Literature Review on Graph Neural Network using Reddit Dataset for Node Classification

Monib Saadi[#1], Aditya Babar[*2]

Applied Data Science
FH Kaernten, Villach

[2]edubabadi001@fh-kaernten.ac.at

[1]edusaamon001@fh-kaernten.ac.at

***Abstract*— Graph data is essential for numerous learning tasks across various domains, including physical systems modeling, molecular fingerprinting, protein interface prediction, and disease classification. Robust graph reasoning models are also critical for tasks involving text and image data. Graph Neural Networks (GNNs) have proven effective in these areas by capturing dependencies through message passing between nodes.**

This study examines node classification in the Reddit dataset using three advanced GNN models: Graph Convolutional Networks (GCNs), Graph Attention Networks (GATs), and GraphSAGE. We aim to identify the most effective model for classifying subreddit communities. Our results all the 3 models perform equally well. A descent accuracy is due to the graph properties, which effectively preserve critical node characteristics from previous layers. The structure of the Reddit dataset, with edges concentrated among similar nodes, further boosts the performance of these algorithms. This study highlights the outcomes of 3 graph algorithms for subreddit community classification, providing a foundation for future research on more expressive and robust graph embeddings, particularly for social media platforms like Reddit.

***Keywords*— GCNN, GAT, GraphSage, Embeddings, Attention, Nodes, Edges**

I. Introduction

Graphs represent objects (nodes) and their relationships (edges), essential for modeling complex systems in domains like social networks, biological networks, and knowledge graphs. This format captures intricate dependencies valuable for tasks such as node classification, link prediction, and clustering. Graph Neural Networks (GNNs) have become powerful tools for analyzing graph data by leveraging the graph's structure to learn rich node and graph representations. GNNs are applied to various problems, from protein interface prediction to social media recommendations.

GNNs evolved from Recursive Neural Networks (RecNNs) in the 1990s to more advanced models like Convolutional Neural Networks (CNNs), leading to the development of Graph Convolutional Networks (GCNs). GCNs capture localized structural information, a significant advancement in graph-based machine learning. Other specialized models include Graph Attention Networks (GATs) and GraphSAGE, each addressing different aspects of graph data processing. Representation learning in graphs shifted from hand-engineered features to automatically learned representations, improving scalability and flexibility. GNNs learn to aggregate information from graph structures, enhancing tasks like node classification, link prediction, and clustering.

This research focuses on node classification in the Reddit dataset using GCN, GAT, and GraphSAGE models. The study aims to:

1. Evaluate the effectiveness of these models in classifying subreddit communities.
2. Identify factors contributing to performance differences.
3. Investigate model scalability with large-scale and dynamic graph data.

The paper is structured to review related work, describe the methodology, present results, discuss findings and conclusions. This comprehensive evaluation of GNN models on the Reddit dataset contributes to understanding graph neural networks and their applications in social media analysis.

II. Related Work

Graph Neural Networks (GNNs) have significantly advanced since their introduction in 2005 by Marco Gori and Franco Scarselli. Initially conceptualized to encapsulate relationships and attributes of nodes, GNNs now play crucial roles in various domains like NLP, computer vision, and healthcare. These networks learn state embeddings summarizing node neighborhoods, facilitating tasks such as node classification, link prediction, and clustering.

Key Milestones in GNN Research:

2005: Marco Gori and Franco Scarselli introduced GNNs, establishing the theoretical framework for learning from graph data.

2017: Tomas Kipf and Max Welling introduced Graph Convolutional Networks (GCNs), applying convolutional operations to graph data, akin to how CNNs work with grid-structured data like images. GCNs marked a significant advancement by capturing localized structural information effectively.

Recent Years: Emergence of Graph Attention Networks (GATs) and Relational Graph Convolutional Networks (R-GCNs). GATs incorporate attention mechanisms to weigh the importance of neighboring nodes, enhancing fine-grained analysis. R-GCNs handle multi-relational data through relation-specific transformations during message-passing managing complex relationships within graphs.

