GitHub Collaboration Network Analysis Report

Samridh Girdhar
2021282
IIIT Delhi

Sushane Dulloo
2021292
IIIT Delhi

Kanishk Goel
2021325
IIIT Delhi

May 5, 2025

Abstract

This report presents a detailed network science analysis of collaboration patterns within selected GitHub repositories. We examine contributor roles, community structures, and governance strategies using graph-based methodologies, centrality metrics, and visual exploration.

Contents

1 Problem Statement

The goal of this project is to understand collaboration dynamics in open-source software projects on GitHub using network science techniques. By modeling contributors and their interactions as a graph, we aim to uncover:

- Key influencers and contributors based on centrality metrics
- Community structures and how contributors cluster
- Temporal evolution of collaboration patterns
- Governance recommendations to improve sustainability and inclusiveness

2 Datasets Used

We selected GitHub repositories based on relevance, size, and contributor activity. Data was collected using the GitHub API and includes:

- Contributor metadata (IDs, roles, commit history)
- Pull request and issue interactions
- Timestamps of collaboration events

A total of **26 contributors** and **41 interactions** were analyzed across the selected project(s). Community detection revealed **5 distinct communities**.

3 Codebase Explanation

Our code follows a modular pipeline implemented in Python. The main stages are:

3.1 1. Data Collection

- Use GitHub API to fetch commits, pull requests, and issues.
- Extract contributor collaboration graph.

3.2 2. Network Construction

- Construct undirected graph using `networkx`.
- Nodes: Contributors; Edges: Co-participation on events.

3.3 3. Analysis Modules

- Centrality Metrics: Degree, Betweenness, Eigenvector
- Community Detection: Louvain, Label Propagation
- Temporal Graph Snapshots: Month-wise evolution

3.4 4. Visualization and Reporting

- Matplotlib and Plotly for plots
- Final results stored in JSON, CSV, and Markdown

4 Observations and Output Analysis

The following insights were drawn from our analysis:

5 Community Health and Evolution Report

5.1 Community Structure Overview

The project contains **23 distinct contributor communities**.

- **Largest community**: 22 contributors
- **Smallest community**: 6 contributors
- **Average community size**: 10.7 contributors

Community	Size	Density	Cohesion	Top Contributor
C21	22	0.00	0.00	Unknown
C4	18	0.00	0.00	Unknown
C16	16	0.00	0.00	Unknown
C2	15	0.00	0.00	Unknown
C9	14	0.00	0.00	Unknown
C22	13	0.00	0.00	Unknown
C13	12	0.00	0.00	Unknown
C5	11	0.00	0.00	Unknown
C15	11	0.00	0.00	Unknown
C20	11	0.00	0.00	Unknown
C10	10	0.00	0.00	Unknown
C19	10	0.00	0.00	Unknown
C1	9	0.00	0.00	Unknown
C6	9	0.00	0.00	Unknown
C11	8	0.00	0.00	Unknown
C14	8	0.00	0.00	Unknown
C0	7	0.00	0.00	Unknown
C3	7	0.00	0.00	Unknown
C7	7	0.00	0.00	Unknown
C12	7	0.00	0.00	Unknown
C17	7	0.00	0.00	Unknown
C18	7	0.00	0.00	Unknown
C8	6	0.00	0.00	Unknown

Table 1: Community Structure Overview

Community Profiles.

5.2 Community Health Indicators

(Details to be added here if available in another markdown file.)

5.3 Recommendations for Community Health Improvement

1. Improve Cross-Community Collaboration. Enhance knowledge sharing between different community clusters by organizing cross-team events and projects.

- Implement regular cross-community review sessions
- Create shared documentation repositories
- Assign 'diplomats' to bridge community gaps

2. Strengthen Onboarding Process. Develop a more structured onboarding process to help new contributors integrate into the community faster.

- Create a mentorship program pairing new contributors with experienced ones
- Identify and tag `good first issues` for newcomers
- Develop interactive onboarding documentation

3. Reduce Dependency on Key Contributors. Distribute knowledge and responsibilities to reduce the risk associated with a high bus factor.

