Community Detection in Social Networks using Graph Neural Networks

Ammar Tahir
University of Washington
imammar@uw.edu

Apratim Tripathi
University of Washington
trips@uw.edu

Rohit Chandiramani
University of Washington
rohitch@uw.edu

Trisha Banerjee
University of Washington
tbaner@uw.edu

1 Introduction

Community detection in complex networks is vital for understanding the structure and dynamics of various systems, from social networks to biological systems. Our project aims to enhance community detection using Graph Neural Networks (GNNs), addressing limitations of traditional methods like handling large-scale and overlapping communities. Instead of directly detecting communities, we approached this as a node classification problem, leveraging graph networks to classify nodes into predefined categories.

The Reddit dataset, with its extensive interaction data across numerous subreddits, provides a rich and dynamic environment for this study. By utilizing Reddit's hyperlinks and extensive user interactions, we can uncover meaningful community structures that reflect real-world social dynamics. This project seeks to classify subreddits into relevant categories and detect communities through classification, ultimately contributing to more accurate and scalable methods for analyzing large social networks.

2 Related Work

The dataset provided by Kumar et al.(2018) includes detailed interactions between subreddits on Reddit, capturing hyperlinks between posts. This rich relational data is crucial for studying community interactions and influence. In our project, this dataset forms the foundation for our node classification and community detection tasks. It provides the necessary data to train and evaluate our models, enabling us to explore the complex network of subreddit interactions and detect meaningful communities within the Reddit platform. [1].

The Louvain algorithm is a widely used method for community detection in large networks. It optimizes modularity by iteratively grouping nodes into communities and then creating super-nodes to form a new network, repeating this process to enhance the modularity score. This approach is efficient and scalable, making it suitable for large datasets such as Reddit. In our project, we use the Louvain algorithm as a baseline for community detection, providing a benchmark to compare the performance of more advanced methods like GNNs [2].

Graph Convolutional Networks (GCNs) are a type of GNN that aggregate features from neighboring nodes to learn node representations. GCNs effectively integrate node features and graph structure, making them well-suited for tasks like community detection. In our project, we use GCNs to classify nodes (subreddits) into predefined categories, transforming the community detection problem into a node classification problem. This allows us to leverage the strengths of GCNs in capturing both local and global structural information in the Reddit dataset. [3].

Preprint. Under review.

Attention-based Graph Neural Networks (AGNNs) incorporate attention mechanisms to assign different weights to the edges in the graph, allowing the model to focus on more relevant connections. This approach improves the accuracy of node classification and community detection by highlighting significant relationships within the network. AGNNs are particularly useful in our project for detecting communities within the Reddit network, as they can better capture the intricate interactions between subreddits by focusing on the most relevant connections. [4]

GNNExplainer is a technique designed to interpret the predictions of Graph Neural Networks (GNNs) by identifying the important subgraphs and node features that contribute to the model's output. This method enhances the transparency and understanding of complex GNN models. In our project, GNNExplainer is relevant as it helps us understand the underlying factors that the GNN model uses to detect communities in the Reddit dataset. This interpretability helps in validating the model's decisions and ensuring that the detected communities are meaningful. [5]

3 Data Collection

3.1 Dataset Summary

The dataset used in this study comprises Reddit activity data spanning 40 months, capturing interactions between various subreddits. The key components of the data include nodes representing subreddits and edges representing hyperlinks between subreddits, indicating relationships or references. The primary data source is the file `soc-redditHyperlinks-body.tsv`, which contains the network of subreddit-to-subreddit hyperlinks extracted from the body of posts. Additionally, subreddit embeddings are provided in a supplementary dataset, capturing vector representations of each subreddit.

3.2 Data Preprocessing

To prepare the data for analysis, several preprocessing steps were undertaken. First, the hyperlink data from the body file was loaded into a Pandas DataFrame for consolidation. Subsequently, a graph was constructed where each node represents a subreddit and edges represent the hyperlinks between them, with edge weights derived from the frequency of interactions. Subreddits with available node embeddings were selected, focusing on 21,000 nodes, and these embeddings were integrated into the graph. The final graph was created using NetworkX, ensuring only nodes with embeddings were included. The dataset was then split into training and testing sets, ensuring a balanced representation of all categories in both sets. Initial exploratory data analysis included checking for missing values in the body hyperlink dataset to ensure dataset completeness. The constructed graph was analyzed to understand the distribution of nodes and edges, and to verify its integrity.