Node Embedding Approaches:

DeepWalk, node2vec, and LINE - These methods learn low-dimensional node representations using techniques from natural language processing, like random walks and matrix factorization-based objectives. While successful, they are typically transductive, requiring retraining for new nodes.

Supervised Learning on Graphs:

Kernel-Based Approaches: Feature vectors derived from graph kernels generate node representations.

Neural Network Approaches: Focus on generating useful representations for individual nodes, enhancing their applicability in various tasks.

Graph Convolutional Networks (GCNs): Designed for semi-supervised learning in a transductive setting, GCNs require the full graph Laplacian during training. Subsequent research has extended GCNs to inductive learning, making them applicable to larger and more dynamic graphs.

This review highlights the rapid evolution and broad applicability of GNNs, emphasizing significant advancements and diverse applications. From initial frameworks to advanced models like GCN, GAT, and R-GCN, GNNs have proven to be powerful tools for analyzing complex graph-structured data, providing a comprehensive overview of the current state of research in this dynamic field.

III. Dataset

Reddit, with around 50 million daily active users in 2022, serves as the focus of our dataset. The dataset includes 232,965 posts from September 2014 across 41 subreddits. Each post is represented as a node in a graph, with edges connecting posts commented on by the same user, allowing us to explore user interactions and community dynamics. The subreddit of each post serves as its node label, and 300-dimensional GloVe word vectors provide semantic node features. The graph consists of 232,965 nodes and 114,615,892 edges, enriched with 602 features per node and spanning 41 classes. For model training, validation, and evaluation, the dataset is divided into a training set, a testing set and a validation set. The dataset is divided through binary masking into these 3 sets. This dataset offers a detailed view of Reddit's sub-communities, enabling researchers to analyze user behavior, interactions, and content trends.

IV. Methodology

Implementation in Google Colab:

https://colab.research.google.com/drive/1gWvavtRkVir2BtcdlIi7D0u6JJX2EB4E?usp=sharing

Overview of Graph Neural Networks:

Graph Neural Networks (GNNs) are specialized neural networks designed for graph-structured data, consisting of nodes and edges. They excel in various applications,

from social network analysis to molecular biology. Central to GNNs is the message passing mechanism, which iteratively updates node representations by aggregating information from neighboring nodes. This process maintains graph symmetries, ensuring the preservation of node relationships and global context.

Message Passing Mechanism in Graph Neural Networks

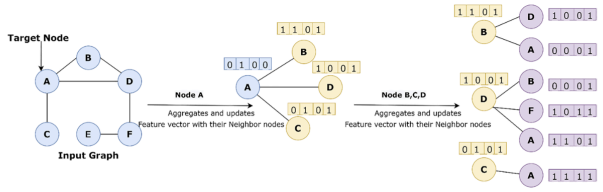


Fig. 6 How a single node aggregates messages from its adjacent neighbor nodes

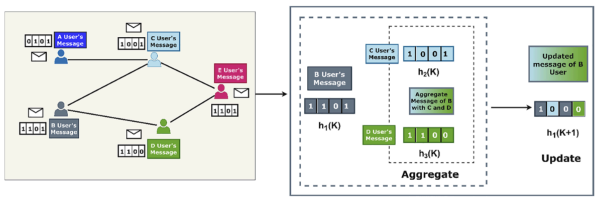


Fig. 7 Message passing mechanism in GNN

During each iteration, nodes aggregate messages from their neighbors, progressively incorporating information from wider contexts. This iterative process captures both structural information (e.g., node degrees) and feature-based information (e.g., node attributes).

