Jaypee Institute of Information Technology, Noida

Semester 7 (2025–2026)

Course: Social Network Analysis

Course Code: 15B1NCI732

Mapping Friendships: Social Network Analysis of Facebook Ego Networks

Submitted To:

Dr. Bhawna Saxena

Submitted By:

Name	Batch	Enrollment No.
VIYOM SHUKLA	B12	22803030
ABHIJEET KUMAR	B12	22803029
N. DEVASAIKUMAR	B11	22103337

Contents

1 Introduction

Social networks play a vital role in understanding how individuals connect, interact, and influence one another. In recent years, the analysis of online social platforms has become a major area of research in data science, sociology, computer science, and network theory. Among these platforms, Facebook provides one of the richest sources of relationship data, where users form connections known as friendships. Analyzing such networks helps reveal important structural patterns, influential users, community formations, and the overall interaction dynamics within a social ecosystem.

This project, titled **Mapping Friendships: Social Network Analysis of Facebook Ego Networks,** focuses on examining the Facebook Combined Ego Network dataset collected from the Stanford SNAP repository. An **ego network** consists of a central user (the ego) along with their immediate friends (the alters) and the connections among those friends. By studying ego networks, we gain insights into individual-centric community structures, local connectivity patterns, communication flows, and influence spread in social systems.

The goal of this project is to apply key **Social Network Analysis (SNA)** techniques using Python to extract meaningful information from the Facebook dataset. Metrics such as degree distribution, clustering coefficients, connected components, and centrality measures (including degree, betweenness, closeness, and eigenvector centrality) allow us to quantify user importance and understand the role each node plays in the network. Various graph visualizations and interactive network diagrams further help in revealing hidden structural patterns that cannot be observed through raw data alone.

Through this analysis, the project demonstrates how graph theory and network science can be used to uncover critical insights from real-world social networks. This analysis not only deepens our understanding of online friendship patterns but also highlights how data-driven approaches can be used to study human social behavior at scale. The outcomes of this project provide a foundation for further research in areas such as community detection, recommendation systems, viral marketing, and network robustness.

2 System Overview

The system developed in this project is designed to analyze and interpret the structural characteristics of **Facebook Ego Networks** using concepts from Social Network Analysis (SNA) and graph theory. The project processes the **Facebook Combined Ego Network dataset**, transforms the raw data into graph structures, and applies analytical methods to uncover patterns within the network. The system operates in several stages, from data ingestion to visualization, resulting in meaningful insights about user relationships and network behavior.

At the core of this system is **NetworkX**, a powerful Python library for graph creation and analysis. The Facebook dataset, consisting of edges representing friendships between users, is first loaded and converted into an undirected graph. This graph represents users as nodes and friendships as edges. From this, various structural properties are computed, such as the number of nodes, edges, connected components, and degree distributions.

The system further extracts more sophisticated insights by calculating **centrality measures**, which help identify influential or well-connected individuals within the network. These measures include degree centrality, betweenness centrality, closeness centrality, and eigenvector centrality. Each of these metrics captures a different aspect of node importance, such as connectivity, influence over shortest paths, and proximity to other nodes.

To support result interpretation, the system incorporates both static and dynamic visualization components. Using libraries like **Matplotlib** and **PyVis**, the system produces visual representations of the network structure. Static plots highlight degree distributions and connectivity, while interactive visualizations allow users to explore the network in a dynamic, zoomable, and user-friendly interface.

The system is modular and scalable, enabling additional features such as community detection, approximate centrality computation for large graphs, and graph sampling to handle high node volumes. This flexible architecture ensures efficient processing and easy extension for future enhancements in network modeling or analysis.

Overall, the system provides a comprehensive pipelinefrom data loading to advanced visual analysisallowing users to conduct an in-depth study of the structure and behavior of social networks using real-world Facebook friendship data.