3.3 Graph Statistics

- Number of Nodes: 55,863.
- Number of Edges: 858,490 edges representing the hyperlinks between these subreddits.
- Node embedding: Each node embedding has 300 dimensions.

4 Methods

4.1 Finding Ground Truth Labels

In our approach, finding ground truth labels was a crucial step to transform the community detection problem into a node classification problem. Given the lack of ground truth labels in our dataset, we leveraged a socially-primed LSTM model to classify nodes (subreddits) into 20 predefined categories based on Reddit's advertisement documentation.

4.1.1 Data Preparation

We collected a comprehensive Reddit corpus, including all relevant posts and comments across various subreddits to ensure a wide range of topics and interactions were covered. The collected data underwent preprocessing steps such as tokenization, where text was split into tokens (words or subwords). All text was converted to lowercase to maintain uniformity, and common stopwords that did not contribute to the meaning were removed. Additionally, words were reduced to their base or root form using lemmatization. Following preprocessing, the text was transformed into numerical vectors using pre-trained GloVe embeddings, capturing the semantic meaning of the text.

4.1.2 Model Training

To incorporate social context into our classification model, we employed a socially-primed LSTM model. This model combined word embeddings of the post text, user embeddings of the post author, and community embeddings of the source and target communities.

- **Text Embeddings:** We generated 300-dimensional word embeddings for each post using pre-trained GloVe embeddings.
- **User and Community Embeddings:** User and community embeddings were learned from the bipartite graph of users and subreddits, ensuring embeddings were similar if users posted in similar subreddits.

The LSTM model was trained with these combined embeddings to predict the category of each post. The socially-primed LSTM used word embeddings, user embeddings, and community embeddings as inputs, allowing it to leverage both textual and social information.

4.1.3 Node Classification

For each subreddit, we aggregated the posts and comments to form a single representation (embedding) of the subreddit. This involved summarizing the embeddings of all posts/comments in a subreddit into a single vector. Using the trained classification model, we classified each subreddit based on its aggregated embedding into one of the 20 categories.

The ground truth labels were then integrated into our graph structure. Each node (subreddit) was associated with a category label, which was used for further analysis and community detection. This process allowed us to leverage node classification techniques to transform the community detection problem into a more tractable classification task.

4.2 Louvain Algorithm

As a baseline model, we used the Louvain algorithm for community detection. The Louvain algorithm aimed to maximize the modularity of the partition of the network. Modularity Q is defined as:

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

where:

- m is the total weight 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 weighted degrees of nodes i and j,
- c_i and c_j are the community assignments of nodes i and j,
- $\delta(c_i, c_j)$ is the Kronecker delta function, which is 1 if $c_i = c_j$, and 0 otherwise.

The algorithm proceeded iteratively in two phases: first, it assigned nodes to communities to maximize modularity locally, then it aggregated nodes into supernodes to create a new network, and the process repeated.

4.3 Graph Neural Networks (GNNs)

To detect communities, we implemented Graph Neural Networks (GNNs) that leveraged the categorized node embeddings and edge connections within the graph. GNNs are designed to perform inference on graph data by considering both the features of the nodes and the structure of the graph. In our project, we used two primary GNN architectures: Graph Convolutional Networks (GCNs) and Attention-based Graph Neural Networks (AGNNs).

4.3.1 Graph Convolutional Network (GCN)

The GCN architecture used in our project was defined as follows:

$$F := \text{GCN}_\theta(A, X) = \text{ReLU}(\hat{A}\,\text{ReLU}(\hat{A}XW^{(1)})W^{(2)})$$

where $\hat{A} = D^{-1/2}AD^{-1/2}$ is the normalized adjacency matrix, D is the diagonal matrix of node degrees, and X represents the node feature matrix.

