Optimizing GNN to Solve Community Detection in Social Networks

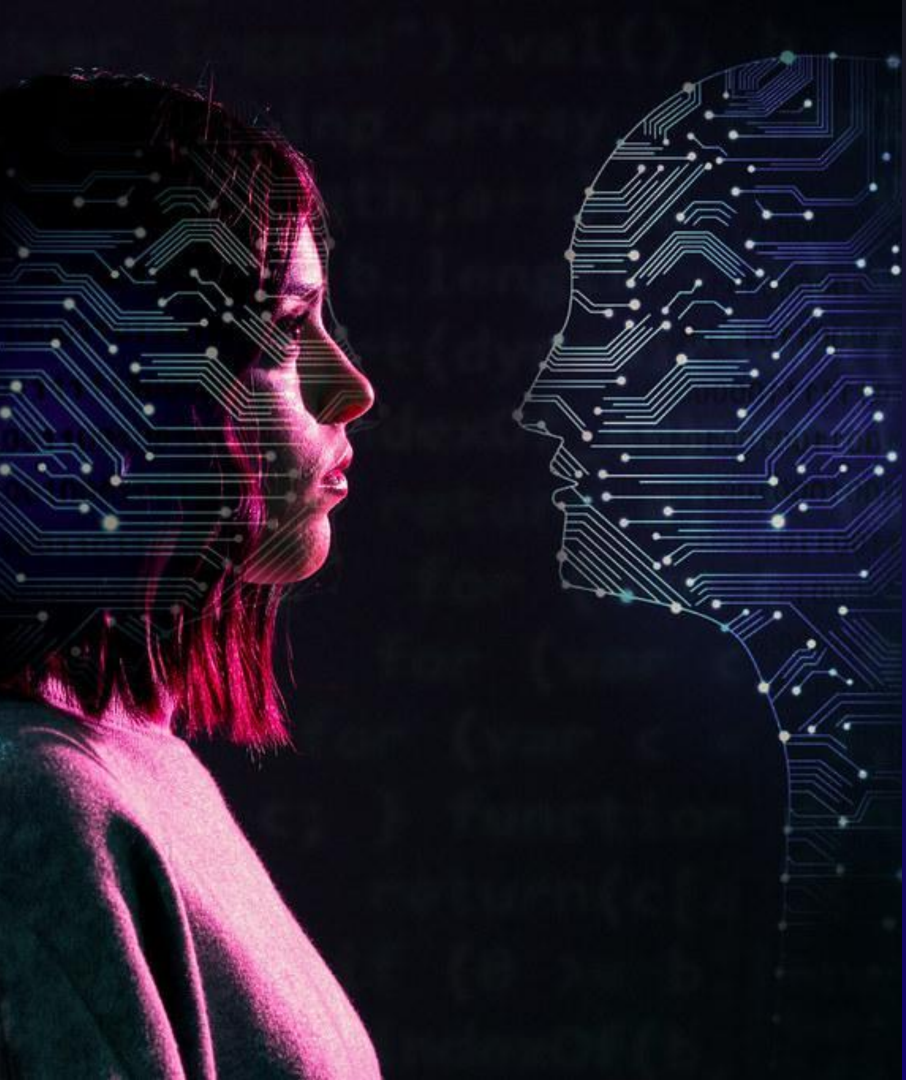
Abdulmoumen Al-Atrash | Parsa Kamalipour | Aniket Roy

TABLE OF CONTENTS

01.

PURPOSE STATEMENTS

Community Detection definition and literature



INTRODUCTION

Community Detection:

- **Definition:** Identifying clusters (communities) in a network where nodes are more densely connected internally than with the rest of the network.

- **Applications:** Social network analysis, biology, and information retrieval, etc.

Major Challenges & Problems

1. Scalability issues in handling large-scale networks.

2. Limitations in detecting detailed community structures with traditional methods.

Related Work

Optimization Methods

Techniques like Learning-Based Genetic Algorithms (LGA) and Quantum-Inspired Optimization achieve impressive accuracy but face challenges in scaling across diverse network types.

Modularity-Based Methods

Known for simplicity and interpretability. However, these methods rely heavily on modularity maximization, which often limits accuracy in detecting varying community structures.

Deep Learning Methods

Advanced models, such as Graph Neural Networks (GNNs), excel at capturing complex relationships but can be computationally demanding for large-scale networks.

> Our Goal: Leverage GNNs to tackle community detection while addressing the shortcomings of existing machine learning approaches.

02.

METHODOLOGY

Trying GNN and Exploring Modifications

Neighbor Sampling

Approach: Sample a fixed number of neighbors for each node during training.

Benefits:

- Reduces computational overhead.
- Enables training on larger graphs that cannot fit entirely in memory.
- Introduces stochasticity, helping prevent overfitting.

Subgraph Partitioning

Approach: Divide the large graph into smaller, manageable subgraphs (clusters).

Benefits:

- Facilitates parallel processing of subgraphs.
- Decreases memory requirements during training.
- Preserves local graph structures within clusters.

Efficient Graph Convolutions - GraphSAGE (Graph Sample and Aggregate)

Approach: Aggregates feature information from a sampled set of neighbors.

Benefits:

- Inductive Capability: Generalizes to unseen nodes or graphs.
- Efficiency: Limits neighborhood size, preventing computational explosion.
- Applicability: Suitable for dynamic graphs where nodes/edges change over time.