3 Technologies Used

This project utilizes a combination of Python libraries, data analysis tools, and visualization frameworks to perform Social Network Analysis on the Facebook Ego Network dataset. Each technology plays a key role in data preprocessing, graph construction, metric computation, and visual interpretation.

1. **Python 3.x** Python serves as the core programming language for implementing graph analysis, data manipulation, and visualization. Its extensive library ecosystem makes it ideal for network analysis and scientific computing.

2. **NetworkX** NetworkX is the primary library used for:

   - Creating and managing graph data structures
   - Computing centrality measures (degree, betweenness, closeness, eigenvector)
   - Extracting connected components and clustering coefficients
   - Performing graph sampling and subgraph analysis

   It provides efficient algorithms that make network computation intuitive and scalable.

3. **NumPy** NumPy is used for:

   - Numerical computations
   - Handling matrix-like structures
   - Supporting intermediate operations in centrality and distribution analysis

   Its optimized array operations improve performance for large-scale graph data.

4. **Matplotlib** Matplotlib is utilized for generating:

   - Degree distribution plots
   - Network diagrams
   - Static visualizations of sampled graphs

   It provides high-quality charts essential for report-ready insights.

5. **PyVis** PyVis is used to build:

   - Fully interactive network graphs
   - Visual interfaces where nodes can be moved, zoomed, and explored dynamically

   This enhances user understanding of structural patterns and node relationships.

6. **Jupyter Notebook** The entire project was developed and executed in a Jupyter Notebook environment, which enables:

- Step-by-step execution
- Inline visualizations
- Easy documentation alongside code
- Reproducible research structure

7. **SNAP Dataset** The Facebook Combined Ego Network dataset from the SNAP repository forms the foundation of the project. It provides:
    - Real-world friendship edges
    - Large and complex social graph structure
    - A suitable dataset for centrality, connectivity, and clustering analysis