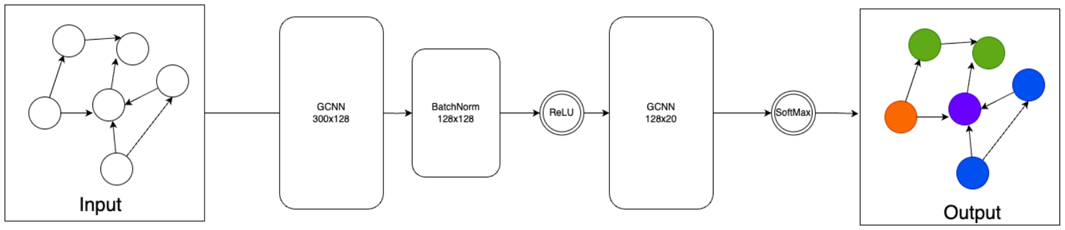

Figure 1: Graph Convolutional Neural Network (GCN) Architecture

The GCN model consists of multiple layers. The input layer takes the node embeddings and adjacency matrix as input. The hidden layers apply graph convolution operations to aggregate features from neighboring nodes. Specifically, the model first applies a graph convolution operation followed by a ReLU activation function, normalizes the adjacency matrix, and then applies another graph convolution operation. The output layer produces a log-softmax output, representing the probabilities of each node belonging to each category. This hierarchical feature aggregation allows the model to capture complex dependencies between nodes in the graph, enabling effective community detection.

4.3.2 Attention-based Graph Neural Network (AGNN)

To enhance the model's performance, we also implemented an AGNN. The AGNN architecture included attention mechanisms to weigh the importance of neighboring nodes during feature aggregation. This attention mechanism assigns different weights to different nodes in the neighborhood, allowing the model to focus on more relevant connections within the graph.

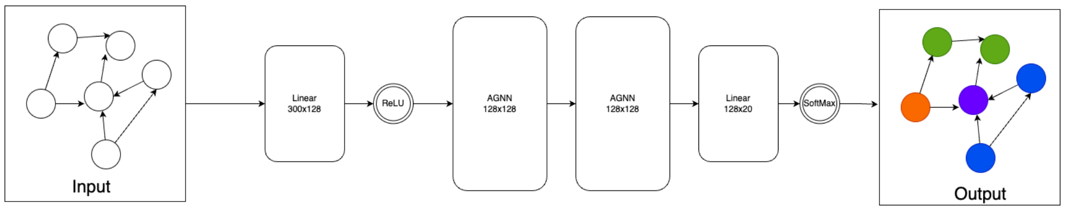

Figure 2: Attention based Graph Neural Network (AGNN) Architecture

The AGNN model begins with a linear transformation of the node features, followed by two layers of AGNNConv, which are attention-based convolutional layers. These layers allow the model to learn the relative importance of neighboring nodes, improving the feature aggregation process.

Mathematically, the AGNN can be expressed as follows:

$$h_i^{(k+1)} = \sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(k)} W^{(k)} h_j^{(k)}$$

where $h_i^{(k+1)}$ is the representation of node i at layer $k+1$, $\mathcal{N}(i)$ represents the neighbors of node i, $W^{(k)}$ is a learnable weight matrix, and $\alpha_{ij}^{(k)}$ is the attention coefficient that determines the importance of node j to node i at layer k.

The attention coefficient $\alpha_{ij}^{(k)}$ is computed using the softmax function:

$$\alpha_{ij}^{(k)} = \frac{\exp(\text{LeakyReLU}(a^\top [W^{(k)} h_i^{(k)} \| W^{(k)} h_j^{(k)}]))}{\sum_{j' \in \mathcal{N}(i)} \exp(\text{LeakyReLU}(a^\top [W^{(k)} h_i^{(k)} \| W^{(k)} h_{j'}^{(k)}]))}$$

where a is a learnable weight vector, $\|$ denotes concatenation, and LeakyReLU is the activation function.

After applying attention-based convolution, the model uses a final linear layer to map the features to the desired output space. The output is then passed through a log-softmax function to obtain the probability distribution over the categories.

In our implementation, the GCN and AGNN models were trained on the categorized node embeddings and the adjacency matrix of the graph. The training process involved minimizing the cross-entropy loss between the predicted and actual node categories using the Adam optimizer. The trained models were evaluated on a separate test set to assess their performance in classifying nodes into the predefined categories.

4.4 Training and Evaluation

The dataset was split into training and testing sets to ensure a balanced representation of all categories. We used stratified sampling to maintain the distribution of categories across both sets. This approach ensured that each category was proportionally represented in both the training and testing datasets, allowing for a more accurate evaluation of the model's performance.

