Analyzing GNN Models for Community Detection Using Graph Embeddings: A Comparative Study

Devesh Kumar V V[1], $SreesankarS$[2], $KrishnpriyaDinesan$[3], $PranavBNair$[4], $DeepthiLR$[5]
Department of Computer Science and Engineering
Amrita School of Computing
Amrita Vishwa Vidyapeetham
Amritapuri, India
deveshkumarvv2002@gmail.com[1], $sreesankar098@gmail.com$[2], $krishnpriyadinesan@gmail.com$[3],
pranavbnair09@gmail.com[4], $deepthilr@am.amrita.edu$[5]

***Abstract*—Graphs are effective models for illustrating complex connections in various contexts, including social networks, citation networks, and biological systems. In this study, the performance of Graph Neural Networks(GNN) for two main tasks is being explored: Node classification and Community detection. Community detection on complex networks is an emerging area that helps to understand the underlying structures and functions of interconnected systems across various domains. The structural and semantic information are captured using the various GNN models. This paper uses four GNN models for node classification: GCN, GAT, GraphSAGE, and Deep Graph Infomax. The embeddings of these models are clustered to detect the communities by using spectral clustering, k-means clustering, and the Louvain algorithm. The GNN model's performance was assessed in terms of computational complexity, accuracy of clustering, and modularity. The results demonstrate that GNN models, particularly GCN, yield competitive performance in community detection tasks on citation networks. The effectiveness of each model varies across different datasets and community detection algorithms.**

***Index Terms*—community detection, clustering, graph neural networks, graph convolutional networks, and graph attention networks.**

I. Introduction

The study of complex networks, including biological, social, and citation networks, has gained attention in the last few years from a variety of disciplines, including computer science, biology, and social sciences. Determining influential nodes, extracting valuable insights, and revealing hidden patterns all require understanding these networks' structure and organization. These find applications in anomaly detection, recommendation systems, and social network analysis. One of the main tasks in network analysis is community detection, which divides nodes into coherent groups or communities according to patterns of connectedness. Traditional community detection algorithms often struggle to capture the complex relationships among the edges, Graph Neural Networks (GNNs) are a powerful class of models that can represent the complex linkages and structures that are present in graph data. Traditional Neural networks are not well suited for processing graph data, given the random and complex topology of graph data, suggesting the absence of spatial locality. It is further made more difficult by the unfixed node ordering. To overcome these drawbacks, GNNs allow representations to be learned for every node in a graph that combines topological structure and node properties.GNNs have been applied widely in a variety of fields, including citation networks, social networks, biological networks, and recommendation systems. Message passing, or the repetitive propagation of information between adjacent nodes in the graph, is the fundamental idea behind GNNs. This allows nodes to generate rich feature representations that capture both local and global graph structures by aggregating data from their immediate surroundings. GNNs are capable of capturing more intricate patterns and relationships inside the graph by utilizing many layers of message passing.

Several GNN models have emerged to learn node embeddings for graphs. In this paper, a comparative analysis of four cutting-edge GNN techniques is done: Graph Convolutional Networks (GCN), Graph Attention Networks (GAT), Deep Graph Infomax, and GraphSAGE. To sum up, all techniques proposed novel ways to learn from graph-structured data. The data from nodes in the neighbourhood is captured differently and the global features of the graph are extracted using various ways to aggregate data. The GNN techniques embeddings, including GCN, GAT, Deep Graph Infomax, and GraphSAGE are used to get the results of the traditional community detection algorithms: Spectral, K-means, and Louvain. We proposed using the traditional algorithms to merge the best forms of both approaches. Our goal is to achieve a comprehensive comparative analysis of the performance properties of different GNN techniques and clustering algorithms. The experimental results offer the main takeaways for the selection of a dataset and method in practice.

II. Related Work

Both academic and industrial domains have had a steep interest in Graph Neural Networks (GNNs) due to their high efficiency in modelling and analyzing graph-structured data. There have been several research papers which have worked on analysing GNNs in various applications as well as highlighting the advantages and disadvantages of each. [1] Several of these include comparative analysis of the multiple GNN designs

like introduced Graph Convolutional Networks (GCN) demonstrated the work on semi-supervised node classification tasks. [1] More recent research done by Veličković [2] introduces Graph Attention Networks (GAT), in this attention mechanism, these capture fine-grained graph patterns, in turn, prioritize data from closer nodes. Various tasks and datasets when evaluating the GNN performance give contrasting metrics. To add-on graph-level metrics like density, modularity, and conductance are used. For classification tasks metrics like accuracy, precision, recall, and F1-score are used. The choice of the right metrics still poses a problem, particularly when underlying graph structures are extremely dynamic or varied. [3]