4 WorkFLow

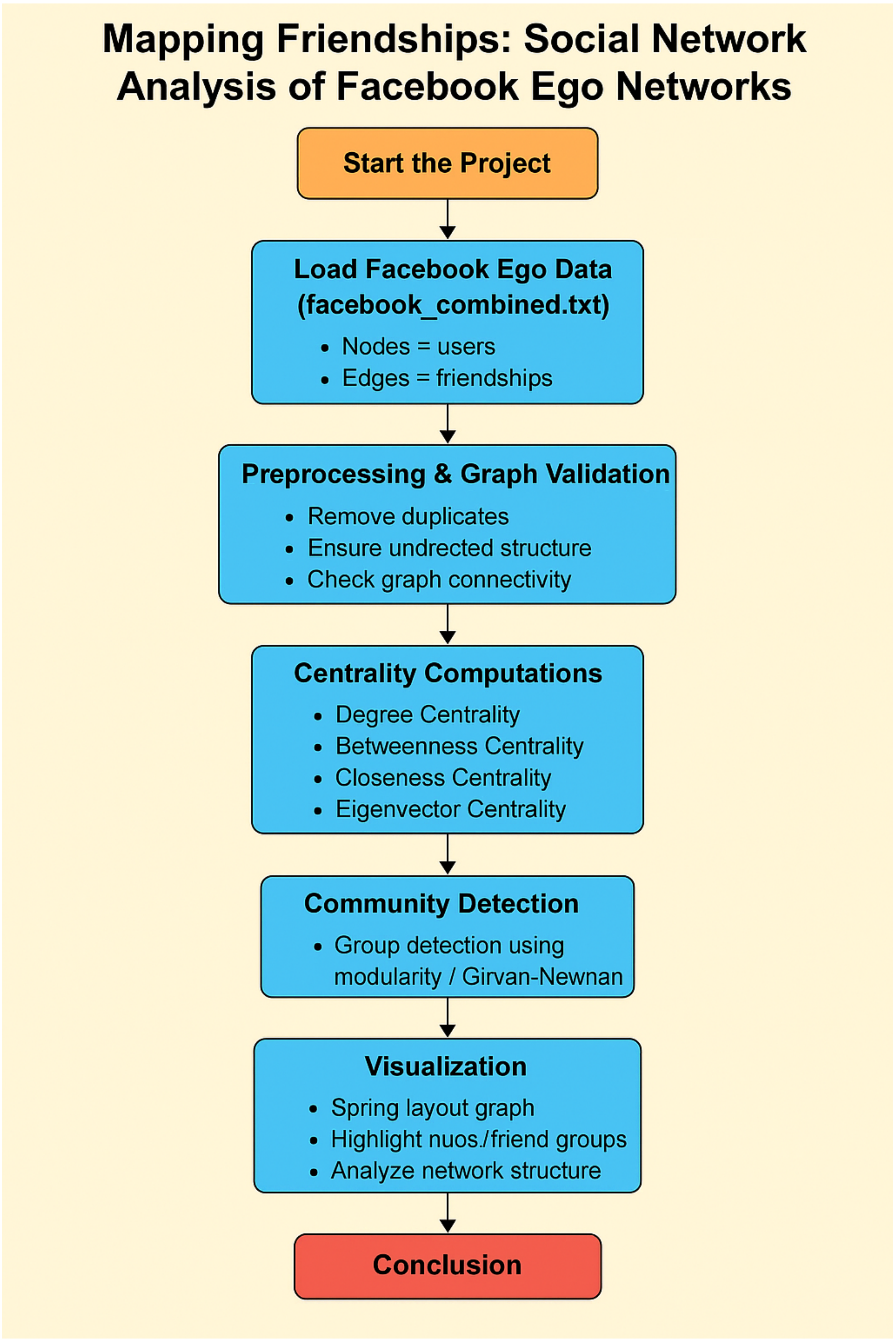
Mapping Friendships: Social Network Analysis of Facebook Ego Networks
Start the Project
Load Facebook Ego Data (facebook_combined.txt)
• Nodes = users
• Edges = friendships
Preprocessing & Graph Validation
• Remove duplicates
• Ensure undrected structure
• Check graph connectivity
Centrality Computations
• Degree Centrality
• Betweenness Centrality
• Closeness Centrality
• Eigenvector Centrality
Community Detection
• Group detection using modularity / Girvan-Newnan
Visualization
• Spring layout graph
• Highlight nuos./friend groups
• Analyze network structure
Conclusion

Figure 1: Work FLow of the Project

5 Implementation

The implementation of this project follows a systematic pipeline beginning with dataset loading and ending with advanced graph visualizations and centrality analysis. All tasks were executed in a Jupyter Notebook environment using Python. The following subsections summarize the core implementation steps.

5.1 Dataset Loading and Preprocessing

The project uses the Facebook Combined Ego Network dataset provided by SNAP. The dataset is an edge list where each line represents a friendship between two users.

```
import networkx as nx

# Load the Facebook ego network dataset
G = nx.read_edgelist("dataset/facebook_combined.txt", nodetype=int)
```

This step converts the dataset into an undirected graph where nodes represent users and edges represent friendships. Basic statistics are computed to verify successful loading:

```
print("Nodes:", G.number_of_nodes())
print("Edges:", G.number_of_edges())
```

5.2 Graph Cleaning and Connected Components

Social networks often contain multiple weakly connected regions. The graph is therefore reduced to its largest connected component (LCC):

```
largest_cc = max(nx.connected_components(G), key=len)
G_cc = G.subgraph(largest_cc).copy()
```

Working with the LCC ensures meaningful centrality computation and visualization.

5.3 Degree Distribution Analysis

Degree measures how many friends a user has. The system computes and visualizes the degree distribution:

```
degrees = [deg for node, deg in G_cc.degree()]
```

A histogram is generated using Matplotlib:

```
plt.hist(degrees, bins=50)
plt.title('Degree Distribution')
plt.xlabel('Degree')
plt.ylabel('Frequency')
```

This reveals that most users have few connections, while some behave as hubs.