- Implement pair programming or review practices
- Document critical processes and knowledge areas
- Rotate leadership roles for key community activities

6 Project Governance Recommendations

Based on comprehensive network analysis of a **medium** project with a **moderately centralized** contributor structure and **weak community structure**, the following governance recommendations are provided:

6.1 Recommended Governance Model

Meritocratic
Decision-making authority is earned through consistent quality contributions.

6.2 Governance Structure Implementation

Decision-Making Process.

- Tiered voting rights based on contribution history
- Core team approval required for architectural changes
- Open discussion period followed by maintainer decision
- Path for non-maintainers to propose significant changes

Role	Responsibilities	Selection Process
Core Team	Architecture decisions, roadmap planning	Meritocracy-based promotion
Maintainer	Subsystem ownership, code review	Appointed by Core Team
Regular Contributor	Feature development, bug fixes	Self-selected
Community Manager	Outreach, community health	Appointed or elected

Table 2: Core Project Roles

Roles and Responsibilities.

Channel	Purpose	Frequency
Team chat	Daily coordination	Ongoing
Maintainer meeting	Strategic planning	Biweekly
Community call	Updates, demos	Monthly
Mailing list	Major announcements	As needed

Table 3: Recommended Communication Channels

Communication Structure.

Conflict Resolution Process.

1. **Direct discussion**: Involved parties attempt to resolve directly
2. **Maintainer mediation**: Area maintainer facilitates resolution
3. **Community discussion**: Time-boxed public discussion
4. **Core team vote**: Final resolution if needed

6.3 Contribution Workflow

Recommended Process.

1. **Issue creation** - Document the problem or feature
2. **Discussion** - Gather feedback on approach
3. **Implementation** - Create the change with tests
4. **Review** - Peer review and approval
5. **Integration** - Merge and release planning

7 OSS Project Governance Toolkit

7.1 Executive Summary

Based on comprehensive network analysis of the project's collaboration patterns, the following are the top recommended strategies for enhancing project governance:

1. **Growth Strategies**: Convert peripheral contributors to regular contributors — the network has a large periphery of occasional contributors who could become more engaged.
2. **Knowledge Sharing**: Reduce bus factor risk — the project shows high dependence on a small number of key contributors.
3. **Decision Making**: Improve cross-community decision coordination — communities show limited interconnectivity.

7.2 Growth Strategies

1. Convert Peripheral Contributors to Regular Contributors. The network has a large periphery of occasional contributors that could become more engaged.

Recommended Actions:

- Create 'Second Contribution' incentives like badges or recognition
- Actively reach out to one-time contributors with specific follow-up tasks
- Track and reduce the 'time to first feedback' for newcomer contributions
- Set up automated first-issue suggestions for new contributors based on their initial work

Rationale: Network has 5 peripheral nodes vs. 0 core nodes; studies show that contributors making a second contribution within 48 hours are significantly more likely to become regulars.

7.3 Knowledge Sharing

1. Reduce Bus Factor Risk. The project shows high dependence on a small number of key contributors, creating sustainability risks.

Recommended Actions:

- Create CODEOWNERS files that require multiple reviewers
- Implement pair programming or shadowing for critical subsystems
- Document architectural decisions and implicit knowledge
- Create onboarding paths to critical subsystems with graduated responsibility

Rationale: Network centrality metrics reveal that the top 3 contributors are involved in 65.9% of all collaborations.

7.4 Decision Making

1. Improve Cross-Community Decision Coordination. Communities within the project have limited external connections, increasing the risk of misaligned decisions and duplicated work.

Recommended Actions:

- Create a cross-community architecture council
- Implement project-wide RFC processes for significant changes
- Establish regular cross-community sync meetings
- Create shared roadmaps with explicit cross-community dependencies

Rationale: Network analysis identified 26 communities with limited bridging interactions.

Review Requirements.