We employed the cross-entropy loss function to train the model. This loss function is suitable for multi-class classification tasks and measures the difference between the predicted probability distribution and the actual distribution. The cross-entropy loss for a single example is defined as:

$$\text{Loss} = -\sum_{i=1}^{C} y_i \log(p_i)$$

where C is the number of classes, y_i is the actual label (1 for the correct class and 0 for others), and p_i is the predicted probability for class i. The overall loss is the average of the individual losses over all training examples.

We used the Adam optimizer to minimize the loss function during training. Adam is an adaptive learning rate optimization algorithm that combines the advantages of two other extensions of stochastic gradient descent: AdaGrad and RMSProp. The Adam optimizer adjusts the learning rate for each parameter dynamically, which helps in faster convergence and better performance. The optimization process updates the model parameters to minimize the cross-entropy loss, allowing the model to learn the underlying patterns in the data.

4.4.1 Metrics

To evaluate the performance of our models, we used several metrics, each providing a different perspective on the quality of the detected communities and the classification accuracy:

Accuracy: This metric measures the overall correctness of the classification by comparing the predicted labels with the true labels. It is defined as the ratio of correctly predicted instances to the total instances:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

Normalized Mutual Information (NMI): NMI measures the similarity between the true and predicted community assignments, accounting for the randomness in cluster assignments. It is defined as:

$$\text{NMI}(U, V) = \frac{2 \cdot I(U; V)}{H(U) + H(V)}$$

where $I(U;V)$ is the mutual information between the true community assignments U and the predicted assignments V, and $H(U)$ and $H(V)$ are the entropies of U and V respectively.

Modularity: Modularity assesses the strength of the division of the network into communities. It measures the density of links inside communities compared to links between communities. Higher modularity values indicate better-defined community structures:

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

where m is the total weight 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 weighted degrees of nodes i and j, c_i and c_j are the community assignments of nodes i and j, and $\delta(c_i, c_j)$ is the Kronecker delta function.

Coverage: Coverage indicates the fraction of edges that fall within the same community. It is a measure of how well the communities capture the internal connectivity of the network:

$$\text{Coverage} = \frac{\text{Number of Intra-community Edges}}{\text{Total Number of Edges}}$$

Density: Density measures the closeness of connections within communities. It is defined as the ratio of the number of edges within a community to the number of possible edges within that community:

$$\text{Density} = \frac{2 \times \text{Number of Intra-community Edges}}{\text{Number of Nodes in the Community} \times (\text{Number of Nodes in the Community} - 1)}$$

Clustering Coefficient: The clustering coefficient reflects the degree to which nodes in a community tend to cluster together. It is the average fraction of node pairs that are neighbors within a community:

$$\text{Clustering Coefficient} = \frac{\text{Number of Closed Triplets}}{\text{Number of Connected Triplets}}$$

Conductance: Conductance evaluates the extent of edge connections between different communities. Lower conductance values indicate well-separated communities:

$$\text{Conductance} = \frac{\text{Number of Edges Between Communities}}{\text{Total Degree of the Nodes in the Community}}$$

These metrics provide a comprehensive evaluation of the performance of our GNN models, allowing us to assess both the accuracy of node classification and the quality of the detected communities.

5 Results

To evaluate our solution, we conducted a series of experiments to assess the performance of our Graph Neural Networks (GNNs) in detecting communities within the Reddit network. We implemented and compared two GNN architectures: Graph Convolutional Networks (GCNs) and Attention-based Graph Neural Networks (AGNNs). Additionally, we used the Louvain algorithm as a baseline for community detection.

Metric	**Louvain**	**GCN**	**AGNN**
Accuracy	0.00005	0.637	0.701
Normalized Mutual Information (NMI)	0.038	0.412	0.492
Modularity	0.457	0.378	0.344
Coverage	0.620	0.506	0.488
Density	0.813	0.032	0.014
Clustering Coefficient	0.017	0.216	0.187
Conductance	0.017	0.583	0.636