5.4 Centrality Measures

Several centrality metrics are calculated to identify influential users.

Degree Centrality

```
deg_cent = nx.degree_centrality(G_cc)
```

Betweenness Centrality

For large graphs, approximate betweenness is used:

```
betw_cent = nx.betweenness_centrality(G_cc, k=500,normalized=True)
```

Closeness Centrality

```
clos_cent = nx.closeness_centrality(G_cc)
```

Eigenvector Centrality

```
eig_cent = nx.eigenvector_centrality(G_cc, max_iter=500)
```

5.5 Ranking Influential Users

The top 10 users are extracted:

```
top_10_betweenness = sorted(betw_cent,key=betw_cent.get,reverse=True)[:10]
```

These rankings are used for further analysis.

5.6 Network Visualization

Static Visualization (Matplotlib)

```
pos = nx.spring_layout(G_cc)
nx.draw(G_cc, pos, node_size=10, edge_color='gray')
```

Interactive Visualization (PyVis)

```
from pyvis.network import Network
net = Network(height="750px", width="100%", notebook=False)
net.from_nx(G_cc)
net.show("facebook_ego_interactive.html")
```

The generated HTML file supports zooming, panning, and node inspection.

5.7 Subgraph Sampling

To improve clarity in visualization, a random subset of nodes is extracted:

```
import random
nodes_sample = random.sample(list(G_cc.nodes()), 500)
H = G_cc.subgraph(nodes_sample)
```

5.8 Summary of Implementation

The system integrates graph creation, statistical analysis, centrality computation, static and interactive visualizations, and subgraph extraction. These combined steps create a robust Social Network Analysis workflow using real Facebook friendship data.

6 Key Features

The project incorporates several powerful analytical and visualization capabilities that make it an effective solution for studying large-scale social networks. The following key features highlight the strengths and uniqueness of the system:

1. **Real-World Social Network Analysis**

   The system analyzes the Facebook Combined Ego Network dataset from SNAP, providing practical insights into user connections, friendships, and influence patterns. This ensures the results reflect real-world social behavior rather than synthetic examples.

2. **Graph Construction and Preprocessing**

   - Efficient conversion of raw edge data into an undirected graph.
   - Automatic extraction of the Largest Connected Component (LCC).
   - Ability to handle thousands of nodes and edges smoothly.

3. **Degree Distribution Insights**

   The system computes and visualizes:

   - Degree vs. frequency histogram
   - Identification of high-degree hubs
   - Understanding of user popularity and connectivity levels

4. **Centrality Metrics for Influential Node Detection**

   Several centrality measures are implemented to identify key users:

   - **Degree Centrality** detects users with the highest number of direct connections.
   - **Betweenness Centrality** (approximate) identifies bridge users connecting communities.
   - **Closeness Centrality** measures user accessibility across the network.
   - **Eigenvector Centrality** identifies users connected to other influential nodes.

5. **Interactive Network Visualization**

   Using PyVis, the project provides:

   - Zooming and panning
   - Node dragging interactions
   - Hover-based data inspection
   - Dynamic exploration of structural patterns

6. **Static Graph Visualizations**

   Static network visualizations generated using Matplotlib include:

   - Force-directed layouts
   - Degree-based color or size variations

7. **Subgraph Sampling for Clarity**

   To handle large networks effectively:

   - Random sampling of 300–500 nodes
   - Clean, lightweight mini-network representations
   - Reduced visual clutter and improved clarity

8. **Modular and Extendable Design**

   The structure supports easy expansion into:

   - Community detection
   - Clustering algorithms
   - Influence propagation modeling
   - Graph embeddings
   - Dynamic and temporal networks

9. **Performance Optimization**

   The system uses multiple techniques to improve performance:

   - Approximate betweenness centrality computation
   - Subgraph-based visualization for efficiency
   - Lazy loading and memory-efficient graph handling

