Cluster Comet

Submitted By:

ARANYA MAJI (21103005)

GARGI JAIN (21103018)

ISHITA SETHI (21103006)

Under the supervision of:

Dr. DHANALEKSHMI G.

December – 2023

Submitted in partial fulfilment of the Degree of

Bachelor of Technology

In

Computer Science Engineering

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING & INFORMATION TECHNOLOGY

JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

TABLE OF CONTENTS

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and beliefs, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma from a university or other institute of higher learning, except where due acknowledgment has been made in the text.

Place: Jaypee Institute of Information Technology, Noida
Date: 2nd December, 2023

Name: Aranya Maji
Enrolment No.: 21103005
Name: Gargi Jain
Enrolment No.: 21103018
Name: Ishita Sethi
Enrolment No.: 21103006

CERTIFICATE

This is to certify that the work titled **Cluster Comet** submitted by **Aranya Maji, Gargi Jain and Ishita Sethi** in partial fulfilment for the award of degree of B. Tech of Jaypee Institute of Information Technology, Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Signature of Supervisor
Dr. Dhanalekshmi G.
Assistant Professor (Senior Grade) CSE/IT

ACKNOWLEDGEMENT

I express my sincere gratitude to Dr. Dhanalekshmi G., Dept. of CSE/IT of Jaypee Institute of Information Technology, Noida for her stimulating guidance, continuous encouragement and supervision throughout the course of present work.

I also wish to extend my thanks to my teachers and classmates for their insightful comments and constructive suggestions to improve the quality of this project work.

Name: Aranya Maji

Enrolment No.: 21103005

Name: Gargi Jain

Enrolment No.: 21103018

Name: Ishita Sethi

Enrolment No.: 21103006

ABSTRACT

This report presents a Python web application that allows users to input graph network csv dataset of a certain format and select the desired and most suitable Community Detection Algorithm from the list based on the type of network. The Frontend of the webapp is made with the help of HTML, CSS and JS and integrated to the Python program through Flask. The project focuses on providing a platform to the users where they can work, experiment, research and compare different Community Detection Algorithms on the same dataset.

The webapp includes the main page that contains all the information regarding the project, case studies that are popular and the one which we have implemented ourselves for a detailed analysis, and also a Try Out section to use the webapp along with a complete guide about how to proceed.

Community Detection plays an important role right from recommending friends in social media to compatible proteins for fusion, from the field of entertainment to the field of deep research, community detection plays an invisible yet one of the most important roles in the realm of Data Science.

LIST OF ABBREVIATIONS

1. HTML:- Hyper Text Markup Language
2. CSS:- Cascading Style Sheets
3. JS:- JavaScript
4. Webapp:- Web Application
5. CSV:- Comma Separated Value
6. URL:- Uniform Resource Locator
7. API:- Application Programming Interface
8. UI:- User Interface
9. UX:- User Experience

CHAPTER 1 – INTRODUCTION

1.1 GENERAL INTRODUCTION

Community detection is a pivotal analytical tool in network science, addressing the intrinsic human desire to understand and categorize complex relationships within intricate systems. The need for community detection spans across diverse domains, from unravelling the intricate web of social interactions in online platforms to discerning the collaborative dynamics of biological entities. By identifying clusters or communities of densely interconnected nodes, this computational technique sheds light on the underlying organizational principles of networks, offering invaluable insights for various applications. Whether optimizing the design of transportation networks, enhancing the efficacy of recommendation systems, or aiding criminal investigations by delineating patterns in illicit activities, community detection serves as a versatile lens through which we can comprehend, simplify, and navigate the complexities inherent in interconnected systems. It not only facilitates a clearer visualization of network structures but also empowers researchers, analysts, and decision-makers to derive meaningful interpretations, contributing to advancements in fields as diverse as epidemiology, sociology, and information technology.

Many individuals, including those without a background in network science or data analysis, may have datasets that represent networks (e.g., social connections, collaboration networks). Our project allows them to gain insights into the community structure of their data without requiring advanced technical knowledge.

Also, in the era of social media, users often have large networks of connections. Our project can help users identify communities within their social networks, revealing patterns of interaction, shared interests, or common affiliations. Along with this our website can serve as an educational tool, allowing students and researchers to explore community detection concepts in a practical and interactive manner. It could be used as a teaching aid in courses related to network science, data analysis, or machine learning.

This report outlines the design, implementation, and evaluation of the website, including its features and functionality, user experience, and performance metrics. It also includes a detailed description of the website's design and development process, including the technologies used and the algorithms implemented. Finally, the report concludes by outlining potential future improvements and areas for further research. Overall, the website provides a convenient and efficient solution for students to swap their elective subjects, enhancing their academic experience.

1.2 PROBLEM STATEMENT

In the era of interconnected data, individuals and organizations often possess datasets representing complex networks, such as social connections, collaborations, or interactions. Analysing these networks to identify meaningful communities or clusters is a challenging task, particularly for users without a background in data science or network analysis. Existing tools may be too complex or lack user-friendly interfaces, hindering accessibility. This project aims to address this gap by developing a web-based community detection tool that empowers users to effortlessly analyse and visualize community structures within their input data. The tool provides an intuitive interface, robust algorithms, and insightful visualizations to cater to users from diverse domains, ranging from social media enthusiasts and business professionals to researchers and educators. The goal is to democratize the understanding of network structures, providing individuals, businesses, and researchers with a tool that facilitates the identification of communities, fosters informed decision-making, and opens avenues for exploration and collaboration.

Our problem statement encapsulates the need for a user-friendly community detection tool, outlines the target audience, and emphasizes the importance of accessibility and applicability across various domains.

To address this problem, we propose a user-friendly webapp Cluster Comet . The aim of this webapp is to provide users with a user-friendly and intuitive platform for community detection in their input data. By leveraging advanced network science algorithms, the website aims to empower users, regardless of their technical expertise, to gain insights into the inherent structures within their networks.

1.3 SIGNIFICANCE OF THE PROBLEM

The significance of the problem statement lies in addressing the growing demand for accessible and user-friendly community detection tools in the context of network analysis. Here are some key aspects highlighting the significance of this problem statement:

1. Democratizing Network Analysis: The problem statement addresses the need to democratize network analysis by creating a platform that is accessible to users with varying levels of technical expertise. This inclusivity enables a broader audience, including individuals, businesses, educators, and researchers, to harness the benefits of community detection in their data.
2. Empowering Non-Technical Users: The significance lies in empowering users who may not have a background in advanced data science or network analysis. By providing a user-friendly interface and clear visualizations, the webapp enables non-technical users to explore and understand the community structures within their datasets.
3. Applicability Across Domains: The problem statement recognizes the diverse applications of community detection in various domains, including social networks, business intelligence, academia, and event planning. This versatility makes the webapp valuable for individuals and organizations across different sectors.
4. Insights for Decision-Making: Community detection provides valuable insights into the organizational and relational structures within networks. The significance of this problem statement lies in offering users actionable insights that can inform decision-making processes, whether in marketing, organizational planning, or research.
5. Educational Impact: The development of an educational tool is significant for promoting understanding and learning in the field of network science. The website's features can contribute to educational initiatives, helping students and learners grasp the concepts of community detection through practical application.

In summary, the significance of this problem statement lies in its potential to make community detection accessible, applicable, and valuable across diverse user groups and domains, ultimately contributing to informed decision-making and a deeper understanding of network structures.

1.4 EMPIRICAL STUDY

In the realm of community detection, our empirical study delved into the intricacies of four key algorithms: Louvain, Girvan-Newman, Modularity Optimization, and Label Propagation. Each algorithm was subjected to a meticulous series of experiments, strategically utilizing diverse network datasets to assess their efficacy in uncovering underlying community structures. The Louvain algorithm, celebrated for its efficiency, demonstrated a remarkable ability to swiftly identify dense and modular communities, making it particularly well-suited for large-scale networks where computational speed is paramount. Girvan-Newman, with its emphasis on edge-betweenness, emerged as a powerful tool for unveiling hierarchical structures within networks, providing a unique perspective on community organization.

In the case of Modularity Optimization, the algorithm showcased robust performance, excelling in the detection of well-defined communities by optimizing modularity scores. This method proved particularly effective in networks with clear modular structures, shedding light on cohesive groups of nodes with dense internal connections. On the other hand, Label Propagation exhibited a versatile adaptability, demonstrating effectiveness across a spectrum of network structures. Its ability to propagate labels through the network allowed for the identification of communities in scenarios with varying degrees of structural complexity.

The diverse range of network datasets employed in our study mirrored real-world scenarios, capturing the nuances of social, biological, and technological networks. The findings from these experiments offer practitioners nuanced insights into the strengths and weaknesses of each algorithm, empowering them to make informed decisions based on the specific characteristics of the network under investigation. This comprehensive understanding of algorithmic behaviour in different contexts contributes significantly to the evolving landscape of community detection, guiding researchers and practitioners towards optimal choices in their pursuit of unravelling the complex tapestry of community structures within networks.

