Community Detection and Analysis on DBLP v9 Dataset: Centrality, Network Metrics, and Predictive Modeling

Xiang Zheng
21307110169

Qin Ma
21307110024

Author n
Address line

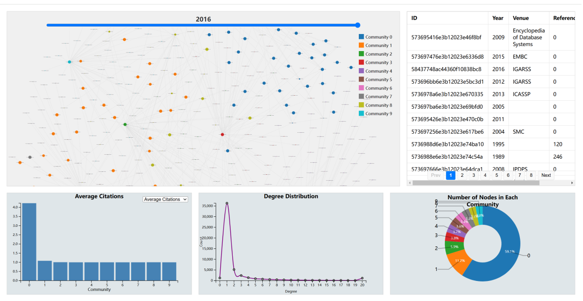
School of Data Science, Fudan University

Abstract

abstract

1 Introduction

Network analysis is a critical methodology for studying complex systems across various domains. The DBLP dataset, a comprehensive bibliographic resource for computer science, forms the foundation for investigating collaboration patterns and scholarly relationships. This study applies advanced graph analysis techniques to explore the structural and functional properties of the DBLP v9 dataset.



ID	Year	Venue	Referenc
573695416e3b12023e46f8bf	2009	Encyclopedia of Database Systems	0
573697476e3b12023e6336d8	2015	EMBC	0
58437748ac44360f10838bc8	2016	IGARSS	0
573696bb6e3b12023e5bc3d1	2012	IGARSS	0
5736978a6e3b12023e670335	2013	ICASSP	0
573697ba6e3b12023e69bfd0	2005		0
573695426e3b12023e470c0b	2011		0
573697256e3b12023e617be6	2004	SMC	0
5736988d6e3b12023e74ba10	1995		120
5736988e6e3b12023e74c54a	1989		246
573697666e3b12023e64dca1	2008	IPDPS	0

Figure 1: Snapshot of the visualization system

The subsequent sections of this report follow the analytical framework outlined below:

Preprocessing: Transforming the raw DBLP data into a structured network through cleaning, filtering, and attribute assignment.

Community Detection: Identifying cohesive subgroups using algorithms like Louvain and Label Propagation to uncover collaboration dynamics and research specializations.

Centrality Analysis: Analyzing metrics such as degree centrality and PageRank to identify influential nodes and assess network topology. **Link Prediction**: Employing the GLACE model on the Cora-ML dataset to predict future connections based on structural features, providing insights into evolving citation trends.

Visualization: Presenting findings through an interactive system developed with **Python** and **D3.js**, highlighting key network characteristics and community structures.

This structured approach provides a comprehensive examination of the DBLP co-authorship and publication network, delivering insights into collaboration patterns, influential figures, and the evolution of academic networks.

2 Preprocessing

The preprocessing phase is a critical step in transforming the raw DBLP V9 dataset into a structured format suitable for detailed analysis. This stage encompasses data cleaning, network construction, feature engineering, and dataset filtering. The DBLP V9 dataset was specifically chosen for its balance between computational feasibility and data comprehensiveness, enabling robust analysis of academic collaboration and citation patterns within computer science.

2.1 Dataset Selection

The DBLP-Citation-network V9 dataset was selected due to its optimal balance between data richness and computational manageability, as summarized in Table 1. Other versions of the DBLP dataset, while valuable, either lacked the depth of information or exceeded practical computational limits for this project. DBLP V9 captures key trends up to July 3, 2017, offering a comprehensive yet tractable dataset with **3,680,007 papers** and **1,876,067 citation relationships**.