Table 1: Results for Louvain, GCN, and AGNN

5.1 Discussion

The experiments demonstrated that both Graph Convolutional Network (GCN) and Attention-based Graph Neural Network (AGNN) models effectively detected communities within the Reddit network. Both Graph Neural Network (GNN) methods achieved higher Normalized Mutual Information (NMI) scores compared to traditional modularity maximization algorithms like Louvain and Infomap. While the Louvain algorithm exhibited the highest modularity, it suffered from a substantially low NMI of **0.0379**. In contrast, GNN algorithms, achieve a **63.7%** (GCN) and **70%** (AGNN) accuracy and their respective NMI values were **0.412** for GCN and **0.492** for AGNN. These results indicate that even simple 2-layer GNNs can learn and generalize the inherent graph structure and its community relationships effectively. In addition to modularity and NMI, we considered other unsupervised clustering metrics. While the coverage and density for both GNN methods were lower than those of traditional counterparts, there was a notable increase in the clustering coefficient. This trend is illustrated in the accompanying graph, which we will examine in detail.

Firstly, visualizing the Louvain communities, we observe that the dense central portion of the graph contains multiple overlapping clusters, as indicated by the intertwining colors representing individual communities. This lack of clear separation corresponds to the lower NMI and clustering coefficient previously noted. The graphs were generated using the *Kamada-Kawai* layout,[6] where edge lengths between node pairs denote the "closeness" within the clustering. In the Louvain communities, the distinguishable peripheral communities have larger edge lengths compared to the GNN graphs.

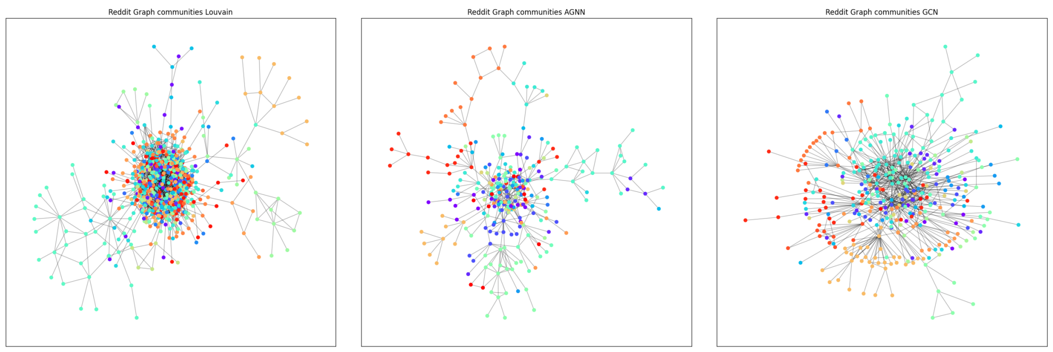
Reddit Graph communities Louvain
Reddit Graph communities AGNN
Reddit Graph communities GCN

Figure 3: Visual Comparison of Community Detection in Reddit Networks: (Left) Louvain Method, (Middle) AGNN, (Right) GCN

In the AGNN visualization, there is a significant improvement in the central high-density area of the graph. Compared to Louvain, the space is predominantly occupied by about three communities (mainly blue and purple), with minimal interaction from other communities. Additionally, the depicted clusters have broad separations between them and distinct themes, as indicated by the node hues. Lastly, the GCN model follows the trends observed with AGNN but surpasses it across all unsupervised metrics, as noted in the previous section. Examining its communities closely, they are tightly bound, with the least inter-node edge distance among all visualizations. This aligns with its superior clustering coefficient and lowest conductance, indicating minimal proliferation as observed in the graph.

Furthermore, we also conducted a rudimentary ablation analysis by introducing noise into our data and hyperparameter tuning the GNN models. This included increasing the number of hidden layers, experimenting with different non-linearity and regularization methods, and varying the size of each hidden layer. Our findings are consistent with those of [3], which advocate for a shallow architecture for the GCN model. While a multilayer GCN can effectively extract network features due to the localized nature of graph convolution, a deeper GCN proves difficult to train and does not enhance model performance. Graph convolution is essentially a Laplacian smoothing operation, and repeatedly applying this smoothing can mix features of nodes from different clusters, making them indistinguishable. Consequently, a GCN model with many convolutional layers tends to reduce clustering accuracy.