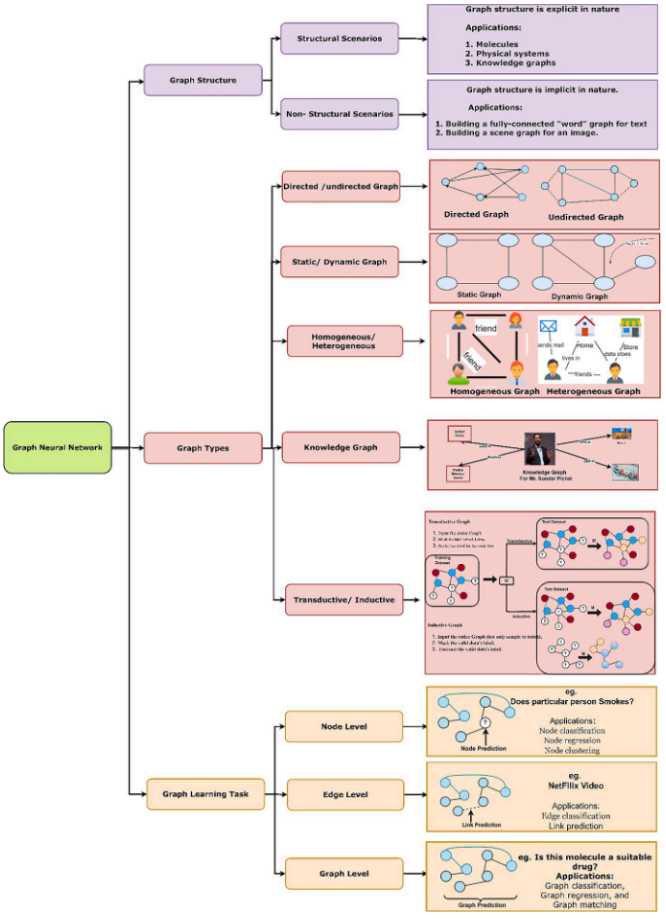
GNNs can be categorized based on their structural characteristics and learning tasks

Graph Structures:
Structural Graphs: Explicitly defined structures, such as molecular systems or knowledge graphs.
Non-structural Graphs: Implicit structures constructed from the task, such as word graphs for text data or scene graphs for images.

Graph Types:
Directed vs. Undirected: Directed graphs have edges with specific directions; undirected graphs have bidirectional edges.
Static vs. Dynamic: Static graphs have fixed structures; dynamic graphs change over time.
Homogeneous vs. Heterogeneous: Homogeneous graphs have uniform nodes and edges; heterogeneous graphs have multiple types.
Knowledge Graphs: Represent real-world entities and their relationships.
Transductive vs. Inductive: Transductive models require the entire graph for prediction; inductive models generalize to unseen nodes or graphs.

Graph Learning Tasks:
Node-level Tasks: Predict properties or labels for individual nodes.
Edge-level Tasks (Link Prediction): Analyze relationships between node pairs.
Graph-level Tasks:Make predictions about entire graphs.

This framework allows GNNs to effectively learn from and make predictions on graph-structured data across diverse domains.

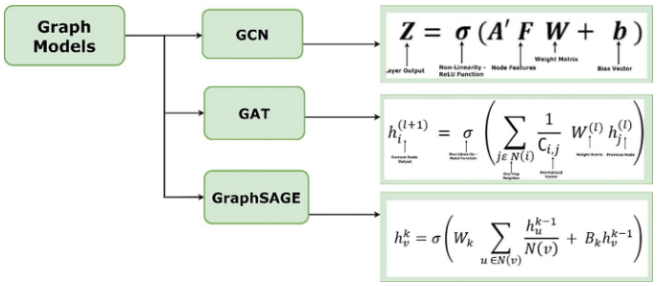


Graph Learning Tasks

GNNs can be applied to various learning tasks:

Node-level Tasks: Predict properties or labels for individual nodes, such as classifying social network users.
Edge-level Tasks (Link Prediction): Analyze relationships between pairs of nodes, such as predicting connections in a social network.
Graph-level Tasks: Make predictions about entire graphs, such as determining molecular properties.

