Expressive Power of Message Passing in Graph Neural Networks

Dissertation submitted in partial fulfilment for the award of the degree

Master of Technology in Computer Science

by

PRATAP DEY

Roll No.: CS2221
M.Tech, 2nd year

Under the supervision of
Dr. Malay Bhattacharyya

Computer and Communication Sciences Division
INDIAN STATISTICAL INSTITUTE

June, 2024

CERTIFICATE

This is to certify that the work presented in this dissertation titled “Expressive Power of Message Passing in Graph Neural Networks”, submitted by Pratap Dey, having the roll number CS2221, has been carried out under my supervision in partial fulfilment for the award of the degree of Master of Technology in Computer Science during the session 2023-24 in the Computer and Communication Sciences Division, Indian Statistical Institute. The contents of this dissertation, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

[illegible]

Dr. Malay Bhattacharyya
Associate Professor, Machine Intelligence Unit
Associate Member, Centre for Artificial Intelligence and Machine Learning
Associate Member, Technology Innovation Hub on Data Science, Big Data Analytics, and Data Curation
Indian Statistical Institute, Kolkata

Acknowledgements

I am profoundly grateful to *Dr. Malay Bhattacharyya* for his exceptional mentorship, support, and encouragement throughout my postgraduate studies and the course of my dissertation. His insightful guidance has been invaluable, and words cannot adequately express my appreciation for his assistance and dedication. The opportunity to work under his supervision has been an immensely rewarding experience for a burgeoning researcher like myself.

My sincere thanks also go to the CSSC and the ISI Library for their indispensable support and resources, especially in times of urgent need.

I am deeply thankful to my parents, teachers, and friends for their constant support at every stage of my life. They were always with me. Their unceasing encouragement has been crucial throughout my academic journey, particularly during the research undertaken in the past year.

Pratap Dey

Date: 10-06-2024

Pratap Dey
Roll No.: CS2221
M.Tech, 2nd year
Indian Statistical Institute

Abstract

Graph neural networks (GNNs) have become essential tools for graph representation learning, with models like Graph Convolutional Networks (GCNs), GraphSAGE, and Graph Attention Networks (GATs). It has achieved notable success in various applications. HSGATv2, a recent advancement, enhances attention mechanisms for nodes with the same class label. However, traditional GNN weight assignment methods, which often depend on node degrees or pair-wise representations, are less effective in heterophilic networks in which the labels or properties of connected nodes differ. It has been shown that most existing models are primarily prone to homophilic graphs and lack generalization to heterophilic settings, and multi-layer perceptrons and other models that neglect the graph structure sometimes exceed these models in terms of performance. This dissertation explores the effectiveness of GNNs in node classification tasks within heterophilic or low-homophily environments, where many common GNNs fail to perform well. So, in this dissertation, we try to address it and introduce a representation learning methodology that is comparatively suitable for both homophilic and heterophilic graphs. By thoroughly examining local structure and heterophily distributions, our approach effectively manages networks with diverse homophily ratios. Additionally, we propose a regularized optimization function to enhance model adaptability to any graph structure. Our evaluations on various node classification datasets demonstrate that the proposed method is competitive to the standard baseline models, and promisingly generalizable.

Contents

List of Figures

List of Tables

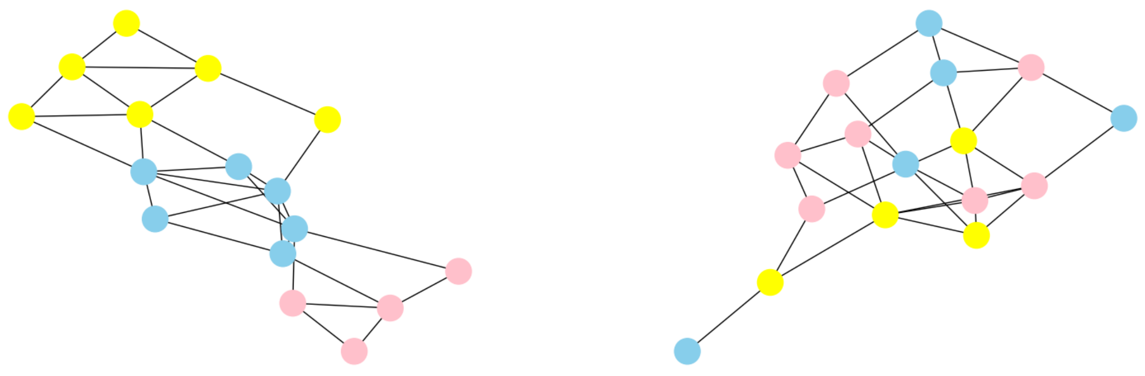
1 Introduction

In learning expressiveness and representations from graph data, Graph Neural Networks (GNNs) have made substantial advancements. Because of their ability to handle irregular graph structures, GNNs are successfully used in various fields across various domains including social analysis, computer vision, and natural language processing, bioinformatics, physics, and many others. For instance, GNNs have been employed to design molecules with particular chemical properties. Most importantly, their applications are mainly divided into three types of tasks: tasks at the node, edge, and graph levels. This dissertation focuses on how effective GNNs are for supervised node classification. The goal is to use partially labeled node attributes and the network topology to predict unknown node labels.

Prior research has focused mainly on networks having strong homophily, in which linked nodes frequently possess same class or identical features. For example, friends often have similar political views or ages, academic papers frequently cite other papers from the same field; Students in the same school or study group are likely to have similar academic interests or performance level; Residents in the same neighborhood often share socio-economic status, cultural background, or lifestyle preferences. This homophily notion is captured by GNNs through feature spreading and aggregation inside graph neighborhoods.

However, real-world networks sometimes display heterophily, where the connected nodes have unique properties or belong to separate classes. Examples include dating networks, where individuals frequently interact with people of the opposite gender; Protein structures where different amino acids connect; In collaborative work environments where in a multidisciplinary project team, members from different professional backgrounds (e.g., engineers, marketers, designers) work together, contributing diverse perspectives and expertise; online purchasing networks where fraudsters connect with accomplices rather than other fraudsters. This dissertation explores GNNs' ability to represent data in networks with varying degrees of homophily and heterophily.

Graph Convolutional Networks (GCNs) [12, 1] as well as their variations use structural information, like symmetric normalization of node degrees, to set edge weights. GraphSAGE [8] expands on this by using more than just averaging for aggregation and treats the pointing node feature and neighbor node features separately in its sampled neighborhoods. Meanwhile, Graph Attention Networks (GATs) [18, 3, 10] use a self-attention mechanism to create attention scores for each node's neighbors, allowing nodes to selectively focus on the most relevant neighbors. Different attention-based weight assignment methods have been developed using various pair-wise attention functions.



(a) High Homophily (b) Low Homophily

Figure 1.1: Depending on the label assignment y_v, the same underlying graph G can exhibit: Figure 1.1 (a) high homophily ($h_G = 0.80$, left) and Figure 1.1 (b) low homophily ($h_G = 0.20$, right).

These models mainly differ in how they aggregate node representations with different weight assignments, helping each node combine its features with those of its neighbors. They are also interpretable because the attention scores indicate the relevance of neighboring nodes. However, recent studies [20, 15] have found that these weight assignment methods do not perform well in heterophilic graphs, in cases where the labels or characteristics of linked nodes vary, as opposed to homophilic graphs. In such cases, even simple models like multilayer perceptrons (MLPs), which ignore the graph structure, can sometimes perform better than existing GNNs.

One recent approach aims to reduce noise in the graph using an attention mechanism. This involves a loss function that promotes nodes to concentrate more on other nodes in their class and less on those of different classes. This method, called HS-GATv2 [4], has shown that when increased edge homophily is present, GNNs function better.

A key interest in GNN research is designing effective weight assignment schemes to determine the importance of different node and neighbor representations. However, in real-world scenarios, calculating edge heterophily is challenging, especially when most node labels are unknown, such as in semi-supervised settings. Previous attempts to estimate edge heterophily using a pre-trained MLP based on node labels have been difficult because accurate labels are hard to predict.

Recent studies [13, 14, 19] indicate that local distribution, which describes the characteristics of the local neighborhood structure, is crucial for handling heterophilic graphs. Additionally, the topological structure, node features, and positional identity are important for estimating effective local distributions and improving the weight assignment scheme.

In this dissertation, we first highlight the limitations of Graph Neural Networks (GNNs) in learning from networks exhibiting heterophily, a challenge often overlooked due to evaluations on a limited set of benchmarks with similar properties. We propose a straightforward solution by designing a GNN that assigns different weights to edges based on their heterophilic types. This allows the GNN to develop an effective local attention to similarities or patterns in the graph, enabling nodes to gather relevant information from both similar and dissimilar nodes.

More specifically, we create an attention mechanism that considers prior edge heterophily, fully utilizing the information of local structure. Our proposed GNN model is comparatively suitable for both homophilic and heterophilic graphs. Additionally, we introduce a regularized optimization function to increase the model's capacity to adjust to different graph configurations. We test the model's effectiveness using benchmark graph datasets with high homophily ratios (e.g., Cora, CiteSeer, Pubmed) and low homophily ratios (e.g., Cornell, Wisconsin, Texas, Actor, Squirrel).