1.5 BRIEF DESCRIPTION OF THE SOLUTION APPROACH

The solution approach for the community detection project involves the integration of robust community detection algorithms into a user-friendly web platform. Here is a description of the key components and steps involved in the solution:

1. Algorithm Selection and Implementation:
    a. Objective: Identify and implement community detection algorithms that cater to the project's goals.
    b. Approach: Chose algorithms based on their ability to handle various network structures, Common algorithms like Louvain, Label Propagation, and Girvan-Newman considered. These algorithms are implemented with attention to efficiency and adaptability to different types of networks.
2. Web Platform Development:
    a. Objective: Create an intuitive and user-friendly web interface for community detection.
    b. Approach: Employed web development frameworks and frontend technologies to build a responsive and visually appealing platform. Designed a user interface that guides users through the process of uploading their data, configuring parameters, and visualizing community structures.
3. Data Input and Preprocessing:
    a. Objective: Enable users to input their network data easily.
    b. Approach: Implemented data upload functionality that accepts data from user
4. Visualizations and Result Display:
    a. Objective: Provide users with clear visualizations of community structures.
    b. Approach: Utilized graph visualization libraries (e.g., Networkx, plotly) to represent detected communities graphically. Included color-coded nodes, edge thickness, and other visual cues to highlight community boundaries.
5. User Guidance and Parameter Configuration:
    a. Objective: Guide users through the community detection process and parameter configuration.
    b. Approach: Implement tooltips, step-by-step instructions, and contextual help to assist users in understanding the significance of community detection parameters.

1.6 COMPARISON OF EXISTING APPROACHES TO THE PROBLEM FRAMED

1. None of the existing platforms of similar kind allow users to work in online mode.
2. Existing platforms of similar kind are mostly analysing network graphs.
3. They are asking user a lot technicalities about the dataset which makes it difficult for a beginner to perform analysis and community detection.
4. They don’t let user choose between algorithms making it unclear to the user about comparative analysis of various algorithms.
5. They are not entirely dedicated to community detection.
6. Also, there are very less tools which let user perform community detection. Until, now community detection is a process which is generally carried out by user by coding for his/her requirements which makes it difficult for a person from non-technical background to understand the meaning and use of community detection.

CHAPTER 2 – LITERATURE SURVEY

2.1 SUMMARIES OF PAPERS STUDIED

1. https://www.researchgate.net/publication/310736499_Community_Detection_Based_on_Girvan_Newman_Algorithm_and_Link_Analysis_of_Social_Media

   a. **SUMMARY:**

   The paper titled "Community Detection Based on Girvan-Newman Algorithm and Link Analysis of Social Media" explores the application of the Girvan-Newman algorithm and link analysis techniques for community detection in social media networks. Published on ResearchGate, this paper delves into the realm of social network analysis, aiming to uncover meaningful communities within complex online interactions. The following is a comprehensive summary of the literature presented in this paper.

   b. **ABSTRACT:**

   The abstract of the paper provides a succinct overview of the research objectives and methodologies employed. It introduces the utilization of the Girvan-Newman algorithm and link analysis in the context of social media community detection. The abstract outlines the significance of community detection in understanding online interactions and hints at the specific contributions of the study.

   c. **INTRODUCTION:**

   The introduction sets the stage by emphasizing the pervasive influence of social media platforms in contemporary society. It underscores the intricate nature of online interactions and the need for effective tools to discern underlying community structures. The Girvan-Newman algorithm is introduced as a prominent method for community detection, and the rationale behind choosing link analysis as a supplementary technique is explained.

d. LITERATURE REVIEW:

The literature review delves into the existing body of knowledge on community detection in social media networks. It discusses prior research that has utilized the Girvan-Newman algorithm, highlighting its strengths and limitations. Additionally, the review touches upon other methodologies used in similar studies and the challenges associated with community detection in dynamic and evolving online environments.

e. GIRVAN-NEWMAN ALGORITHM:

A significant portion of the paper is dedicated to elucidating the Girvan-Newman algorithm. The authors described the underlying principles of the algorithm, emphasizing its edge-betweenness centrality concept and iterative edge removal strategy. The section also delves into the algorithm's ability to uncover communities by detecting bridges and bottlenecks in the network.

f. LINK ANALYSIS TECHNIQUES:

The paper introduces link analysis techniques as a complementary approach to community detection. This involves exploring the interconnectedness of nodes and analysing the structure of links between them. The authors discussed the rationale behind incorporating link analysis and how it enhances the overall accuracy of community detection in social media networks.

g. METHODOLOGY:

The methodology section outlines the specific steps taken in the research. This includes details on data collection from social media platforms, preprocessing steps to clean and structure the data, and the application of the Girvan-Newman algorithm and link analysis techniques. The authors justified their choices regarding parameter settings and any adaptations made to suit the characteristics of social media networks.

h. RESULTS AND DISCUSSION:

The results and discussion section presents the outcomes of applying the Girvan-Newman algorithm and link analysis to social media data. The authors showcased visual representations of detected communities and provide quantitative metrics to evaluate the performance of their approach. Discussions revolved around the effectiveness of the method, insights gained from the detected communities, and potential real-world applications.

i. CONCLUSION:

In the conclusion, the paper summarizes key findings, reiterates the significance of the research, and discusses the implications of the results. The conclusion serves as a wrap-up, emphasizing the practical relevance of the Girvan-Newman algorithm and link analysis in uncovering community structures within social media networks.

In essence, this paper contributes to the growing body of literature on community detection in social media by leveraging the Girvan-Newman algorithm and link analysis techniques. Its focus on real-world applications and the analysis of online interactions provides valuable insights for researchers, practitioners, and stakeholders interested in understanding the complex dynamics of social media communities.

2. https://www.researchgate.net/publication/51929918_Generalized_Louvain_Method_for_Community_Detection_in_Large_Networks

   a. **SUMMARY:**

      The paper titled "Generalized Louvain Method for Community Detection in Large Networks" explores an extension of the Louvain method for community detection, aiming to enhance its scalability for large networks. Published on ResearchGate, this research contributes to the field of network science by addressing the computational challenges posed by the increasing size and complexity of modern networks. The following is a comprehensive summary of the literature presented in this paper.

   b. **ABSTRACT:**

      The abstract provides a concise overview of the research objectives, methods, and findings. It introduces the concept of a Generalized Louvain method and highlights its application in the context of community detection in large networks. The abstract outlines the significance of addressing scalability issues and hints at the specific contributions of the study.

   c. **INTRODUCTION:**

      The introduction sets the stage by emphasizing the ubiquity of large-scale networks in various domains, such as social media, biological systems, and transportation networks. It highlights the importance of community detection as a fundamental task in network analysis and introduces the Louvain method as a widely-used algorithm. The motivation for a generalized version of the Louvain method is rooted in the need to overcome limitations in handling large-scale networks efficiently.

   d. **LITERATURE REVIEW:**

      The literature review provides a thorough examination of existing community detection methods, with a focus on the challenges associated with scalability in the context of large networks. The authors discussed prior research on the Louvain method, its strengths in terms of modularity optimization, and its limitations concerning computational efficiency for networks of substantial size.

e. LOUVAIN METHOD OVERVIEW:

This section delves into the foundational aspects of the Louvain method. The authors explained the algorithm's modularity optimization objective, iterative node movement between communities, and the concept of small communities merging into larger ones. The strengths of the Louvain method in detecting communities of varying sizes and its application in diverse network domains are likely highlighted.

f. CHALLENGES IN SCALABILITY:

The paper addresses the challenges posed by large networks and the computational bottlenecks associated with the standard Louvain method. Issues such as memory constraints and processing time for extensive datasets are discussed, laying the groundwork for the need for a generalized approach.

g. GENERALIZED LOUVAIN METHOD:

The core of the paper introduces the Generalized Louvain method. The authors described the modifications made to the original algorithm to enhance its scalability. This included strategies for optimizing memory usage, parallelization techniques, or adaptive parameter tuning. The generalized approach may also consider aspects like overlapping communities or dynamic networks to make the method versatile.

h. METHODOLOGY:

The methodology section outlines the specific steps taken to generalize the Louvain method. This includes details on algorithmic modifications, parameter choices, and any benchmark datasets used for testing and validation. The authors justified their approach by discussing its advantages over existing methods for large networks.

i. RESULTS AND DISCUSSION:

In this section, the paper presents the results of applying the Generalized Louvain method to large-scale networks. The authors showcase performance metrics, computational efficiency gains, and visual representations of the detected communities. The discussion interprets the results, compares them with the standard Louvain method and potentially with other existing algorithms, and addresses the implications of the findings.

j. CONCLUSION:

The conclusion summarizes the key contributions of the paper. The authors emphasized the effectiveness of the Generalized Louvain method in overcoming scalability challenges and providing accurate community detection in large networks. They discussed potential applications, acknowledge any limitations, and suggest avenues for future research, contributing to the ongoing advancement of community detection methodologies.

In summary, this paper extends the legacy of the Louvain method by proposing a Generalized Louvain approach, specifically tailored to address the computational complexities associated with large networks. Its contributions to the field lie in offering an efficient and scalable solution for community detection, with potential implications for a wide range of real-world applications across diverse network domains.

