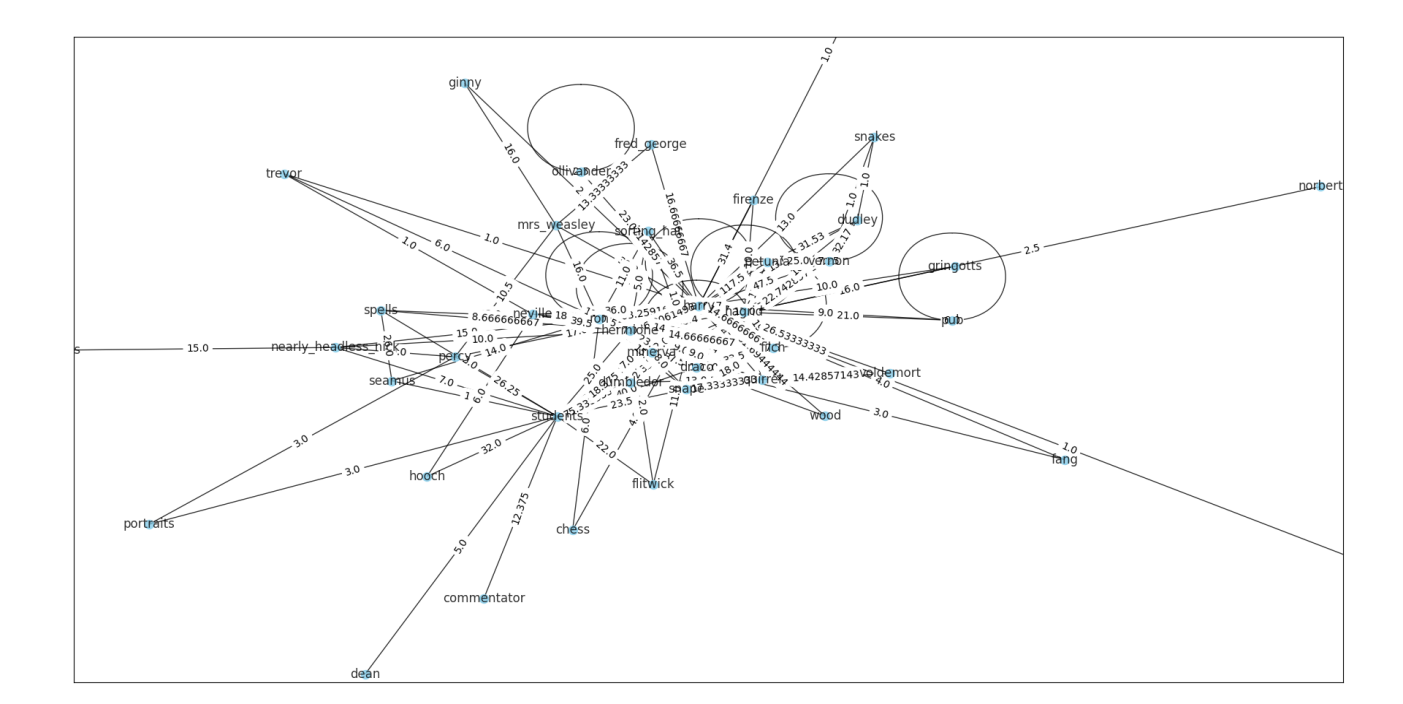


Figure 13:2D visualization of Harry Potter movie data

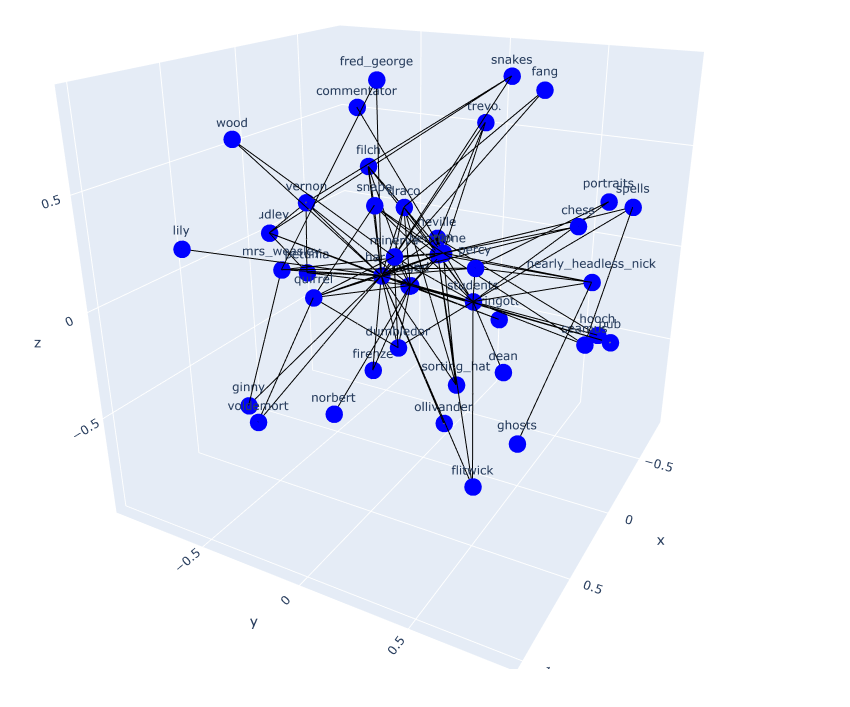


Figure 14:3D visualization of Harry Potter movie data

After running the algorithm we get the communities:

**Community 1**

dumbledor, percy, spells, students, nearly\_headless\_nick, portraits, seamus, hooch, dean, ghosts, flitwick, commentator

**Community 2**

minerva, ron, hermione, neville, trevor, chess

**Community 3**

draco, filch, fang

**Community 4**

hagrid, harry, vernon, dudley, snakes, petunia, quirrel, ollivander, pub, gringotts, sorting\_hat, snape, wood, lily, james, norbert, firenze, voldemort