2 Related Work

2.1 Graph Attention Networks

The Graph Attention Network (GAT) [18] is an effective graph neural network architecture. designed for learning some interactions and relationship in graph-structured data. It produces an attention mechanisms to prioritize important nodes. By dynamically weighting the influence of neighboring nodes, GAT offers enhanced performance in various graph tasks. After GAT, several variations have been proposed that use different ways to find these attention coefficients. Additionally, other approaches have been proposed to enhance GAT further. For example, SuperGAT which employs a self-supervised strategy to learn attention coefficients based on edge information. CS-GNN improves GAT by predicting attention coefficients using separate scoring representations instead of combined ones. GATv2 [2] points out that the attention scores learned by GAT do not account for the node's own representation and offers an improved version by altering the sequence of operations. Brody et al. showed that GATv2 [2] implements dynamic attention, whereas the earlier GAT model depicts static attention mechanism. These attention based Graph Neural Networks (GNNs) work well under the assumption of homophily, meaning they measure similarities or distances using various pair-wise attention functions, but they are less effective in handling networks with heterophily. In addition, one recent work with the goal of mitigating noise within graphs by employing an attention mechanism. This method encourages nodes to pay more attention to other nodes that are similar to them and less attention to those that are different. It's called HSGATv2 [4], and it has shown that when connections between nodes are more similar, graph neural networks work better. Previous studies [21, 9] use a pre-trained MLP to predict the edge heterophily based on node labels.

2.2 Beyond Homophily GNN Designs

According to recent studies, graph neural networks (GNNs) face challenges when trying to classify nodes, especially in networks where the labels and attributes of connected nodes can often be different. This is called heterophily. To address these challenges, the authors of the H2GCN [22] model have identified several key design strategies that enhance GNN learning in heterophilous settings. These strategies include separating pointing node and neighbor node embeddings, utilizing neighborhoods that are in higher order, and combining intermediate representations. The integration of these strategies in the H2GCN model has led to significant performance improvements. Empirical tests reveal that these design enhancements can increase accuracy by up to 40% on synthetic networks and 27% on real-world networks with heterophily, while also delivering competitive performance in homophilous networks. Also there are some path based approaches for designing GNN models which give better results over graphs with various homophily ratios.

Recent research [19, 13] suggests that understanding the local distribution, or the traits of nearby nodes, is essential for effectively dealing with graphs that exhibit heterophily. A recent work [19] uses node class labels and heterophily preference matrix to assign attention weights to overcome the limitation stated. Moreover, factors like the graph's structure, the features of individual nodes, and where they are located play significant roles in accurately determining these local distributions and enhancing how weights are assigned.

3 Preliminary and Notations

3.1 Notation

Consider the undirected graph $G = (V, E)$, where V represents nodes and E represents edges [7], where M, number of edges and N , number of vertices, i.e.,$M = |E|$ and $N = |V|$. Explicitly, $V = \{v_1, v_2, \ldots, v_N\}$. Each node v_i is associated with feature $x_i \in \mathbb{R}^d$. The set of all the features in all the nodes is denoted by $X \in \mathbb{R}^{N\times d}$. Let $A = [a_{ij}] \in \{0,1\}^{N\times N}$ be the associated adjacency matrix corresponding to graph G, where nodes v_i and v_j are linked if $a_{ij} = 1$, and else $a_{ij} = 0$. Neighbors of node v that are precisely i hops distant are indicated by $N_i(v)$(Note that, edges of G may have self-loops). For instance, $N_1(v) = \{u : (u, v) \in E\}$ denotes v's adjacent neighbors. Let the vector $Y = [y_1, y_2, \ldots, y_N]^\top \in \{1, \ldots, C\}^N$ hold the nodes corresponding class labels, and C denotes the number of classes.

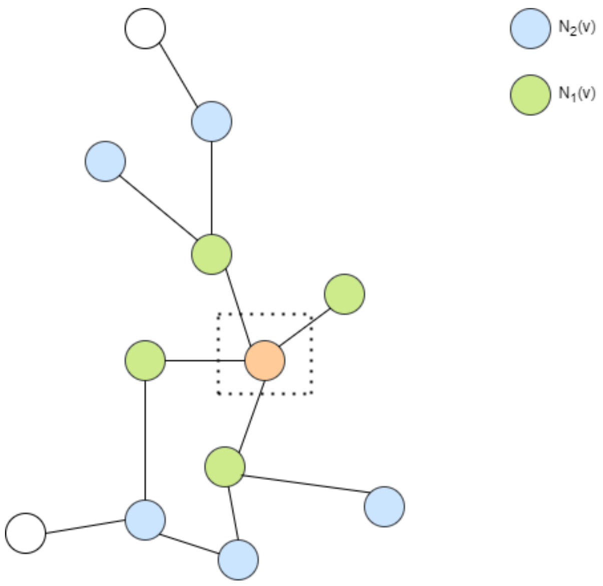


Figure 3.1: Visualization of the neighborhoods around a central node v. The immediate neighbors $N_1(v)$ are shown in light green, and the neighbors at exactly two hops away $N_2(v)$ are shown in sky-blue. The central node v is displayed in yellow.

The degree of node v_i is given by $d_i = \sum_j a_{ij}$, and the degree matrix D is defined as $D = \text{diag}(d_1, d_2, \ldots, d_N)$. Additionally, $\hat{N}_1(v_i) = N_1(v_i) \cup \{v_i\}$ represents the set containing the node v_i and its immediate neighbors.

The target in the supervised node classification is to find a function that maps nodes to labels given a collection of training nodes. Our primary goal is to acquire the knowledge of a mapping $f : \{v_1, v_2, \ldots, v_N\} \to \{0, 1, \ldots, C\}$.

3.2 Message Passing in Graph Neural Networks

In this section, we provide the mathematical formulations for message passing in three popular designs of graph neural networks: (i) Graph Convolutional Networks (GCN) [12, 1], (ii) GraphSAGE [8], and (iii) Graph Attention Networks (GAT) [18]. First, we initialize $h_v^{(0)} = x_v$ for every node v. As additional notations, An activation function is σ. The parameters of layer k that can be learned are $M^{(k)}$.

Graph Convolutional Networks (GCN) [12]: Message Passing at Layer k is formulated as

$$h_v^{(k)} = \sigma\left(\sum_{u\in\hat{N}_1(v)} M^{(k)} \cdot \frac{h_u^{(k-1)}}{\|\hat{N}_1(v)\|}\right), \quad \forall v \in V$$

GraphSAGE [8]: Message Passing at Layer k is formulated as

$$h_v^{(k)} = \sigma\left(W^{(k)} \cdot \text{CONCATENATION}\left(h_v^{(k-1)}, \text{AGGRG}\left(\{h_u^{(k-1)} \forall u \in N(v)\}\right)\right)\right), \quad \forall v \in V$$

Graph Attention Networks (GAT): Message Passing at Layer k is formulated as

$$h_v^{(k)} = \sigma\left(\sum_{u\in N_1(v)} \alpha_{vu} \cdot M^{(k)} \cdot \left(h_u^{(k-1)}\right)\right), \quad \forall v \in V,$$

where α_{vu} represents the normalized attention coefficient between nodes u and v. More specifically calculating the normalized attention weights has the following procedure

$$e_{vu} = a^T \cdot \text{CONCATENATION}(W^k \cdot h_u^{k-1}, M^k \cdot h_v^{k-1})$$

$$\alpha_{vu} = \frac{\exp(e_{vu})}{\sum_{j\in N_1(v)} \exp(e_{vj})}$$

3.3 Edge Homophily Ratio

In our study, within class labels, we investigate the idea of heterophily. To measure the degree of homophily inside a graph, we provide the edge homophily ratio h_G, use this term in graphs to delineate characterized by distinct levels of homophily or heterophily:

Definition: The percentage of connections in a graph that connect nodes with the same class label, or intra-class edges, is known as the edge homophily ratio h_G, over the graph's entire number of edges. Mathematically edge homophily ratio can be explicitly written as following:

$$h_G = \frac{|\{(v_p, v_q) : (v_p, v_q) \in E \land y_p = y_q\}|}{M}$$

We note that,$h_G \in [0, 1]$. Graphs with strong homophily possess a lot of edges connecting nodes such that it has the same class label. So, the edge homophily ratio is nearly equal to one, showing that most edges link nodes of the same class. On the other hand, graphs with strong heterophily have fewer edges between nodes of the same class. This means edge homophily ratio is nearly equal to zero, indicating that there are nodes with the same class label are connected by fewer edges.

The edge homophily ratio provides an overview of how often nodes with the same class label are connected by edges in the graph. However, It's essential to note that heterophily, which we're exploring in this study, isn't the same as heterogeneity. Heterogeneity, as defined in network science, refers to networks with two or more different kinds of nodes, as well as different connections between them. For instance an example could be, knowledge graphs typically exhibit heterogeneity.Conversely, homogeneous networks consist of only one type of node, like users, and one type of relationship, like friendship. Every node and every edge in these networks is of the same kind.