Apart, from GNNs, graph clustering techniques play a role, in uncovering community structures within networks. Graph Laplacians' eigenstructure is utilised by spectral clustering techniques, like the ones put out by von Luxburg [4], to divide graphs into coherent clusters. In the meanwhile, the popular centroid-based approach K-means clustering provides scalability and simplicity, however, its performance can be affected by initialization and the number of clusters selected (MacQueen) [5]. The integration of GNNs with conventional clustering algorithms has also been investigated recently to make use of the complementing advantages of both strategies. For community detection in social networks, Zhang et al. [6] developed a framework that integrates spectral clustering and graph convolutional networks, showing better performance than standalone techniques. Similarly, Liu et al. [7] achieved competitive results on benchmark datasets by integrating K-means clustering with Graph Neural Networks for node classification tasks.

Even though understanding and utilising the power of GNNs and clustering algorithms has advanced significantly, a number of issues still need to be resolved. These include interpretability of learnt representations, robustness to noisy or missing data, and scalability challenges with large-scale graphs. Taking on these problems offers fascinating prospects for graph analytics and machine learning research in the future.

III. Methodology

A. Graph Neural Network Techniques

Using graph convolutional layers to spread information throughout the graph, we applied GCN. The implementation of GATs included attention methods to aggregate input from nearby nodes alone. The goal of the unsupervised graph representation learning technique known as Deep Graph Infomax is to maximize the mutual information between graph topologies and node representations. Based on Hamilton et al. methodology, GraphSAGE was put into practice by combining data from selected neighbourhood nodes to create node embeddings through the use of inductive learning techniques.
The high-Level Design of the Experiment is shown in the Fig 1

1) Graph Convolutional Network (GCN): Designed to function on graph-structured data, Graph Convolutional Networks (GCNs) is a collection of deep learning models that

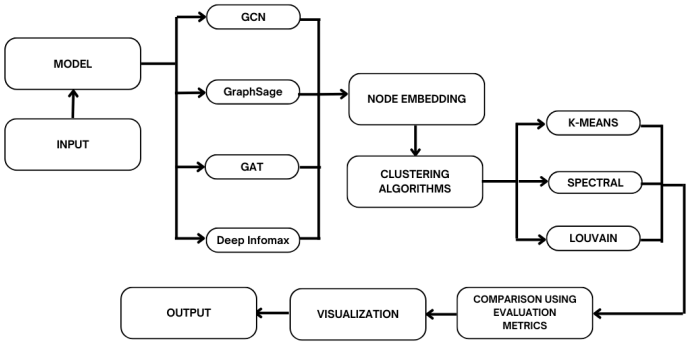

Fig. 1. Proposed Methodology

provide significant insights and predictions over a range of disciplines, including social networks, recommendation systems, and biological networks. Fundamentally, GCNs take advantage of the interconnections and underlying structures found in graphs to derive meaningful depictions of nodes.
An attributed network is represented as $G = (V, E, A)$ where V denotes the set of nodes, E is, the set of edges and A represents the attributes associated with the nodes. GCNs execute convolutions on the graph data, much like convolutional layers do in image processing. Unlike images, graphs do not have a fixed structure, and additional convolutional methods are needed to manage irregular connectivity, unlike conventional grids in images. The convolutional layer in GCN aggregates information from a node's neighbours to update its representation.

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}) \tag{1}$$

where:

- $H^{(l)}$ is the feature matrix at layer l.
- $\tilde{A} = A + I$ is the adjacency matrix with self-connections added (I is the identity matrix).
- $\tilde{D}$ is the degree matrix of $\tilde{A}$, where $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$.
- $W^{(l)}$ is the weight matrix at layer l.
- σ is an activation function, typically ReLU or similar.

