Exploring the Potential of Large Language Models for Heterophilic Graphs

Yuxia Wu[1]*, Shujie Li[2]*, Yuan Fang[1], Chuan Shi[2]
[1]Singapore Management University, [2]Beijing University of Post and Telecommunication
yieshah2017@gmail.com, shujieli@bupt.edu.cn, yfang@smu.edu.sg, shichuan@bupt.edu.cn

Abstract

Large language models (LLMs) have presented significant opportunities to enhance various machine learning applications, including graph neural networks (GNNs). By leveraging the vast open-world knowledge within LLMs, we can more effectively interpret and utilize textual data to better characterize heterophilic graphs, where neighboring nodes often have different labels. However, existing approaches for heterophilic graphs overlook the rich textual data associated with nodes, which could unlock deeper insights into their heterophilic contexts. In this work, we explore the potential of LLMs for modeling heterophilic graphs and propose a novel two-stage framework: LLM-enhanced edge discriminator and LLM-guided edge reweighting. In the first stage, we fine-tune the LLM to better identify homophilic and heterophilic edges based on the textual content of their nodes. In the second stage, we adaptively manage message propagation in GNNs for different edge types based on node features, structures, and heterophilic or homophilic characteristics. To cope with the computational demands when deploying LLMs in practical scenarios, we further explore model distillation techniques to fine-tune smaller, more efficient models that maintain competitive performance. Extensive experiments validate the effectiveness of our framework, demonstrating the feasibility of using LLMs to enhance node classification on heterophilic graphs.

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities across a wide range of applications, from natural language processing (Brown et al., 2020) to computer vision (Wang et al., 2024b), leveraging the extensive open-world knowledge LLMs encode. Inspired by these successes, recent efforts have extended the application of LLMs to the graph domain, particularly in text-attributed graphs where node attributes are composed of textual information (Li et al., 2023b; Zhang et al., 2024). In the context of heterophilic graphs, where connected nodes often exhibit contrasting features or class labels (Bo et al., 2021; Sun et al., 2022; Liang et al., 2024), LLMs offer a unique opportunity to enhance the understanding of complex semantic relationships between these connected nodes, which remains largely unexplored.

Existing approaches to addressing heterophily in GNNs typically involve extracting shallow embeddings from textual information, using them as initial node features without fully exploiting their rich semantic content. They can be broadly categorized into two main strategies: non-local neighbor extension and architectural refinement (Zheng et al., 2022; Gong et al., 2024). The former extends the node's receptive field to include distant, high-order neighbors (Abu-El-Haija et al., 2019; Song et al., 2023) or potential connections (Jin et al., 2021; Wang and Zhang, 2022; Zou et al., 2023), thereby enhancing node representations through a broader scope of information integration. The latter modifies the core functions of GNNs, such as the message aggregation and updating functions, to better suit heterophilic contexts (Du et al., 2022).

In summary, current methods for heterophilic graphs largely overlook the rich textual content associated with the nodes in real-world graphs, which can provide deeper insights into heterophilic contexts. For instance, textual content on hyperlinked webpages can enrich the understanding and prediction of heterophilic links. Traditionally, GNNs employ bag-of-words or shallow embeddings to incorporate textual attributes, which are inadequate for capturing complex semantics. While LLMs (Zhao et al., 2023b) have been used to empower GNNs for text-attributed graphs (Liu et al., 2023; Li et al., 2023b; Yu et al., 2024a; Mao et al., 2024), existing efforts focus on homophilic graphs, leaving heterophilic graphs largely unexplored.

* Co-first authors with equal contribution.

In this work, we delve into the potential of LLMs for heterophilic graphs. To the best of our knowledge, this is the first investigation into exploiting LLMs for heterophilic graphs. We aim to bridge the gap between the general capabilities of LLMs and the unique characteristics of heterophilic graphs. Specifically, we aim to address the following research questions.