4 Limitation in earlier GNN Models

In this section we point out the weakness of Using Graph Neural Networks to learn from networks with heterophily, a problem often overlooked because most evaluations are done on benchmarks with similar characteristics. While many GNN models have been developed, most assume homophily and struggle with heterophily. Table 1 [22], for instance, shows the average accuracy on classification of a number of the best graph neural network models on simulated benchmark data (syn-cora), in which we may change the edge homophily ratios. We look at two edge homophily ratios situations. All current methods perform worse than a Multilayer Perceptron (MLP), which is a base model that is independent of graphs and just classifies nodes based on their attributes only. We can clearly see, GCN [12] and GAT [18] perform significantly worse than MLP, demonstrating that techniques that work well in settings of high edge homophily (h = 0.7) could not perform well in networks that have low edge homophily.

Models	$h_G = 0.1$	$h_G = 0.7$
GCN	37.14	84.52
GraphSAGE	70.89	85.56
GAT	33.11	84.03
MLP	74.85	88.28

Table 1: An use-case of a heterophilic environment with an edge homophily score of $h_G = 0.1$. In such a scenario, traditional Graph Neural Networks (GNNs) often struggle to generalize effectively. On the other hand, a typical homophilic setting, where the edge homophily score is $h_G = 0.7$, tends to be more favorable for these models.

Comprehensive analysis of benchmark datasets is presented in the appendix section(9.1, 9.2).

5 Proposed Method

This section introduces our proposed model, which uses local structures and heterophily distribution patterns to handle graphs with different levels of similarity among nodes so that it can address the limitations stated. In the Empirical Analysis section, we have shown performances of our models in comparison with the existing models, with respect to F1 scores as stable accuracy measure in graphs with different homophily ratios. To begin with, we introduce an attention mechanism in the heterophily settings, and state how modeling the differences in edges can be beneficial. Specifically, our Graph Neural Network (GNN) gives different weights to edges based on their category types so that it can learn effective local attention strategies, helping nodes gather relevant information from their diverse neighbors.

5.1 Proposed Attention Scheme

This section develops an attention mechanism that recognizes heterophily by allowing it to adaptively assign weights by fully utilizing the differences in edge types.

5.1.1 Motivation

We have discovered two essential design strategies that can enhance the the effectiveness of GNN's in heterophily settings:

(i) Separation of Node-Embedding and Neighbor-Embedding:

In this approach, The embedding of every node is represented independently of the sum of the embeddings of its neighbors, acknowledging the differences that may exist in heterophily contexts. Formally, for a node v, the representation learning at layer k can be written as:

$$h_v^{(k)} = \text{COMBINE}\left(h_v^{(k-1)}, \text{AGGRG}\left(\{h_u^{(k-1)} : u \in N_1(v)\}\right)\right)$$

Here, important thing is to note that the neighborhood $N_1(v)$ exclude the node v.

Intuition 1: By definition, in heterophily contexts, the class label y_v and initial features x_v of a node differ from those in its surrounding neighbors $\{(y_v, x_u) : u \in N_1(v)\}$. In any way, The final embeddings produced by typical GCN designs, which combine embeddings by averaging as the COMBINE function, are comparable for adjacent nodes. This is suitable for graphs with high homophily but problematic for high heterophily settings. In such cases, differentiating between neighbors belonging to distinct classes with identical learnt representations is challenging. Hence, employing a COMBINE function that maintains separate representations so that it is more expressiveness for each node v and its neighbors $N_1(v)$.

(ii) Utilizing Higher-Order Neighborhoods:

Information from higher-order neighborhoods in each layer k, which go beyond one-hop neighbors, are aggregated in this scheme.:

$$h_v^{(k)} = \text{COMBINE}\left(h_v^{(k-1)}, \text{AGGRG}\left(\{h_u^{(k-1)} : u \in N_1(v)\}\right), \text{AGGRG}\left(\{h_u^{(k-1)} : u \in N_2(v)\}\right), \dots\right)$$

In this context, Neighbors of node v that are precisely i hops distant are indicated by $N_i(v)$.

Intuition 2: Higher-order neighborhoods enable GNNs to perform better in heterophily settings. Although immediate neighbors might have different labels, more distant It's conceivable that nodes with the same label can be found in neighbors. This pattern aids GNNs in making more accurate predictions. Recent studies [22] indicate that leveraging extended neighborhoods significantly improves GNN performance, particularly in binary classification tasks.

5.1.2 Adaptive Attention: Preprocessing

Adaptive attention mechanism that we propose in this section which is designed to dynamically allocate weights by thoroughly utilizing edge heterophily. The local distribution concept is employed to characterize the distribution based on certain features within each local neighborhood. Recent research uses node labels to calculate these distributions. In this work we calculate it using local attributes.

The local distribution for a node v is generally derived from its neighborhood using the following generalizable expression:

$$d_v = \Phi\left(x_v, \{x_{u_1}\}_{u_1 \in N_1(v)}, \{x_{u_2}\}_{u_2 \in N_2(v)}, \dots\right)$$

where the function $\Phi(\cdot)$ is tasked with estimating the appropriate local distribution based on specific measurements; $d_v \in \mathbb{R}^h$ denotes the local distribution of node v, and Neighbors of node v that are precisely i hops distant are indicated by $N_i(v)$.

Each element $d_{v_k} \geq 0$ in d_v indicates the probability of node v being assigned to category k, with the sum of all elements in d_v equal to 1 i.e., $\sum_k d_{v_k} = 1$.

This estimation enhances the node classification process by accurately capturing the characteristics of the local neighborhood.

Now, in this stage our goal is to generate local distribution. We delve into extracting local information from the smoothed features within a local neighborhood. This approach is driven by the following main considerations.

Recent studies demonstrate that local distributions generated from node features and topological information can enhance weight assignment schemes in practical applications where heterophily is a common pattern to observe.

As we just discussed in motivation subsection, We have discovered two crucial designs that, when properly combined, can enhance the functionality of GNN models in heterophily scenarios, multi-hop graphSAGE [8] can be one of such choice to be in use. More specifically, GraphSAGE [8] can perform well by capturing these smoothed local features, as the local distributions derived from node labels are quite distinct. By using two-layer GraphSAGE, we can better utilize the local information and topological context to enhance the model's adaptability and accuracy in diverse network structures. Additionally we apply SOFTMAX to each of the smoothed features corresponding to each node.

So we assume, $D = \{d_1, d_2, \ldots, d_N\} \in \mathbb{R}^{N \times h}$ which represents the local distributions for each node in the network. Note that hidden dimension h of each smoothed local distribution may not be the same as node class labels C.

As mentioned earlier, we operate under the assumption that the local distribution reflects the corresponding node categories (classification). Thus, by utilizing D, we can approximately model the underlying heterophily by combining the learnable parameter matrices associated with GraphSAGE [8]. Consequently, each d_v represents the category distribution among the underlying categories (classification) of node v.

The underlying heterophily matrix of each edge is formulated as follows:

$$\mathbf{P}_{vu} = d_v \otimes d_u$$

where $\mathbf{P}_{vu} \in \mathbb{R}^{h \times h}$ represents the probability distribution of the $h \times h$ heterophilic types for the edge e_{vu}. The element in $\mathbf{P}_{vu}$ can be expressed as $(\mathbf{P}_{vu})_{ij} = d_{vi} \cdot d_{vj}$, signifies the heterophily matrix of the edge e_{vu} connecting two nodes with categories (classification) i and j respectively.

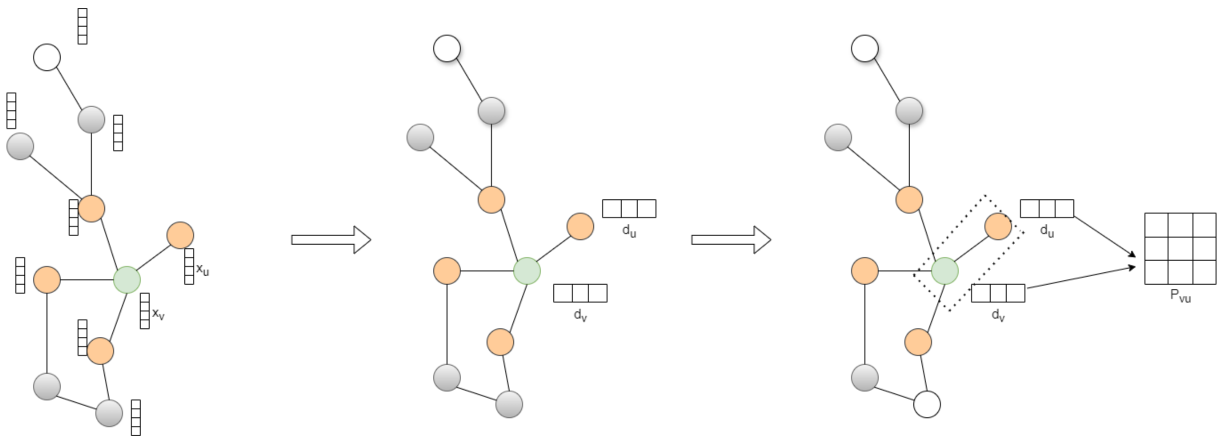


Figure 5.1: The diagram illustrates how our model leverages the local structure and distribution patterns within the data.