**Community 5**

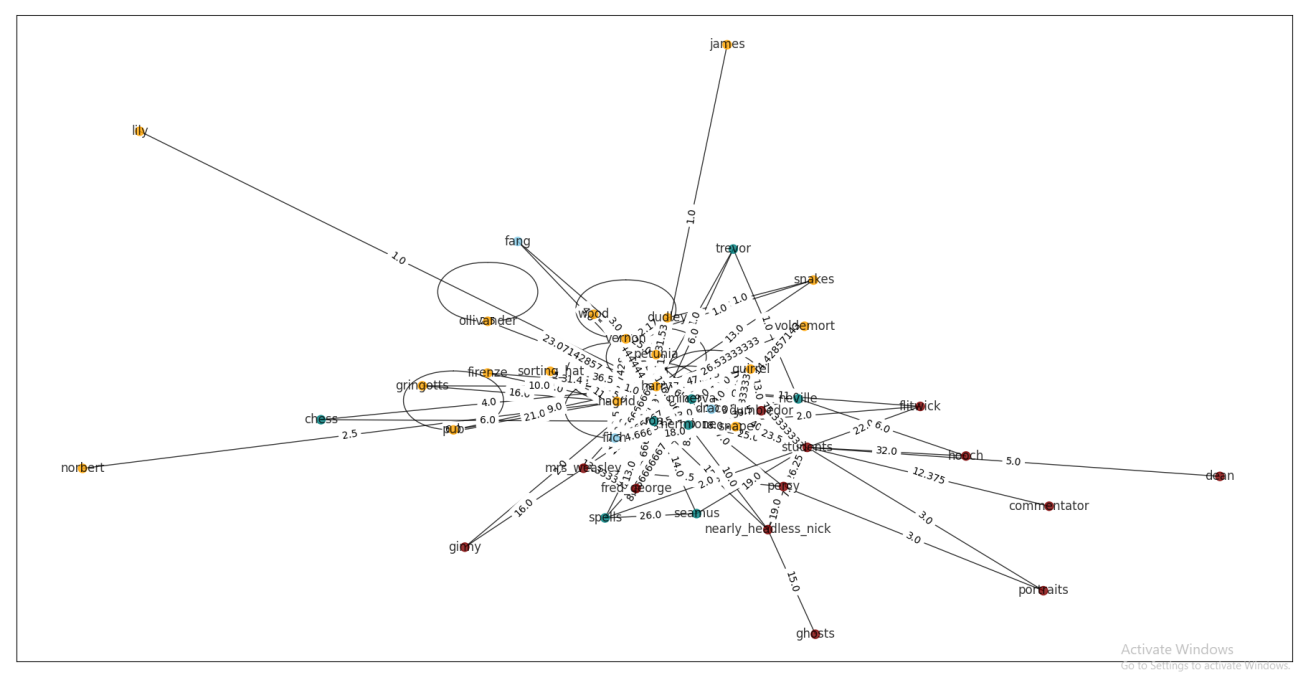
mrs\_weasley,ginny, fred\_george

Figure 15:2D visualization of communities in Harry Potter movie data

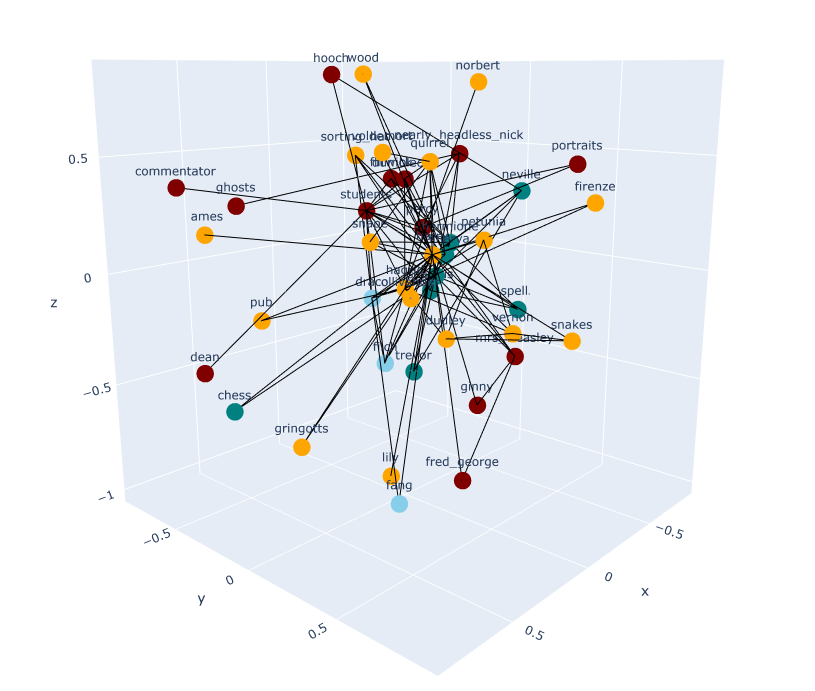


Figure 16:3D visualization of communities in Harry Potter movie data

Pearson Correlation Coefficient analysis

For interactions in between Community 1:

dumbledor, percy, spells, students, nearly\_headless\_nick, portraits, seamus, hooch, dean, ghosts, flitwick, commentator