2.2 Data Cleaning and Integration

The raw DBLP dataset includes metadata such as titles, authors, venues, and citations. Data cleaning involved:

Dataset Version	Number of Papers	Number of Citation Relationships
Citation-network V1	629,814	>632,752
Citation-network V2	1,397,240	>3,021,489
DBLP-Citation-network V3	1,632,442	>2,327,450
DBLP-Citation-network V4	1,511,035	2,084,019
DBLP-Citation-network V5	1,572,277	2,084,019
DBLP-Citation-network V6	2,084,055	2,244,018
DBLP-Citation-network V7	2,244,021	4,354,534
DBLP-Citation-network V8	3,272,991	8,466,859
DBLP-Citation-network V9	**3,680,007**	**1,876,067**
DBLP-Citation-network V10	3,079,007	25,166,994
DBLP-Citation-network V11	4,107,340	36,624,464
DBLP-Citation-network V12	4,894,081	45,564,149
DBLP-Citation-network V13	5,354,309	48,227,950
DBLP-Citation-network V14	5,259,858	36,630,661

Table 1: Summary of DBLP Citation Network Versions

- **Grouping Records:** Papers were grouped using unique identifiers to ensure each record was correctly structured.
- **Resolving Missing Data:** Missing fields, such as titles, authors, or venues, were assigned default values to maintain consistency.

These steps ensured the dataset was standardized and ready for subsequent analysis.

2.3 Network Construction

The dataset was represented as two interconnected networks:

1. **Co-authorship Network:** In this network, authors are represented as nodes, with weighted edges denoting co-authorship relationships. The weight of each edge corresponds to the frequency of collaborations between authors. This network contains a total of 3,680,007 nodes (authors) and 1,876,067 edges (co-authorship relationships).
2. **Citation Network:** The citation network is constructed by representing papers as nodes, with directed edges indicating citation relationships between them. Each edge direction signifies the citing paper and the cited paper. This network also comprises 3,680,007 nodes (papers) and 1,876,067 edges (citation relationships).

This dual representation allows for a comprehensive study of collaboration and citation dynamics.

2.4 Feature Engineering

Key features were engineered to enhance the dataset's analytical capabilities:

- **Co-authorship Features:** Unique author identifiers were assigned, and collaboration frequencies were calculated.
- **Citation Metrics:** In-degree (citations received) and out-degree (references) were computed for each paper.
- **Venue Indexing:** Publication venues were standardized and indexed for uniform representation.

2.5 Dataset Filtering

Papers with no citations and references were flagged as "isolate" and excluded to improve computational efficiency. Additionally, thresholds were applied to focus on significant collaborations and impactful papers.

2.6 Exploratory Analysis

Exploratory analyses were conducted using Python libraries such as `pandas` and `matplotlib`. The key findings are visualized in Figures 2–5:

- **Authors per Paper:** Most papers have few authors, with fewer multi-author publications.
- **Citation Distribution:** Citations are highly skewed, with a small number of papers receiving the majority of citations.

- **References per Paper:** Papers with more citations tend to reference more works.
- **Co-authors per Author:** A small group of authors collaborate extensively, while most have limited collaborations.

These analyses offer critical insights into academic collaboration and citation patterns, laying the groundwork for deeper exploration of academic network structures.

3 Community Mining

In academic networks, community mining aims to identify cohesive subgroups reflecting collaborative dynamics, research specializations, and the structural foundation of scholarly interactions. Three common algorithms - Louvain, Label Propagation, and Multi - level - are used to discover potential communities based on node relationships.

After detection, community proportions and modularity are calculated to assess the quality of community division. They provide insights into the network's structure, indicating whether detected communities have strong internal and weak external connections, characteristic of well - formed community structures.

3.1 Overview of Community Mining

Community detections operate by grouping nodes in a graph based on their connectivity. In the context of academic networks, nodes represent entities such as papers or authors, and edges represent relationships between these entities, such as co-authorship or citations. Below are the specifics of Paper Networks and Author Networks.