Given that $\sum_i d_{vi} = 1$ and $\sum_j d_{uj} = 1$, therefore, $\mathbf{P}_{vu}$ is also a probability matrix, i.e., $\sum_i \sum_j (\mathbf{P}_{vu})_{ij} = 1$. So, in one sentence we can say, $\mathbf{P}_{vu}$ indicates the heterophily matrix of the edge e_{vu}, and this can be is used in the adaptive attention scheme to generate the attention coefficients.

5.1.3 Attention Scheme

As we have discussed in the earlier section $\mathbf{P}_{vu} \in \mathbb{R}^{h\times h}$ be regarded as the probability distribution of the $h \times h$ heterophilic types for the edge e_{vu}. The heterophily matrix $\mathbf{P}_{vu}$ for each edge is then explained using another learnable parameter matrix to generate the attention coefficient in each layer. In the k-th layer, the parameter matrix $\mathbf{W_1}^{(k)} \in \mathbb{R}^{h\times h}$ is optimized during training. The attention coefficient for each edge e_{vu} is adaptively predicted as follows:

$$w_{vu}^{(k)} = \langle \mathbf{P}_{vu}, \psi(\mathbf{W_1}^{(k)})\rangle_F = \mathrm{tr}\left(\mathbf{P}_{vu}^T \psi(\mathbf{W_1}^{(k)})\right),$$

where $\langle\cdot,\cdot\rangle_F$ represents the Frobenius inner product, an inner product operation between two matrices, and The symbol $\mathrm{tr}(\cdot)$ represents a matrix's trace. The function $\psi(\cdot)$ performs element-wise computation to determine the each heterophilic edge type's significance depending on real-valued parameters. For each parameter $w^{(k)}$ in $\mathbf{W_1}^{(k)}$, $\psi(w^{(k)}) = \max(\lambda \cdot w^{(k)}, 0)$, where $\lambda > 0$ is a gradient scaling factor used to adjust the gradient magnitude of each element in the learnable matrix $\mathbf{W_1}^{(k)}$. If L is the associated loss function that is aimed to minimize as the optimization problem, this gradient will indicate how L changes with changes in $\mathbf{W_1}^{(k)}$.

So, in the final stp, unlike traditional normalization techniques, we introduce a Neighbor Norm to normalize the attention coefficients. This is defined as:

$$\alpha_{vu}^{(k)} = \frac{w_{vu}^{(k)}}{\sum_{i\in\hat{N}_1(u)} w_{ui}^{(k)}}$$

,where $\hat{N}_1(u) = N(u) \cup \{u\}$ represents the set containing the node u and its neighbors. Therefore each attention coefficient $\alpha_{vu}^{(k)}$ for the edge e_{vu} is normalized by the adjoining nodes' weighted degree. This approach assumes these connections to general neighbors (nodes with lower weighted degrees) are more reflective of the node's characteristics compared to connections to popular neighbors (nodes with higher weighted degrees).

5.2 Our Attention Network

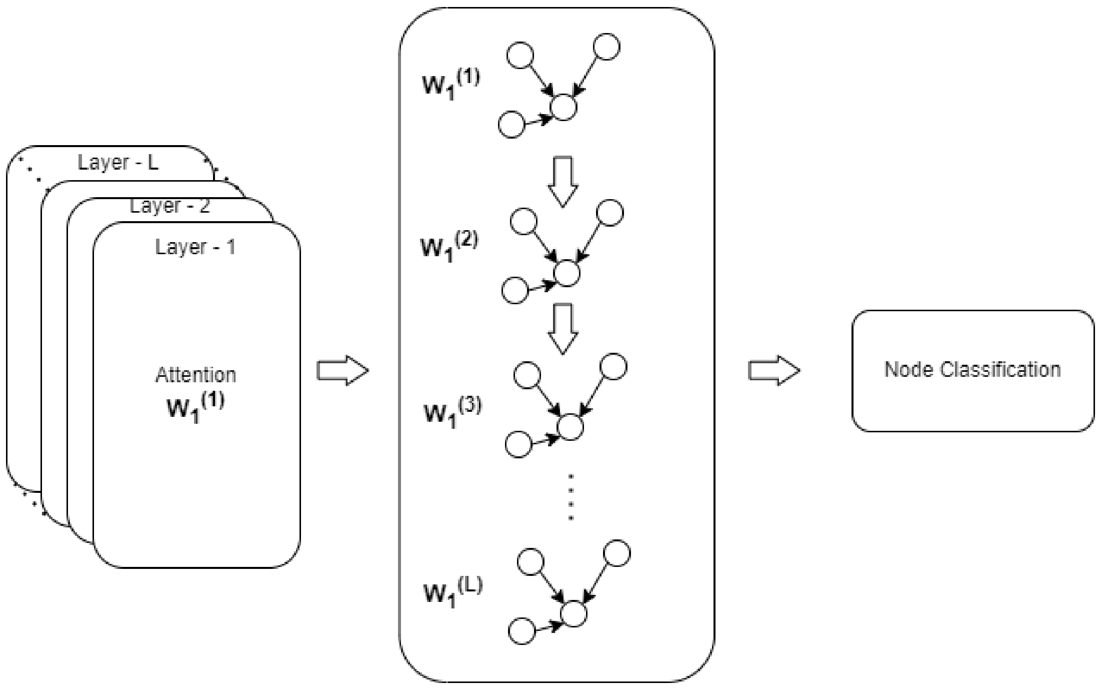


Figure 5.2: The figure illustrates the overall structure of the network and demonstrates its application to the node classification task.

Now we are ready to introduce the network layer updation strategy. The normalized attention coefficients are determined in the earlier section. The message aggregation step is similar to the existing GAT [18] mechanism message passing and aggregation. So, the following is the execution of aggregation for the $(k+1)$-th layer:

$$h_v^{(k+1)} = \sigma\left(\sum_{u\in\hat{N}_1(v)} \alpha_{vu}^{(k+1)} W_2{}^{(k)} h_u^{(k)}\right),$$

where $W_2{}^{(l)}$ is a parameter matrix, and the activation function is represented by σ.

5.3 Custom Loss Function

Overall, we want a model that should give better accuracy over any graph with different edge homophily ratios. Our attention mechanism is good to capture several distinct heterophilies in the graphs but to ensure it is performing well for high homophily graphs as well, For the training, we suggest a personalized loss function.

By giving labels that indicate whether an edge is between two separate category nodes or the same category node, we are able to oversee the attentions across the edges. Denote by V_{train} a set containing the nodes in the training set. Consider E_{train} as well. Any edge that can be found where both nodes in the V_{train} correspond is included in the set that contains the training edge set, Specifically,

$$E_{\text{train}} = \{(v_p, v_q) : (v_p, v_q) \in E \wedge v_p \in V_{\text{train}} \wedge v_q \in V_{\text{train}}\}.$$

In particular, our custom loss function consists of the next two terms: (i) the loss due to overseeing averaging attention for edges (between training set nodes) [let's say L_E]; and (ii) the cross-entropy loss (between the ground truth of node class labels and model predictions, let us say L_V). Let us write it mathematically as follows:

$$L = L_V + f(h_G).L_E$$

where the two terms are formally written as,

$$L_V = -\frac{1}{|V_{\text{train}}|} \sum_{v_i \in V_{\text{train}}} \sum_{c=1}^{C} y_{(v_i,c)} \log(p_{(v_i,c)})$$

$$L_E = -\frac{1}{L|E_{\text{train}}|} \sum_{l=1}^{L} \sum_{e \in E_{\text{train}}} \left(y_e \log(\sigma(w_e^{(l)})) + (1 - y_e) \log(1 - \sigma(w_e^{(l)})) \right)$$

$y_{(v_i,c)}$ shows whether node v_i is a member of class (let us say) c or not, $p_{(v_i,c)}$ denotes the probability that node v_i is a member of class c, $w_e^{(l)}$ denotes The edge e's unnormalized attention score in the l-th message passing layer, and y_e is symbolizes the source and target nodes are in the same class, this indicates the edge's label, which is 1; otherwise, it indicates that they are in separate classes. Interestingly, $f(h_G)$ is monotonically increasing continuous function of homophily ratio corresponding to that graph or network, with a constraint such that f ranges to $(0, 1)$. Note that h_G varies in the open interval $(0, 1)$.