First, *can LLMs be effectively adapted to characterize and identify heterophilic contexts?* As LLMs encompass general open-world knowledge, they can be utilized for the semantic understanding of nodes' textual content. However, it is not sufficient to merely extract features from the textual content. Unlike homophilic graphs, a key distinction of heterophilic graphs is that edges frequently form between dissimilar nodes. Therefore, distinguishing heterophilic edges from homophilic ones is crucial for subsequent aggregation on graphs. To address this, we leverage LLMs (or any pre-trained language model in general) to identify heterophilic edges. Specifically, we propose **LLM-enhanced edge discrimination**, where we fine-tune an LLM using Low-Rank Adaptation (LoRA) (Hu et al., 2022) to discriminate heterophilic and homophilic edges based on a limited amount of ground truth labels. This module focuses on adapting the general semantic capabilities of LLMs to the specific task of predicting heterophily between nodes. The fine-tuned LLM is subsequently used to infer heterophilic edges on the graph to facilitate the integration of heterophilic contexts in the next stage.

Second, *can LLMs effectively guide the fine-grained integration of heterophilic contexts into graph models?* With respect to a target node, nodes with a potential heterophilic (or homophilic) edge provide valuable heterophilic (or homophilic) contexts for the target node. Given the diverging characteristics of homophilic and heterophilic contexts, it is important to differentiate them when integrating into a GNN. Specifically, heterophilic contexts of a target node can be identified by our LLM-enhanced edge discrimination. Building on this, we further propose **LLM-guided edge reweighting** to further aggregate heterophilic and homophilic contexts through GNNs. In this module, we aim to learn adaptive weights for both heterophilic and homophilic edges. These weights are adapted to each edge based on its features, structure, and heterophilic or homophilic characteristics, thereby guiding the fine-grained, edge-sensitive aggregation in GNNs.

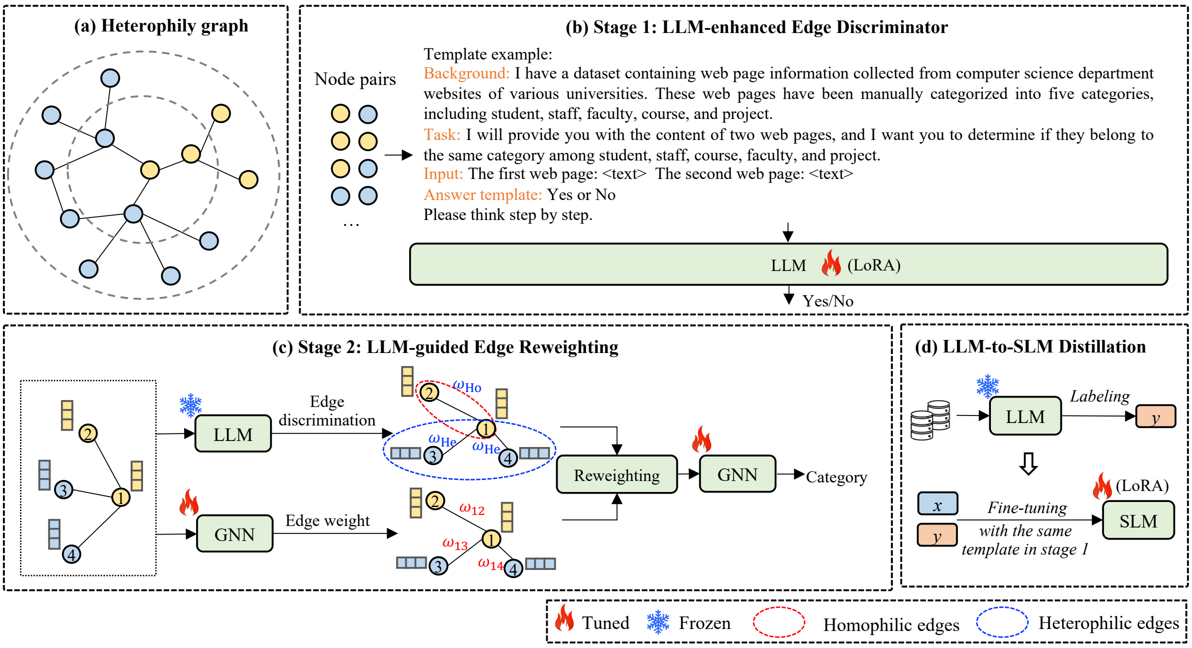
Additionally, LLMs often incur high computational costs even for inference, limiting their practical deployment for real-world applications. To this end, we further explore model distillation techniques (Xu et al., 2024) to condense the knowledge from fine-tuned LLMs into small language models (SLMs) (Schick and Schütze, 2021; Li et al., 2023a; Gu et al., 2024; Pan et al., 2024) to speed up the inference stage required for edge discrimination and reweighting. Specifically, we utilize the LLM well-tuned for edge discrimination to generate high-quality pseudo-labels for heterophilic and homophilic edges. These pseudo-labels supplement the limited ground truth labels, forming an expanded label set that further enables the fine-tuning of SLMs. The fine-tuned SLM then replaces the LLM for conducting inference for the edge discrimination and reweighting, maintaining effectiveness while significantly reducing inference time.