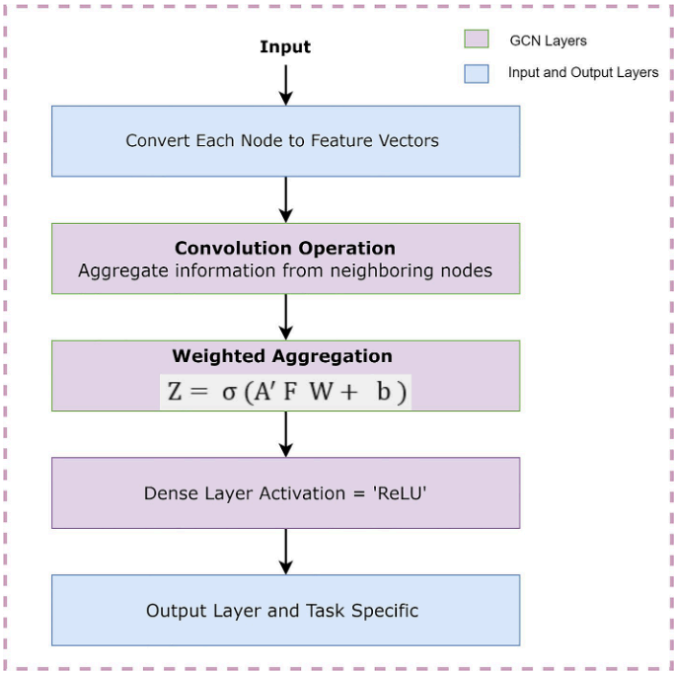
GNN Models and Comparative Analysis



Graph Convolutional Networks (GCNs)

Graph Convolutional Networks (GCNs), introduced by Tomas Kipf and Max Welling, extend convolution operations from traditional Convolutional Neural Networks (CNNs) to graph data. While CNNs handle grid-like data structures (like images), GCNs operate on irregular, non-Euclidean data structures (such as social networks).

How GCNs Work:
GCNs operate by aggregating information from neighboring nodes using convolutional operations based on the graph's adjacency matrix. These operations, performed iteratively across multiple layers, refine node representations by incorporating features from local neighborhoods. Finally, task-specific outputs are generated from these refined representations for tasks like node classification and link prediction.



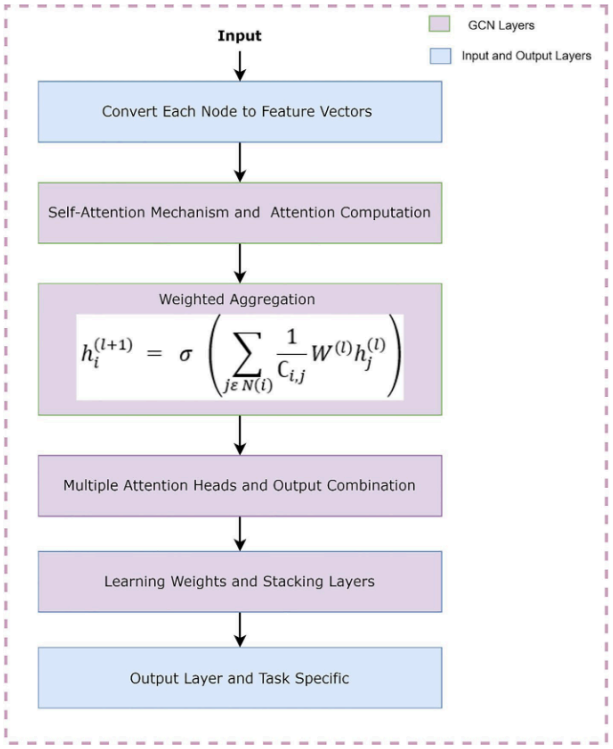
Applications: GCNs are effective for tasks involving graph-structured data, such as social networks, citation networks, and recommendation systems.