2) Graph Attention Network (GAT): One kind of neural network intended for learning on graph-structured data is the Graph Attention Network (GAT). It incorporates attention techniques to enable nodes to selectively aggregate information from their neighbours, making graph representations flexible and versatile. The attention mechanism in GAT computes attention coefficients for each neighbour of a node based on the node's features and the features of its neighbours. It then aggregates neighbour features weighted by these attention coefficients. Let's consider a node i in a graph with neighbours j. The attention mechanism computes attention coefficients.
Attention Mechanism:

$$\alpha_{ij} = \text{softmax}(\text{LeakyReLU}(\vec{a}^T[W\vec{h}_i \| W\vec{h}_j])) \tag{2}$$

Where:

- $\vec{h}_i$ and $\vec{h}_j$ are the feature representations of nodes i and j respectively.
- $[W\vec{h}_i \| W\vec{h}_j]$ denotes concatenation of the transformed feature vectors of nodes i and j with weight matrix W.
- $\vec{a}$ is a shared learnable parameter vector.
- LeakyReLU is the leaky rectified linear unit activation function.
- softmax computes the softmax function to normalize the attention coefficients across neighbours of node i.

3) GraphSAGE: Using a graph neural network architecture called GraphSAGE (Graph Sample and Aggregation), node embeddings are learned by sampling and aggregating data from each node's vicinity. Using subgraphs rather than the full graph makes learning on large-scale graphs scalable and effective. The GraphSAGE algorithm performs two tasks: Sampling and Aggregation.

1) **Sampling**: For each node v in the graph, a fixed-size neighbourhood subgraph G_v is sampled by performing random walks or other neighbourhood sampling techniques.
2) **Aggregation**: Features from the sampled neighbourhood subgraph G_v are aggregated to compute the updated embedding for node v. This aggregation process is typically done using a neural network, such as a mean aggregator or a pooling aggregator.

$$h_v^{(l+1)} = \text{AGG}\left(\{h_u^{(l)}, \forall u \in \mathcal{N}(v)\}\right) \tag{3}$$

where:

- $h_v^{(l)}$ represents the embedding of node v at layer l.
- $h_v^{(l+1)}$ is the updated embedding of node v at layer $l+1$.
- $\mathcal{N}(v)$ is the set of neighbours of node v in the sampled subgraph.
- AGG is the aggregation function, which aggregates embeddings of neighbouring nodes to compute the updated embedding of node v.

4) Deep Graph Infomax: A method, of learning, known as Deep Graph Infomax is used to learn graph representations. By maximizing the information between the structures of graphs and node representations the model can recognize connections and patterns, within the graph. As Deep Graph Infomax does not require explicit supervision to develop informative representations, it is especially helpful in situations when labelled data is limited or nonexistent.

The output feature from each of these models is fed to three clustering algorithms that make it easier to find significant groupings or clusters. Clustering algorithms can effectively divide the feature space into discrete clusters, exposing innate relationships and characteristics, by utilizing the model outputs, which contain important information about the underlying data. This approach emphasizes the combination of unsupervised learning methods and predictive modelling, providing a thorough framework for finding patterns and insights in large, complicated datasets.

B. Graph Clustering Algorithms

Based on the spectrum characteristics of the graph Laplacian matrix, spectral clustering was used to divide the datasets into disjoint clusters. As a centroid-based partitioning approach, K-means clustering was applied, which iteratively assigned nodes to clusters based on closeness to cluster centroids. Using Louvain clustering, a measure of the quality of the resulting partitions, communities within the graphs were found by optimising modularity.

1) Spectral Clustering: The graph-based clustering method known as "spectral clustering" makes use of the Laplacian matrix of a graph's eigenvalues and eigenvectors. Using spectral embedding, it converts the data into a lower-dimensional space where clusters are easier to distinguish and isolate. This method works well with graph data that have complicated structures because it may be used to discover clusters with complex geometries or non-linear separability.

2) K-means Clustering: By repeatedly assigning each data point to the closest cluster centroid, the centroid-based K-means clustering algorithm divides data into K clusters. By reducing the within-cluster sum of squares, it successfully divides data points into clusters with the least amount of intra-cluster variance. K-means works well with huge datasets that contain numerical features since it is simple to implement and computationally efficient. It doesn't work well with non-globular or non-convex clusters, though, and it needs to know the number of clusters (K) ahead of time.

3) Louvain Algorithm: A modularity-based methodology for community discovery in networks is the Louvain method. It maximizes the modularity quality function, which quantifies how strongly a network is divided into communities according to the density of connections inside communities relative to connections between communities. There are two stages to the Louvain method: locally optimizing modularity within communities and globally optimizing modularity through iterative community mergers or splits. It is a well-liked option for community identification jobs across a variety of fields because of its scalability and capacity to identify communities in massive networks.