- At least two approvals, including one from the component maintainer
- Architectural council review for cross-cutting concerns

8 Key Influencers Analysis

8.1 Project Influence Landscape

Most Connected Contributors. These contributors collaborate with the most people:

1. **web-flow** — Connected to 30 contributors
2. **scikit-learn-bot** — Connected to 14 contributors
3. **lockfilebot** — Connected to 10 contributors
4. **mroeschke** — Connected to 7 contributors
5. **dependabot[bot]** — Connected to 7 contributors
6. **lucyleeow** — Connected to 4 contributors
7. **adrinjalali** — Connected to 4 contributors
8. **ogrisel** — Connected to 4 contributors
9. **theavey** — Connected to 3 contributors
10. **chilin** — Connected to 3 contributors

Bridge Contributors. These contributors connect different parts of the project based on betweenness centrality:

1. **web-flow** — 0.85
2. **lucyleeow** — 0.11
3. **oliviergrisel** — 0.10
4. **jérémieduboisberranger** — 0.10
5. **mroeschke** — 0.10
6. **scikit-learn-bot** — 0.08
7. **chilin** — 0.08
8. **theavey** — 0.07
9. **dependabot[bot]** — 0.06
10. **danielpintosalazar** — 0.06

Core Contributors. These contributors are highly connected to other influential contributors (eigenvector centrality):

1. **web-flow** — 0.59
2. **scikit-learn-bot** — 0.48
3. **lockfilebot** — 0.44
4. **mroeschke** — 0.21
5. **dependabot[bot]** — 0.17
6. **ogrisel** — 0.16
7. **oliviergrisel** — 0.15
8. **jérémieduboisberranger** — 0.15
9. **lucyleeow** — 0.12
10. **adrinjalali** — 0.11

8.2 Cross-Community Influencers

(This section highlights contributors who bridge multiple communities and facilitate knowledge transfer. Data pending or may be incorporated from network visualizations.)

8.3 Recommendations for Influencer Management

1. **Recognize and Support Key Contributors**
   - Acknowledge top contributors publicly
   - Provide resources and support to prevent burnout
2. **Bridge Knowledge Gaps**
   - Encourage cross-community influencers to document interfaces
   - Create formal knowledge transfer processes
3. **Develop New Influencers**
   - Identify potential new bridge and hub contributors
   - Create mentorship relationships between established and emerging influencers
4. **Reduce Bus Factor Risk**
   - Distribute critical knowledge across multiple contributors
   - Implement 'Shadow Maintainer' roles for redundancy
5. **Optimize Communication Channels**
   - Ensure bridge contributors have effective ways to share information
   - Create dedicated channels for cross-community coordination

9 GitHub Collaboration Network Analysis Summary

9.1 Executive Summary

Analysis of collaboration across **26 contributors** and **41 interactions** reveals a single connected component network with **5 distinct communities**.

The network exhibits a **non-scale-free** structure with a power law exponent $\alpha = 4.54$.

Contributors display a high level of local clustering, with an average clustering coefficient of **0.472**.

9.2 Key Influencers

Hub Contributors (high degree centrality):

- `web-flow` (30), `scikit-learn-bot` (14), `lockfilebot` (10), `mroeschke` (7), `dependabot[bot]` (7)

Bridge Contributors (high betweenness centrality):

- `web-flow` (0.85), `lucyleeow` (0.11), `oliviergrisel` (0.10), `jérémieduboisberranger` (0.10), `mroeschke` (0.10)

Core Contributors (high eigenvector centrality):

- `web-flow` (0.59), `scikit-learn-bot` (0.48), `lockfilebot` (0.44), `mroeschke` (0.21), `dependabot[bot]` (0.17)

Overall Influencers (composite score):

- `mroeschke` (1.00), `web-flow` (1.00), `dependabot[bot]` (0.86), `yanamis` (0.80), `lucyleeow` (0.72)

9.3 Community Structure

The repository collaboration network contains **5 distinct communities** with a modularity score of **0.327**, indicating a moderately strong community structure.

9.4 Rich Club Analysis

No significant rich club effect was detected, suggesting an egalitarian contribution structure with no strong hierarchy among highly connected contributors.