- Paper Network:In a paper network, nodes represent individual papers, and edges represent relationships such as citations or co-authorships. Edges are weighted by the number of citations or co-authors shared. Community detection aims to identify research areas or subfields where papers are closely linked by citations or co-authors, revealing thematic connections or shared academic topics.
- Author Network:In an author network, nodes represent authors, and edges represent co-authorships. The edge weight reflects the number of joint papers between authors, with higher weights indicating stronger collaboration. Community detection in this network uncovers collaborative groups or research teams, offering insights into research partnerships and collaboration patterns.

The goal of community detection is to find subsets of nodes (communities) where the internal connectivity is higher than expected by random chance, and the external connectivity is relatively lower. Mathematically, this is often expressed in terms of modularity.

3.2 Community Detection Algorithms

3.2.1 Louvain Algorithm

The Louvain algorithm is a method for detecting communities by optimizing modularity. The modularity Q for a community division is calculated as:

$$Q = \frac{1}{2m} \sum_{i,j} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(c_i, c_j)$$

Where: m is the total number of edges in the network. A_{ij} is the weight of the edge between nodes i and j. k_i and k_j are the degrees of nodes i and j. $\delta(c_i, c_j)$ is 1 if nodes i and j are in the same community and 0 otherwise. c_i and c_j are the community labels of nodes i and j.

The Louvain algorithm works in two phases:

1. Local optimization: Each node is assigned to the community of its neighbor that maximizes modularity.

2. Community aggregation: Communities are treated as super-nodes, and the algorithm repeats the process of modularity optimization on this new, aggregated graph.

The algorithm is computationally efficient and is well-suited for large-scale networks, making it a popular choice in detecting academic collaboration communities.

3.2.2 Label Propagation Algorithm

Label Propagation (LP) is a simple and efficient algorithm where each node is initially assigned a unique label. At each iteration, each node updates its label to the most frequent label among its neighbors. This process continues until the labels stabilize.

Mathematically, Label Propagation is expressed as:

$$l_i^{(t+1)} = \arg \max_{l_j \in N(i)} \sum_{k \in N(i), l_k = l_j} \frac{1}{d_k}$$

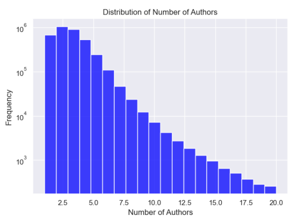


Figure 2: Number of Authors per Paper

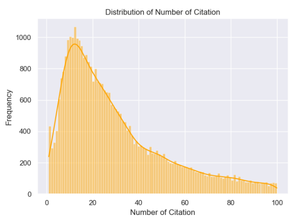


Figure 3: Citation Distribution of Papers

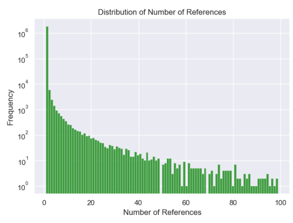


Figure 4: Reference Distribution of Papers

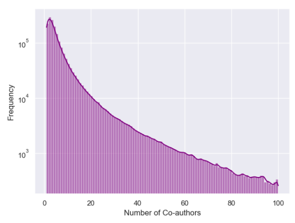


Figure 5: Number of Co-authors per Author

Where: $l_i^{(t+1)}$ is the new label for node i after iteration t. $N(i)$ is the set of neighbors of node i. d_k is the degree of neighbor k.

The algorithm is highly parallelizable and does not require predefined parameters like the number of communities. It is particularly efficient for large networks with many nodes and edges.

3.2.3 Multi-level Algorithm

The Multi-level algorithm is based on a hierarchical approach, where the graph is coarsened into a smaller graph by iteratively merging nodes that are highly connected. The algorithm detects communities at multiple levels by optimizing modularity at each level. After the graph is coarsened, the community detection process is applied to the smaller graph, and the solution is refined by uncoarsing the graph back to its original size.

The Multi-level algorithm follows these general steps:

1. Coarsing: Repeatedly reduce the graph by merging highly connected nodes, creating a smaller graph at each level.

2. Community Detection: Perform community detection on the coarsed graph (typically using modularity maximization).

3. Uncoarsing: Refine the community structure by uncoarsing the graph and applying the community structure from the coarser level to the finer levels.