IV. Experimentation Results And Observations

The performance of four popular GNN architectures: Deep Graph Infomax, GraphSAGE, Graph Convolutional Networks (GCN), and Graph Attention Networks (GAT) is analyzed in this section. Furthermore, the evaluation of the effectiveness of three clustering algorithms: the Louvain algorithm, spectral clustering, and k-means clustering, in removing communities from the embeddings produced by these models is done in this paper.

A. Dataset

In this study, GCN, GAT, GraphSAGE, and Deep Graph Infomax are compared on PubMed (19,717 nodes, 44,338

edges), Citeseer (3,312 nodes, 4,732 edges), and Cora (2,708 nodes, 5,429 edges). In graph-based machine learning, these datasets serve as important benchmarks, providing a wide range of scholarly and biomedical literature to assess model performance.

B. Experimental Setup

PyTorch and NetworkX are two libraries that we used to create GNNs and manipulate graphs in our Python-based framework. Using grid search or random search strategies, performance metrics on validation sets were optimized for GNN algorithms. The full network or selected subgraphs were subjected to clustering techniques, and the resolution parameter for Louvain clustering and the elbow method for K-means were used to calculate the number of clusters (K value).

C. Evaluation Metrics

Classification accuracy was calculated for GNN techniques to assess node classification task performance. The quality of the community structures found by clustering techniques was gauged by their modularity. By calculating conductance, which is the ratio of edges exiting clusters to total cluster volume, graph partition quality was evaluated.

1) Modularity: A quality function called modularity computes how firmly a network is divided into communities within communities relative to connections between communities by looking at the density of connections. It determines the degree to which a particular graph partitioned into communities is superior to a random partition. Stronger community structures with sparse links between communities and dense connections within them are shown by higher modularity scores. Modularity is an objective function that should be enhanced in the clustering condition to divide the graph into meaningful clusters and make it effortless to find logical communities in the given data.

$$\text{Modularity} = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(c_i, c_j) \tag{4}$$

2) Conductance: The proportion of edges exiting clusters to the total volume of the clusters is the key indicator of conductance, It is a metric used to determine the quality of graph partitions. Lower conductance values indicate tighter and more internally connected clusters. It measures the connection between clusters. High conductance values point to a high number of edges spanning cluster boundaries, which is indicative of a poor-quality partition. By assessing how well-separated and internally cohesive a cluster is, conductance can be used to evaluate the success of partitioning algorithms in the context of clustering. This can provide valuable insights into the structure and coherence of the resulting partitions.

$$\text{Conductance} = \frac{\sum_{i \in S, j \notin S} w(i,j)}{\min(\text{vol}(S), \text{vol}(\bar{S}))} \tag{5}$$

D. GNN Models Performance

Using the Cora, Citeseer, and PubMed datasets, we first assess each GNN model's node classification performance. Each model's accuracy ratings are used as a starting point to evaluate how well it can represent nodes in the graphs and extract relevant characteristics. A comparison Graph of the Accuracy of the GNN models is shown in "Fig. 1".We create graphs showing the relationship, between loss and epochs for each model and dataset pairing to gain insights, into how the training progresses. These visual representations help us grasp the performance patterns and convergence behaviour of GNN models thoroughly.

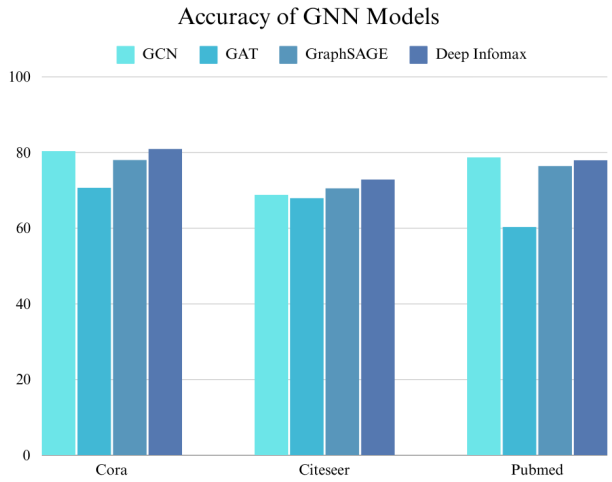

Fig. 2. Accuracy of GNN Models

TABLE I
MODEL EVALUATION METRICS

MODEL	DATASET	ACCURACY	RECALL	PRECISION
GCN	Cora	0.806	0.8206	0.787
	Citeseer	0.686	0.6659	0.6629
	Pubmed	0.784	0.7837	0.7791
GAT	Cora	0.725	0.7646	0.7274
	Citeseer	0.665	0.637	0.6344
	Pubmed	0.668	0.695	0.7081
GraphSAGE	Cora	0.786	0.8071	0.7649
	Citeseer	0.707	0.6829	0.6761
	Pubmed	0.766	0.7661	0.7616
Deep Graph Infomax	Cora	0.791	0.791	0.8118
	Citeseer	0.731	0.731	0.7235
	Pubmed	0.783	0.783	0.7936