Graph Attention Networks (GATs)

Graph Attention Networks (GATs) introduce attention mechanisms to GNNs, allowing nodes to focus on the most relevant neighbors during information aggregation. This flexibility makes GATs suitable for both inductive and transductive tasks.

How GATs Work:
GATs use attention mechanisms to assign different weights to neighboring nodes based on their features. These attention scores determine the importance of each neighbor's contribution to the node's representation. By aggregating features based on these scores, GATs

effectively capture complex relationships within the graph.



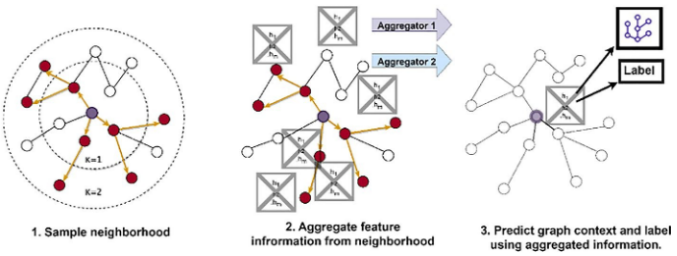
Advantages Over GCNs: GATs' attention mechanisms provide a flexible and powerful mechanism for aggregating information from neighbors, potentially improving performance.

GraphSAGE:

GraphSAGE (Graph Sample and Aggregation) is designed for inductive learning on large graphs, allowing it to generalize to unseen nodes or graphs, unlike transductive models.

How GraphSAGE Works:

GraphSAGE operates by sampling a fixed-size set of neighbors for each node and aggregating their features using various functions (such as mean or LSTM). These aggregated features, combined with the node's own features, are then passed through multiple layers to capture complex patterns and higher-level features.



Applications: GraphSAGE is useful for dynamic graphs where structure and node features change over time, such as in social networks and recommendation systems.

Comparative Analysis

All the models are defined in pyTorch in the following way.

a) A layer for each model(GCN, GAT, GraphSage)
b) Activation Function (ReLU)
c) Dropout layer
d) Normalization layer

Each of these GNN models offers unique advantages based on the specific requirements of the task at hand:

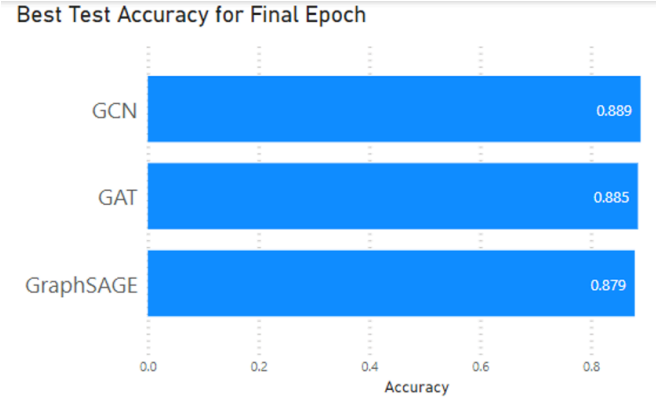
- GCNs are well-suited for tasks where the entire graph is available during training and inference, offering robust performance on node and graph classification tasks.
- GATs enhance the flexibility and accuracy of GCNs by incorporating attention mechanisms, allowing for more nuanced aggregation of neighbor information.
- GraphSAGE excels in scalability and inductive learning, making it ideal for applications involving large, dynamic graphs.

By understanding the strengths and mechanisms of GCNs, GATs, and GraphSAGE, we can choose the most appropriate model for a given graph-based problem, enhancing the predictive power and efficiency of our analyses.