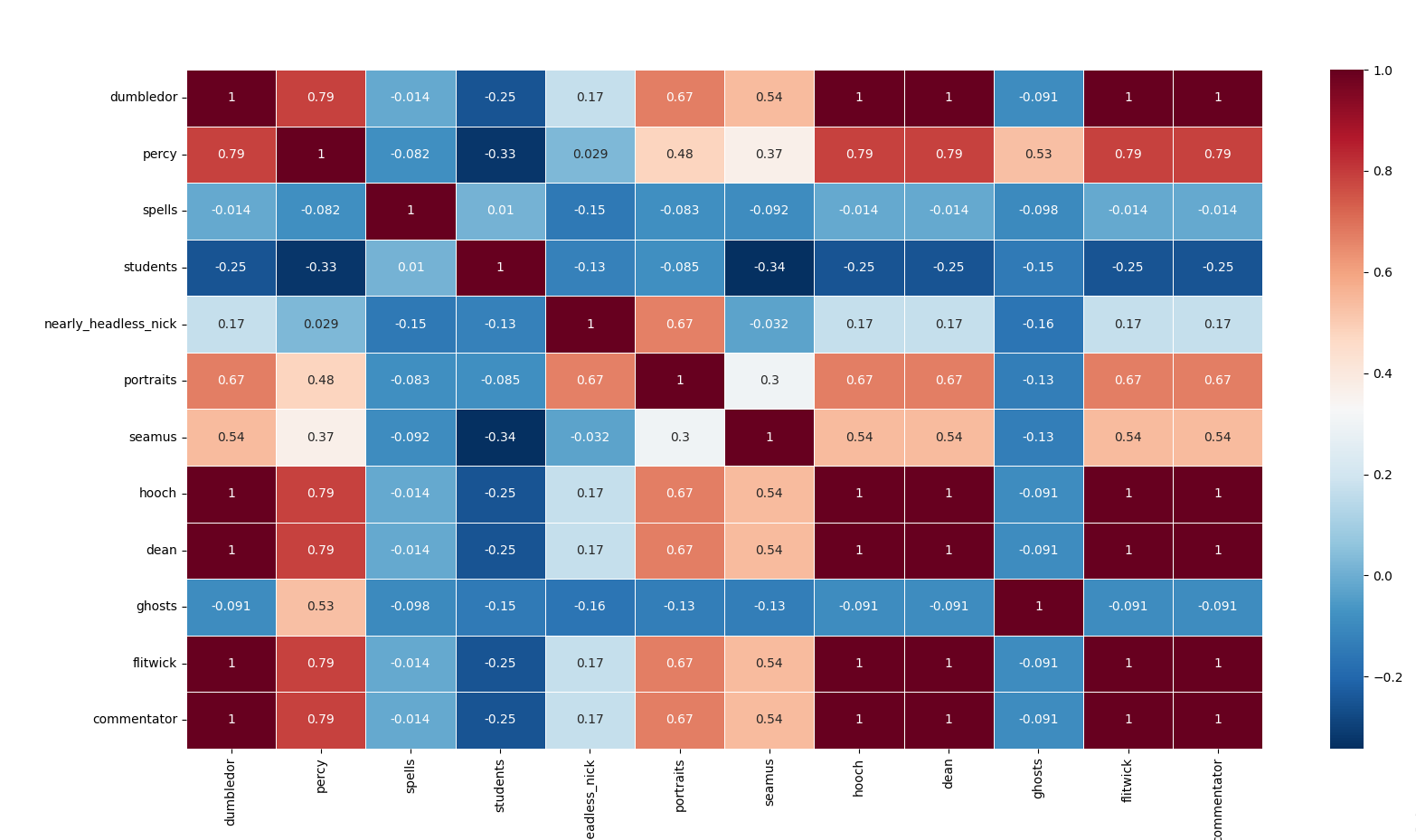


Figure 17:Pearson Correlation Coefficient for Characters in Community 1

For interactions in between Community 2:

minerva, ron, hermione, neville, trevor, chess



Figure 18:Pearson Correlation Coefficient for Characters in Community 2

For interaction in between Community 3:

draco, filch, fang

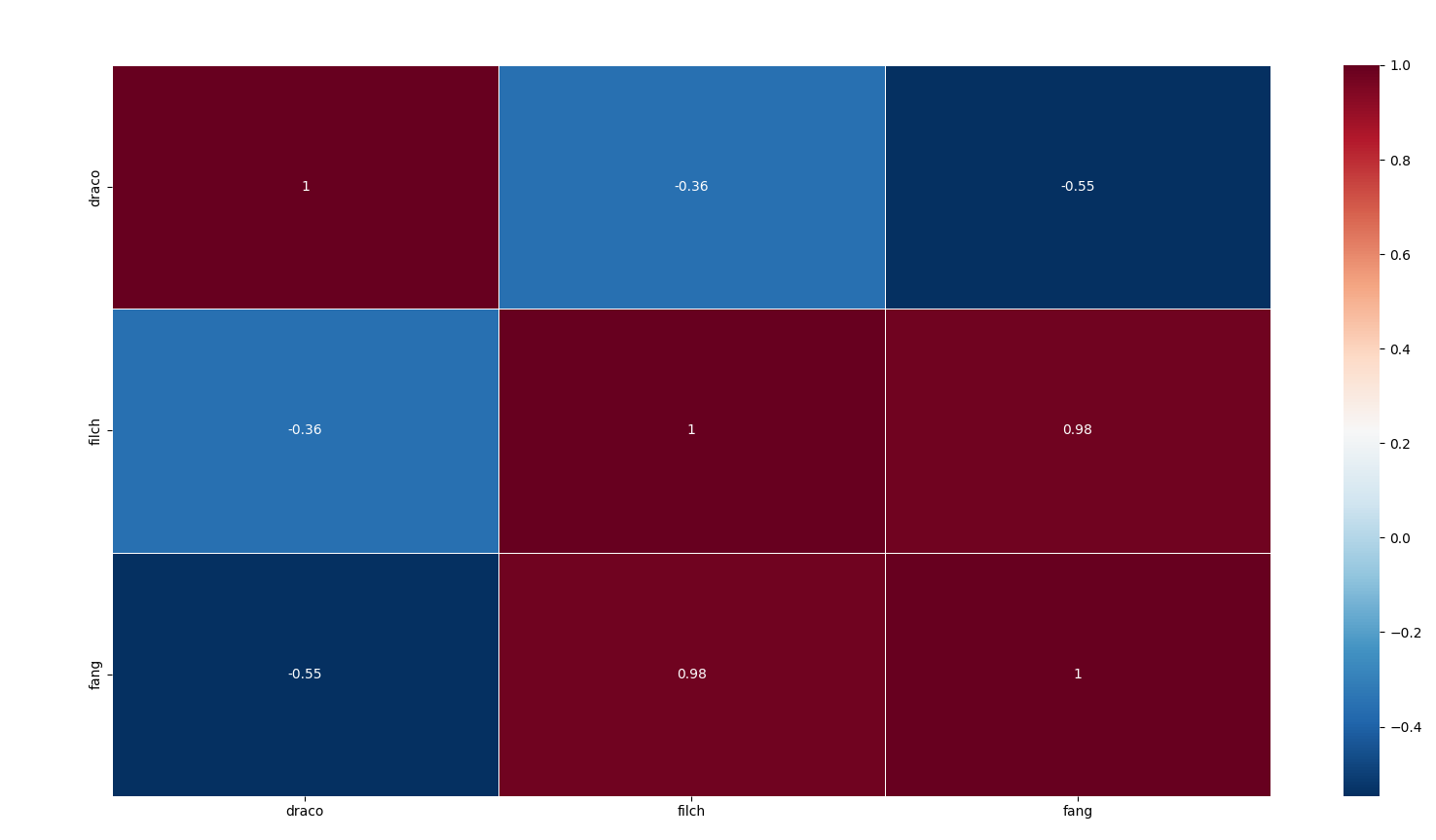


Figure 19:Pearson Correlation Coefficient for Characters in Community 3

For interaction in between Community 4:

hagrid, harry, vernon, dudley, snakes, petunia, quirrel, ollivander, pub, gringotts, sorting\_hat, snape, wood, lily, james, norbert, firenze, voldemort