1) GCN:

- In the Cora dataset GCN achieved an accuracy of 80.40% showcasing its effectiveness, in classifying nodes within citation networks.
- However In Citeseer the Performance of GCN was comparatively Less when compared to Cora. In the Citeseer dataset, GCN got an accuracy of 68.90%.
- In the Pubmed dataset, GCN achieved an accuracy of 78.80%.GCN got the Highest Accuracy Score in Pubmed Dataset when compared to the other models.

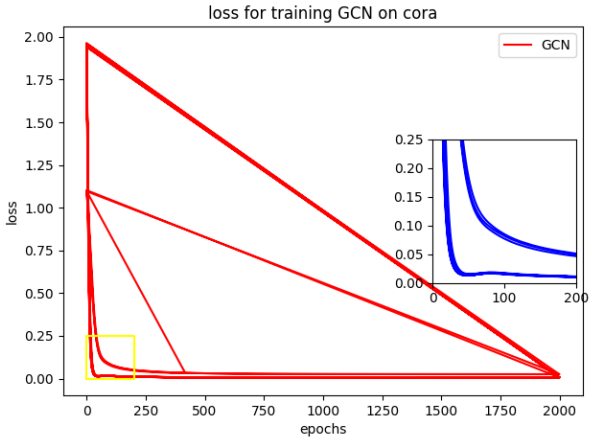

Fig. 3. Loss Vs Epochs For GCN on Cora

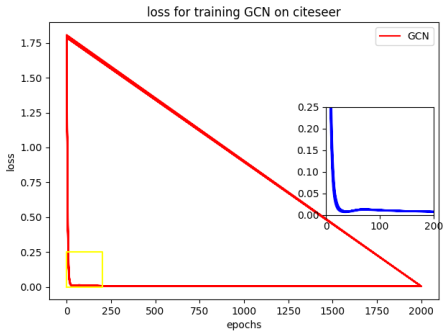

Fig. 4. Loss Vs Epochs For GCN on Citeseer

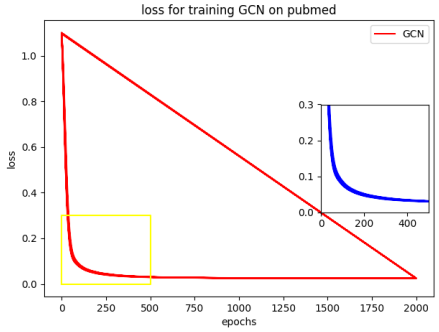

Fig. 5. Loss Vs Epochs For GCN on Pubmed

2) GAT:

- GAT performed less accurately on all three datasets than GCN. GATs accuracy on Cora was a reasonable 70.70%, while it performed far worse on Citeseer (68.00%) and Pubmed (60.40%).
- The attention mechanism of GAT may be the cause of its comparatively poorer performance in some domains, since it may find it difficult to efficiently extract informative features from nearby nodes.

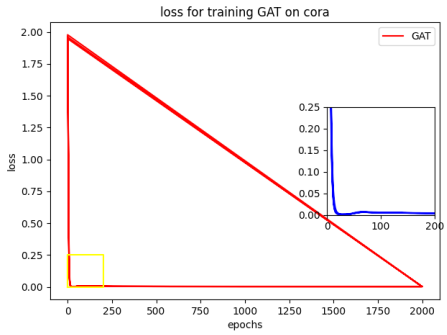

Fig. 6. Loss Vs Epochs For GAT on Cora

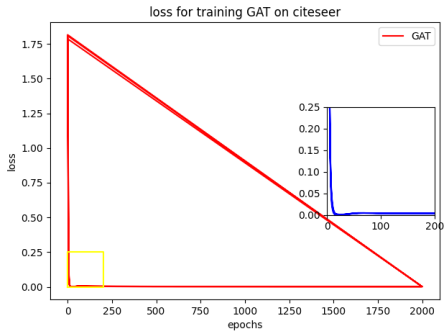

Fig. 7. Loss Vs Epochs For GAT on Citeseer

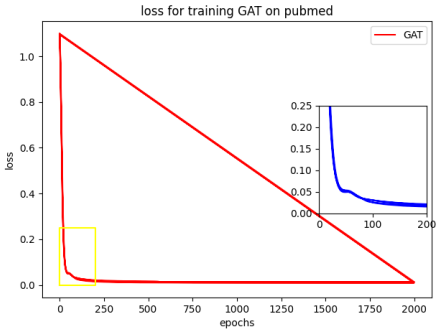

Fig. 8. Loss Vs Epochs For GAT on Pubmed

3) GraphSAGE:

- With accuracy scores of 78.10%, 70.60%, and 76.50% on Cora, Citeseer, and Pubmed, respectively, GraphSAGE showed competitive performance across all three datasets.
- GraphSAGE's strong performance was probably aided by its capacity to create node embeddings by combining data from a sample of nearby nodes, especially in situations when the graph structures were different.

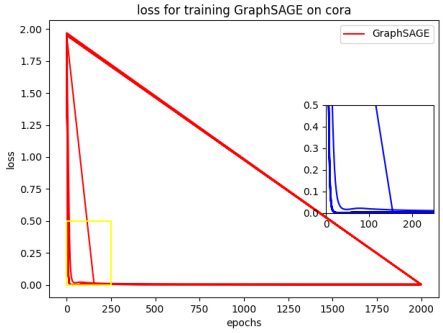

Fig. 9. Loss Vs Epochs For GraphSAGE on Cora

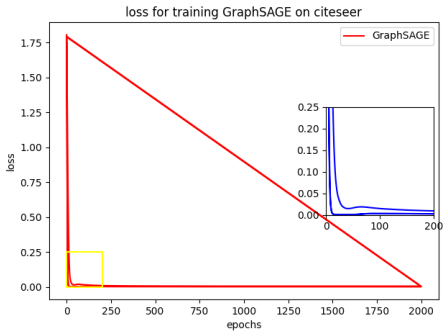

Fig. 10. Loss Vs Epochs For GraphSAGE on Citeseer

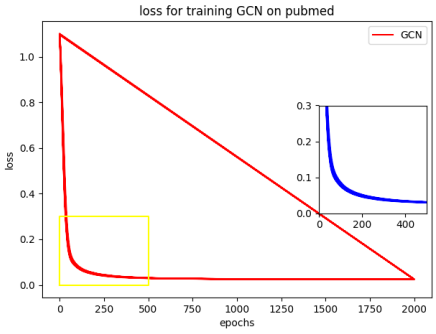

Fig. 11. Loss Vs Epochs For GraphSAGE on Pubmed

4) Deep Graph Infomax:

- Out of all the GNN techniques examined in this study, Deep Graph Infomax obtained the highest accuracy on the Cora dataset (81.01%). This demonstrates how well-unsupervised learning techniques, like Deep Graph Infomax, can capture insightful graph representations.
- Similar to this, Deep Graph Infomax showed its capacity to learn useful representations from graph-structured data across several domains by demonstrating competitive performance on the Citeseer (72.92%) and Pubmed (78.00%) datasets.

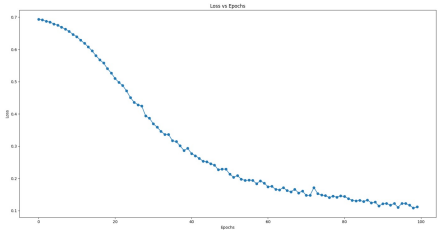

Fig. 12. Loss Vs Epochs For Deep Graph Infomax on Cora

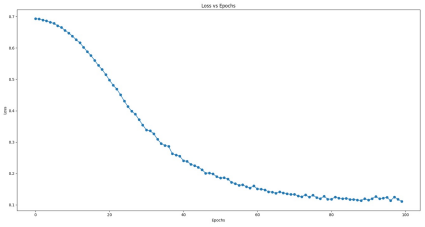

Fig. 13. Loss Vs Epochs For Deep Graph Infomax on Citeseer

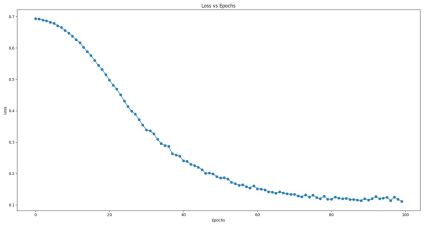

Fig. 14. Loss Vs Epochs For Deep Graph Infomax on Pubmed

E. Comparing the GNN Techniques

- Our findings show that Deep Graph Infomax and GCN fared better in terms of accuracy over the three datasets than other GNN techniques. Leveraging unsupervised learning approaches, Deep Graph Infomax demonstrated impressive performance in developing informative representations and capturing hidden graph structures.