This method can efficiently handle large-scale networks and is suitable for discovering communities at different scales.

3.2.4 Reasons for Choosing These Algorithms

These three algorithms represent different approaches to community detection:

- Louvain emphasizes modularity optimization and is efficient for large networks.
- Label Propagation uses local information propagation and is highly scalable and efficient.
- Multi-level works at multiple scales, ensuring that communities are detected at both coarse and fine levels.

Each algorithm brings unique strengths, making them well-suited for the diverse and complex academic networks that we are analyzing.

3.3 Community Proportions

The community proportions represent the distribution of nodes across different communities. By calculating the size of each community (i.e., the number of nodes within it) and dividing by the total number of nodes, we can get the proportion of each community in the overall network.

Let N_c be the number of nodes in community c, and N_{total} be the total number of nodes in the network. The proportion of community c is given by:

$$\text{Proportion of community } c = \frac{N_c}{N_{\text{total}}}$$

This helps to understand the size and importance of each community within the entire network. Larger communities may represent more significant areas of academic collaboration, while smaller ones could represent more specialized or niche research areas.

3.4 Modularity

Modularity is a key measure used to evaluate the quality of community detection. It quantifies the strength of division of a network into communities by comparing the number of edges within communities to the number of edges expected by random chance.

The modularity Q is calculated using the formula mentioned earlier. A modularity value greater than 0 indicates that the community division is better

Algorithm	Modularity
Louvain	0.9939834022020653
Label Propagation	0.9909479458016406
Multi-level	0.9938523610510882

Table 2: Modularity values for community detection in the paper network using different algorithms

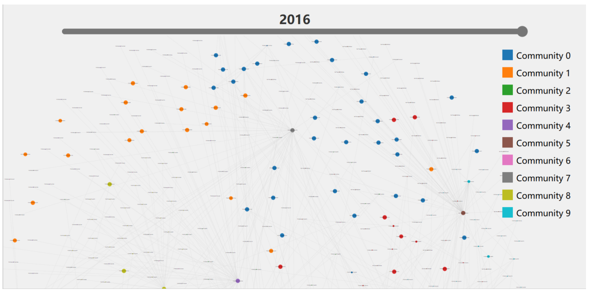


Figure 6: Community partitioning of the paper network using the Louvain algorithm

than a random distribution of edges, with higher values indicating stronger community structures.

The following presents the modularity values obtained from community detection in the paper network using the Louvain, Label Propagation, and Multi-level algorithms in this study.

The results indicate that the modularity for all three algorithms exceeds 99%, suggesting that the community partitions identified by these methods exhibit strong internal cohesion and weak external connections. This high modularity score reflects the algorithms' effectiveness in detecting well-structured communities with clear, well-defined boundaries, underscoring their suitability for analyzing academic collaboration networks.

3.5 Community Detection Results

The results of community detection in paper network are as follows: using the Louvain algorithm, a total of 29,898 communities were detected; the Label Propagation algorithm identified 30,549 communities; and the Multi-level algorithm found 29,589 communities. The following figures illustrate the community partitioning of the paper network using the Louvain algorithm.

In these visualizations, nodes represent individual papers, and edges represent relationships such as citations or co-authorships. Different colors indicate distinct communities within the network. These visualizations offer a clear representation of how papers are grouped into cohesive subgroups, reflecting collaborative research themes or areas of study.

4 Centrality Measurement

After detecting communities within the academic network, it is essential to assess the importance and influence of individual nodes within each community. In this analysis, centrality measures such as Degree Centrality and PageRank Centrality were calculated to identify influential nodes, revealing key hubs and authoritative figures within the network. Additionally, structural characteristics of the network, including community diameters and average citations per author, were analyzed to understand the overall network topology and functional properties. The following sections describe the methodology and results of these calculations:

4.1 Centrality Calculation

4.1.1 Degree Centrality

This measure counts the number of direct connections (edges) a node has, reflecting its immediate popularity or connectivity within the network. A higher degree centrality indicates a node that is well-connected, potentially influencing many other nodes within the community. Degree centrality is particularly useful for identifying the most central or influential nodes based on direct relationships.