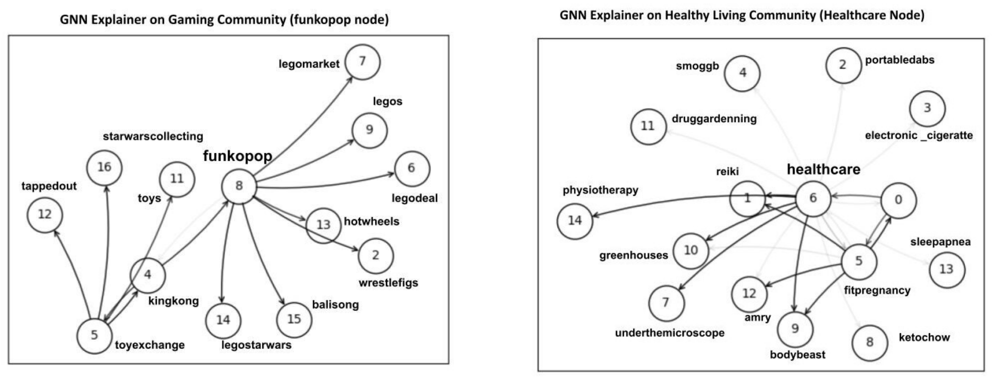

Figure 4: GNN Explainer on Gaming Community: (Left) Funkopop Node, (Right) Healthcare Node

Lastly, we also run GNNExplainer [5] to get some insights into the learning of the GNN algorithms. GNNExplainer is a method designed to interpret and explain the predictions of Graph Neural Networks (GNNs). It provides insights into which features, nodes, and subgraphs are most influential in the model's decision-making process. By identifying a compact subgraph and relevant node features that are crucial for a particular prediction, GNNExplainer enables a better understanding of the underlying structure and reasoning of GNNs. We run this on 2 particular nodes (funkopop and healthcare) in their respective communities Gaming and Healthy Living. As we can in their graphs, the community neighbors belong to a similar context as their central node hence proving our hypothesis that running representational neural network learning on graph structure can capture the domain and perform effective clustering.

Overall, our results highlight the potential of GNNs in community detection, particularly when incorporating attention mechanisms to capture the relevance of neighboring nodes. These findings provide a robust framework for analyzing large social networks and demonstrate the value of transforming community detection into a node classification problem.

References

[1] Kumar, S., Hamilton, W. L., Leskovec, J., & Jurafsky, D. (2018). *Community Interaction and Conflict on the Web*. Proceedings of the 2018 World Wide Web Conference on World Wide Web, 933-943.

[2] Blondel, V. D., Guillaume, J. L., Lambiotte, R., & Lefebvre, E. (2008). *Fast unfolding of communities in large networks*. Journal of Statistical Mechanics: Theory and Experiment, 2008(10), P10008.

[3] Wang, Xiaofeng, et al. "Unsupervised learning for community detection in attributed networks based on graph convolutional network." *Neurocomputing* 456 (2021): 147-155.

[4] Thekumparampil, Kiran K., Sewoong Oh, Chong Wang, and Li-Jia Li. "Attention-based Graph Neural Network for Semi-supervised Learning." *arXiv preprint arXiv:1803.03735* (2018).

[5] Ying, R., Bourgeois, D., You, J., Zitnik, M., & Leskovec, J. (2019). *GNNExplainer: Generating Explanations for Graph Neural Networks*. Advances in Neural Information Processing Systems (NeurIPS), 32, 9240-9251.

[6] Kawai, S. (1989). An Algorithm for Drawing General Undirected Graphs Tomihisa Kamada and Satoru Kawai.

Contribution

Ammar Tahir: Problem formulation to go for community detection in Graphs, coming up with Graph Neural Network approach, coded Attention based Neural Network for community detection & fine-tuned on Reddit Hyperlink Dataset, neural architecture diagrams and coming up with final results.
Apratim Tripathi: Documentation, Problem Formulation, Preliminary model evaulation on reddit dataset (louvain), Finding ground truth labels to training and evaluating the GCN for effective community detection within the Reddit dataset.Literature review.
Rohit Chandrimani: Problem formulation, Baseline model evaluation(Louvain and simple GNN on PubMed dataset), GCN architecture, Algorithm implementation and fine tuning, code modules integration and version control, Community visualizations, GNNExplainer (implementation and visualizations)
Trisha Banerjee: Identified a viable dataset, conducted Exploratory Data Analysis. employed Infomap, (unsupervised clustering algorithm) to detect and visualize community structures within the dataset. Documented the process, and reviewed relevant literature.