To summarize, we propose a two-stage framework that leverages LLMs for Heterophilic Graph modeling (LLM4HeG). Our main contributions are as follows. (1) To the best of our knowledge, this is the first study to explore LLMs specifically for modeling heterophilic graphs. This exploration not only opens new research avenues but also provides valuable insights into the capabilities of LLMs in addressing unique graph characteristics such as heterophily. (2) We introduce LLM4HeG, a novel two-stage framework that fine-tunes LLMs to enhance GNNs for heterophilic graphs. The two stages, LLM-enhanced edge discrimination and LLM-guided edge reweighting, accurately identify heterophilic edges and adaptively integrate heterophilic contexts into GNNs, respectively. (3) We further investigate the distillation of LLMs fine-tuned for heterophilic edge discrimination into SLMs, achieving faster inference time with minimal performance degradation. (4) Finally, we conduct extensive experiments on five real-world datasets and demonstrate the effectiveness and efficiency of our work.

2 Related Work

We review the literature on LLM-based and heterophilic graph learning and highlight the key distinctions of our work from existing studies.

Existing research on LLMs for graph learning includes LLM-based methods that adopt LLM as the backbone and GNN+LLM-based methods that integrate the advantages of both GNNs and LLMs (Liu



Template example:
Background: I have a dataset containing web page information collected from computer science department websites of various universities. These web pages have been manually categorized into five categories, including student, staff, faculty, course, and project.
Task: I will provide you with the content of two web pages, and I want you to determine if they belong to the same category among student, staff, course, faculty, and project.
Input: The first web page: <text> The second web page: <text>
Answer template: Yes or No
Please think step by step.

Figure 1: Overall framework of the proposed method LLM4HeG.

et al., 2023; Chen et al., 2024). The former works focus on aligning graph data with natural language via graph-to-token and graph-to-text approaches (Liu et al., 2023). The graph-to-token approach involves tokenizing graph data to align it with natural language, enabling joint understanding with data from other modalities (Zhao et al., 2023a; Ye et al., 2024). Graph-to-text focuses on describing graph information using natural language (Liu and Wu, 2023; Wang et al., 2024a; Guo et al., 2023). The latter harnesses the strengths of both language understanding from LLMs and structural analysis from GNNs by using GNN-centric methods utilizing LLMs to extract node features from raw data and make predictions using GNNs (He et al., 2024; Xie et al., 2023) or LLM-centric methods utilizing GNNs to enhance the performance of LLM (Tang et al., 2024; Zhang et al., 2024).