Datasets Utilized for Evaluation

- Stochastic Block Models (SBM)
  - Purpose: Synthetic benchmark for community detection.
  - Configuration:
    - Nodes: 1,000
    - Communities: 4
    - Intra-community Edge Probability (p_intra): 0.05
    - Inter-community Edge Probability (p_inter): 0.005
    - Features: 16-dimensional random vectors
  - Advantages: Controlled environment to assess baseline performance and scalability.

- CORA Dataset
  - Description: Real-world citation network.
  - Nodes: 2,708 scientific publications.
  - Edges: 5,429 citation links.
  - Features: 1,433-dimensional bag-of-words vectors.
  - Classes: 7 research topics.
  - Significance: Represents complex, large-scale networks with inherent noise and irregular patterns.

03.

Results

Results Looking Good!

Comparison on SBM

Model	Accuracy (Change from GCN)	Training Time (Change from GCN)
Graph Convolutional Network (GCN)	0.83	1.26s
GCN with Neighbor Sampling	0.29 (-0.54)	13.11s (+11.85s)
Graph Sample and Aggregate with Neighbor Sampling	0.83 (+0.00)	12.29s (+11.03s)
GCN with Subgraph Partitioning	0.84 (+0.01)	2.21s (+0.95s)
Graph Sample and Aggregate Network	0.96 (+0.13)	0.96s (-0.30s)
Graph Sample and Aggregate with Subgraph Partitioning	0.90 (+0.07)	2.03s (+0.77s)

Observation:

- **Accuracy:** The Graph Sample and Aggregate Network achieves the highest accuracy (+0.13) compared to GCN.
- **Training Time**: GCN is faster than most models, except the Graph Sample and Aggregate Network, which is the fastest.

Comparison on CORA

Model	Accuracy (Change from GCN)	Training Time (Change from GCN)
Graph Convolutional Network (GCN)	0.88	1.47s
GCN with Neighbor Sampling	0.88 (+0.00)	22.23s (+20.76s)
Graph Sample and Aggregate with Neighbor Sampling	0.86 (-0.02)	43.76s (+42.29s)
GCN with Subgraph Partitioning	0.89 (+0.01)	3.79s (+2.32s)
Graph Sample and Aggregate Network	0.90 (+0.02)	5.00s (+3.53s)
Graph Sample and Aggregate with Subgraph Partitioning	0.89 (+0.01)	7.40s (+5.93s)

Observation:

- **Accuracy:** The Graph Sample and Aggregate Network achieves slightly higher accuracy (+0.02).
- **Training Time:** GCN remains the most efficient in training time, with Neighbor Sampling and Subgraph methods being significantly slower.

Complete Table

Model	Dataset	Accuracy	Training Time
Graph Convolutional Network	SBM	0.83	1.26s
Graph Convolutional Network	CORA	0.88	1.47s
Graph Convolutional Network with Neighbor Sampling	SBM	0.29	13.11s
Graph Convolutional Network with Neighbor Sampling	CORA	0.88	22.23s
Graph Sample and Aggregate Network with Neighbor Sampling	SBM	0.83	12.29s
Graph Sample and Aggregate Network with Neighbor Sampling	CORA	0.86	43.76s
Graph Convolutional Network with Subgraph Partitioning	SBM	0.84	2.21s
Graph Convolutional Network with Subgraph Partitioning	CORA	0.89	3.79s
Graph Sample and Aggregate Network	SBM	0.96	0.96s
Graph Sample and Aggregate Network	CORA	0.9	5.00s
Graph Sample and Aggregate Network with Subgraph Partitioning	SBM	0.9	2.03s
Graph Sample and Aggregate Network with Subgraph Partitioning	CORA	0.89	7.40s

Key Takeaways

1. Best Accuracy:
   - SBM: Graph Sample and Aggregate Network (0.96)
   - CORA: Graph Sample and Aggregate Network (0.90)
2. Fastest Training:
   - SBM: Graph Sample and Aggregate Network (0.96s)
   - CORA: Graph Convolutional Network (1.47s)
3. Trade-Offs:
   - Neighbor Sampling methods generally increase training time without significantly improving accuracy.
   - Subgraph Partitioning adds some overhead but improves accuracy and training efficiency.

Key Takeaways for SBM and CORA

1. GCN Strengths:
   - Maintains competitive accuracy.
   - Very efficient in training time.
2. Graph Sample and Aggregate:
   - On SBM, it outperforms all models in accuracy and training speed.
   - On CORA, it achieves the best accuracy but takes slightly longer to train.
3. Neighbor Sampling Methods:
   - Generally introduce longer training times without significant accuracy improvements.
4. Subgraph Partitioning:
   - Provides a reasonable balance between accuracy and training time, making it a solid enhancement for both datasets.

04.

CONCLUSIONS

What did we achieve?

Recap

Goal

To overcome the limitations present in traditional community detection approaches

How?

Advanced machine learning methods: Graph Neural Network

Why?

Struggle with generalizability and the ability to adapt to networks of varying sizes and structures

ML increases both accuracy and scalability, which aims at making community detection more useful for large, complex networks.

What Was the benefit of this project?

A chance to understand network structures better and how advanced machine learning models help improve community detection

THANKS!

Hope this presentation was useful to you :D