3. https://www.researchgate.net/publication/264817879_Label_propagation_algorithm_A_semi-synchronous_approach

   a. **SUMMARY :**

      The paper titled "Label Propagation Algorithm: A Semi-Synchronous Approach" explores an extension and refinement of the Label Propagation Algorithm (LPA), introducing a semi-synchronous approach. Published on ResearchGate, this research contributes to the field of network science by enhancing the adaptability and performance of the Label Propagation Algorithm in the context of community detection and node labelling in networks. The following is a comprehensive summary of the literature presented in this paper.

   b. **ABSTRACT:**

      The abstract provides a concise overview of the research objectives, methods, and findings. It introduces the semi-synchronous extension to the Label Propagation Algorithm (LPA) and underscores its significance in the context of community detection. The abstract outlines the challenges addressed by the semi-synchronous approach and hints at the specific contributions of the study.

   c. **INTRODUCTION:**

      The introduction begins by highlighting the ubiquity of complex networks in various domains, such as social networks and information networks. It emphasizes the importance of community detection as a fundamental task in network analysis and introduces the Label Propagation Algorithm as a popular and simple method for this purpose. The motivation for a semi-synchronous approach is rooted in addressing the limitations of the original algorithm, particularly its sensitivity to initial conditions and asynchronous nature.

   d. **LITERATURE REVIEW:**

      The literature review provides a comprehensive examination of existing community detection methods and node labelling algorithms, with a focus on the strengths and weaknesses of the Label Propagation Algorithm. Prior research on LPA's performance in various network types and its challenges in handling certain network structures is discussed. The authors also touched upon related work that explores synchronous or semi-synchronous strategies in other algorithms.

e. **LABEL PROPAGATION ALGORITHM OVERVIEW:**

This section delves into the foundational aspects of the Label Propagation Algorithm. The authors explained the algorithm's simplicity, where nodes adopt the majority label of their neighbours iteratively until a stable state is reached. The strengths of LPA, including its efficiency and effectiveness in detecting communities with balanced structures, are highlighted.

f. **CHALLENGES WITH ASYNCHRONOUS LPA:**

The paper addresses the challenges associated with the asynchronous nature of the original Label Propagation Algorithm. Issues such as sensitivity to initial conditions, potential bias introduced by the order of node processing, and the impact on the stability of detected communities are discussed. These challenges set the stage for the introduction of the semi-synchronous approach.

g. **SEMI-SYNCHRONOUS LABEL PROPAGATION ALGORITHM:**

The core of the paper introduces the semi-synchronous extension to the Label Propagation Algorithm. The authors described the modifications made to the original algorithm to introduce a more controlled and stable update mechanism. This involved incorporating elements of synchronicity into the label updating process, providing a compromise between full synchrony and asynchrony.

h. **METHODOLOGY:**

The methodology section outlines the specific steps taken to implement and evaluate the semi-synchronous Label Propagation Algorithm. Details on parameter choices, benchmark datasets used for testing, and performance metrics employed are likely provided. The authors justified their approach by discussing its potential advantages over asynchronous methods, especially concerning the stability of results.

i. **RESULTS AND DISCUSSION:**

In this section, the paper presents the results of applying the semi-synchronous Label Propagation Algorithm to various datasets. The authors showcased performance metrics, stability gains, and comparisons with the original LPA and potentially with other community detection methods. The discussion interprets the results, addresses the implications of the semi-synchronous approach, and explores potential use cases or scenarios where it outperforms the asynchronous version.

j. CONCLUSION:

The conclusion summarizes the key contributions of the paper. The authors emphasized the effectiveness of the semi-synchronous Label Propagation Algorithm in mitigating the challenges associated with the asynchronous nature of the original algorithm. They discussed potential applications, acknowledge any limitations, and suggest avenues for future research, contributing to the ongoing refinement and adaptation of label propagation algorithms for community detection.

In summary, this paper extends the applicability and stability of the Label Propagation Algorithm by introducing a semi-synchronous approach. By addressing the challenges associated with the asynchronous nature of the original algorithm, the research contributes to the ongoing efforts in enhancing the efficiency and accuracy of community detection methods in complex networks.

4. https://www.researchgate.net/publication/223956042_Modularity_optimization_in_community_detection_of_complex_networks

   a. **SUMMARY:**

      The paper titled "Modularity Optimization in Community Detection of Complex Networks" focuses on the critical concept of modularity in community detection within complex networks. Published on ResearchGate, this research significantly contributes to the field of network science by exploring the application of modularity optimization techniques in uncovering community structures. The following is a comprehensive summary of the literature presented in this paper.

   b. **ABSTRACT:**

      The abstract provides a concise overview of the research's objectives, methods, and findings. It introduces the central theme of modularity optimization in community detection within complex networks. The abstract outlines the significance of modularity as a metric and optimization goal in network analysis and hints at the specific contributions of the study.

   c. **INTRODUCTION:**

      The introduction begins by emphasizing the ubiquity of complex networks in various domains, from social networks to biological systems and technological infrastructures. It highlights the importance of community detection as a fundamental task in network analysis and introduces the concept of modularity as a quantitative measure for community structure. The motivation for exploring modularity optimization techniques is rooted in the desire to enhance the accuracy and effectiveness of community detection methods.

   d. **LITERATURE REVIEW:**

   The literature review provides a comprehensive survey of existing community detection methods and the role of modularity in evaluating community structures. The authors discussed prior research on the significance of modularity as a metric, its strengths, and its limitations. Other optimization methods and algorithms used for community detection are explored to set the stage for the specific modularity optimization techniques proposed in this paper.

e. MODULARITY METRIC OVERVIEW:

This section delves into the foundational aspects of the modularity metric. The authors explained how modularity measures the quality of community structures by comparing the density of edges within communities to that expected by chance. The strengths of modularity, including its interpretability and broad applicability, are highlighted. The challenges associated with resolution limits and sensitivity to community size variations are also discussed.

f. CHALLENGES IN MODULARITY OPTIMIZATION:

The paper addresses challenges associated with modularity optimization, including the computational complexity of exploring the vast solution space and the potential for algorithmic convergence to suboptimal solutions. These challenges set the stage for the introduction of specific modularity optimization techniques aimed at improving the efficiency and effectiveness of the community detection process.

g. MODULARITY OPTIMIZATION TECHNIQUES:

This is the core of the paper, where the authors introduce and detail their proposed modularity optimization techniques. The modifications or enhancements made to existing algorithms or the development of novel algorithms are explained. This involved strategies for exploring the solution space more efficiently, considering resolution limits, and addressing issues related to the sensitivity of modularity to community size.

h. METHODOLOGY:

The methodology section outlines the specific steps taken to implement and evaluate the proposed modularity optimization techniques. Details on parameter choices, benchmark datasets used for testing, and performance metrics employed are likely provided. The authors justified their approach by discussing its potential advantages over traditional modularity optimization methods.

i. **RESULTS AND DISCUSSION:**

In this section, the paper presents the results of applying the proposed modularity optimization techniques to various datasets. The authors showcased performance metrics, comparisons with traditional modularity optimization methods, and insights gained from the detected community structures. The discussion interprets the results, addresses the implications of the proposed techniques, and explores potential use cases or scenarios where they outperform existing methods.

j. **CONCLUSION:**

The conclusion summarizes the key contributions of the paper. The authors emphasized the effectiveness of the proposed modularity optimization techniques in addressing challenges associated with traditional methods. They discussed potential applications, acknowledge any limitations, and suggest avenues for future research, contributing to the ongoing refinement and adaptation of modularity optimization in community detection.

In summary, this paper significantly contributes to the field of community detection within complex networks by focusing on modularity optimization. By addressing the challenges associated with traditional modularity optimization methods, the research offers novel techniques that enhance the accuracy and efficiency of community detection. The exploration of modularity as a key metric in this context reflects the ongoing efforts to improve our understanding of complex network structures.

5. https://ieeexplore.ieee.org/document/8340627

   a. **SUMMARY :**

      The paper introduces the context of community detection algorithms for complex networks, emphasizing the significance of understanding these networks for comprehending real-life systems. It highlights the challenges posed by the enormous size of complex networks and the role of community detection in simplifying their comprehension. The paper mentions the diversity of proposed community detection techniques and their utility in revealing network structures with reduced effort. The concept of modularity is introduced as a crucial parameter for evaluating the quality of detected communities, with higher values indicating better community structures. The paper outlines the organization of subsequent sections, including an introduction to chosen algorithms, a description of modularity, details of datasets and experimental setup, results analysis, and a concluding section.

   b. **SECTION II OF PAPER: COMMUNITY DETECTION ALGORITHMS**

      The paper provides an overview of various community detection algorithms, their methodologies, and their applications. The importance of community detection in understanding complex networks is emphasized, as it has practical implications in fields such as recommendation systems, innovation diffusion, and viral marketing. The study conducted by S. Emminos et al. is referenced, where a comparison of Louvain, Infomap, Label Propagation, and Smart Local Moving algorithms was performed using modularity and information recovery metrics.