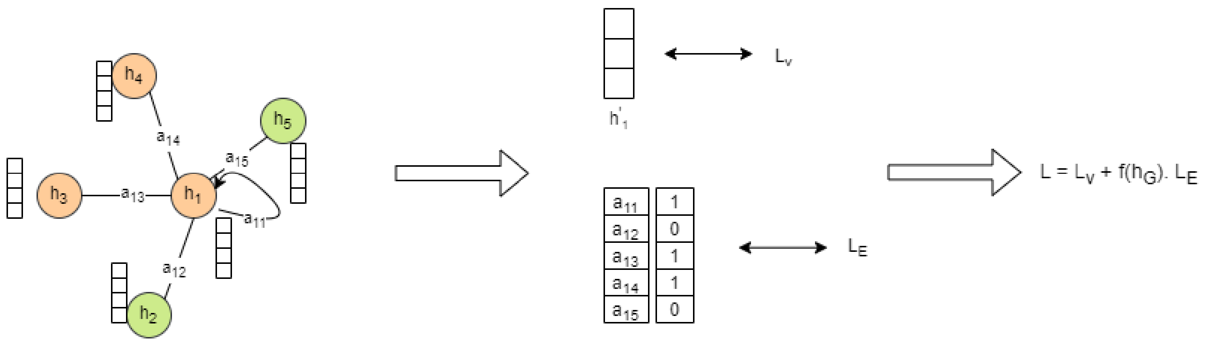


Figure 5.3: The diagram illustrates our custom loss function, highlighting how it integrates two key components L_V and L_E.

The supervision will be regularized using this parameter $f(h_G)$ with the goal of reducing the loss function. Accordingly, the corresponding weight for edges connected to the same category in a network with a high homophily ratio should be larger, and vice versa.

5.4 Comparison with GATs

Several graph attention networks (GATs) have been examined by us. This section presents a comparison between the current variants of GATs models and our proposed attention-based approach. At the moment, attention-based GNNs typically take advantage of a self-attention strategy, in which the representations of the nodes to which the edges are correlated with the attention coefficients on the edges are as follows:

$$w_{vu}^{(k)} = \Phi^{(k)}\left(h_v^{(k-1)}, h_u^{(k-1)}\right),$$

where $h_v^{(k)}$ is the representation of node v in the k-th layer and $w_{vu}^{(k)}$ is the attention coefficient of edge e_{vu} in the k-th layer. Also, the well-designed attention function $\Phi^{(k)}(\cdot)$ differs from the designs used in the current approaches.

For example in simple GATv2 [2] proposed in 2021, the un-normalized attention score calculated as,

$$w_{vu}{}^{(k)} = a^T \cdot \text{LeakyReLU} \cdot (W^k \text{CONCAT}(h_u^{k-1}, h_v^{k-1}))$$

Then, the attention coefficient is normalized by a softmax function, i.e.,

$$\alpha_{vu}{}^{(k)} = \frac{\exp w_{vu}{}^{(k)}}{\sum_{i \in N(v)} \exp(e_{vi})}$$

Conversely, the underlying heterophily matrix P_{vu} for every edge e_{vu} in our suggested attention method is correlated with the local neighborhood in the attributed graph. Generally, one can get the heterophily matrix P_{vu} from its neighbors by applying the following generalizable expression:

$$P_{vu} = \Phi\left(x_v, \{x_{u_1}\}_{u_1 \in N_1(v)}, \{x_{u_2}\}_{u_2 \in N_2(v)}, \ldots\right)$$

The attention coefficients in each layer are then calculated using P_{vu} as,

$$w_{vu}^{(k)} = \varphi^{(k)}(P_{vu}),$$

where $\varphi^{(k)}(\cdot)$ is the function to be used in the k-th layer to construct the adaptive attention coefficient from the heterophily matrix, as previously shown in the section.

6 Empirical Analysis

We thoroughly assess our approach in this part. We used standard node classification datasets with different homophily ratios and Compare it with the state of the art graph neural networks (GNNs) available.

6.1 Dataset Details

Below is a list of the datasets that we used for the experiment: First three dataset Cora, CiteSeer and PubMed [17] are high homophilic graph data whereas rest of the following are low homophilic or heterophilic graph data.

Cora: A network of scientific literature citations makes up the Cora dataset. A paper is represented by each node, and a citation from one paper to another is represented by each edge. The nodes are labeled with one of seven classes, representing the research area of the paper. It has 2708 nodes (papers) and 5429 many edges (citation links). And Every document is represented by a word vector with a value of 0/1, signifying the existence of certain terms from a lexicon including 1,433 distinct terms. Additionally, the seven classes/categories that each node belongs to are: Reinforcement Learning, Rule Learning, Probabilistic Methods, Neural Networks, Genetic Algorithms, Case Based, and Theory.

CiteSeer: CiteSeer is another citation network dataset, similar to Cora but typically larger and more complex. It also represents a network of scientific publications with citation links. It has 3327 nodes (papers) and 4732 many edges (citation links). A binary-valued word vector based on a vocabulary of 3,703 distinct words characterizes each document. Additionally each node belongs to one of the following six groups or classes: information retrieval, machine learning, database, artificial intelligence, human-computer interaction, and algorithms.

PubMed: The PubMed dataset is derived from the PubMed database of biomedical literature. It consists of a citation network of scientific papers from the biomedical domain.It has 19717 nodes (papers) and 44338 many edges (citation links). And a Term Frequency-Inverse Document Frequency (TF-IDF) weighted word vector drawn from a lexicon of 500 distinct terms characterizes each document. Additionally each node is either one of diabetes mellitus experimental, diabetes mellitus type 1 or diabetes mellitus type 2 among these three groups or classifications.

Wisconsin, Texas, and Cornell: Wisconsin, Texas and Cornell are three small-scale datasets derived from university web pages. Each dataset represents a network of web pages (nodes) linked by hyperlinks (edges). Each dataset has a few hundred nodes representing web pages. It consists of typically around five classes, representing different types of web pages (e.g., student, professor, project). A feature vector with a sparse bag of words is used to describe each node, derived from the text content of the web pages.

Actor: Based on the actor co-occurrence network, the Actor dataset (also called the Film dataset) was created. Edges indicate co-occurrences on the same Wikipedia page, while nodes denote actors. It consists of 7600 nodes (actors) and 26752 edges (co-occurrences). Additionally each node is one of five classes based on the actors' age groups.A bag-of-words feature vector taken from the actor's Wikipedia page describes each actor.

Squirrel: The Squirrel dataset is part of the WebKB dataset collection, specifically from the Squirrel [16] web graph. It represents web pages connected by hyperlinks, similar to the Cornell, Texas, and Wisconsin datasets but generally larger and more complex. It consists of 5201 nodes (web pages) and 198353 edges (hyperlinks). A bag-of-words feature vector characterizes each web page.

	Cora	**CiteSeer**	**PubMed**	**Cornell**	**Texas**	**Wisconsin**	**Actor**	**Squirrel**
Nodes	2708	3327	19717	183	183	251	7600	5201
Edges	5429	4732	44338	277	295	466	26752	198353
Feature	1433	3703	500	1703	1703	1703	932	2089
Classes	7	6	3	5	5	6	5	5
h_G	0.82	0.70	0.76	0.31	0.11	0.21	0.16	0.22

Table 2: Datasets description in detail: (i)Cora, (ii)CiteSeer, (iii)PubMed, (iv)Cornell, (v)Texas, (vi)Wisconsin, (vii)Actor and (viii)Squirrel

6.2 Experimental Set-Up

We are primarily interested in exploring and implementing methods in a supervised manner to achieve high accuracy and reliability in our predictive GNN model. For the supervised node classification tasks, we follow a robust and standardized strategy to ensure the effectiveness and fairness of our model training and evaluation processes. This involves a systematic approach to splitting our dataset, optimizing our training process, and selecting hyperparameters.

60% of the nodes are used for training the model and this partition of the data is utilized to fit the model parameters. 20% of the nodes are used for validation and this partition helps in tuning the model's hyperparameters and prevents overfitting. And the remaining 20% of the nodes are used for testing. This partition is used to evaluate the model's performance on unseen data (independent test set). This 60%/20%/20% split [5] is crucial for a balanced and comprehensive evaluation of the model's performance.

We make use of our custom loss function throughout the training procedure, which is specifically adaptively designed to optimize the performance of our node classification tasks in graphs with different homophily ratios. This particular loss function plays an eventual role in guiding the parameters optimization process, ensuring that the model learns the patterns in the data effectively. For this use-case, we have chosen $f(h_G)$ as $(h_G)^2$ to make sure the requirements are satisfied. All the hyperparameters are determined using a grid-search strategy. Also we note that we used Glorot initialization [6] to initialize all the trainable parameters.

We use the Adam optimizer [11] to optimize the model parameters, which efficiently handles sparse gradients and noisy data. Adam [11] incorporates the benefits of two more stochastic gradient descent improvements.: RSMProp and AdaGrad.

During the preprocessing phase, we initialize each of the parameter in ${W_1}^{(k)}$ to $\frac{1}{\lambda}$, ensuring that $\psi(\lambda \cdot {W_1}^{(k)})$ equals one. This setup ensures that all heterophilic edge types are initially assigned equal importance at the start of training.