- Although GCN demonstrated competitive accuracy scores, its performance differed throughout datasets, indicating possible difficulties in generalizing across other domains.
- Strong performance was shown by GraphSAGE on all datasets, demonstrating how well inductive learning methods work to produce node embeddings for a variety of graph architectures.
- GAT demonstrated comparatively lower accuracy when compared to other approaches while incorporating attention mechanisms for adaptive feature aggregation. This suggests that GAT may have limits in specific domains or datasets.

F. Effectiveness of Community Detection

We use clustering techniques to extract communities from the embeddings produced by the GNN models after node categorization. In particular, we apply the Louvain method, spectral clustering, and k-means clustering to divide the nodes into discrete communities according to their feature representations.

G. Evaluation Metrics

To evaluate the quality of the discovered communities, we compute a set of evaluation measures for each clustering algorithm. Modularity and conductance are some of these metrics. The degree to which the communities are internally dense with few links between them is measured by their modularity. Conductance measures the percentage of edges connecting nodes throughout various communities.

TABLE II
RESULTS OF EVALUATION METRICS FOR GCN

Graph Convolutional Network (GCN)			
Dataset	Clustering	Modularity	Conductance
Cora	K-Means	0.137419983	0.999621069
	Spectral	0.957516506	0.298917
	Louvain Algorithm	0.878873655	0.028193795
Citeseer	K-Means	0.156777612	0.999780316
	Spectral	0.908841901	0.6382432
	Louvain Algorithm	0.937362782	0.005153839
Pubmed	K-Means	0.48425385	0.999966158
	Spectral	0.992438492	0.4782737
	Louvain Algorithm	0.848046833	0.848046833

TABLE III
RESULTS OF EVALUATION METRICS FOR GAT

Graph Attention Network (GAT)			
Dataset	Clustering	Modularity	Conductance
Cora	K-Means	0.3984	0.5671
	Spectral	0.4144	0.2427
	Louvain Algorithm	0.8098	0.0491
Citeseer	K-Means	0.0027	0.7025
	Spectral	0.0023	0.7809
	Louvain Algorithm	0.8871	0.0607
Pubmed	K-Means	0.3876	0.592
	Spectral	0.3865	0.4319
	Louvain Algorithm	0.7672	0.2446

TABLE IV
RESULTS OF EVALUATION METRICS FOR GRAPHSAGE

GraphSAGE			
Dataset	Clustering	Modularity	Conductance
Cora	K-Means	0.527893925	0.003196712
	Spectral	0.141992916	0.5
	Louvain Algorithm	0.8146	0.0288
Citeseer	K-Means	0.581548943	0.00301365
	Spectral	0.198883264	0.5
	Louvain Algorithm	0.8897	0.0043
Pubmed	K-Means	0.427404355	0.000722478
	Spectral	0.238930523	0.5
	Louvain Algorithm	0.7726	0.1435

TABLE V
RESULTS OF EVALUATION METRICS FOR DEEP GRAPH INFOMAX

Deep Graph Infomax			
Dataset	Clustering	Modularity	Conductance
Cora	K-Means	0.8631	0.0968
	Spectral	0.8631	0.0013
	Louvain Algorithm	0.9171	0.5834
Citeseer	K-Means	0.9303	0.0518
	Spectral	0.9303492	0.00109
	Louvain Algorithm	0.968	0.49094
Pubmed	K-Means	0.7431	0.0898
	Spectral	0.8421	0.00101
	Louvain Algorithm	0.9092	0.51094

Tables II to V show the outcomes of the evaluation metrics, which include conductance and modularity, for each algorithm.

In particular, the metrics for GCN are shown in Table II, those for GAT in Table III, GraphSAGE in Table IV, and Deep Graph Infomax in Table V.

H. Discussion

- Our experimental results provide several important conclusions about how well GNN models and clustering algorithms work in community discovery tasks. First, we note differences in node classification accuracy between various datasets and GNN architectures, suggesting that model complexity and dataset properties affect classification performance. Furthermore, our investigation shows how the choice of clustering technique affects the results of community discovery, with different algorithms displaying different advantages and disadvantages.
- Additionally, we address the scalability and computational efficiency of the techniques used while considering runtime and memory needs. Moreover, we also investigate how interpretable GNN-based community detection is, by evaluating how robust the identified communities match domain-specific knowledge or ground truth labels.