      The analysis presented in the paper focuses on community detection algorithms available in Python's igraph library, including Newman2006, Infomap, Louvain, Fast Greedy, Label Propagation, Spin-Glass, and Random-Walktrap algorithms.

      i. **Newman2006:**

         Utilizing spectral partitioning, Newman2006 calculates leading eigenvectors of the modularity matrix to partition the network. The process involves subdividing the network based on maximizing modularity, with modularity contribution calculated at each step until a negative contribution is reached.

ii. **Infomap Community Detection:**

Proposed by Martin Rosvall et al., this algorithm employs the map equation to find community structure. The map equation represents the description length of a random walker in a network, with partitions having smaller description lengths indicating better modular structures. The algorithm starts by considering each node as a separate module and iteratively combines nodes to minimize the map equation.

iii. **Louvain Community Detection:**

Proposed by Blondel et al., the Louvain algorithm works in multiple passes, utilizing modularity as the stopping criterion. It identifies local maxima of modularity and merges adjacent nodes to form communities. The process continues by treating these communities as nodes in subsequent phases.

iv. **Fast Greedy Community Detection:**

Clauset et al. proposed this algorithm, which focuses on networks with sparse adjacency matrices. It efficiently utilizes data structures to speed up community detection, starting with each node as a community and iteratively combining communities to maximize modularity.

v. **Label Propagation Community Detection:**

Raghavan et al. introduced a near-linear time algorithm where each node is assigned a unique label, and in each step, a node adopts the label most owned by its neighbours. Ties are resolved by randomly selecting a label among neighbours. The strong community measure is used as a stop condition.

vi. **Spin-Glass Community Detection:**

Proposed by J Reichardt, this algorithm considers the spin state of nodes as communities and aims to minimize the spin energy. It operates on the concept that nodes with the same spin state should be connected, while those with different spin states should be disconnected, seeking the ground state of the spin-glass model.

vii. **Random-Walk Community Detection:**

This method, based on random walks, aims to confine random walkers to denser regions of a network (communities). Starting from a non-clustered area, the random walker calculates distances between adjacent nodes, chooses adjacent communities to merge, and repeats the process.

The comparative study conducted in the paper utilizes Python's igraph library and considers modularity and execution time as evaluation parameters for the community detection algorithms. The organization of the rest of the paper is outlined, with subsequent sections delving into detailed descriptions of the algorithms, modularity parameters, datasets used for analysis, experimental setups, results analysis, and concluding remarks. The comprehensive overview provided serves as a valuable resource for researchers and practitioners in the field of community detection in complex networks.

c. SECTION III OF PAPER: EVALUATION METRICS

The paper utilizes modularity and execution time as evaluation metrics for community detection algorithms. Modularity, a key metric, measures the quality of partitions by assessing the difference between the fraction of edges within a community and that in a random network. The fraction of edges inside a community is calculated by summing the product of the adjacency matrix elements and the Kronecker delta function, divided by the total number of edges in the network. This approach quantifies the goodness of partitions, offering insights into the effectiveness of community detection algorithms in capturing network structures.

d. SECTION IV OF PAPER: DATASETS AND EXPERIMENTAL SETUP

The paper employs two types of datasets, medium and large, to evaluate the performance of community detection methods. Medium datasets, including karate, dolphin, polbooks, netscience, facebook, powergrid, hiEnCo, and Cond-2003, are utilized for comparing and selecting the best-performing community detection methods. The large datasets are sourced from Stanford datasets and include YouTube, Amazon, and DBLP. The datasets cover diverse network structures, such as social networks, co-authorship networks, and e-print archives.

The medium datasets encompass networks representing a karate club, dolphin associations, political book co-purchases, co-authorship in network theory, anonymous Facebook connections, power grid topology, high-energy theory co-authorship, and condensed matter co-authorship between 1995 and 2003. The large datasets involve YouTube user groups, Amazon product co-purchases, and a co-citation network from DBLP.

For experimentation, the igraph Python library is employed for community detection algorithms, executed on a system with a Core i7 processor and 4GB RAM. This experimental setup ensures consistency and allows for a systematic comparison of algorithmic performance across diverse datasets, enabling a comprehensive evaluation of community detection methods.

e. SECTION V & SECTION VI OF PAPER: RESULTS AND CONCLUSIONS:

In the analysis conducted, the first phase revealed that Louvain, Newman2006, Label Propagation, and Fast Greedy algorithms outperformed others in terms of modularity and execution time. Label Propagation, while efficient in community detection, exhibited poor performance in terms of execution time on the Cond-2003 dataset. The Spin-Glass community detection method demonstrated similar modularity performance to the aforementioned algorithms but with significantly higher running times, and it struggled with datasets requiring fully connected graphs.

In the case of complex networks, Louvain surpassed Newman2006 and Fast Greedy algorithms, providing a balance of high modularity and faster execution. Newman2006 exhibited low modularity, and while Fast Greedy produced higher modularity, its execution time was considerably longer.

The conclusion drawn from the analysis is that the Louvain community detection algorithm outperforms other methods in terms of efficiency and modularity, particularly on complex networks. However, it acknowledges that methods like Spin-Glass, Fast Greedy, and Infomap, while working approximately as well as Louvain, are computationally expensive. The first phase of the analysis enabled the selection of computationally efficient and high modular structure-producing community detection methods (Newman2006, Louvain, Fast Greedy) for further investigation. Overall, Louvain emerged as the most effective community detection algorithm across both phases of the study.

2.2 INTEGRATED SUMMARY OF LITERATURE STUDIED

Here is an integrated summary of all the research papers studied:

They have a basic discussion of community detection and its various algorithms:

1. Paper 1: This paper discusses the importance of social networks and the challenges of finding communities within them. The authors propose a method for community detection using the Girvan-Newman algorithm. They also discuss the results of their experiments.
2. Paper 2: This paper proposes a method for detecting communities in networks called the Louvain Method. The authors show that their method is more accurate than other methods.
3. Paper 3: This paper proposes an approach for community detection named label propagation algorithm. The authors prove that their algorithm converges to a stable labelling. They also show that their algorithm is efficient and stable.
4. Paper 4: This paper discusses what modularity is and why it is important. It also details two methods: Q and D. The article then shows how these two methods can be used to find communities in networks. However, it is important to note that these methods do not always work.
5. Paper 5 : This paper discussed a comparative analysis of various algorithms of community detection.

KEY FINDINGS:

1. Community detection is an important task in social network analysis.
2. There are a number of different methods for community detection.
3. The best method for a particular network will depend on the characteristics of the network.
4. Modularity is a common measure of the quality of a community detection algorithm.

CHAPTER 3 – Requirement Analysis and Solution Approach

3.1 Overall description of the project

The proposed project aims at providing a free and online platform to the masses to perform Community Detection freely and without any constraints. There is no need for any pre-requisite knowledge for operating the webapp, so a user without any technical skills and knowledge about it can also use it. The main page contains every detail about the project, how to use the tool, the direct access to the tool and case studies upon which all the provided algorithms are implemented (mentioned later in this section).

There are 2 case studies that are provided in the webapp:

1. Zachary Karate Club

   A very famous network graph that originated due to a brawl at a Karate Club at a University. True story, that this brawl made the 34-member club to divide into 2 groups, one who are loyal to their coach, Mr. Hi, and the other a new division. This became a playground for Mr. Wayne W. Zachary to find communities and their formation, forming a different fanbase for Community Detection and this is where the craze for this field started to rise. This network is used in the project to verify proper functioning of the algorithms and precise outputs as per expectations.
2. Highschool network

   It is a dataset attained from KONECT PROJECT. This dataset contains students as nodes and friendship as the edges between these nodes. This network is used as a custom case study for this tool.

The users then get to select a choice of 12 different network types that suite the best for the network provided by the user and the app automatically suggests the Community Detection Algorithms most suitable for that specific network. And if the user wants to use any algorithm for any kind of network, they get to do that too.

The Community Detection Algorithms provided are:-

1. Louvain’s Algorithm
2. Girvan-Newman Algorithm
3. Modularity Optimisation Algorithm
4. Label Propagation Algorithm

3.2 Requirement Analysis

The requirements for the proposed web apps are as follows:

1. Immersive and Easy to Use:
    a. The user interface should be intuitive and engaging, ensuring a seamless and enjoyable experience.
    b. Use modern design principles, clear navigation, and interactive elements to enhance user engagement.
2. Platform for Multiple Algorithms:
    a. The web app should support various algorithms.
    b. Users should be able to easily choose between different algorithms based on their needs and preferences.
3. Comparison of Network Graphs:
    a. Implement a feature that allows users to compare network graphs side by side.
    b. Provide visualization tools that displays different algorithm outputs.
4. Case Studies:
    a. Include a section or feature that presents case studies to illustrate real-world applications of the algorithms.
    b. These case studies can help users understand how the algorithms perform in different scenarios and industries.
5. User Guide:
    a. Develop a comprehensive guide or tutorial to assist users in navigating and utilizing the web app effectively.
    b. Include step-by-step instructions.
6. Customization Options:
    a. Allow users to customize the type of network and choose specific algorithms based on their requirements.
    b. Implement a user-friendly interface that enables easy selection and configuration of network types and algorithms.