Existing heterophilic graph learning approaches generally fall into two main strategies: non-local neighbor extension and architectural refinement (Zheng et al., 2022; Gong et al., 2024). Non-local neighbor extension approaches aim to extend the neighbors to include non-local nodes in the graph that may share similar labels or features. These methods often involve high-order neighbor mixing (Abu-El-Haija et al., 2019; Song et al., 2023; Yu et al., 2024b) or discovering potential neighbors based on various distance measurements, such as feature-based distance (Jin et al., 2021; Bodnar et al., 2022), structure-based distance (Pei et al., 2020), or hybrid approaches (Wang and Zhang, 2022; Wang et al., 2022; Li et al., 2022; Zou et al., 2023; Bi et al., 2024). Architectural refinement approaches enhance the GNN architecture by employing identifiable message aggregation to discriminate and amplify messages from similar neighbors while minimizing the influence of dissimilar ones (Bo et al., 2021; Zhu et al., 2021; Du et al., 2022; Liang et al., 2024), or by leveraging inter-layer combinations to capture information from different neighbor ranges (Xu et al., 2018; Chien et al., 2021; Zhu et al., 2020), thereby improving the model's representation power in heterophilic graphs.

The key distinctions of our method lie in two aspects. First, we explore LLMs to enhance text-attributed heterophilic graphs' modeling specifically. While prior works leverage LLMs for text-attributed graphs, including edge reweighting (Sun et al., 2023; Ling et al., 2024), they do not explicitly target heterophilic graphs. Additionally, existing heterophilic graph methods often overlook rich textual node attributes, which provide essential semantic contexts. Second, our two-stage framework employs an LLM-enhanced edge discriminator to predict edge types, followed by adaptive message propagation in GNNs using a comprehensive suite of information, including node semantics, structural contexts, and LLM-inferred edge characteristics. While GBK-GNN (Du et al., 2022) follows a similar two-stage approach, it does not leverage the power of LLMs.

3 Proposed Model: LLM4HeG

In this section, we first introduce some preliminaries on the problem formulation and classic GNNs. Then we introduce the overview of the proposed model followed by details of different components.

3.1 Preliminaries

Problem formulation. Let $G = (V, E, X, C)$ denotes a text-attributed graph with a set of nodes V and a set of edges E, where each node $v \in V$ is associated with a text document $x_v \in X$. C is the set of node classes. In this paper, we address the task of semi-supervised transductive node classification for heterophilic graphs. Specifically, a subset of the nodes is designated as the training/validation nodes with known class labels, while the goal is to predict the unknown labels of the remaining nodes in the graph.

Classic GNNs. GNNs typically employ a multi-layer approach to neighborhood aggregation, wherein each node incrementally gathers and aggregates contexts from its neighboring nodes. At the l^{th} layer, the representation $\mathbf{h}_v^l \in \mathbb{R}^{d_l}$ of a node v is derived as follows:

$$\mathbf{h}_v^{(l)} = \text{AGGR}(\mathbf{h}_v^{(l-1)}, \{\mathbf{h}_u^{(l-1)} : u \in \mathcal{N}(v)\}), \quad (1)$$

where d_l is the dimension of the node representations at the l^{th} layer. The function $\text{AGGR}(\cdot)$ denotes an aggregation mechanism combining the feature vectors of the neighboring nodes, where $\mathcal{N}(v)$ denotes the set of neighboring nodes of v.

3.2 Overall Framework

Fig. 1 illustrates the overall framework of our approach LLM4HeG, with a two-stage framework leveraging LLMs for heterophilic graph modeling. As shown in Fig. 1(b), Stage 1 involves our LLM-enhanced edge discriminator, where we fine-tune an LLM to distinguish between homophilic and heterophilic edges, utilizing the rich textual data associated with nodes and a limited amount of ground truth label. Following this, in Fig. 1(c), Stage 2 involves LLM-guided edge reweighting to learn adaptive weights for both homophilic and heterophilic edges. These weights are adapted to individual edges by integrating node features, graph structures, and edge types, enabling fine-grained aggregation within GNNs.