V. Results and Discussion

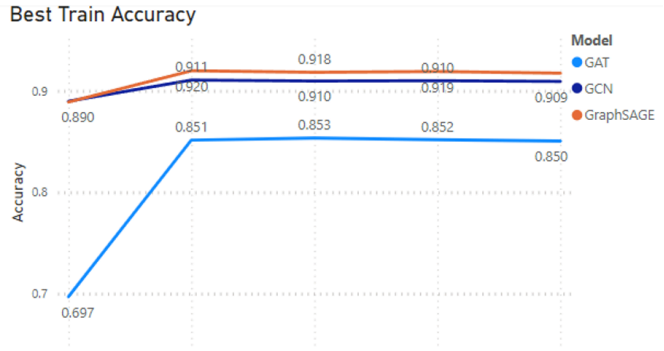
In our study, we evaluated the performance of three Graph Neural Network (GNN) models—GCN, GAT, and GraphSAGE—on a node classification task using the Reddit dataset. The models were trained and tested under different conditions where the best hyperparameters were found to be of a batch size of 1024, two layers, two attention heads for GAT, a hidden dimension of 256, and using the Adam optimizer. The results are summarized below.

Best Test Accuracy



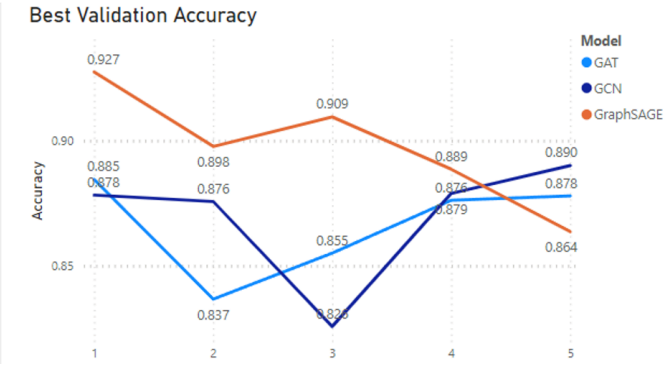
The first graph shows the best test accuracy for each model at the final epoch: GCN (88.9%), GAT (88.5%), and GraphSAGE (87.9%). GCN slightly outperforms the other models.

Training Accuracy vs. Epochs



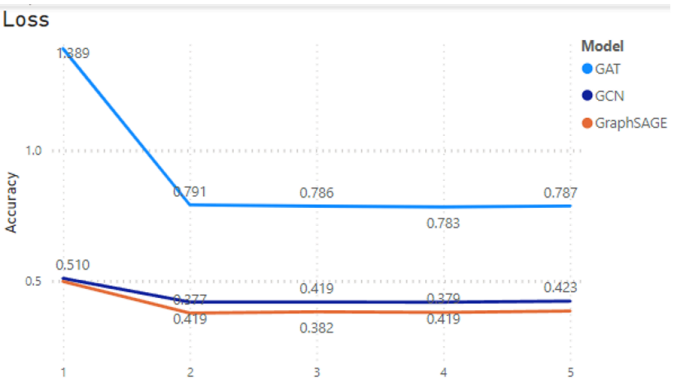
The second graph displays training accuracy over epochs. GCN and GraphSage achieve high training accuracy than GAT, indicating their efficiency in learning from the data.

Validation Accuracy vs. Epochs



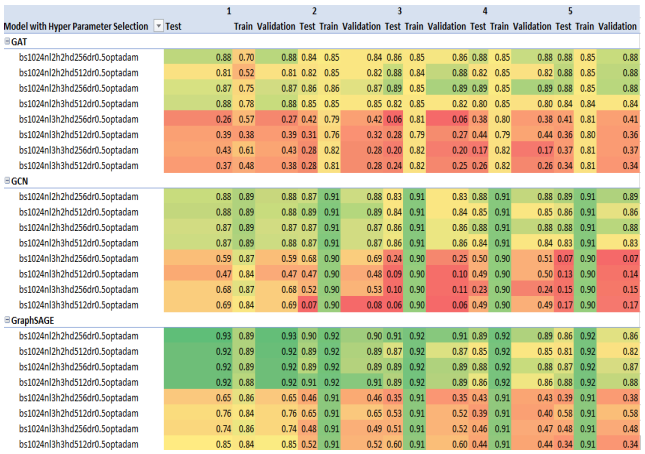
The third graph presents validation accuracy over epochs. GCN maintains higher validation accuracy compared to GAT and GraphSAGE, suggesting better generalization capabilities.