3.3 Solution Approach

The project functions from start to end in the following manner:-

1. The Python program deploys and launches the main page (index.html) at localhost port http://127.0.0.1:5002
2. The HTML file follows the URL to the index.css (CSS file), index.js and particles.js (JS files) for the stylesheet and other functionalities. It also accesses the URL to google fonts API to load the desired font and Particles.js for its interactive network-based background used from CodePen, creating an immersive experience for the users.
3. Users choose the .csv file from their system directory, choose the type of network and the desired Community Detection Algorithm and then hit the Execute button to submit the form.
4. The data entered by the user in the HTML file is now transferred back to the Python program, thanks to Flask app route. The data is then calculated as per the below given procedure:
    a. The Python program has all the 4 algorithms (mentioned in the previous section) in the conditional statement.
    b. If the user selects Louvain's Algorithm, the HTML form returns value as 0 to the Python program, telling it to go for Louvain's Algorithm for the provided dataset.
    c. Same happens for the remaining 3 algorithms as well, value as 1 for Girvan-Newman Algorithm, 2 for Modularity Optimisation Algorithm and at last, 3 for Label Propagation Algorithm.
    d. The final graph is then output on a new webpage in that web browser, allowing the user to get multiple outputs side-by-side and go for comparison and analysis.
5. The main page is then refreshed, ready to be used again.

How each algorithm works

Louvain's Algorithm

Commencing with the initialization phase, each node is initially designated as a standalone community. This establishes a foundational state, setting the stage for the algorithm's subsequent actions.

Modularity, a pivotal metric in the algorithm's methodology, undergoes continuous evaluation. The algorithm iteratively refines the community structure by redistributing nodes among neighbouring communities to optimize modularity—a measure of the network's inherent structural organization.

The algorithm orchestrates strategic mergers of communities, consolidating nodes with strong internal connections. This process aims to transcend random chance, fostering communities with a notable level of internal cohesion.

Throughout its iterative refinement, the algorithm persists until convergence is achieved. Convergence signifies a stable community structure, as further iterations yield marginal gains in modularity.

The denouement of the algorithmic process yields a partition of nodes into communities. This output is meticulously crafted to maximize modularity, providing a comprehensive representation of the underlying social fabric within the network.

Girvan – Newman Algorithm

The algorithm's approach hinges on the concept of betweenness centrality—a measure of the node's influence on communication between other nodes. Edges with high betweenness centrality are systematically identified and removed, acting as a surgical excision of critical communication links.

As the algorithm continues its iterative process of edge removal, the network gradually fragments into distinct communities. The notion here is that by strategically severing key communication channels, the natural community structures are unveiled.

The iterative pruning of edges persists until a desired criterion is met, such as the optimal identification of communities or a significant increase in modularity.

In the denouement of this algorithmic journey, the output manifests as a delineation of nodes into communities, each reflective of the algorithm's discerning dissection of the network's communication architecture.

Modularity Optimisation Algorithm

Initiating the algorithmic process involves assigning nodes to communities, setting the initial stage for subsequent refinement.

At the heart of the algorithm lies the pursuit of modularity—an evaluative metric that gauges the quality of community assignments. The algorithm systematically iterates through potential community configurations, aiming to maximize the modularity score.

Through a series of adjustments to the community composition, nodes are strategically reassigned to enhance the overall modularity. This iterative optimization process is akin to a sculptor refining their work, each adjustment shaping the community structure with precision.

The algorithm perseveres through these iterations until a convergence criterion is met, indicating stability in the community assignments. Convergence typically occurs when further iterations yield minimal improvement in modularity.

The culmination of this algorithmic journey yields a refined partition of nodes into communities, each indicative of the algorithm's prowess in discerning the optimal community structure based on modularity.

Label Propagation Algorithm

Commencing with the algorithmic initiation, each node is assigned an initial label, marking the starting point for community assignment.

The algorithm's principle revolves around the propagation of labels through the network. Nodes iteratively adopt the most prevalent label among their neighbours, fostering a dynamic process akin to the spread of influence within a social network.

As labels propagate through successive iterations, nodes naturally gravitate towards communities with similar labels. This self-organizing mechanism captures the intrinsic community affiliations embedded in the network.

The algorithm continues this iterative propagation until a stable state is reached, where label assignments cease to change significantly. This stability indicates the formation of coherent communities within the network.