The degree centrality $C_d(i)$ for a node i is calculated as:

$$C_d(i) = \deg(i)$$

Where $\deg(i)$ is the number of edges connected to node i.

4.1.2 PageRank Centrality

PageRank, developed by Google to rank web pages, measures the influence of a node by considering both the number and quality (weight) of its incoming edges. In the context of academic networks, PageRank helps identify authoritative nodes, where a high PageRank centrality indicates that a node is not only well-connected but also highly regarded by other influential nodes.

PageRank centrality $PR(i)$ for node i is computed iteratively based on the network structure:

$$PR(i) = \frac{1-d}{N} + d \sum_{j \in \mathcal{N}(i)} \frac{PR(j)}{|\mathcal{N}(j)|}$$

Where: d is the damping factor (usually set to 0.85), N is the total number of nodes in the network,

$\mathcal{N}(i)$ is the set of neighbors of node i, $|\mathcal{N}(j)|$ is the number of outgoing edges from node j.

4.1.3 Why Degree Centrality and PageRank Centrality?

The selection of Degree Centrality and PageRank Centrality as centrality measures is based on their ability to capture different yet complementary aspects of node influence and importance within a network.

- Degree Centrality measures the number of direct connections a node has. In academic networks, it identifies the most connected authors or papers, making it useful for spotting collaboration hubs and active contributors.
- PageRank Centrality considers both the number and quality of connections, giving more importance to nodes linked to influential ones. This helps identify authoritative figures and key players in the network.
-

Together, these measures provide a well-rounded view of node influence, capturing both direct connectivity (Degree) and overall importance (PageRank).

4.2 Community Diameters

The diameter of a community refers to the longest shortest path between any two nodes within that community. A smaller diameter indicates that nodes within the community are relatively close to one another, suggesting a more cohesive community. In contrast, a larger diameter may indicate the presence of disconnected subgroups or more sparse relationships between community members.

5 Link Prediction

Traditional link prediction methods for graph data mainly rely on structural features and similarity metrics of the graph. By calculating the similarity between nodes, utilizing path information, or applying statistical models, these methods predict potential links. They are simple and efficient, suitable for various types of graph data. Although they may face challenges in computational efficiency and accuracy when dealing with large and complex graphs, they lay the foundation for more advanced machine learning and deep learning-based approaches.

5.1 Similarity-Based Metrics

These methods predict potential links by calculating similarity scores between pairs of nodes. Common similarity metrics include:

5.1.1 Common Neighbors (CN)

Measures the number of shared neighbors between two nodes. The more common neighbors two nodes have, the higher the likelihood of a link forming between them.

$$\text{CN}(x, y) = |\Gamma(x) \cap \Gamma(y)|$$

where $\Gamma(x)$ denotes the set of neighbors of node x.

5.1.2 Jaccard Coefficient

Measures the ratio of the intersection to the union of the neighbor sets of two nodes, with values ranging from 0 to 1.

$$J(x, y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|}$$

5.2 Introduction of GLACE

With the advancement of deep learning technologies, neural network-based models have shown remarkable performance in link prediction tasks. This report introduces the GLACE (Gaussian Latent Attribute-based Contrastive Embedding) model and its application in link prediction. The GLACE model is a Gaussian-based graph embedding method designed for link prediction tasks. Unlike the LACE model, GLACE learns Gaussian distribution embeddings (mean μ and variance σ) for each node, enabling it to better capture the uncertainty and complex relationships between nodes. The model minimizes the symmetric Kullback-Leibler (KL) divergence between node pairs, aligning their distributions in the embedding space to enhance link prediction accuracy.

