CS 510: Homophily and Heterophily Analysis of Node Classification Projec Report

Jou-Chi Huang

1 Project Introduction and Background

This project mainly aims to analyze the impact of high homophily or high heterophily on classifying nodes within datasets. Three datasets/networks from Planetoid were utilized for this analysis: Cora, Citeseer, and Pubmed. These datasets are commonly used for research purposes in node classification tasks. In these datasets, the nodes represent documents, and the edges represent citation links. Training, validation, and test splits are provided through binary masks. Cora and Pubmed exhibit high homophily, with values of 81.00% and 80.24%, respectively. In contrast, the Citeseer dataset shows relatively lower homophily at 73.55%. These characteristics make these datasets suitable for investigating the relationship between node classification performance and homophily.

To provide a clearer understanding of this project, it is essential to introduce some background concepts:

- **Homophily**: Nodes with similar features are more likely to connect, as seen in social networks.
- **Heterophily**: Nodes with dissimilar features are more likely to connect, such as in protein interaction networks.

2 Research Objective

The primary objective of this research is to explore the correlation between the degree of nodes in a network and their link prediction performance. Additionally, this project aims to investigate whether homophily is essential for the effectiveness of Graph Neural Network (GNN) models, such as Graph Convolutional Networks (GCN), Graph Attention Networks (GAT), and GraphSAGE. Specifically, the study seeks to address the following questions:

1. Can GNN models be applied directly to homophily datasets, or is it necessary to preprocess the dataset before applying GNN?
2. If preprocessing is required, do we need to use other AI models for classifying or predicting heterophily datasets?
3. What is the valid range of homophily that allows GNN models to outperform other methods?
4. How does the efficiency of GNN models on homophily datasets compare to their performance on heterophily datasets? (Something extra)

3 Research Method

The dollowing tools and techniques are involved in this project:

1. **Planetoid Dataset**: The Planetoid dataset is used for this research, which includes popular citation networks such as Cora, Citeseer, and Pubmed. These datasets are commonly employed for node classification tasks in graph-based machine learning research. The datasets are downloaded and preprocessed using the `torch_geometric.datasets.Planetoid` module.

2. **GCN Model**: The Graph Convolutional Network (GCN) model is implemented using PyTorch and PyTorch Geometric. GCN is a widely used GNN model for semi-supervised node classification tasks.

3. **GAT Model**: The Graph Attention Network (GAT) model is another GNN architecture utilized in this research. GAT uses attention mechanisms to weigh the importance of neighbors when aggregating information. This allows the model to focus on the most relevant neighbors during training.

4. **GraphSAGE Model**: The GraphSAGE (Graph Sample and Aggregation) model is implemented to learn node representations by sampling and aggregating information from neighbors. This model is especially useful for large-scale graphs and enables inductive learning.

5. **Additional Libraries**: Several additional libraries are employed for data processing, visualization, and graph analysis:

    - `torch`: PyTorch is the core deep learning framework used for building and training GNN models. It provides the necessary tools to define and optimize the models.
    - `torch.nn` and `torch.nn.functional`: These modules are used to define the neural network layers and activation functions for the GNN models.
    - `networkx` (imported as `nx`): This library is used for additional graph-related operations and analysis, such as checking if a graph is undirected using `is_undirected`.
    - `torch_geometric.datasets`: The `torch_geometric.datasets` module is used to load and preprocess graph datasets, including Planetoid, which consists of Cora, Citeseer, and Pubmed. This module handles data preprocessing tasks such as feature extraction and graph construction, enabling efficient graph-based learning.
    - `matplotlib` and `numpy`: The `matplotlib` and `numpy` libraries are used for visualizing the results of the experiments. `matplotlib` is utilized to generate plots, such as accuracy vs. homophily scatter plots, while `numpy` is employed to handle numerical operations, such as calculating mean accuracy values.

4 Research Method

To carry out this research, the following methodology was employed to explore the relationship between homophily and the performance of Graph Neural Network (GNN) models:

1. Dataset Preparation: Download and preprocess the datasets (Pubmed, Cora, and Citeseer). For each dataset, calculate the homophily and heterophily percentages based on node feature similarities and edge connectivity. This involves computing the proportion of edges between nodes with similar (homophily) or dissimilar (heterophily) features.

2. Model Implementation: Implement and train GNN-based models, specifically Graph Convolutional Networks (GCN), Graph Attention Networks (GAT), and GraphSAGE, for node classification tasks. Each model is trained and evaluated on the datasets to assess its performance in relation to the dataset's homophily characteristics.

3. Performance Analysis: Generate visualizations comparing the test accuracy, validation accuracy, and training accuracy with corresponding homophily values for each dataset. Additionally, set up a timer to estimate the computation time required for each dataset, taking into account its homophily percentage. This allows for assessing both performance and efficiency.

4. Insight Extraction: Analyze the results and summarize the key insights and findings derived from the visualizations. Specifically, focus on understanding how varying levels of homophily affect the performance of the GNN models, and what implications this has for model selection in real-world applications.