The learning rate and dropout rate are pivotal hyperparameters that substantially impact the model's performance. We here pre-define the weight decay rate as 5×10^{-5}. To find the other best hyperparameter values, we extensively follow a grid-search strategy. The hidden layer sizes are selected from the set {8, 16, 32, 64}, the learning rate options include {0.001, 0.003, 0.005}, and the dropout rate choices are {0.0, 0.2, 0.4, 0.6}. The combination that achieves the highest validation performance is selected for training the final model.

6.2.1 baselines

We compare our proposed model against five leading methods in the field. which we have re-implemented from scratch. (i) Muti-layer Perceptron, (ii) Graph Convolution Network (GCN) [12], (iii) GraphSAGE [8], (iv) Graph Attention Network (GAT) [18], and (v) Graph Attention Network v2 (GATv2) [2]

6.3 Results

As we are already known, the Benchmark datasets were utilized for the purpose of our model. In situations where the prior edge heterophily is available, our approach extracts effective local distributions. Considering that our proposed approach fails to attain state-of-the-art performance in overall comparisons, but here, we use real-world graphs with varying homophily ratios to illustrate the efficacy of our suggested strategy for supervised categorization and we note that our proposed model is able to provide stable results on node classification tasks on different edge homophily ratios.

	Cora	**CiteSeer**	**PubMed**
MLP	0.749	0.706	0.792
GCN	0.847	0.749	0.859
GraphSAGE	0.856	0.744	0.876
GAT	0.874	0.776	0.843
GATv2	0.865	0.774	0.855
HSGATv2	0.892	0.791	0.859
Our Method	0.843	0.738	0.860

Table 3: This table displays the test accuracy (F1 score) results for high homophily graph benchmarks in node classification, including models such as MLP, GCN, GraphSAGE, GAT, and our proposed method.

The figures (see Fig. 9.4) present evaluations on the datasets (Cora, CiteSeer, PubMed): the confusion matrix is displayed on the left side, while the AUC-ROC curve is shown on the right side.

	Wisconsin	**Texas**	**Cornell**	**Actor**	**Squirrel**
MLP	0.725	0.648	0.702	0.357	0.296
GCN	0.451	0.621	0.459	0.302	0.260
GraphSAGE	0.664	0.620	0.567	0.349	0.321
GAT	0.372	0.540	0.459	0.283	0.271
GATv2	0.451	0.594	0.378	0.288	0.261
HSGATv2	0.588	0.513	0.351	0.289	0.279
Our Method	0.684	0.648	0.756	0.379	0.310

Table 4: This table displays the test accuracy (F1 score) results for low homophily graph benchmarks in node classification, including models such as MLP, GCN, GraphSAGE, GAT, HSGATv2, and our proposed method.

The figures(9.6) present evaluations on the datasets (Actor and Squirrel): the confusion matrix is displayed on the left side, while the AUC-ROC curve is shown on the right side.

7 Conclusion

This dissertation has presented an approach for addressing the challenges posed by heterophily in graph neural networks (GNNs). Our proposed model incorporates an adaptive attention mechanism that leverages local distribution patterns to enhance the performance of GNNs across graphs with varying levels of node similarity. By differentiating between node-embedding and neighbor-embedding separation, and by incorporating higher-order neighborhoods, we have enabled our model to more effectively capture and utilize the heterophilic nature of certain graphs.

The empirical analysis demonstrates that our approach maintains stable accuracy across datasets with different homophily ratios. This stability is achieved through the innovative use of adaptive attention weights, which are dynamically allocated based on the heterophilic types of edges. The local distribution-based attention scheme provides a significant improvement in the ability to distinguish between nodes in heterophilic settings, which is a key limitation of traditional GNNs.

The evaluation of our model on standard datasets, including both high-homophily and low-homophily graphs, reveals its robustness and adaptability. In particular, the results indicate that Our model's performance on high-homophily datasets is close to that of recent techniques that are discussed, while significantly outperforming them on low-homophily datasets. Considering that our proposed approach fails to attain state-of-the-art performance in overall comparisons. This performance is attributed to the custom loss function designed to optimize node classification tasks by considering the homophily ratio, further enhancing the model's capability to generalize across different graph structures.

To put everything in a nutshell, this dissertation tried to contribute to the field of graph neural networks by providing a robust and adaptive solution for handling heterophily, a common challenge in real-world networks. The proposed methodology attempted to bridge the performance gap in any graph with different homophily ratios.

8 Future Work

Our proposed method, though effective, is more complex than existing models. This added complexity can be a downside, especially when dealing with large datasets or networks. Also, due to time limitations, we only explored the method using supervised learning.

A key part of our method is a special loss function that relies on knowing the homophily ratio of the network beforehand. The homophily ratio helps measure how similar connected nodes are. However, in real-life networks, this ratio is usually unknown and hard to determine in advance. Some recent studies suggest that it's possible to estimate this ratio through initial analysis of the network, but this is not yet a common practice.

In the future, our method could be applied in semi-supervised settings. Semi-supervised learning uses both labeled and unlabeled nodes, making it easier to handle large graphs where it is difficult to get all the nodes labeled. This approach could help reduce complexity and improve efficiency. Additionally, exploring how our method works with unsupervised learning could be valuable. Unsupervised learning does not rely on labeled nodes, which could make our method more flexible and applicable in situations where labeling data is not feasible.

Another important direction is improving our loss function to estimate the homophily ratio automatically, without needing to know it beforehand. This would make our method more adaptable and useful for a wider range of real-world networks, where the homophily ratio is often unknown and can change over time. Additionally, our work is limited to the node classification task. Therefore, a significant direction for future research is to expand our approach to more benchmark challenges, including edge-level or graph-level tasks. This expansion would help demonstrate the versatility and robustness of our approach across various types of network analyses.

References

[1] Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing, 2019.

[2] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks?, 2022.

[3] Michail Chatzianastasis, Johannes Lutzeyer, George Dasoulas, and Michalis Vazirgiannis. Graph ordering attention networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(6):7006–7014, Jun. 2023.

[4] Michail Chatzianastasis, Giannis Nikolentzos, and Michalis Vazirgiannis. Supervised attention using homophily in graph neural networks, 2023.

[5] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network, 2021.

[6] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.

[7] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, 2:729–734 vol. 2, 2005.

[8] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[9] Dongxiao He, Chundong Liang, Huixin Liu, Mingxiang Wen, Pengfei Jiao, and Zhiyong Feng. Block modeling-guided graph convolutional neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(4):4022–4029, Jun. 2022.

[10] Dongkwan Kim and Alice Oh. How to find your friendly neighborhood: Graph attention design with self-supervision, 2022.

[11] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[12] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.

[13] Xiaojun Ma, Qin Chen, Yuanyi Ren, Guojie Song, and Liang Wang. Meta-weight graph neural network: Push the limits beyond global homophily. In *Proceedings of the ACM Web Conference 2022*, WWW '22. ACM, April 2022.

[14] Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. Is homophily a necessity for graph neural networks?, 2023.

[15] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks, 2020.

[16] Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding, 2021.

[17] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93, Sep. 2008.

[18] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.

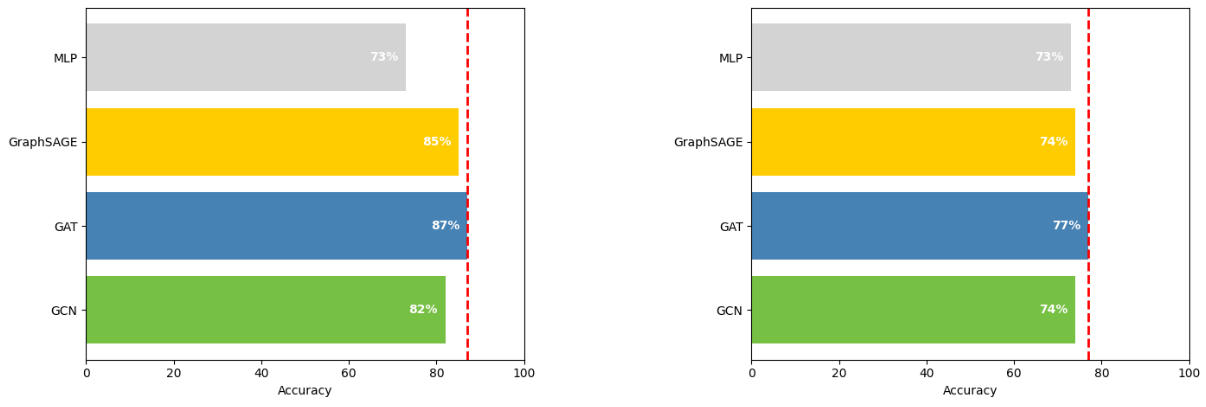
[19] Junfu Wang, Yuanfang Guo, Liang Yang, and Yunhong Wang. Heterophily-aware graph attention network, 2023.

[20] Liang Yang, Mengzhe Li, Liyang Liu, bingxin niu, Chuan Wang, Xiaochun Cao, and Yuanfang Guo. Diverse message passing for attribute with heterophily. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 4751–4763. Curran Associates, Inc., 2021.