5.3 Model Architecture

The main components of the GLACE model include:

- **Input Processing**: Handles the sparse adjacency matrix by converting it into a format suitable for PyTorch sparse tensors.
- **Encoder**: A multi-layer fully connected neural network that extracts latent features of nodes.

- **Mean and Variance Embedding Layers**: Linear layers that generate the mean μ and log variance $\log \sigma$ for node embeddings.
- **Context Encoder**: Used when considering second-order proximity, it generates context embeddings for nodes.
- **Optimizer**: Utilizes the Adam optimizer for training the model parameters.

5.4 Gaussian Embeddings and KL Divergence

GLACE learns Gaussian distribution embeddings for each node, represented as (μ, σ). During link prediction, the symmetric KL divergence between node pairs is computed to measure their similarity. The specific steps are as follows:

1. For a node pair (u_i, u_j), retrieve their means μ_i, μ_j and variances σ_i, σ_j.
2. Calculate the KL divergence $KL(P||Q)$ and $KL(Q||P)$, where P and Q represent the Gaussian distributions of nodes u_i and u_j, respectively.
3. Average the two divergences to obtain the distance metric $KL\_distance$ between the node pair.

5.5 Model Training and Optimization

The training process of the GLACE model involves several key steps:

5.5.1 Loss Function

The model employs a log-sigmoid loss function, suitable for binary classification tasks in link prediction. It is defined as:

$$\text{Loss} = -\frac{1}{N}\sum_{i=1}^{N} \log \sigma(label_i \cdot energy_i)$$

where $label_i$ is the true label (1 for positive samples, -1 for negative samples), and $energy_i$ is the energy value computed by the model (negative KL divergence).

5.5.2 Optimization Process

The Adam optimizer is used to update the model parameters, with the learning rate specified by the experimental setup. The optimization goal is to minimize the loss function, thereby improving the model's performance in link prediction tasks.

5.5.3 Experimental Results

Dataset For our experiments, we utilized the **Cora_ML** dataset, a widely recognized benchmark in the field of link prediction and graph-based learning. The Cora_ML dataset consists of scientific publications classified into various topics, with citation links representing the relationships between these publications. Specifically, the dataset contains 2,708 nodes (publications), 5,429 edges (citations), and 1,433 features representing the presence of specific words in the documents. This dataset is well-suited for evaluating the performance of graph embedding models like GLACE in predicting missing or potential links within the citation network.

Results The GLACE model was trained and evaluated on the Cora_ML dataset over multiple batches. The key performance metrics recorded during the training process include loss, validation AUC (Area Under the Curve), and validation AP (Average Precision). Table 3 presents a summarized view of the results across different training batches, with intermediate batches omitted for brevity.

Table 3: GLACE Model Performance on Cora_ML Dataset

Batch	Loss	Val AUC	Val AP
49	0.303630	0.849437	0.828072
99	0.258379	0.891329	0.872213
149	0.266745	0.910746	0.896074
⋮	⋮	⋮	⋮
1749	0.172694	0.952421	0.949143
1799	0.182493	0.953662	0.948588
1849	0.207456	0.954621	0.950908

Analysis of Results The experimental results on the Cora_ML dataset demonstrate the effectiveness of the GLACE model in link prediction tasks. As observed from Table 3, several key trends emerge:

- **Loss Reduction**: The loss consistently decreases as training progresses, indicating that the model is effectively learning to minimize the discrepancy between predicted and actual links. For instance, the loss decreased from 0.303630 at batch 49 to 0.172694 at batch 1749.
- **Performance Metrics**: Both validation AUC and validation AP show an overall upward

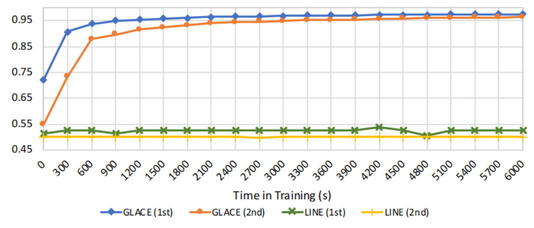


Figure 7: GLACE's link prediction performance

trend, reaching values above 0.95 towards the later batches. This signifies that the model's ability to distinguish between positive and negative links improves with training. For example, the validation AUC increased from 0.849437 at batch 49 to 0.954621 at batch 1849, and the validation AP similarly rose from 0.828072 to 0.950908.