5 Delivered Experiment

Dataset	Homophily/Heterophily	GCN Performance	GCN Time	GAT Time	GraphSAGE Time
Cora	81% / 19%	80.8%	6.61 sec	8.94 sec	16.70 sec
Citeseer	73.55% / 26.45%	68.8%	19.04 sec	19.45 sec	62.90 sec
Pubmed	80.24% / 19.76%	78.9%	26.28 sec	36.45 sec	70.80 sec

Table 1: Homophily, GCN Performance, and Execution Times for Different Datasets

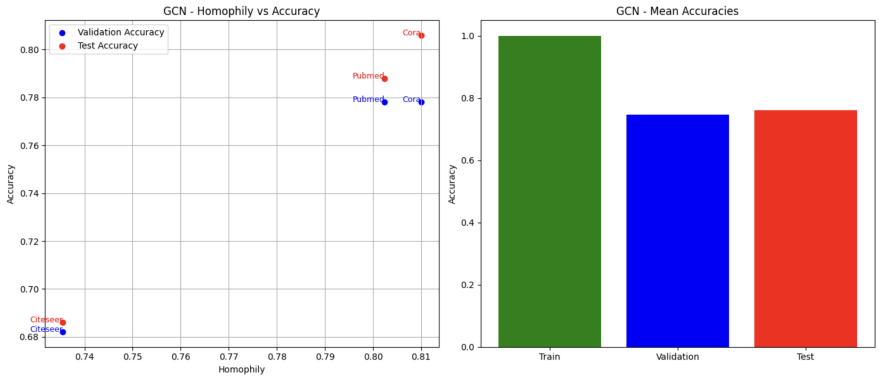

Figure 1: GCN Performace on Test/Validation set

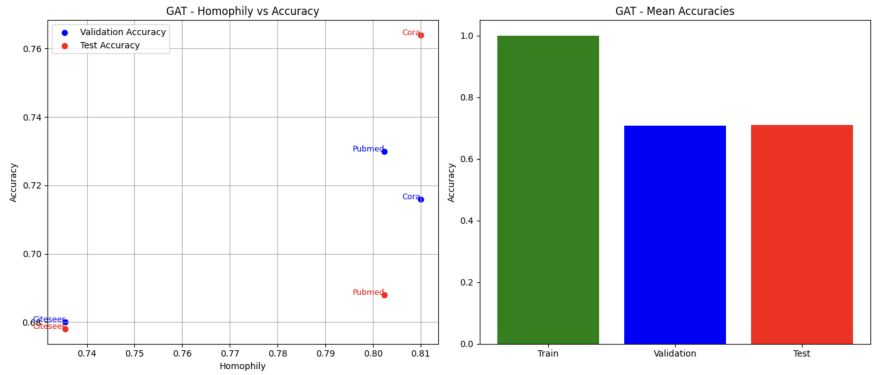

Figure 2: GAT Performace on Test/Validation set

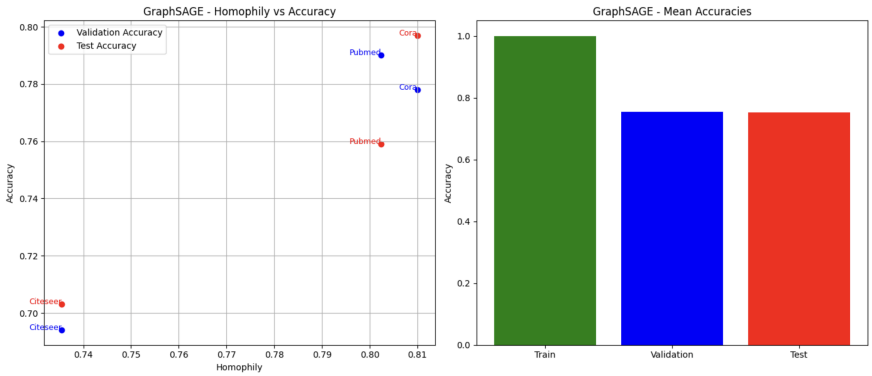

Figure 3: GraphSAGE Performace on Test/Validation set

First, as shown in Table 1, the Cora dataset has the highest homophily value and also achieves the highest performance. Among the three datasets, Cora and Pubmed generally exhibit higher validation accuracy

compared to Citeseer, which has a lower homophily rate. However, when implementing the GAT model, as seen in Figure 3, the validation accuracy of Pubmed is actually closer to that of Citeseer, despite Pubmed having a higher homophily. This suggests that while homophily typically contributes to improved performance in GNN models (in 2 out of 3 experiments), it is not always a guarantee. This could be because nodes with similar features are easier and more consistent to learn from.

Additionally, I found that Citeseer has a lower homophily and also more features in the network, which could explain why the model might become easily distracted during training and classification. The Citeseer dataset has approximately 73% homophily, so we can infer that when homophily is lower than roughly 75% in a network, the performance of GNN models is typically worse. However, homophily does not always correlate with better performance, as demonstrated in the GAT experiments where the performance of the Pubmed and Citeseer datasets, with high and low homophily respectively, was quite similar.

Regarding the computation time for node classification, Table 1 reveals that Pubmed consistently takes longer than the other datasets. Therefore, I conclude that the computation time is more influenced by the number of nodes in the dataset/network rather than the homophily rate.

6 Research Conclusion

In conclusion, we found that high homophily can lead to better performance of GNN models, but it is not always the case. Sometimes, the performance may be similar despite differences in homophily rates, depending on factors such as the characteristics of the dataset, the advantages of the model, and other considerations. This suggests an area for future research: exploring the most significant factors, beyond homophily, that can impact the performance of GNN models. Another potential direction for future work is to investigate the effect of eliminating minor variant nodes within a certain range to see if this strategy can help improve performance, and also to examine how we define variant nodes.