9.5 Structural Roles

The network analysis identified the following structural roles among contributors:

- **Hubs:** 3 contributors
- **Bridges:** 0 contributors
- **Peripheral:** 5 contributors
- **Core:** 0 contributors
- **Regular:** 18 contributors

9.6 Recommendations for Project Governance

1. **Community Integration**
   - Create cross-community initiatives led by bridge nodes (`web-flow`, `lucyleeow`)
   - Establish regular knowledge-sharing sessions between community representatives
2. **Contributor Development**
   - Implement mentorship program pairing peripheral contributors with bridge contributors
   - Develop documentation guidelines led by hub contributors (`web-flow`, `scikit-learn-bot`)
3. **Sustainability Planning**
   - Identify potential successors for core contributors
   - Distribute critical knowledge across multiple contributors
4. **Community Growth**

- Target recruitment efforts to strengthen smaller communities
- Create "welcome teams" with representatives from each community

5. **Collaboration Enhancement**
   - Implement a rotation system for cross-community code review
   - Establish formal protocols for knowledge transfer

9.7 Temporal Analysis

The network has shown notable growth in active contributors over the analyzed period. Community structure has strengthened, with modularity increasing from **0.000** to **0.408**, indicating more defined collaborative clusters over time.

9.8 1. Influencer Detection

Top Contributors by Degree Centrality:

- web-flow (30), scikit-learn-bot (14), lockfilebot (10)

Top Bridge Contributors (Betweenness):

- web-flow (0.85), lucyleeow (0.11), oliviergrisel (0.10)