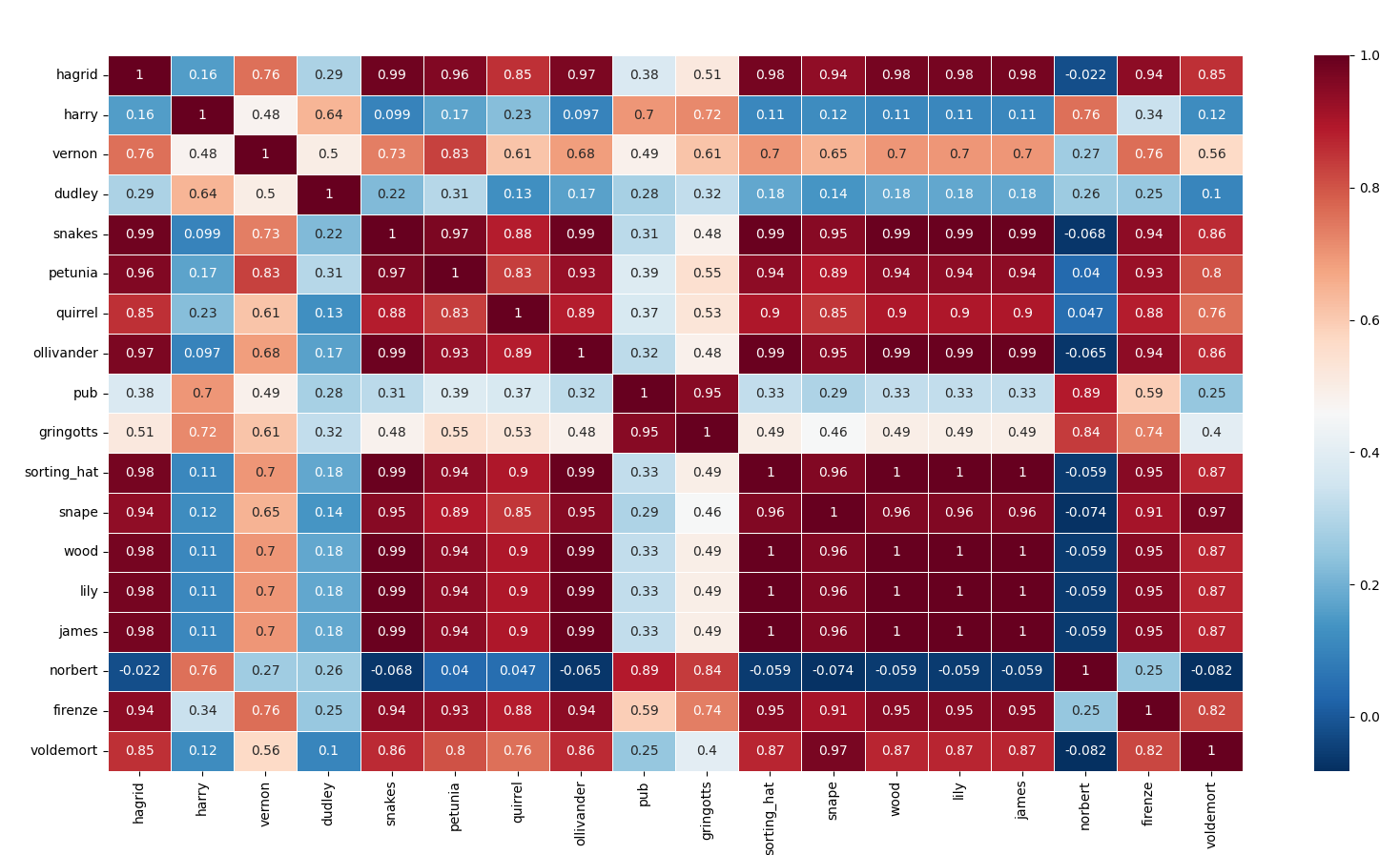


Figure 20:Pearson Correlation Coefficient for Characters in Community 4

For interactions in between community 5:

mrs\_weasley,ginny, fred\_george

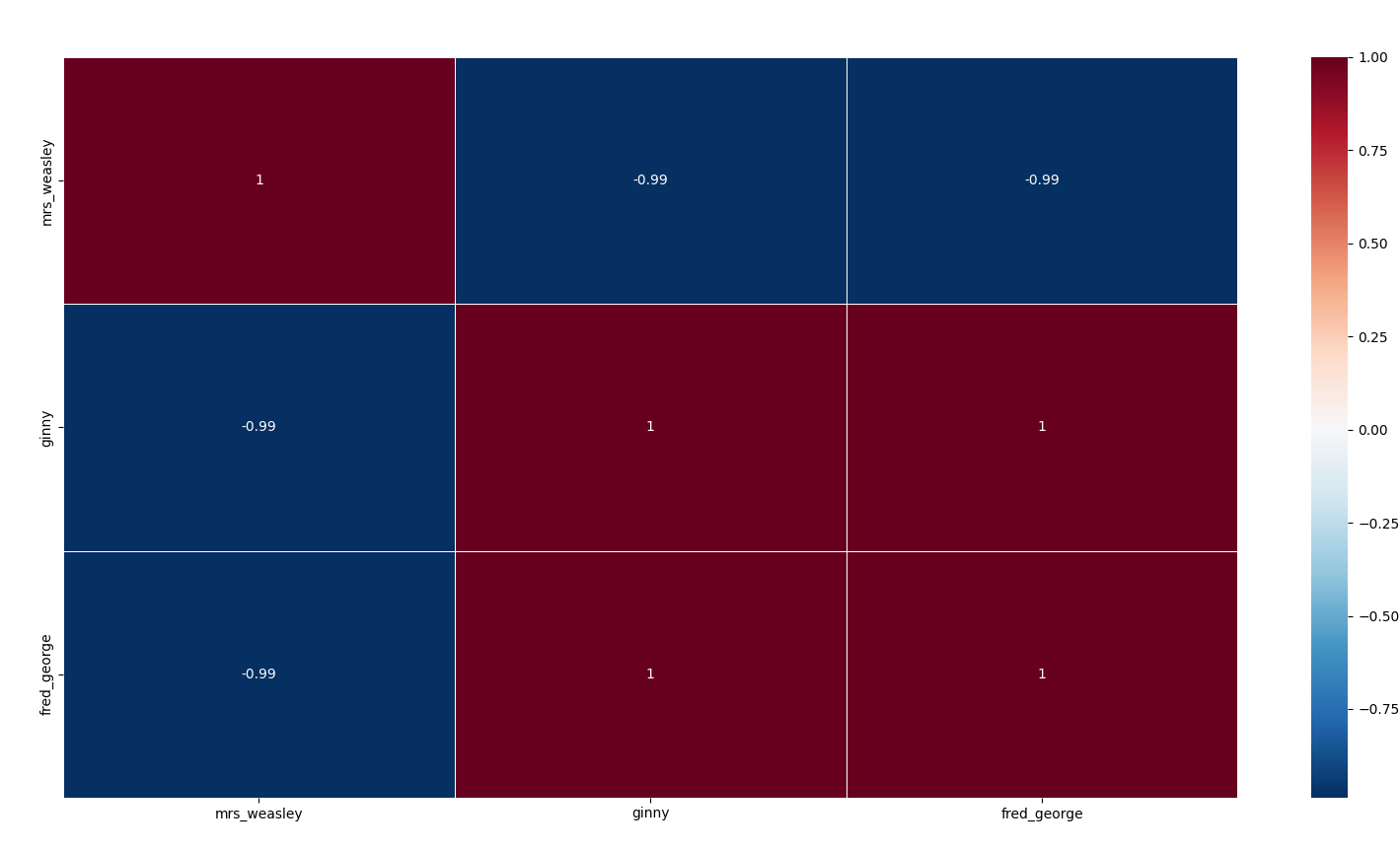


Figure 21:Pearson Correlation Coefficient for Characters in Community 5

For interactions in between communities i.e. actors from different communities interacting with each other