[21] Jiong Zhu, Ryan A. Rossi, Anup Rao, Tung Mai, Nedim Lipka, Nesreen K. Ahmed, and Danai Koutra. Graph neural networks with heterophily. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):11168–11176, May 2021.

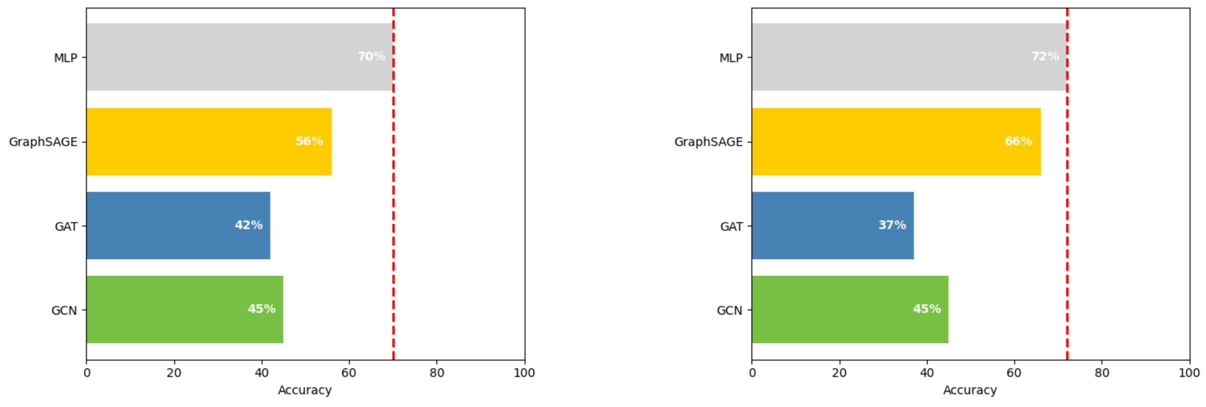
[22] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs, 2020.

9 Appendix



(a) (b)

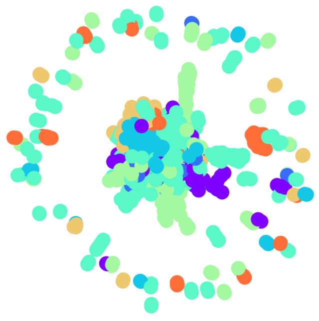
Figure 9.1: Figure (a) shows the accuracy comparison between MLP, GraphSAGE, GAT and GCN on 'Cora' dataset and Figure (b) shows the accuracy comparison between MLP, GraphSAGE, GAT and GCN on 'Citeseer' dataset.



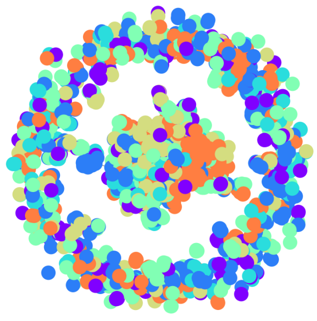
(a) (b)

Figure 9.2: Figure (a) shows the accuracy comparison between MLP, GraphSAGE, GAT and GCN on 'Cornell' dataset and Figure (b) shows the accuracy comparison between MLP, GraphSAGE, GAT and GCN on 'Wisconsin' dataset.

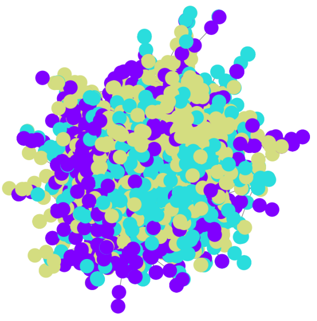
From these figures, it can be observed that MLP outperforms traditional state-of-the-art graph neural network models, especially in scenarios where graphs with low homophily ratios (h_G) are provided as input.



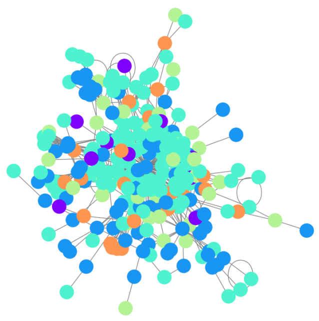
(a) Cora



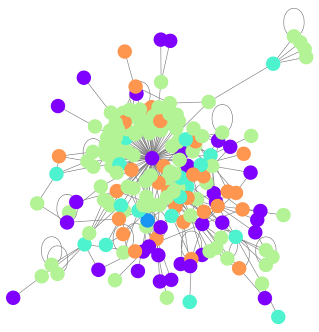
(b) Citeseer



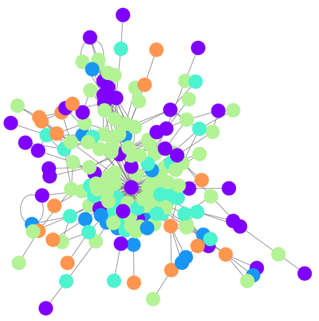
(c) PubMed



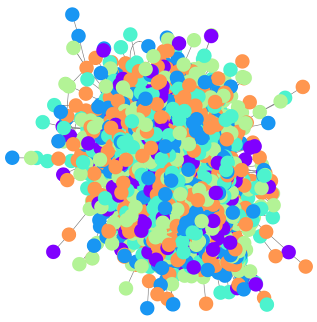
(d) Wisconsin



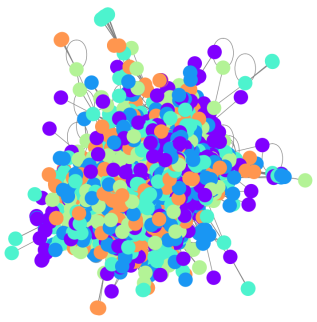
(e) Texas



(f) Cornell



(g) Actor



(h) Squirrel

Figure 9.3: The benchmark datasets that have been used in this dissertation are depicted in these figures. While figures (d), (e), (f), (g), and (h) are of low homophilic type, figures (a), (b), and (c) are of high homophilic type.

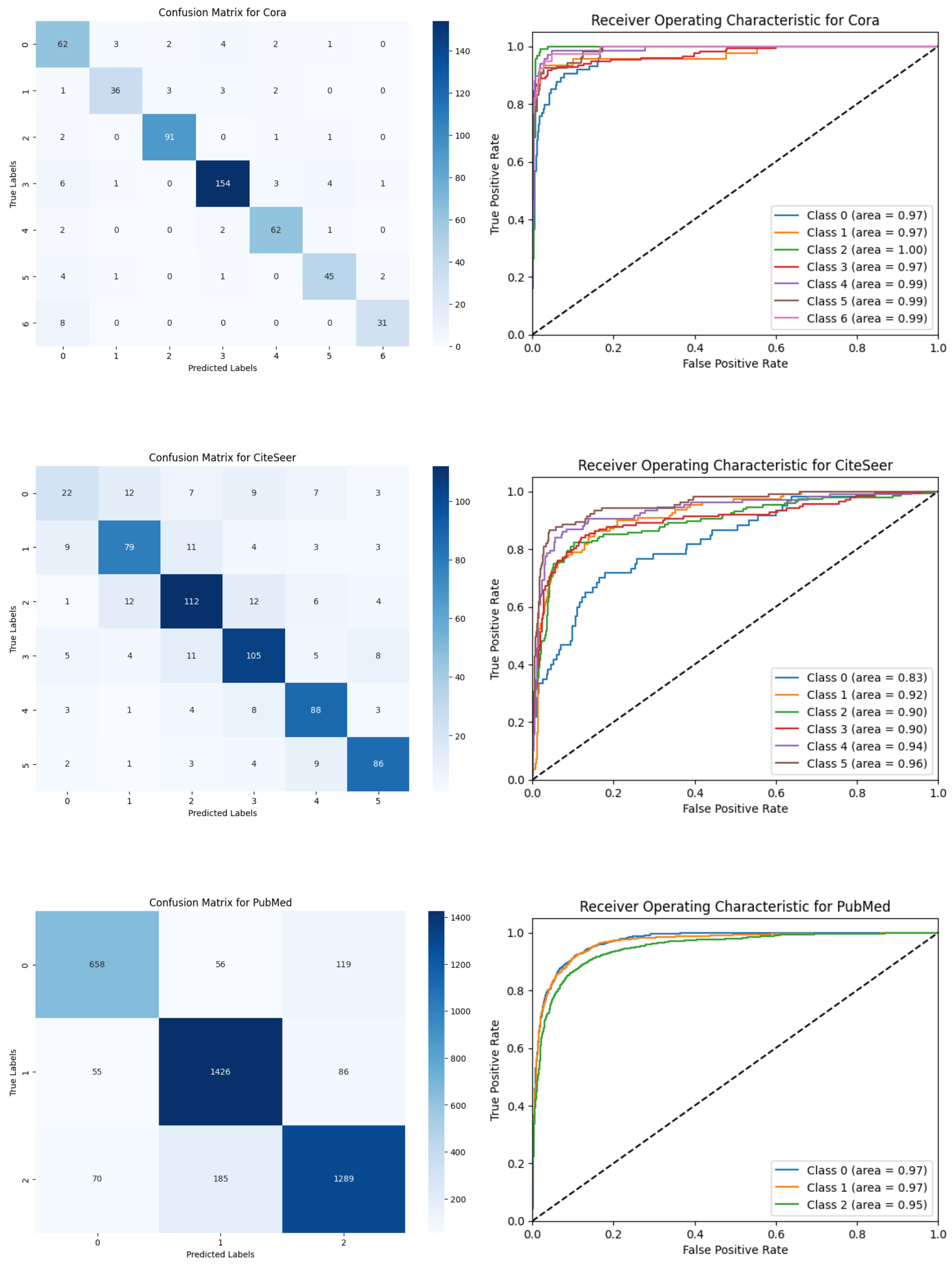


Figure 9.4: The figures present evaluations on the datasets (Cora, CiteSeer, PubMed): the confusion matrix is displayed on the left side, while the AUC-ROC curve is shown on the right side.