CHAPTER 4 – MODELLING AND IMPLEMENTATION DETAILS

4.1 DESIGN DIAGRAMS

4.1.1 USE CASE DIAGRAM

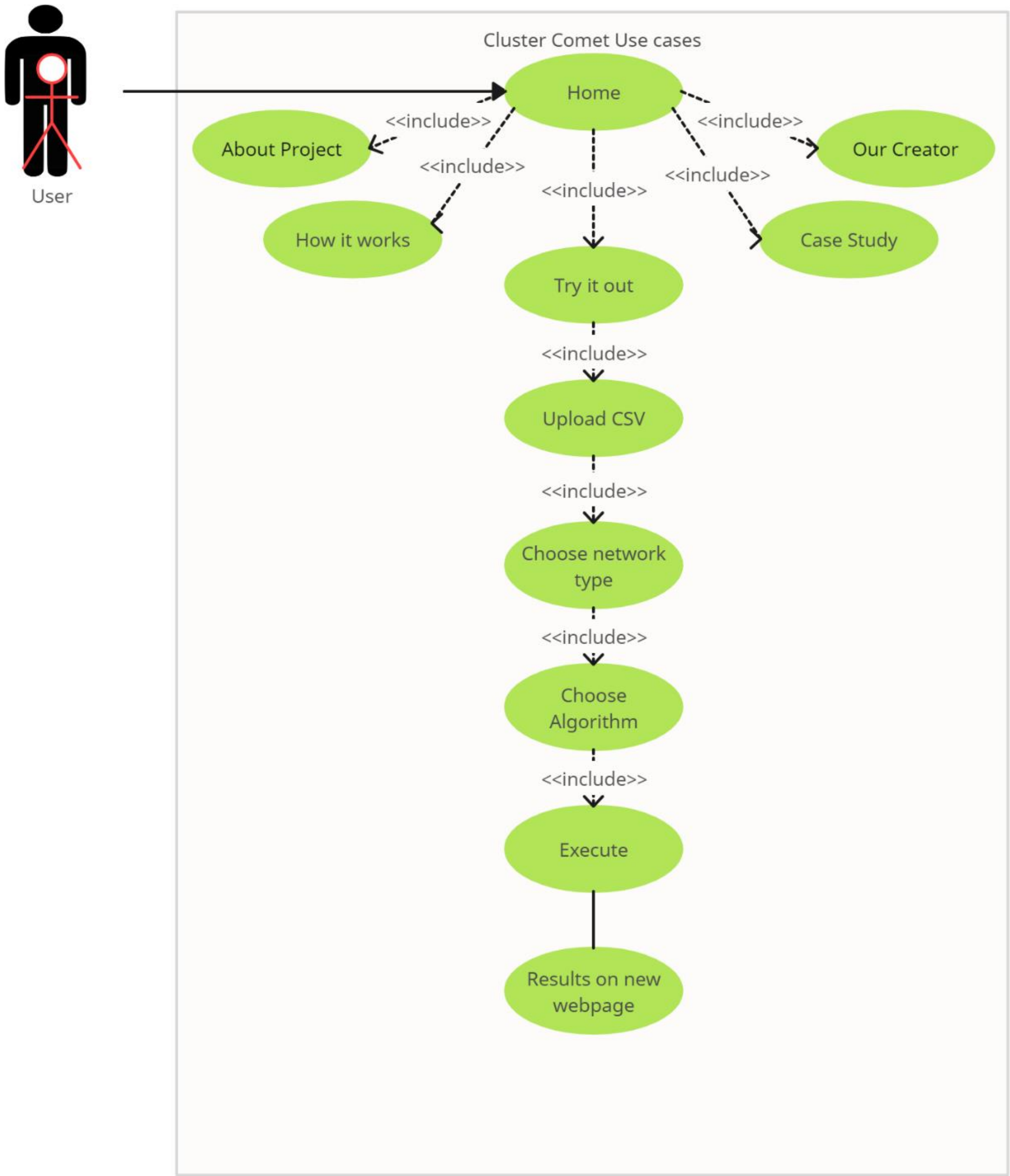


Figure 1

4.1.2 CONTROL FLOW DIAGRAM

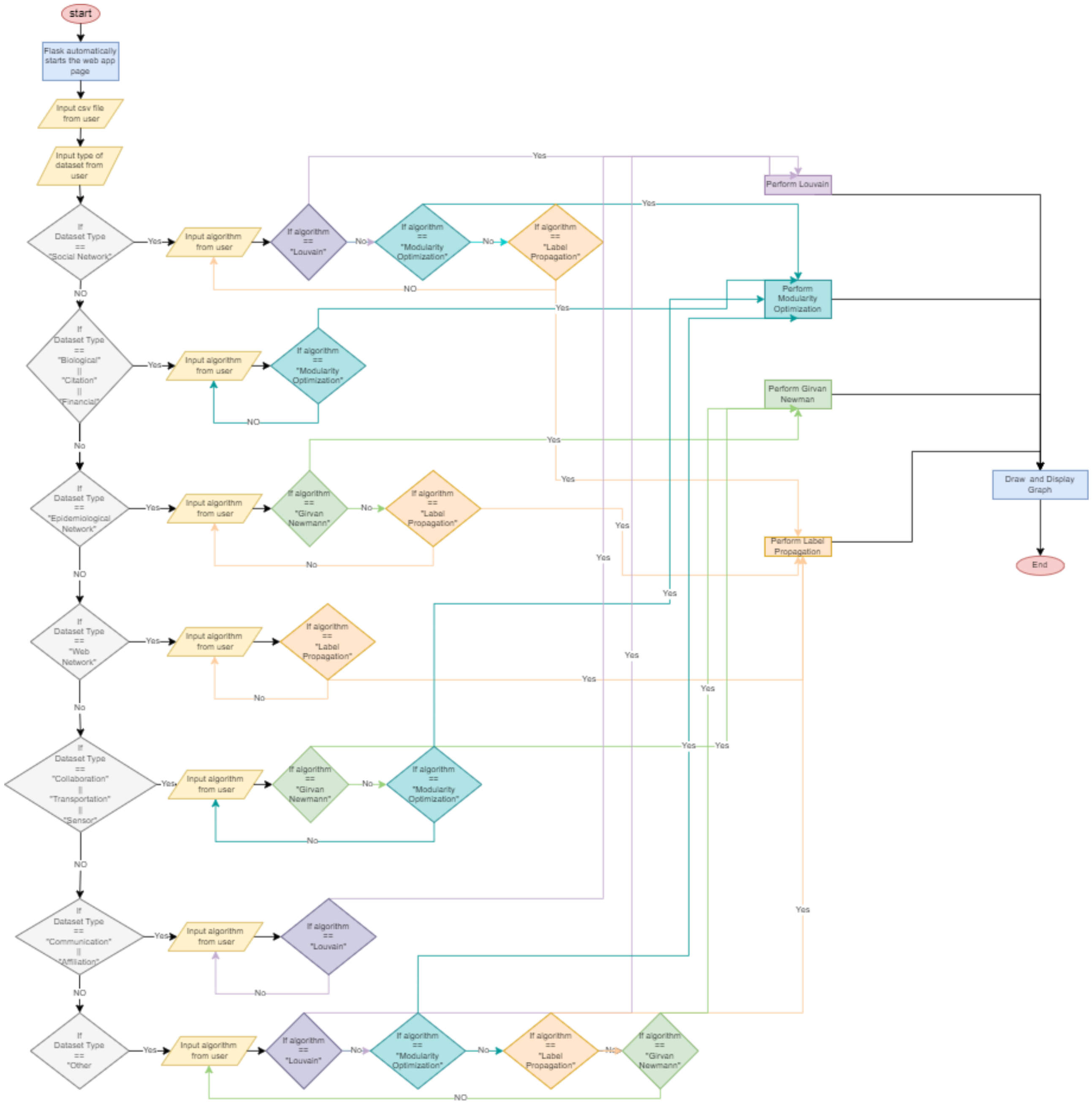


Figure 2

4.1.3 SEQUENCE DIAGRAM

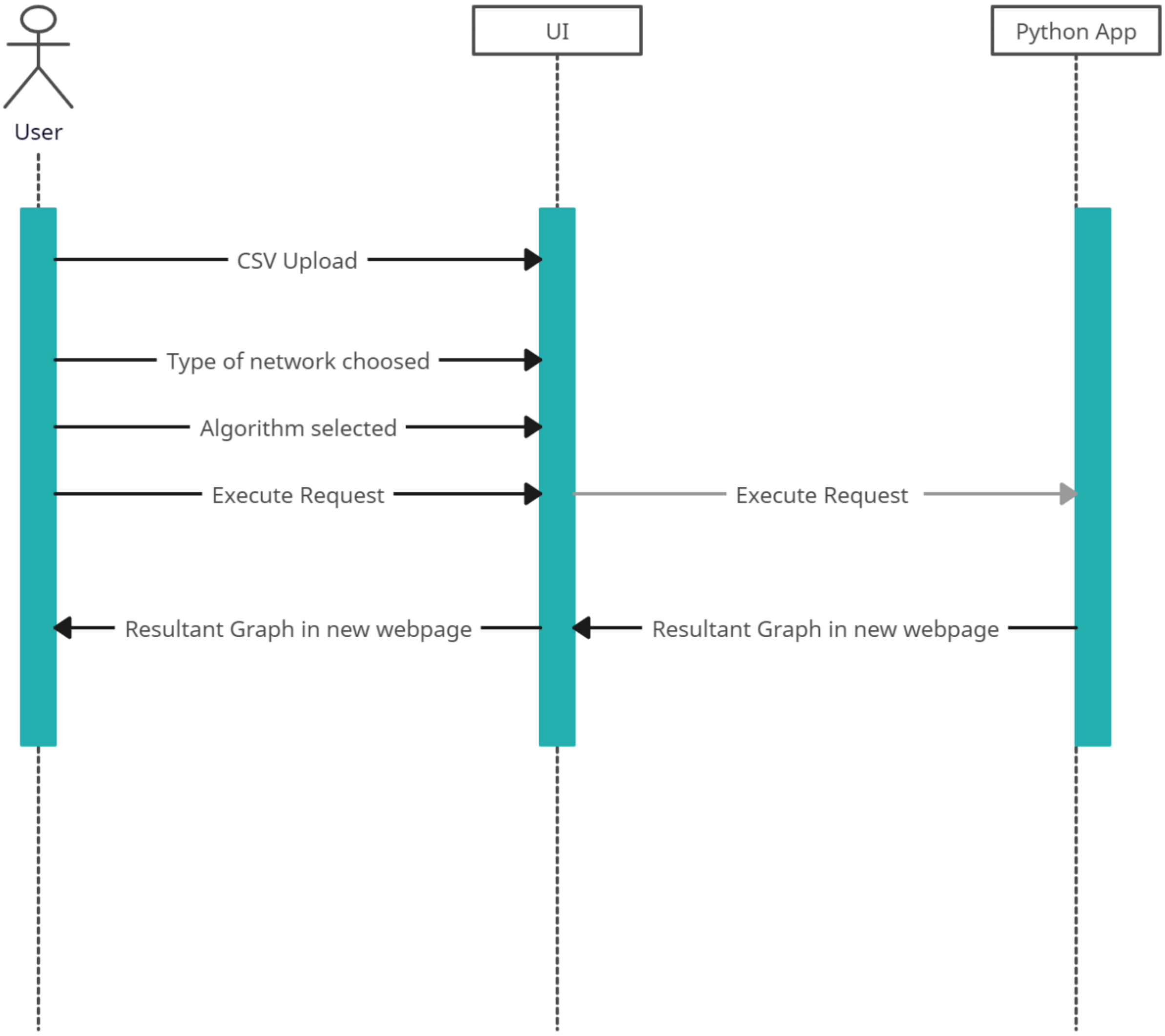


Figure 3

4.2 IMPLEMENTATION DETAILS AND ISSUES

1. **FRONTEND (USER INTERFACE):**
    - **a.** Developed using HTML, CSS, and JavaScript.
    - **b.** Provides information about the project.
    - **c.** Guides user throughout the project.
    - **d.** Handles user input, file uploads, algorithm selections and form submission.

2. **BACKEND (FLASK APPLICATION):**
    - **a.** Built using Python and Flask.
    - **b.** Manages file uploads, validates input, and triggers community detection processes.
    - **c.** Communicates with community detection algorithms and graph generation modules.

3. **COMMUNITY DETECTION MODULE:**
    - **a.** Contains the logic for executing community detection algorithms.
    - **b.** Utilizes libraries like Networkx for Community Detection and graph-related operations.

4. **GRAPH GENERATOR:**
    - **a.** Responsible for generating visual representations of the community structure.
    - **b.** Uses a plotting library, Plotly.

5. **WORKFLOW:**
    - **a.** User uploads a CSV file through the UI.
    - **b.** Flask app handles the file upload and validates the network type and algorithm selections.
    - **c.** When the user presses "Execute," the Flask app triggers the community detection process:
    - **d.** Executes the chosen algorithm on the uploaded data.
    - **e.** Generates the community structure and passes it to Plotly library.
    - **f.** It then creates a visual representation of the community structure.
    - **g.** The result is displayed on a separate webpage.

6. **POTENTIAL ISSUES:**
    a. **ALGORITHM COMPATIBILITY:**
        i. Different community detection algorithms may require different inbuilt libraries that may not be installed in the users' system.
    b. **SCALABILITY:**
        i. Handling large datasets might impact performance. Considerations should be made for scalability, especially when dealing with complex networks.
    c. **USER INPUT VALIDATION:**
        i. Ensuring valid user input and preventing malicious file uploads is essential to avoid security issues.

4.3 RISK ANALYSIS AND MITIGATION

1. **ALGORITHM PERFORMANCE:**
    a. **Risk:** Some algorithms may not perform well on certain types of networks.
    b. **Mitigation:** Perform extensive testing with various algorithms and network types to identify the most suitable ones. Provide users with guidance on algorithm selection.
2. **DATA SECURITY:**
    a. **Risk:** Malicious file uploads or security vulnerabilities.
    b. **Mitigation:** Implement robust input validation and file type verification.
3. **SCALABILITY CHALLENGES:**
    a. **Risk:** Performance degradation with large datasets.
    b. **Mitigation:** Optimize algorithms for efficiency, allow users to use only those algorithms that go well with large datasets, unless it is used as experimentation where users are free to choose any algorithm for dataset of any size.
4. **USER EXPERIENCE:**
    a. **Risk:** Users may find the interface confusing or encounter usability issues.
    b. **Mitigation:** Provide internal hyperlinks to directly jump to the sections in the page. Single page website for simplicity and performance. Full guide of how to use the webapp.