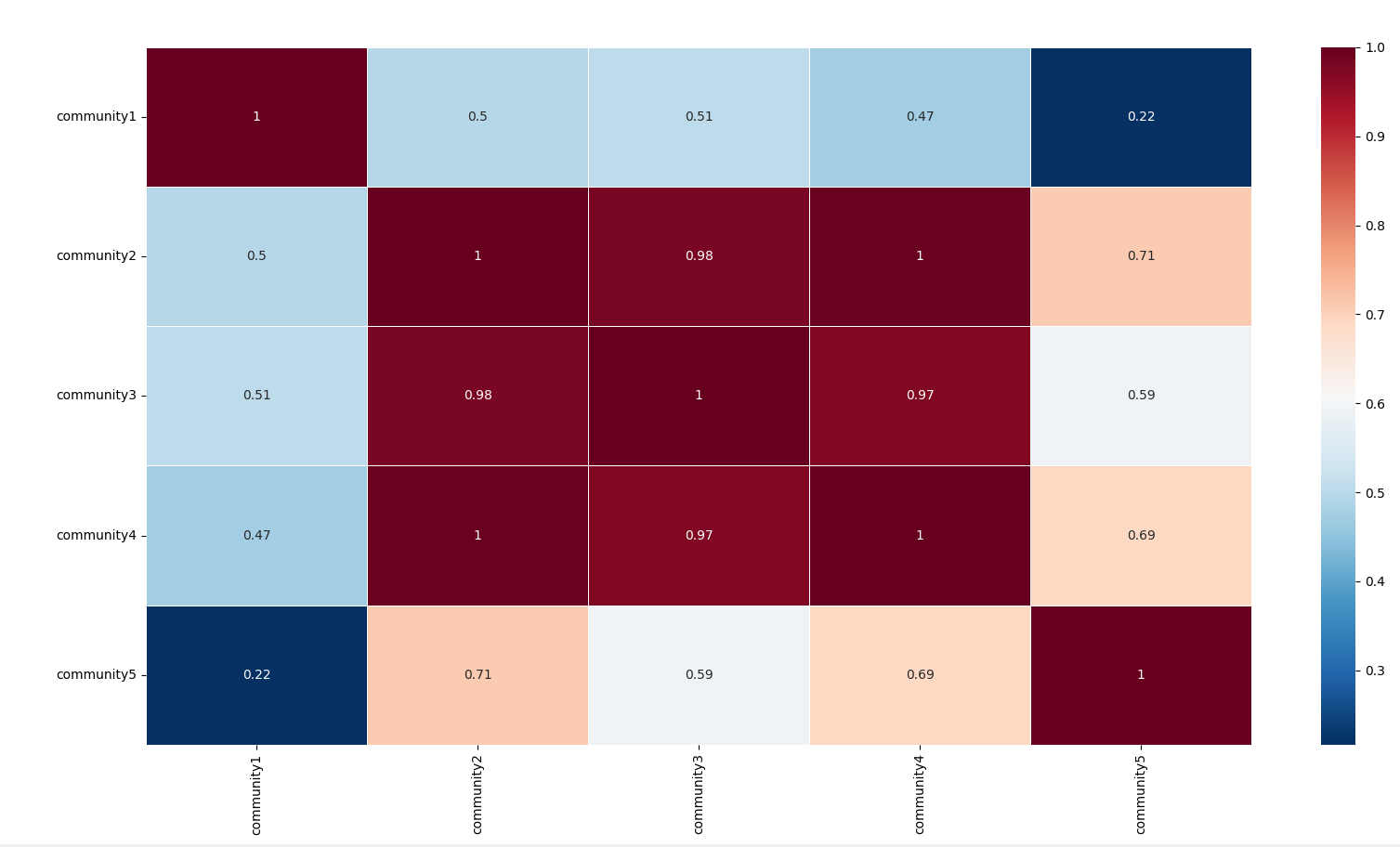


Figure 22:Pearson Correlation Coefficient for inter-community interactions

We can clearly see that with higher interaction in between communities the Pearson Correlation coefficient also increases.

It also means the 2 different communities with almost 1 coefficient could act like a single community and their actors have highers engagement with other actors from the second community.

# Conclusion

## 8.1 Project Benefits

Project Benefit: Enhanced Analysis of Inter-Layer Clusters in Real-Life Networks

The project's focus on utilizing multi-layer graphs and calculating metrics such as NMI (Normalized Mutual Information) and purity index brings several benefits for analyzing inter-layer clusters in real-life networks with large datasets. Here's a breakdown of the project benefit:

Comprehensive Modeling of Multiple Relationship Types: In real-life networks, entities often have multiple types of relationships with one another. By modeling these relationships using a multi-layer graph, the project captures the complexity and diversity of interactions more effectively. This comprehensive modeling allows for a more accurate representation of the underlying community structure in the network.

Analysis of Inter-Layer Clusters: The use of multi-layer graphs enables the identification and analysis of inter-layer clusters, which are communities that span across different relationship types or aspects of the entities. These inter-layer clusters provide valuable insights into the interconnectedness and dependencies among various aspects of the network. By analyzing these clusters, researchers and analysts can gain a deeper understanding of the complex relationships and dynamics within the network.

Application to Real-Life Networks with Huge Data: The project's benefits extend to real-life networks that have massive amounts of data. Traditional analysis methods may struggle to handle such large datasets efficiently. However, by leveraging the multi-layer graph representation and the calculated metrics (such as NMI and purity index), the project provides a scalable approach for analyzing inter-layer clusters in these networks. This scalability allows for the exploration and understanding of complex network structures, even in cases where the data is extensive and diverse.