Loss vs. Epochs



The fourth graph depicts loss values over epochs. GCN and GraphSage show a rapid decrease in loss initially, stabilizing at lower values, while GAT shows a slower but steady reduction in loss.

Comparison of test, train and validation accuracy for all the models



	1			2			3			4			5		
Model with Hyper Parameter Selection	Test	Train	Validation	Test	Train	Validation	Test	Train	Validation	Test	Train	Validation	Test	Train	Validation
GAT															
bs1024nl2h2hd256dr0.5optadam	0.88	0.70	0.88	0.84	0.85	0.84	0.86	0.85	0.86	0.88	0.85	0.88	0.88	0.85	0.88
bs1024nl2h2hd512dr0.5optadam	0.81	0.52	0.81	0.82	0.85	0.82	0.88	0.84	0.88	0.82	0.85	0.82	0.88	0.85	0.88
bs1024nl2h3hd256dr0.5optadam	0.87	0.75	0.87	0.86	0.86	0.87	0.89	0.85	0.89	0.89	0.85	0.89	0.88	0.85	0.88
bs1024nl2h3hd512dr0.5optadam	0.88	0.78	0.88	0.85	0.85	0.85	0.82	0.85	0.82	0.80	0.85	0.80	0.84	0.84	0.84
bs1024nl3h2hd256dr0.5optadam	0.26	0.57	0.27	0.42	0.79	0.42	0.06	0.81	0.06	0.38	0.80	0.38	0.41	0.81	0.41
bs1024nl3h2hd512dr0.5optadam	0.39	0.38	0.39	0.31	0.76	0.32	0.28	0.79	0.27	0.44	0.79	0.44	0.36	0.80	0.36
bs1024nl3h3hd256dr0.5optadam	0.43	0.61	0.43	0.28	0.82	0.28	0.20	0.82	0.20	0.17	0.82	0.17	0.37	0.81	0.37
bs1024nl3h3hd512dr0.5optadam	0.37	0.48	0.38	0.28	0.81	0.28	0.24	0.82	0.25	0.26	0.82	0.26	0.34	0.81	0.34
GCN															
bs1024nl2h2hd256dr0.5optadam	0.88	0.89	0.88	0.87	0.91	0.88	0.83	0.91	0.83	0.88	0.91	0.88	0.89	0.91	0.89
bs1024nl2h2hd512dr0.5optadam	0.88	0.89	0.88	0.89	0.91	0.89	0.84	0.91	0.84	0.85	0.91	0.85	0.86	0.91	0.86
bs1024nl2h3hd256dr0.5optadam	0.87	0.89	0.87	0.87	0.91	0.87	0.86	0.91	0.86	0.88	0.91	0.88	0.88	0.91	0.88
bs1024nl2h3hd512dr0.5optadam	0.87	0.89	0.88	0.87	0.91	0.87	0.86	0.91	0.86	0.84	0.91	0.84	0.83	0.91	0.83
bs1024nl3h2hd256dr0.5optadam	0.59	0.87	0.59	0.68	0.90	0.69	0.24	0.90	0.25	0.50	0.90	0.51	0.07	0.90	0.07
bs1024nl3h2hd512dr0.5optadam	0.47	0.84	0.47	0.47	0.90	0.48	0.09	0.90	0.10	0.49	0.90	0.50	0.13	0.90	0.14
bs1024nl3h3hd256dr0.5optadam	0.68	0.87	0.68	0.52	0.90	0.53	0.10	0.90	0.11	0.23	0.90	0.24	0.15	0.90	0.15
bs1024nl3h3hd512dr0.5optadam	0.69	0.84	0.69	0.07	0.90	0.08	0.06	0.90	0.06	0.49	0.90	0.49	0.17	0.90	0.17
GraphSAGE															
bs1024nl2h2hd256dr0.5optadam	0.93	0.89	0.93	0.90	0.92	0.90	0.91	0.92	0.91	0.89	0.92	0.89	0.86	0.92	0.86
bs1024nl2h2hd512dr0.5optadam	0.92	0.89	0.92	0.89	0.92	0.89	0.87	0.92	0.87	0.85	0.92	0.85	0.81	0.92	0.82
bs1024nl2h3hd256dr0.5optadam	0.92	0.89	0.92	0.89	0.92	0.89	0.89	0.92	0.89	0.88	0.92	0.88	0.87	0.92	0.87
bs1024nl2h3hd512dr0.5optadam	0.92	0.88	0.92	0.91	0.92	0.91	0.89	0.92	0.89	0.86	0.92	0.86	0.88	0.92	0.88
bs1024nl3h2hd256dr0.5optadam	0.65	0.86	0.65	0.46	0.91	0.46	0.35	0.91	0.35	0.43	0.91	0.43	0.39	0.91	0.38
bs1024nl3h2hd512dr0.5optadam	0.76	0.84	0.76	0.65	0.91	0.65	0.53	0.91	0.52	0.39	0.91	0.40	0.58	0.91	0.58
bs1024nl3h3hd256dr0.5optadam	0.74	0.86	0.74	0.48	0.91	0.49	0.51	0.91	0.52	0.46	0.91	0.47	0.48	0.91	0.48
bs1024nl3h3hd512dr0.5optadam	0.85	0.84	0.85	0.52	0.91	0.52	0.60	0.91	0.60	0.44	0.91	0.44	0.34	0.91	0.34