CHAPTER 5 – TESTING

5.1 TESTING PLAN

To ensure that the Cluster Comet webapp is functioning as intended, a variety of testing techniques were employed. The testing procedure were:-

1. The 4 algorithms (mentioned earlier) were made in separate .py files to check working and debugging of each individual algorithm. This made the algorithm building very much time efficient.
2. A dummy and bland webpage was created with just the form where the user will input the csv file and choose network type and algorithm.
3. Another .py file (the current app.py) was made that was responsible for checking connection between the webpage and the program that was supposed to be in the file with the help of Flask.
4. After successful integration and debugging of webpage and app.py, it was checked if all the values entered in the webpage was piped to app.py.
5. The 4 algorithms were now included in app.py for the final integration, and then rigorous testing and debugging was done.
6. The UI/UX designing was done on Figma and then a refined and immersive webpage was made using HTML, CSS and JS which was used in place of its dummy counterpart.
7. Rigorous testing was done to check integration status among all of the components.

5.2 LIST OF ALL TEST CASES

1. **exp.csv**

   An experimental dataset manually created to check and debug the algorithms. The data in it does not have any actual relevance.

2. **amazon.csv**

   It is a dataset that contains data regarding products and co-purchases. Basically, the nodes are products and the edge between the nodes are co-purchases, i.e., recommendations by amazon regarding product purchase when a certain product is purchased.

3. **chess.csv**

   It is a dataset that contains data regarding players and games. Basically, the nodes are players, and the edge between the nodes are the games the players participated in.

4. **facebook.csv**

   It is a dataset that contains data regarding users and posts. Basically, the nodes are users and the edges between the nodes are posts they interacted with.

5. **highschool.csv**

   It is a dataset that contains data regarding students and friendship. Basically, the nodes are students and the edges between the nodes are the friendship among them.

6. **infection.csv**

   It is a dataset that contains data regarding people and infection. Basically, the nodes are people and the edges between the nodes are infection spread among them.

7. **karateclub.csv**

   It is a very famous dataset. It is a dataset that contains data regarding people and infection. Basically, the nodes are people and the edges between the nodes are friendship among them.

5.3 ERROR AND EXCEPTION HANDLING

1. **MANDATORY CSV INPUT WITH EXCEPTION HANDLING:**
    a. **Frontend Check:** The frontend of the application enforces the requirement of uploading a CSV file from the user's directory. This prevents incorrect file types from being submitted.
    b. **Python Code Exception Handling:** Within the backend Python code, there's additional exception handling to ensure that if, for any reason, the CSV file is not selected or the file format is incorrect, appropriate error messages or prompts are displayed to guide the user.

2. **RECOMMENDATION OF APPROPRIATE ALGORITHM:**
    a. **User Guidance:** The system provides users with recommendations for the most suitable algorithm based on their specified type of network. However, if users still choose an incorrect network type or algorithm, the system processes the input but alerts users that the output might not be accurate or as expected due to mismatched selections.

3. **REQUIREMENT FOR SPECIFIC CSV FORMAT:**
    a. **Format Compatibility:** Users are required to input a CSV file that adheres to a specified format. This ensures that the data provided is compatible with the algorithms available for analysis. This adherence to a defined format streamlines the processing and avoids issues related to data compatibility.

4. **ERROR NOTIFICATION FOR CSV FORMAT MISTAKES:**
    a. **User-Friendly Error Handling:** In case of any mistakes or discrepancies in the CSV file format, the system generates user-friendly error notifications. This helps users identify and rectify issues promptly, ensuring that the data is correctly formatted before processing. By providing clear error messages, users can easily address any formatting mistakes.

5.4 LIMITATIONS OF THE SOLUTION

1. **Algorithm Availability:** When starting a project, the availability of algorithms might be limited. This could be due to various reasons like time constraints, resources, or the complexity of certain algorithms.
2. **Hosting on Cloud Services:** Sometimes, projects are initially developed or tested locally before being hosted on a cloud service platform. Reasons for this might include cost considerations, security concerns, or the need for fine-tuning before deployment.
3. **Specific CSV Format:** Datasets often need to adhere to specific formats for smooth processing. This requirement might be due to the data ingestion process, compatibility with existing tools or libraries, or to ensure consistency across different datasets.
4. **Dataset Size and Algorithm Complexity:** Large dataset sizes can pose challenges for complex algorithms like the Girvan-Newman Algorithm, which might have high computational requirements. Balancing algorithmic efficiency and the dataset's scale is crucial for timely and resource-efficient analysis.
5. **User Selection of Network Type:** Determining the network type might require domain-specific knowledge or context. The project is still not capable of figuring out the type of network by itself.
6. **Customization and Weighted Graphs:** While not considered currently, these aspects might become essential in the future. Customization often enhances the adaptability of algorithms to diverse use cases. Similarly, dealing with weighted graphs might require specialized handling and could significantly impact algorithmic outcomes.

CHAPTER 6 – FINDINGS, CONCLUSION AND FUTURE WORK

6.1 FINDINGS

1) **ALGORITHM RECOMMENDATION SYSTEM:**

   a) The algorithm recommendation system based on network type is a valuable feature. It provides users with guidance on suitable algorithms, making the application user-friendly and accessible for users with varying levels of expertise in community detection.

2) **DYNAMIC ALGORITHM SUGGESTIONS:**

   a) Dynamically suggesting algorithms based on the uploaded dataset's network type enhances user experience. It streamlines the process for users who might not be familiar with the characteristics of different network types.

3) **ALGORITHM SUITABILITY:**

   a) The suggested algorithms align well with the characteristics of specific network types. For example, recommending Louvain for social networks and modularity optimization for biological/citation or financial networks demonstrates an understanding of the strengths of each algorithm.

4) **EXECUTION WORKFLOW:**

   a) The workflow of uploading a CSV, selecting the network type, receiving algorithm suggestions, and executing the chosen algorithm is clear and intuitive. Users are guided through the process, reducing potential confusion.

6.2 CONCLUSION:

1) **EFFECTIVE COMMUNITY DETECTION:**
   a) The implemented system enables users to perform effective community detection by providing algorithm recommendations tailored to the characteristics of the uploaded network. This ensures that users can make informed choices and obtain meaningful insights from their data.

2) **USER-FRIENDLY INTERFACE:**
   a) The user interface promotes a positive user experience by guiding users through each step of the process. The dynamic algorithm suggestions simplify the decision-making process for users and contribute to the overall usability of the web application.

3) **ALGORITHM DIVERSITY:**
   a) The diverse set of suggested algorithms caters to various types of networks, acknowledging that different community detection methods may be more suitable for specific domains. This versatility enhances the application's utility across different fields.

6.3 FUTURE WORK:

1) **ALGORITHM PERFORMANCE METRICS:**
   a) Integrate performance metrics to evaluate the effectiveness of the executed algorithms. Metrics such as modularity, community size, and computational time can be displayed to users, providing additional insights into the quality of the detected communities.

2) **USER CUSTOMIZATION:**
   a) Allow users to customize algorithm selections based on their preferences or specific requirements. This could involve providing advanced options for algorithm parameters, allowing more experienced users greater control over the community detection process.

3) **VISUALIZATION ENHANCEMENTS:**
   a) Enhance the visualization capabilities to provide more interactive and informative representations of community structures. Features such as node colouring, size adjustments, and network metrics visualization can contribute to a richer user experience.

4) **SUPPORT FOR ADDITIONAL NETWORK TYPES:**
   a) Expand the system's capability to recognize and suggest algorithms for additional network types as the field of community detection evolves. This could involve incorporating machine learning techniques to adapt to emerging trends in network analysis.

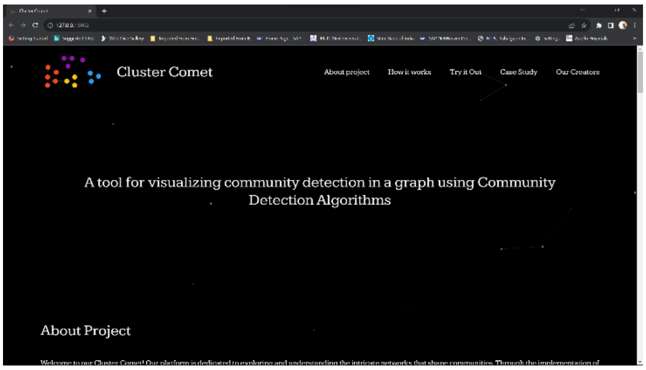
5) **COMMUNITY EVALUATION TOOLS:**
   a) Develop tools for users to evaluate the quality of detected communities, allowing them to assess the meaningfulness of the results. This could involve community overlap analysis, stability testing, and comparison with ground truth datasets where available.

By addressing these areas in future iterations, the community detection web application can continue to evolve, providing users with advanced features, customization options, and a deeper understanding of their network data.