- **Stability of Training**: The training process progresses smoothly without significant interruptions or delays, ensuring a steady training flow. The consistent decrease in loss and increase in performance metrics reflect the model's stable and effective learning dynamics.

Overall, the GLACE model exhibits robust performance in link prediction on the Cora_ML dataset, achieving high accuracy and precision. The incorporation of Gaussian embeddings allows the model to capture nuanced relationships between nodes, resulting in superior predictive capabilities compared to traditional methods.

6 Visualization System

This study introduces a comprehensive visualization system designed to represent a paper citation network, where nodes correspond to academic papers and edges represent citations. The system is organized into five main sections: community network, bibliometric data, degree distribution, node distribution, and additional metrics. An overview of the system is shown in Figure 1.

Given the modularity of author communities (0.66) and paper communities (0.99), the focus is placed on paper-level data, which offers a clearer and more structured representation of the citation network. The higher modularity of paper communities facilitates more detailed analysis. For optimal visualization, data is derived from the top 10 paper communities and the 50 highest centrality papers, ensuring the most influential communities and papers are prominently displayed. This approach enhances both clarity and effectiveness.

6.1 Top Left: Community Network Visualization with Interactive Features

The top-left section visualizes the paper communities using an interactive network graph. Nodes represent individual papers, and edges indicate citations. Users can zoom, pan, and select nodes, which dynamically updates the corresponding sections (top-right and bottom-middle) with community-specific data. A year slider allows users to track the temporal evolution of the citation network, highlighting changes in structure over time.

6.2 Top Right: Bibliometric Data Table

The top-right section presents a dynamic bibliometric table containing key attributes such as paper ID, publication year, venue, references, and citation count. This table updates in response to node selections from the network graph, ensuring that users view relevant bibliometric data for the selected community. Pagination and dynamic updates allow efficient browsing of large datasets.

6.3 Bottom Left: Dynamic Bar Chart Visualization

The bottom-left section features a dynamic bar chart for metrics such as "Average Citations," "Average Centrality," and "Diameter." Users can select a metric from a dropdown menu, and the bar chart visually compares these metrics across communities. The chart includes smooth transitions, tooltips, and highlighted bars on hover, enhancing interactivity.

6.4 Bottom Middle: Degree Distribution Visualization

The bottom-middle section visualizes the degree distribution of communities within the network using a smooth line chart with spline curves. This chart depicts the frequency of nodes with each degree. Upon selecting a node in the network graph, the degree distribution for the corresponding community is displayed dynamically. Tooltips and adjustable dot sizes further improve visibility, and degree values exceeding a specified threshold are grouped for clarity.

6.5 Bottom Right: Node Distribution with Donut Chart

The bottom-right section uses a donut chart to show the distribution of nodes across communities. Each slice represents a community, with its size proportional to the number of nodes. Interactive features such as hover effects and percentage labels allow users to explore the relative sizes of the communities and gain insights into the network's composition.

6.6 System Overview and Interactivity

The visualization system integrates these components into a cohesive interface that enables detailed exploration of the citation network. Key interactive features include zooming, panning, node selection, and year slider adjustments. Selecting a node updates the bibliometric data and degree distribution sections with community-specific information. The year slider adds a temporal dimension, revealing the evolution of the network over time. Powered by the `D3.js` library, the system ensures responsive, high-quality visualizations and an engaging user experience.

7 Conclusion

Acknowledgements