10. **Reproducible Notebook Workflow**

    The entire project is implemented in Jupyter Notebook, enabling:

    - Step-by-step execution
    - Inline visualization
    - Easy experimentation and modification
    - Fully reusable and reproducible workflow

7 Findings

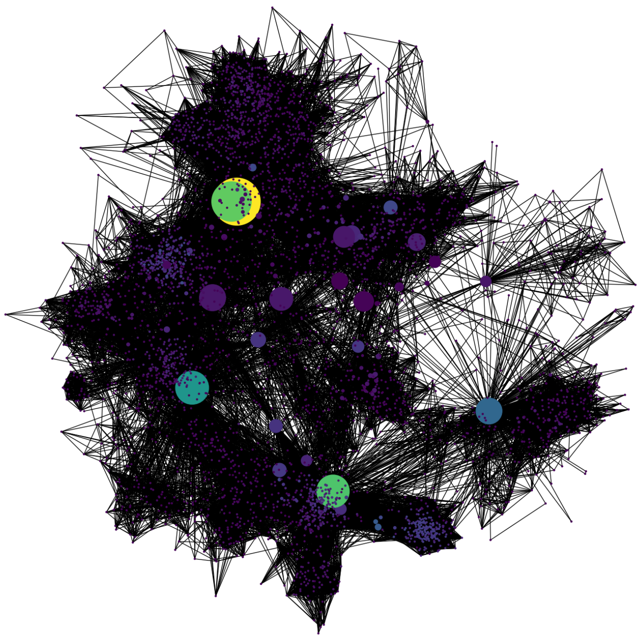

Figure 2: Graph plotted after calculating Betweenness Centrality

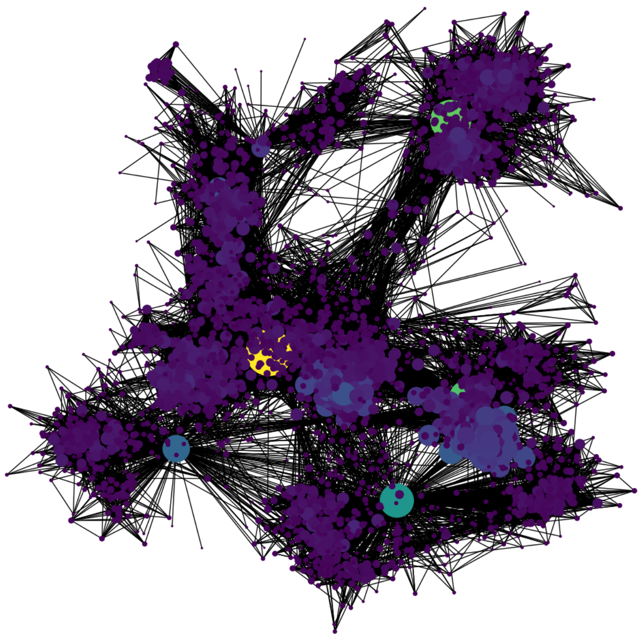

Figure 3: Graph plotted after calculating Degree Centrality

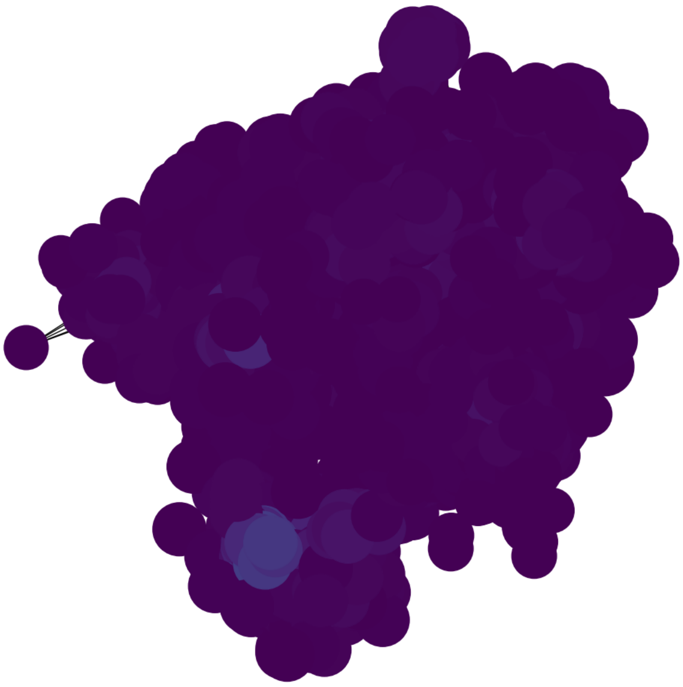

Figure 4: Graph plotted after calculating Closeness Centrality

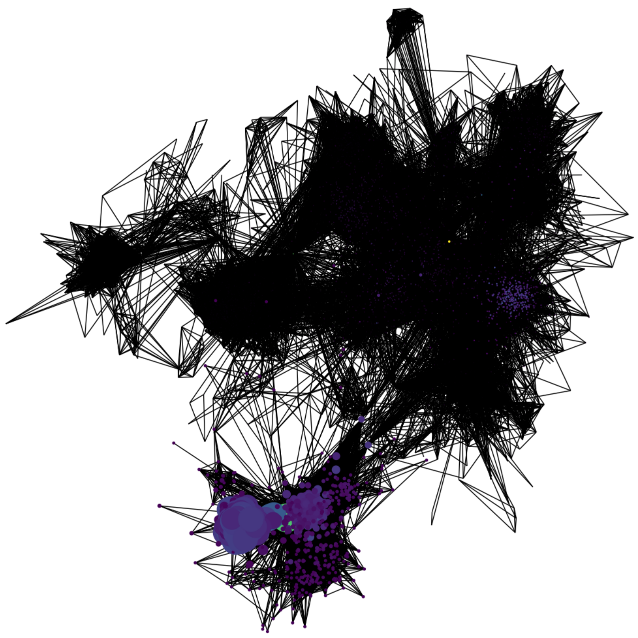

Figure 5: Graph plotted after calculating Eigen Vector Centrality

8 Conclusion

The project *"**Mapping Friendships: Social Network Analysis of Facebook Ego Networks**"* successfully demonstrates how graph theory and computational methods can be applied to understand the structural characteristics of online social relationships. By analyzing the Facebook ego network dataset, this study highlights how individuals are interconnected, how influence flows through the network, and how tightly knit communities emerge within larger social structures.

Using **Python** and the **NetworkX** library, the project computes multiple centrality measures that help identify the most important users in the network. The results indicate that some nodes act as critical connectors with high betweenness centrality, others serve as influential hubs through high eigenvector values, and some maintain numerous direct connections reflected by high degree centrality. These observations mirror real-world social dynamics, where a small group of individuals often plays a disproportionately significant role in communication and information distribution.

Community detection further strengthens the analysis by uncovering groups of users who share dense internal connections. These modular structures often represent real-life communities such as friends, colleagues, classmates, or interest-based circles. Identifying these clusters demonstrates the relevance of Social Network Analysis (SNA) in understanding human behavior, digital interactions, and the formation of social bonds.

Visualization is also a key component of the project. Force-directed layouts and centrality-based node mapping reveal structural patterns that are not immediately apparent in raw data. By illustrating hubs, bridges, and community clusters, the graphical representations provide intuitive insights that support the analytical findings.

Overall, the project integrates data processing, centrality computation, community detection, and visualization into a cohesive analytical workflow. It provides meaningful insights into how online friendships are structured and how influential users contribute to network connectivity. The study also emphasizes the practical utility of SNA techniques for analyzing large-scale social datasets. Future work may include advanced visual dashboards, interactive tools, or machine learningbased predictions to uncover even deeper insights from social network data.