I. Future Directions and Implications

- Our findings give the importance of considering domain-specific factors and dataset attributes when selecting GNN algorithms for graph analysis tasks.

- For improving performance over a wide range of datasets and domains future research should go through hybrid systems that combine the benefits of many GNN methods or implement ensemble methods.
- Furthermore, investigating the interpretability and resilience of GNN approaches in real-world scenarios remains an important path for growing the graph analytics community.

J. Conclusion

This comparison study demonstrates how well clustering approaches and GNN models perform for community detection in complex networks. We provide major contributions to a complete understanding of the benefits and drawbacks of different GNN architectures and clustering techniques by implementing them on a variety of datasets. Along several application domains, our research can help professionals, as well as researchers, select the most effective method for performing community detection. Along with that, it also highlights the importance of choosing GNN-based evaluation methods for efficient community detection and performance during node classification. Overall, our findings provide the leading edge in identifying communities based on graph embedding and also lay the foundation for future research in this field. Overall, our work establishes the foundation for further studies in this area and advances the state-of-the-art in graph embedding-based community discovery.

REFERENCES

[1] Kipf, Thomas N., and Max Welling. "Semi-supervised classification with graph convolutional networks." arXiv preprint arXiv:1609.02907 (2016).

[2] Velickovic, Petar, et al. "Graph attention networks." stat 1050.20 (2017): 10-48550.

[3] Hamilton, Will, Zhitao Ying, and Jure Leskovec. "Inductive representation learning on large graphs." Advances in neural information processing systems 30 (2017).

[4] Von Luxburg, Ulrike. "A tutorial on spectral clustering." Statistics and Computing 17 (2007): 395-416.

[5] MacQueen, James. "Some methods for classification and analysis of multivariate observations." Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability. Vol. 1. No. 14. 1967.

[6] Zhang, Tao, Hao-Ran Shan, and Max A. Little. "Causal GraphSAGE: A robust graph method for classification based on causal sampling." Pattern Recognition 128 (2022): 108696.

[7] Liu, Ziqi, et al. "Geniepath: Graph neural networks with adaptive receptive paths." Proceedings of the AAAI conference on artificial intelligence. Vol. 33. No. 01. 2019.

[8] Liu, Ziqi, et al. "Geniepath: Graph neural networks with adaptive receptive paths." Proceedings of the AAAI conference on artificial intelligence. Vol. 33. No. 01. 2019.

[9] Menta Sai Vineeth, Krishnappa RamKarthik, M. Shiva Phaneendra Reddy, Namala Surya and L. R. Deepthi ,. "Comparative analysis of graph clustering algorithms for detecting communities in social networks." Ambient Communications and Computer Systems: RACCCS 2019. Springer Singapore, 2020

[10] Srisurya, Ippatapu Venkata, K. Mukesh, and I. R. Oviya. "Metabolic Pathway Class Prediction Using Graph Convolutional Network (GCN)." International Conference on Communication and Intelligent Systems. Singapore: Springer Nature Singapore, 2022.

[11] Sujitha Venkatapathy, Mikhail Votinov, Lisa Wagels, Sangyun Kim, Munseob Lee, Ute Habel, In-Ho Ra, Han-Gue Jo, " Ensemble Graph Neural Network Model for Classification of Major Depression Disorder using Whole-Brain Functional Connectivity", Frontiers in Psychiatry

[12] C. Selvi and Elango, S., "A novel Adaptive Genetic Neural Network (AGNN) model for recommender systems using modified k-means clustering approach", Multimedia Tools and Applications, pp. 1–28, 2018.

[13] Medicherla, Abhishek Bharadwaj, Rithvik Vukka, and Sampat Srivatsav Jakkinapalli. "Influential Node Identification on an Multilayered Attributed Network." Proceedings of the 2023 Fifteenth International Conference on Contemporary Computing. 2023.

[14] Manoju, T., and G. P. Sajeev. "Conductivity based agglomerative spectral clustering for community detection." 2021 sixth international conference on image information processing (ICIIP). Vol. 6. IEEE, 2021.

[15] Panicker, Christy Alex, and M. Geetha. "Exploring Graph Partitioning Techniques for GNN Processing on Large Graphs: A Survey." 2023 4th International Conference on Communication, Computing and Industry 6.0 (C216). IEEE, 2023.