Additionally, to cope with the computational demands of deploying LLMs, we explore a distillation method that condenses the heterophily-specific knowledge of a fine-tuned LLM into a more compact SLM. As shown in Fig. 1(d), we leverage the LLM as a teacher model to generate pseudo labels for additional examples, which can be used to fine-tune an SLM that can perform inference more efficiently without compromising performance.

3.3 LLM-enhanced Edge Discriminator

In heterophilic graphs, accurately discriminating heterophilic edges from homophilic ones is pivotal for effectively tailoring context aggregation strategies across neighboring nodes. We propose an LLM-enhanced edge discriminator, tapping on the semantic capabilities and open-world knowledge of LLMs beyond conventional shallow feature-based approaches. We first construct the ground truth labels from the training set and then prepare a language template that describes the edge discrimination task for fine-tuning a given LLM. We elaborate on these steps below.

First, to adapt the LLM into an edge discriminator model for heterophilic graphs, we construct ground truth labels to indicate whether a potential edge between two nodes is homophilic or heterophilic. Specifically, if the two nodes have different attributes or categories, their potential relationship is considered heterophilic (Yan et al., 2022). Hence, we select node pairs from the training set and label them as homophilic or heterophilic by comparing their known class labels. The selection of node pairs depends on the size of the graph. For small graphs, we choose all node pairs; for larger graphs, we choose node pairs within one or two-hop neighborhoods of each other. We will elaborate on the details of node pair selection in Appendix B. Note that we use "heterophilic/homophilic edge" to describe a potential relationship between two nodes, even if no explicit edge exists between them.

Next, given such a node pair with a ground-truth label on their homophilic or heterophilic nature, we design a language template to describe the task of heterophilic edge discrimination. We utilize textual information of the node pairs to construct the template as the input text to the LLM, including the *background*, *task*, *input* and *answer template* (Fig. 1(b)). Notably, the *background* also includes a set of node category names specific to the heterophilic graph, which can be regarded as semantic anchors to enhance the LLM's ability to understand the contexts of the given heterophilic graph.

Fine-tuning. The template, together with the ground-truth labels, enables the fine-tuning of

LLMs. For efficiency, we adopt a parameter-efficient fine-tuning technique called LoRA, which strategically updates only a small fraction of the LLM's parameters (Hu et al., 2022). For fine-tuning, we use a typical cross-entropy loss to align the model output with the ground-truth responses:

$$\mathcal{L}_{\text{fine-tune}} = -\tfrac{1}{N}\textstyle\sum_{i=1}^{N} \log P_\theta(x_i|x_{<i}), \quad (2)$$

where N is the number of tokens in the sentences, x_i is the i-th token to be predicted, and $x_{<i}$ indicates previously generated tokens. $P_\theta(x_i|x_{<i})$ represents the probability of the token x_i given the previous tokens generated by the model.

Inference. After fine-tuning the LLM for edge discrimination, we can employ it to infer the relationship between any two nodes on the graph, determining whether they have a potentially heterophilic or homophilic edge. During inference, we use the same template to generate input text for each node pair, which is then fed into the fine-tuned LLM to produce a "*Yes*" or "*No*" answer regarding the edge type.

3.4 LLM-guided Edge Reweighting

Building on the edge types identified by Stage 1, we proceed to LLM-guided edge reweighting to integrate heterophilic contexts into GNNs. This process leverages the semantic insights acquired from the LLM-enhanced edge discriminator, allowing us to adjust the weight of every individual edge, taking into account various edge-specific factors, including node features and structures, as well as its homophilic or heterophilic nature.

Specifically, for a node $v \in V$, we first extract the representation from the LLM based on the associated textual information:

$$\mathbf{h}_v^{(0)} = \sigma(\texttt{LLM}(x_v)\mathbf{W}_e), \quad (3)$$

where x_v denotes the raw text of node v, `LLM` is an LLM encoder, $\mathbf{W}_e$ is a learnable weight matrix and σ is the activation function.