VI. Conclusion

From our results, it's clear that all the algorithms perform equally good on sub-reddit community classification. This performance boost can be attributed to the fact that both these algorithms use feature embeddings from previous node layers in the new layer embeddings. This approach helps in preserving node characteristics that were embedded earlier.

One possible reason for this superior performance is the expressive power of the GLOVE word embeddings. These embeddings might already provide strong initial representations, which GAT and GraphSAGE effectively leverage. Specifically, GAT uses edge scoring to retain previous node features, while GraphSAGE uses concatenation in the aggregate step to achieve the same.

Another important factor could be the concentration of edges among nodes of a certain type. On Reddit, users often congregate in similar subreddit types based on their interests, leading to numerous cross-posts within these communities. By using graph attention mechanisms, GAT may effectively focus on these numerous types of cross-posts, enhancing classification accuracy.

These findings open up new avenues for enhancing graph-based models on Reddit and similar platforms. With further investigation and optimization, we can develop even more powerful methods for node classification in social networks.

VII. Reference

[1] Inductive Representation Learning on Large Graphs William L. Hamilton* wleif@stanford.edu Rex Ying* rexying@stanford.edu Jure Leskovec jure@cs.stanford.edu Department of Computer Science Stanford University Stanford, CA, 94305

[2] A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions Bharti Khemani1 , Shruti Patil2*, Ketan Kotecha2 and Sudeep Tanwar3

[3] Graph neural networks: A review of methods and applications Jie Zhou a,1 , Ganqu Cui a,1 , Shengding Hu a , Zhengyan Zhang a , Cheng Yang b , Zhiyuan Liu a,* , Lifeng Wang c , Changcheng Li c , Maosong Sun a a Department of Computer Science and Technology, Tsinghua University, Beijing, China b School of Computer Science, Beijing University of Posts and Telecommunications, China c Tencent Incorporation, Shenzhen, China

[4]https://medium.com/red-buffer/implementation-and-understanding-of-graph-neural-networks-gnn-54084c8a0e24

[5]https://medium.com/@hoangben/classifying-reddit-posts-using-gnns-a9108395e2b7