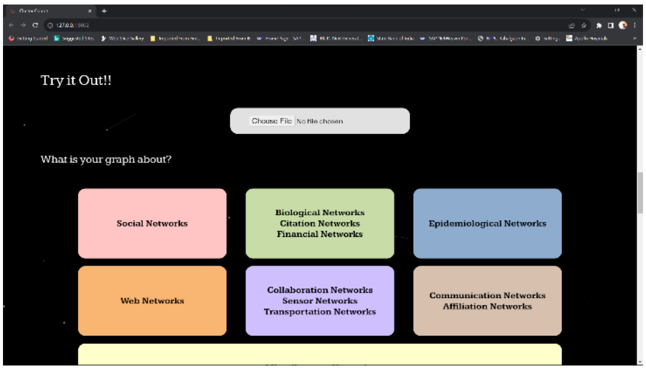
REFERENCES

1. Sathiyakumari, K. and Vijaya, M.S. (2016) Community detection based on Girvan Newman algorithm and link analysis ..., ResearchGate. Available at: https://www.researchgate.net/publication/310736499_Community_Detection_Based_on_Girvan_Newman_Algorithm_and_Link_Analysis_of_Social_Media (Accessed: 29 November 2023).
2. Meo, P.D. et al. (2011) Generalized Louvain method for community detection in large networks, ResearchGate. Available at: https://www.researchgate.net/publication/51929918_Generalized_Louvain_Method_for_Community_Detection_in_Large_Networks (Accessed: 29 November 2023).
3. Cordasco, G. and Gargano, L. (2012) Label propagation algorithm: A semi-synchronous approach - ResearchGate, ResearchGate. Available at: https://www.researchgate.net/publication/264817879_Label_propagation_algorithm_A_semi-synchronous_approach (Accessed: 29 November 2023).
4. Zhang, X. et al. (2009) Modularity optimization in community detection of complex networks, ResearchGate. Available at: https://www.researchgate.net/publication/223956042_Modularity_optimization_in_community_detection_of_complex_networks (Accessed: 29 November 2023).
5. Chejara, P. and Godfrey, W.W. (2017) Comparative analysis of community detection algorithms | IEEE ..., IEEE Xplore. Available at: https://ieeexplore.ieee.org/document/8340627/ (Accessed: 29 November 2023).
6. Mohammad, M. (2018) 'community detection in social networks through Girvan Newman algorithm', Medium. Available at: https://medium.com/@96mehakmohammad/community-detection-in-social-networks-through-girvan-newman-algorithm-78f303912907 (Accessed: 29 November 2023).
7. Yan, M. and Guoqiang, C. (2022) Label propagation community detection algorithm based on density peak optimization, Wireless Communications and Mobile Computing. Available at: https://www.hindawi.com/journals/wcmc/2022/6523363/ (Accessed: 29 November 2023).
8. Zhang, J. et al. (2021) An improved Louvain algorithm for community detection, Mathematical Problems in Engineering. Available at: https://www.hindawi.com/journals/mpe/2021/1485592/ (Accessed: 29 November 2023).

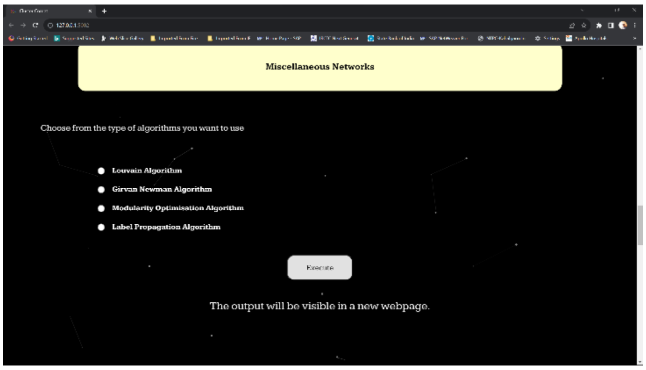
APPENDIX



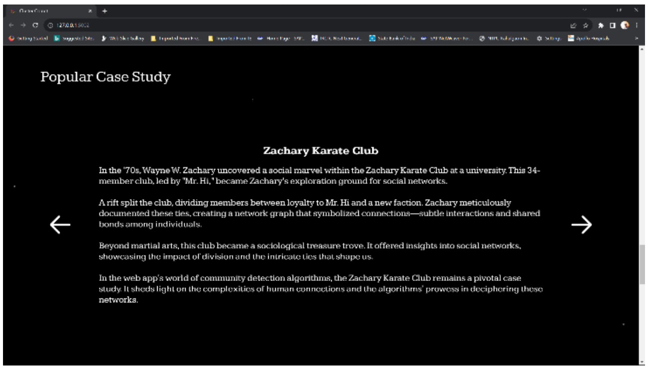
Cluster Comet
About project
How it works
Try it Out
Case Study
Our Creators
A tool for visualizing community detection in a graph using Community Detection Algorithms
About Project



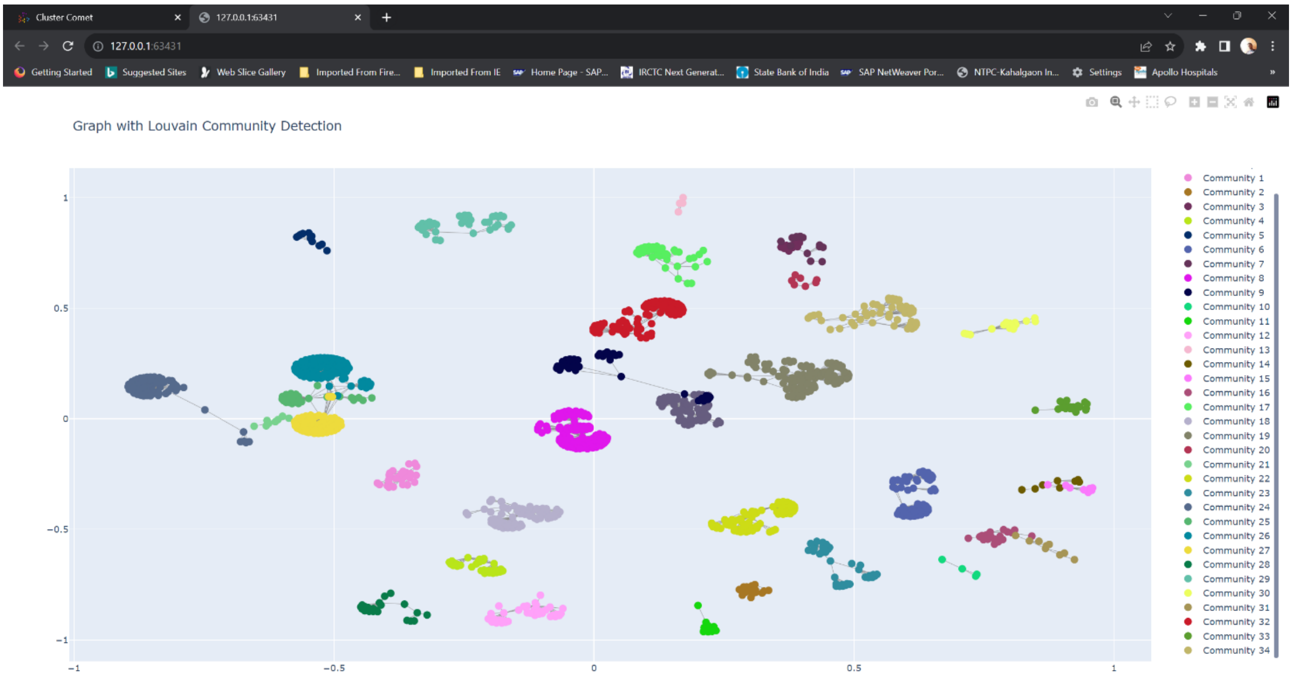
Try it Out!!
Choose File No file chosen
What is your graph about?
Social Networks
Biological Networks
Citation Networks
Financial Networks
Epidemiological Networks
Web Networks
Collaboration Networks
Sensor Networks
Transportation Networks
Communication Networks
Affiliation Networks



Miscellaneous Networks
Choose from the type of algorithms you want to use
Louvain Algorithm
Girvan Newman Algorithm
Modularity Optimisation Algorithm
Label Propagation Algorithm
Execute
The output will be visible in a new webpage.



Popular Case Study
Zachary Karate Club
In the '70s, Wayne W. Zachary uncovered a social marvel within the Zachary Karate Club at a university. This 34-member club, led by "Mr. Hi," became Zachary's exploration ground for social networks.
A rift split the club, dividing members between loyalty to Mr. Hi and a new faction. Zachary meticulously documented these ties, creating a network graph that symbolized connections—subtle interactions and shared bonds among individuals.
Beyond martial arts, this club became a sociological treasure trove. It offered insights into social networks, showcasing the impact of division and the intricate ties that shape us.
In the web app's world of community detection algorithms, the Zachary Karate Club remains a pivotal case study. It sheds light on the complexities of human connections and the algorithms' prowess in deciphering these networks.



Cluster Comet
127.0.0.1:63431
Graph with Louvain Community Detection
Community 1
Community 2
Community 3
Community 4
Community 5
Community 6
Community 7
Community 8
Community 9
Community 10
Community 11
Community 12
Community 13
Community 14
Community 15
Community 16
Community 17
Community 18
Community 19
Community 20
Community 21
Community 22
Community 23
Community 24
Community 25
Community 26
Community 27
Community 28
Community 29
Community 30
Community 31
Community 32
Community 33
Community 34