Generalizability and Transferability: The insights and findings derived from this project can be extended to other real-life networks with similar characteristics. By demonstrating the effectiveness of the approach on a specific dataset, researchers and practitioners can apply the knowledge and methodologies to various domains, such as social networks, online communities, biological networks, and more. This generalizability allows for the widespread applicability of the project's outcomes and contributes to the advancement of network analysis in diverse fields.

In summary, the project benefit lies in the enhanced analysis of inter-layer clusters in real-life networks through the use of multi-layer graphs and calculated metrics. This approach enables a more comprehensive modeling of multiple relationship types, facilitates the analysis of interconnected communities, addresses scalability challenges in handling large datasets, and offers generalizable insights for various domains. Ultimately, it contributes to a deeper understanding of complex network structures and dynamics, leading to more informed decision-making and further advancements in network analysis.

## Future Scope for improvements

Collecting data from different users' social networking sites: Currently, the project is using a publicly available dataset. In the future, there is a plan to gather data from various social networking sites used by different users. This would involve obtaining the necessary permissions and consent from users to access their data. By collecting data from multiple sources, the project can have a more diverse and comprehensive dataset, which can potentially lead to better community detection results.

Utilizing underlying strategies for community detection: The scope includes exploring and implementing various community detection strategies such as cluster expansion, matrix factorization, unified distance, model-based approaches, and pattern mining. Each strategy has its own algorithms and techniques for identifying communities within a given dataset. By experimenting with different strategies, the project can compare their effectiveness and determine which approaches work best for the specific dataset and community detection goals.

Comparison of different community detection strategies: Once multiple strategies for community detection are implemented, a comparison analysis can be conducted. This involves evaluating the performance, accuracy, and efficiency of each strategy. By comparing the results obtained from different techniques, it becomes possible to identify the strengths and weaknesses of each method. This analysis can help in selecting the most suitable strategies for future applications or for improving the existing community detection system.

Multi-layer graph analysis for discovering qualified communities: Entities in social networks often have multiple relations or aspects associated with them. For example, a user might have connections based on their interests, geographical location, profession, or social activities. The scope includes exploring the concept of multi-layer graphs, where each layer represents a specific aspect or relationship. By leveraging the information available in multi-layer graphs, the project aims to discover more qualified communities that can exploit and fuse various aspects of information related to the entities. This can lead to a more comprehensive understanding of the community structure and dynamics within the dataset.

Overall, the future scope aims to enhance the community detection process by incorporating real user data, exploring different strategies, comparing their performance, and leveraging multi-layer graph analysis. These improvements can potentially lead to more accurate and meaningful community detection results, enabling better insights and applications in social network analysis.

# References / Bibliography

* Meena, Pawan, Mahesh Pawar, and Anjana Pandey. "A Survey on Community Detection Algorithm and Its Applications." Turkish Journal of Computer and Mathematics Education (TURCOMAT) 12.6 (2021): 4807-4815.
* Huang, Xinyu, et al. "A survey of community detection methods in multilayer networks." Data Mining and Knowledge Discovery 35.1 (2021): 1-45.
* Kim, Jungeun, and Jae-Gil Lee. "Community detection in multi-layer graphs: A survey." ACM SIGMOD Record 44.3 (2015): 37-48.
* <https://snap.stanford.edu/data/ego-Facebook.html>
* Blondel, Vincent & Guillaume, Jean-Loup & Lambiotte, Renaud & Lefebvre, Etienne. (2008). Fast Unfolding of Communities in Large Networks. Journal of Statistical Mechanics Theory and Experiment. 2008. 10.1088/1742-5468/2008/10/P10008.
* Interdonato, R., Tagarelli, A., Ienco, D. et al. Local community detection in multilayer networks. Data Min Knowl Disc 31, 1444–1479 (2017). <https://doi.org/10.1007/s10618-017-0525-y>
* M. Chen, K. Kuzmin and B. K. Szymanski, "Community Detection via Maximization of Modularity and Its Variants," in IEEE Transactions on Computational Social Systems, vol. 1, no. 1, pp. 46-65, March 2014, doi: 10.1109/TCSS.2014.2307458.
* P. De Meo, E. Ferrara, G. Fiumara and A. Provetti, "Generalized Louvain method for community detection in large networks," 2011 11th International Conference on Intelligent Systems Design and Applications, Cordoba, Spain, 2011, pp. 88-93, doi: 10.1109/ISDA.2011.6121636.
* Annamaria Ficara & Giacomo Fiumara & Salvatore Catanese & Pasquale De Meo & Xiaoyang Liu, 2022."The Whole Is Greater than the Sum of the Parts: A Multilayer Approach on Criminal Networks," Future Internet, MDPI, vol. 14(5), pages 1-21, April.
* T. Aynaud. python-louvain x.y: Louvain algorithm for community detection. 2020. https://github.com/taynaud/python-louvain