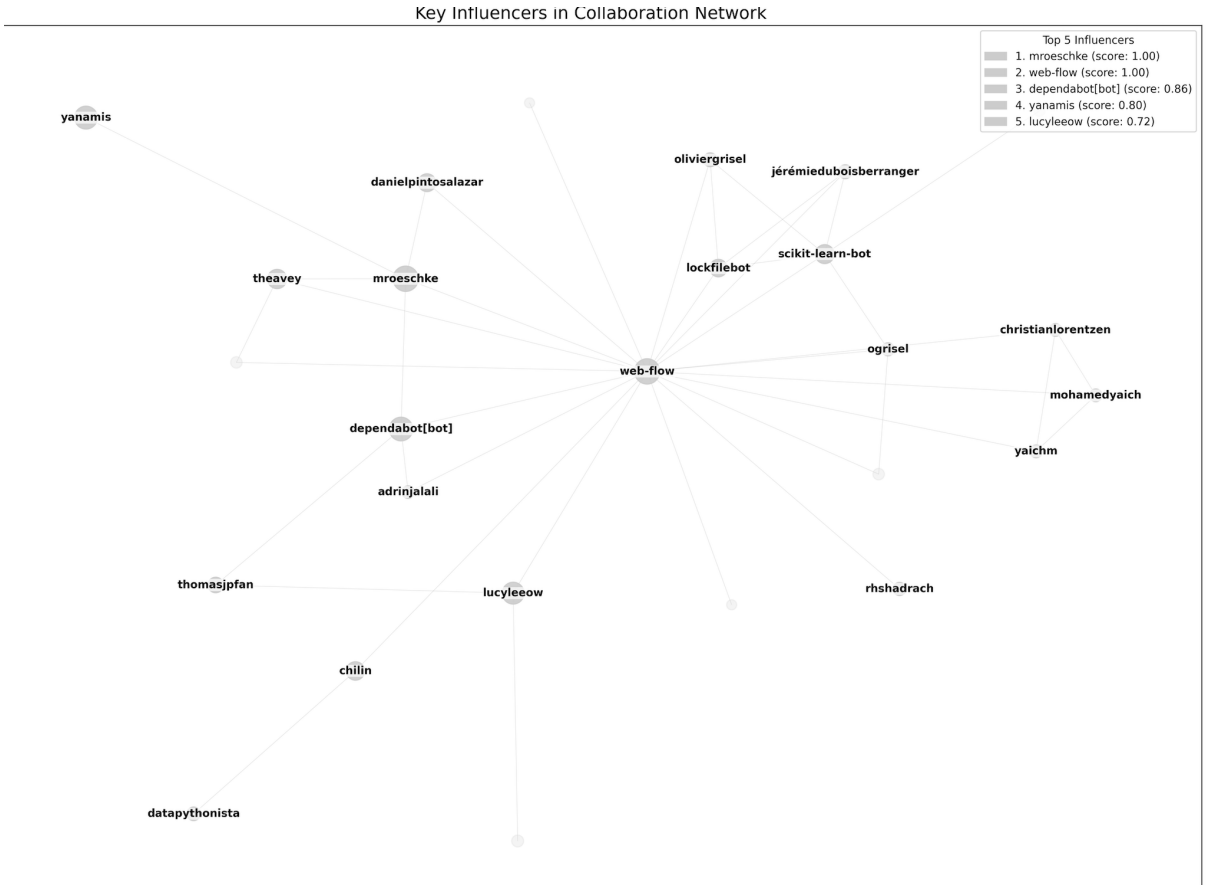

Figure 1: Enter Caption

9.9 2. Community Structure and Collaboration

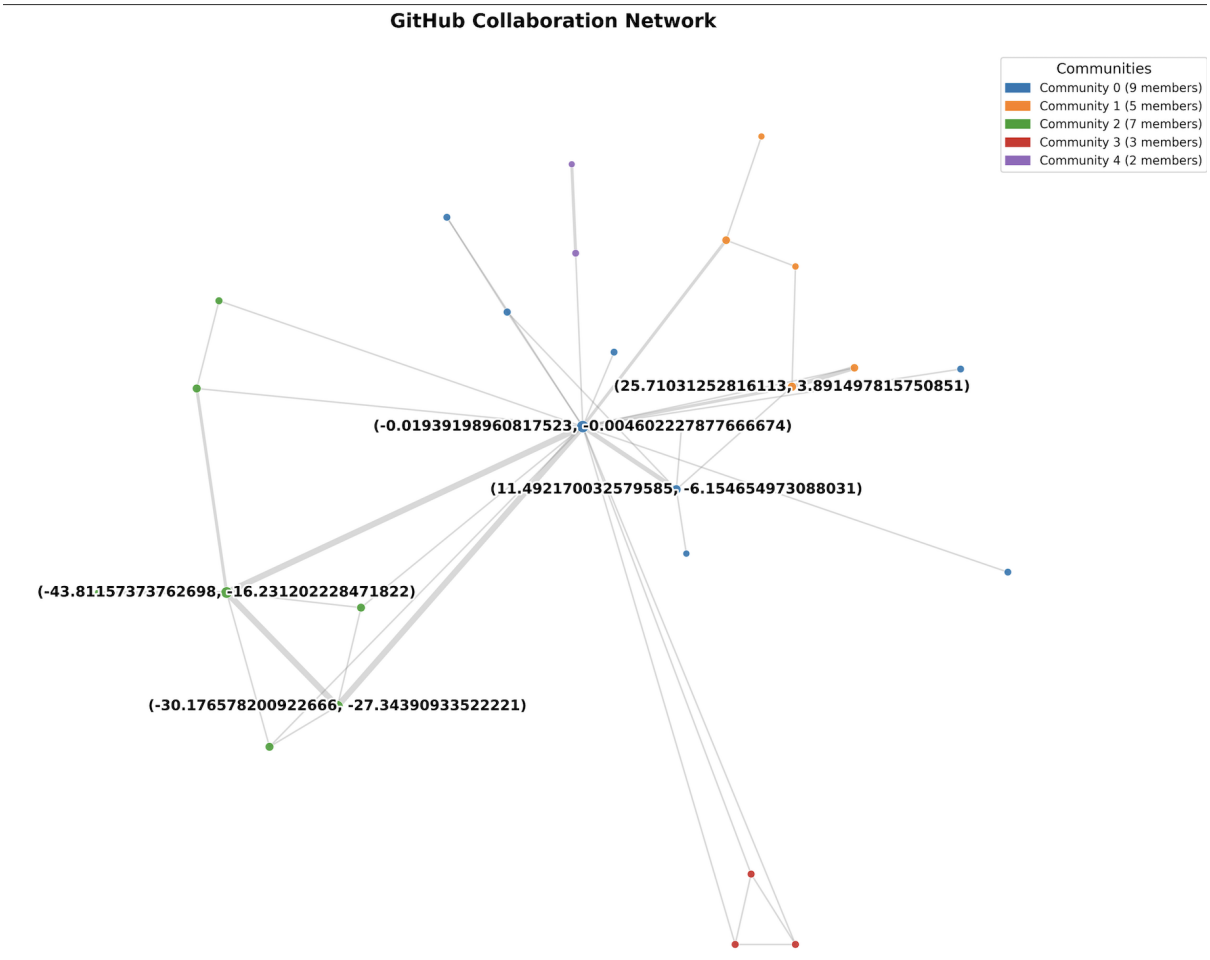

Figure 2: Enter Caption