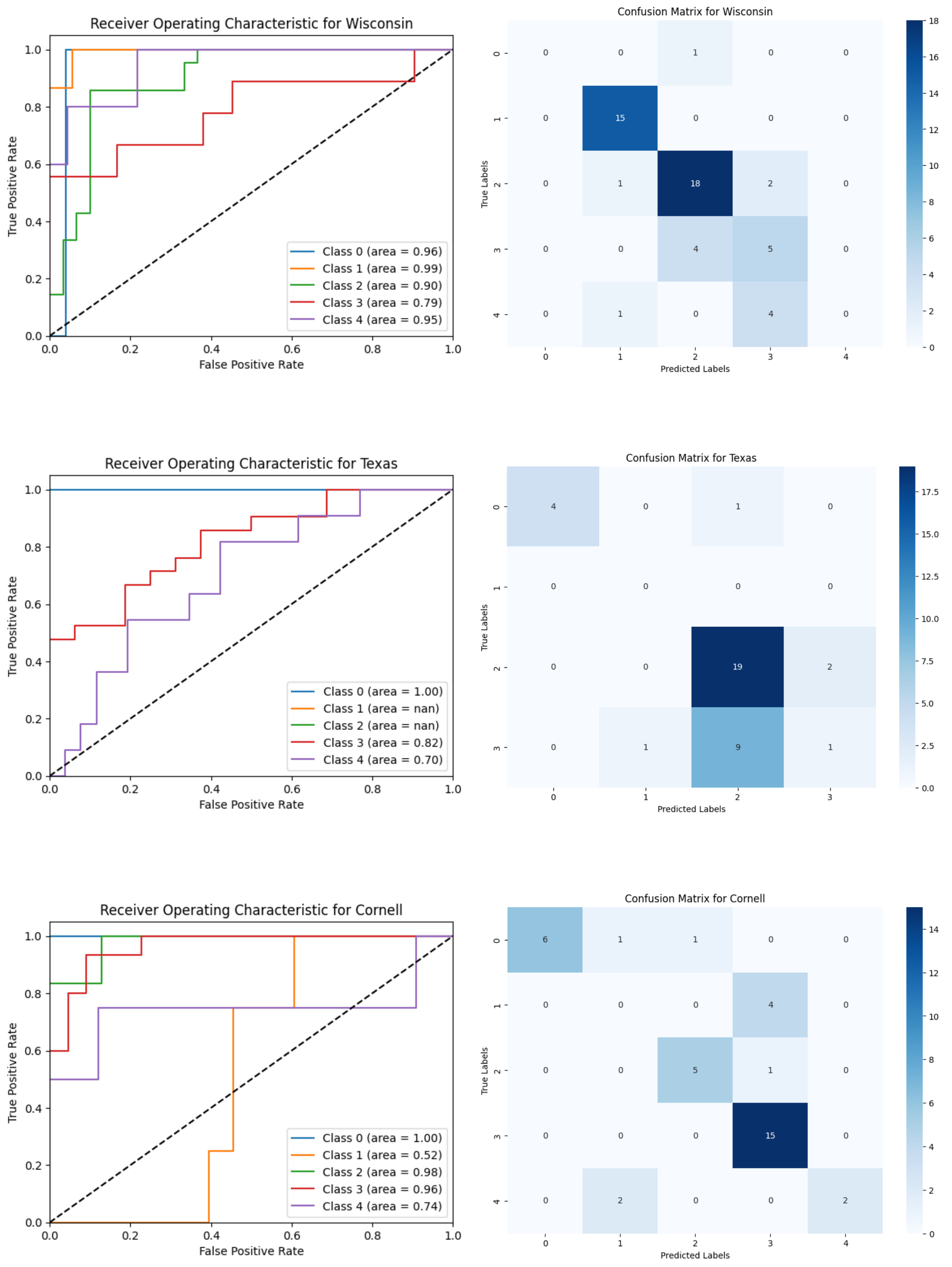


Figure 9.5: The figures present evaluations on the datasets (Wisconsin, Texas and Cornell): the confusion matrix is displayed on the right side, while the AUC-ROC curve is shown on the left side. (Contd.)

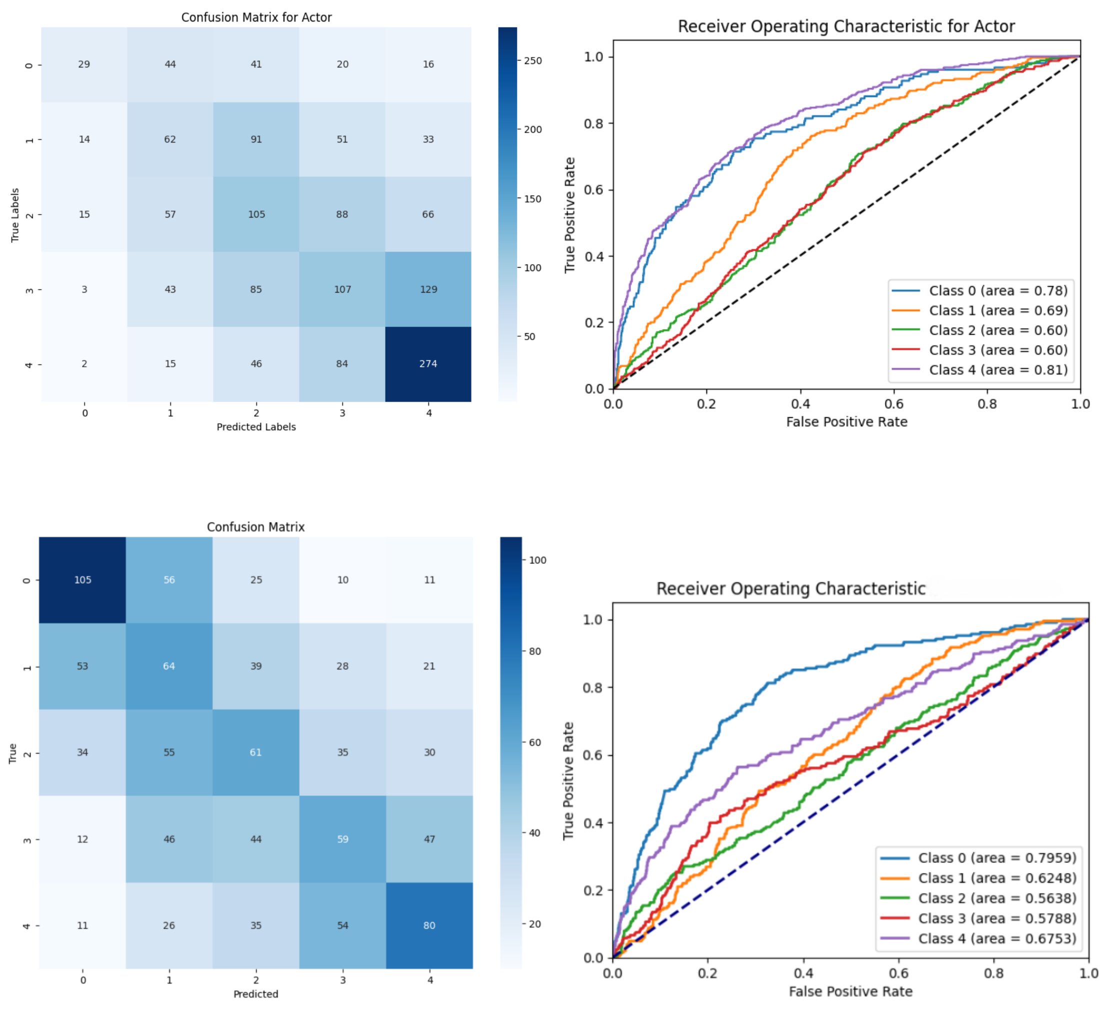


Figure 9.6: The figures present evaluations on the datasets (Actor and Squirrel): the confusion matrix is displayed on the left side, while the AUC-ROC curve is shown on the right side. (Contd.)