9 Future Scope

Although this project provides a comprehensive analysis of Facebook ego networks using centrality measures, community detection, and visualization, there are numerous opportunities to enhance and extend the system. As social network analysis continues to evolve, several modern techniques and technologies can be integrated to improve the depth, accuracy, and scalability of the analysis.

1. Integration of Machine Learning Models

Future work can include machine learning algorithms to predict:

- Influential users
- Potential new friendships
- Community evolution over time
- Viral content propagation paths

Graph Neural Networks (GNNs), such as GraphSAGE and Graph Convolutional Networks (GCNs), can be used to generate node embeddings and classify user roles within the network.

2. Dynamic Network Analysis

The current project analyzes a static network. In real social platforms, relationships evolve over time. Extending the model to support temporal graphs would enable:

- Tracking changes in friendships
- Identifying periods of rapid growth or decline
- Modeling real-time influence spread

This would significantly strengthen the understanding of long-term user behavior.

3. Sentiment and Content-Based Analysis

If posts, comments, or interaction datasets are included, the system could be enhanced to:

- Analyze user sentiments
- Detect influential content creators
- Measure emotional influence within communities

Combining SNA with natural language processing (NLP) would provide deeper insights.

4. Enhanced Visualizations and Interactive Dashboards

Advanced tools such as Gephi, Plotly, Power BI, or Streamlit can be used to create:

- Interactive 3D network visualizations
- Dynamic node and edge filtering
- Community heatmaps
- User-controlled exploration of graph attributes

These visual improvements would make the analysis more intuitive.

5. Real-Time Network Monitoring

By integrating social media APIs, the system can support:

- Live network updates
- Real-time detection of trending communities
- Instant changes in centrality metrics

Such features are valuable for social media analytics and monitoring.

6. Big Data Scalability

For networks with millions of nodes and edges, future versions can incorporate:

- Distributed graph processing (Apache Spark GraphX, Neo4j, TigerGraph)
- GPU-accelerated centrality computation
- Parallel processing to reduce computation times

This would make the system suitable for enterprise-scale networks.

7. Anomaly and Fraud Detection

SNA can also reveal abnormal or malicious behavior such as:

- Fake profiles
- Coordinated activity
- Bot networks
- Suspicious isolated or dense clusters

Integrating anomaly detection algorithms would greatly enhance the real-world usefulness of the system.

10 References

1. Jure Leskovec and Andrej Krevl. *SNAP Datasets: Stanford Large Network Dataset Collection.* Retrieved from `https://snap.stanford.edu/data`
(Facebook Combined Ego Network Dataset)

2. Newman, M. E. J. (2010). *Networks: An Introduction.* Oxford University Press.
(Foundational concepts of graph theory and social networks)

3. Wasserman, S., & Faust, K. (1994). *Social Network Analysis: Methods and Applications.* Cambridge University Press.
(Standard reference for SNA techniques and metrics)

4. Freeman, L. C. (1977). "A Set of Measures of Centrality Based on Betweenness." *Sociometry.*
(Introduced betweenness centrality)

5. Bonacich, P. (1987). "Power and Centrality: A Family of Measures." *American Journal of Sociology.*
(Introduced eigenvector centrality)

6. NetworkX Developers. *NetworkX Documentation.* Retrieved from `https://networkx.org/documentation/stable/`
(Algorithms and graph computation tools used)

7. Matplotlib Developers. *Matplotlib Documentation.* Retrieved from `https://matplotlib.org/`
(Graph visualization tools used)

8. Clauset, A., Newman, M. E. J., & Moore, C. (2004). "Finding Community Structure in Very Large Networks." *Physical Review E.*
(Important work on modularity-based community detection)

9. Girvan, M., & Newman, M. E. J. (2002). "Community Structure in Social and Biological Networks." *PNAS.*
(Introduced the Girvan–Newman community detection algorithm)

10. Barabsi, A.-L. (2016). *Network Science.* Cambridge University Press.
(Covers scale-free networks, hubs, and network structure)