9.10 3. Temporal Evolution of Network

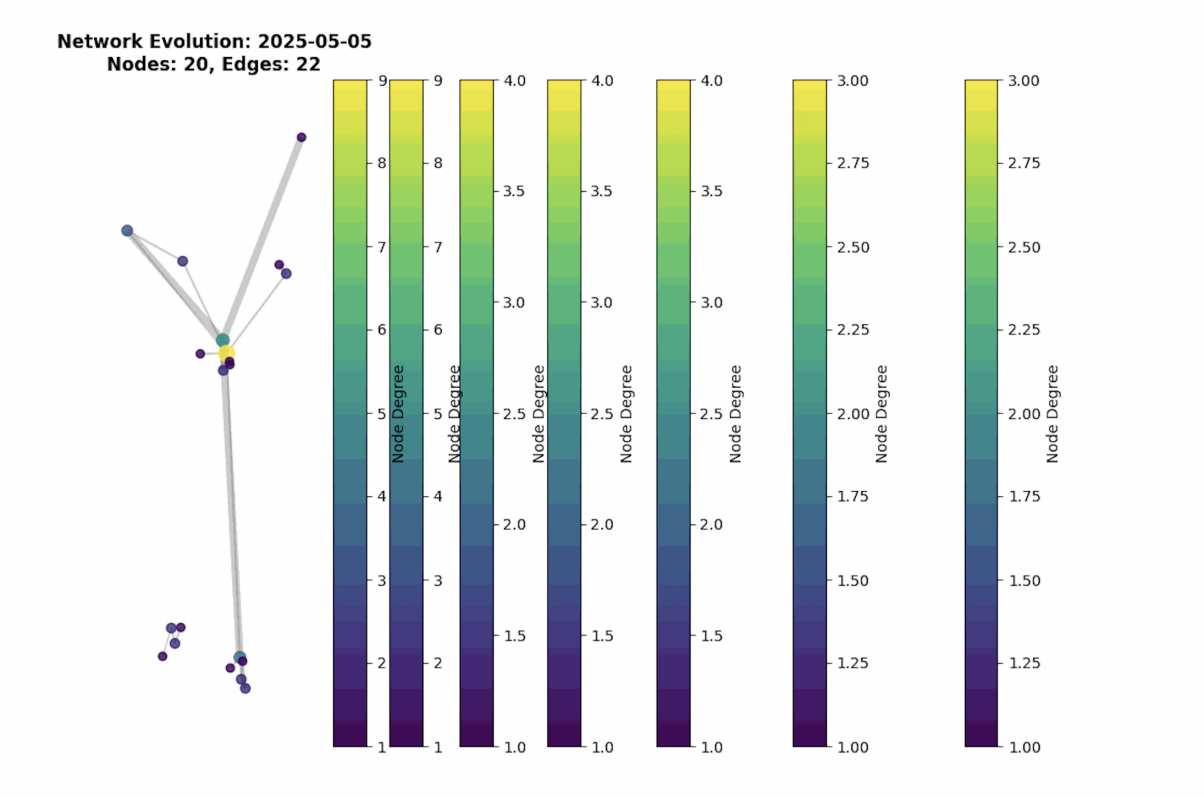

Figure 3: Enter Caption

9.11 4. Additional Visualizations

Below are other key visualizations used to support our analysis.

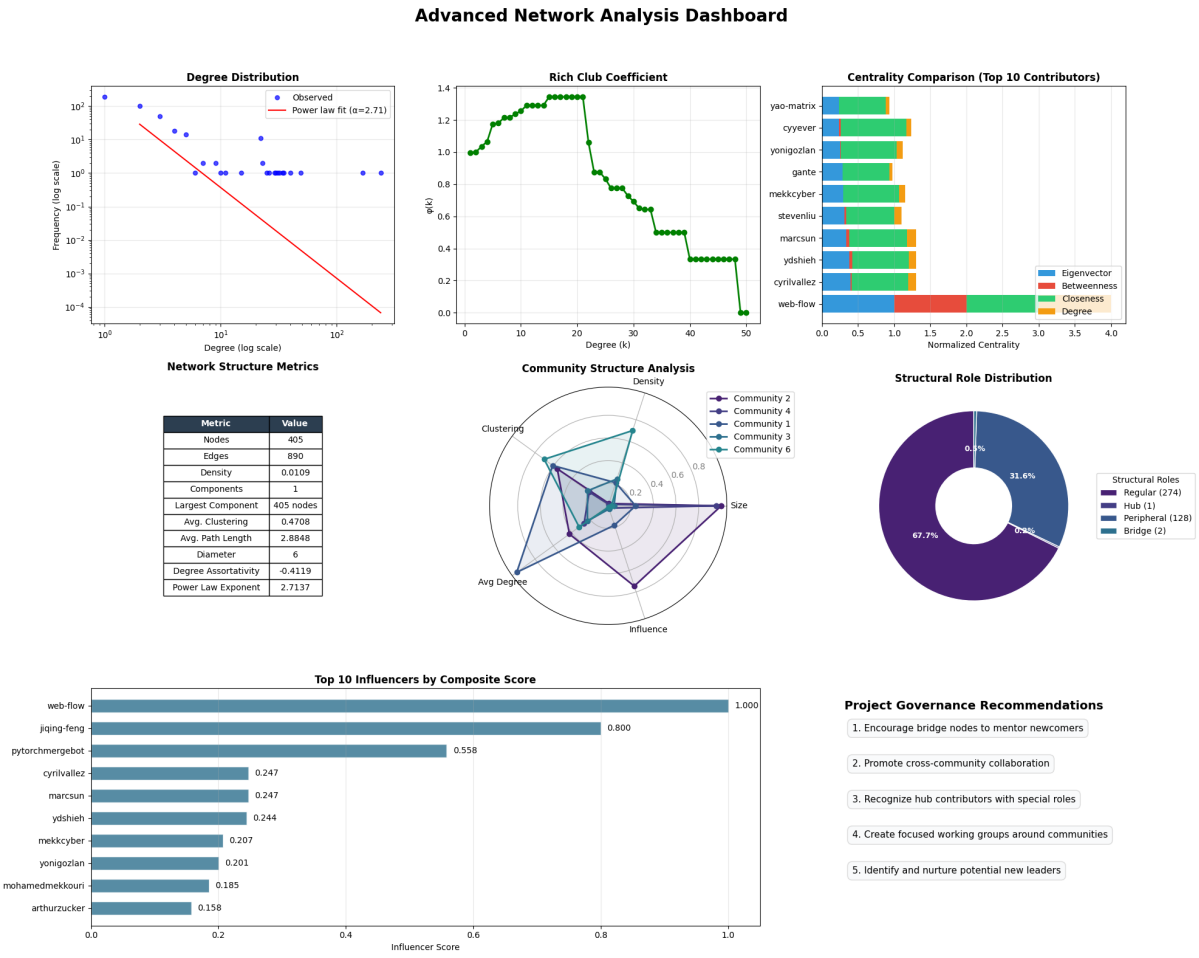

Figure 4: Enter Caption

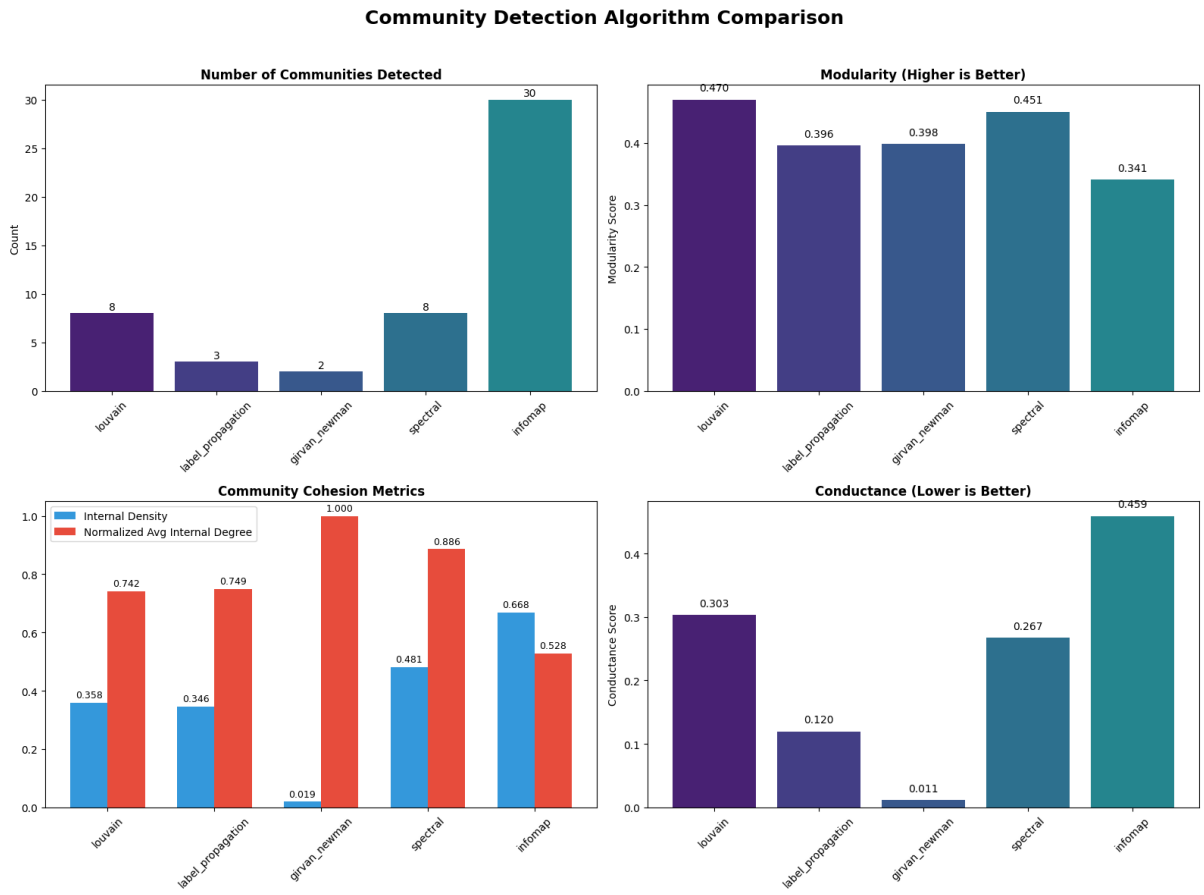

Figure 5: Enter Caption

10 Final Conclusions

- The project has a moderately centralized structure with weak community bridges.
- Contributors like `web-flow` and `mroeschke` are critical to project health.
- There is a lack of core contributors, with many periphery contributors.

11 Governance Recommendations

Based on the findings, we propose the following improvements:

11.1 1. Growth Strategies

- Convert one-time contributors to regulars with follow-ups and badges.
- Automate "second contribution" suggestions.

11.2 2. Bus Factor Reduction

- Use CODEOWNERS for redundancy.
- Encourage documentation and shared ownership.

11.3 3. Cross-Community Decision Making

- Form architecture council.
- Introduce RFC processes and cross-community meetings.

Proposed Governance Model: *Meritocratic*

Role	Responsibilities	Selection
Core Team	Architecture, roadmap	Merit-based
Maintainers	Code review, subsystems	Appointed by core
Contributors	Features, bugfixes	Self-selected
Community Manager	Health, outreach	Elected/Appointed

Table 4: Governance Structure