For a node $u \in \mathcal{N}_i(v)$, the i-hop neighborhood of v (Zhu et al., 2020), we infer the edge type of (u, v) using the predictions from Stage 1. Based on the LLM prediction, we formulate an initial weight for (u, v), denoted by w_{uv}, as follows.

$$w_{uv}^{\text{LLM}} = \begin{cases} \tanh(w_{\text{Ho}}) & \text{if } O_{\text{LLM}}(u,v) = \textit{Yes}, \\ \tanh(w_{\text{He}}) & \text{if } O_{\text{LLM}}(u,v) = \textit{No}, \end{cases} \quad (4)$$

where $O_{\text{LLM}}(\cdot)$ represents the output from the LLM-enhanced edge discriminator. The output "*Yes*" implies a homophilic edge, whereas "*No*" indicates a heterophilic edge. This initial LLM-based weight w_{uv}^{LLM} is defined based on two learnable parameters: w_{Ho} for homophilic edges and w_{He} for heterophilic edges. The two parameters are crucial for modulating the strength and influence of each edge type within the graph. For homophilic edges, where stronger connectivity is often beneficial, the learned parameter may increase the weight, thus amplifying the coherence and communication within similar node clusters. Conversely, for heterophilic edges, which often bridge diverse node groups, the parameter might be adjusted to achieve a balance between reducing noises and maintaining critical cross-group information.

On the other hand, graph-based information, including node features and structures, also provides crucial insights on determining the edge weight. While our framework LLM4HeG is designed to be flexible, allowing for the adoption of various GNN backbones, we showcase FAGCN (Bo et al., 2021) as an example of how our framework can be effectively applied. This method learns the edge-specific aggregation weight via a self-gating mechanism:

$$w_{uv}^{\text{G}} = \tanh\left(\mathbf{g}^\top [\mathbf{h}_u \parallel \mathbf{h}_v]\right), \quad (5)$$

where $\parallel$ denotes the concatenation operation, and $\mathbf{g}$ is a linear layer to map the concatenated feature into a scalar value, which can be seen as a shared convolutional kernel (Veličković et al., 2018).

Finally, we integrate the LLM-based weight w_{uv}^{LLM} with the graph-based weight w_{uv}^{G}. While there are many ways to achieve this, we use a simple yet effective method that takes the average of the two weights as the final weight, denoted by w_{uv}, as follows.

$$w_{uv} = \tfrac{1}{2}\left(w_{uv}^{\text{LLM}} + w_{uv}^{\text{G}}\right). \quad (6)$$

The edge-specific weights further enable fine-grained context aggregation, as detailed below.

$$\mathbf{h}_v^{(l)} = \epsilon \mathbf{h}_v^{(0)} + \textstyle\sum_{u \in \mathcal{N}_i(v)} \frac{w_{uv}}{\sqrt{d_u d_v}} \mathbf{h}_u^{(l-1)}, \quad (7)$$

$$\mathbf{h}_{\text{out}} = \mathbf{W}_o \mathbf{h}_v^{(L)}, \quad (8)$$

where ϵ is a scaling hyper-parameter, $d_v = |N_i(v)|$, $\mathbf{W}_o$ is a weight matrix, and L is the total number of GNN layers.

Training. We adopt a typical cross-entropy (CE) loss to train node classification. Additionally, to ensure the learned weights w_{Ho} and w_{He} show sufficient separation, we introduce a margin-based regularization term into the CE loss.

$$\mathcal{L} = \texttt{CE}(\hat{y}, y) + \lambda \max(0, w_{\text{He}} - w_{\text{Ho}} + \alpha), \quad (9)$$

where $\texttt{CE}(\hat{y}, y)$ represents the cross-entropy loss between the predicted node label $\hat{y}$ and the ground-truth label y, λ controls the influence of the regularization term and α represents the margin that enforces a minimum difference between the weights w_{Ho} and w_{He}. During training, we freeze the parameters of the LLM and only update the weights of the GNN model.

Inference. During the inference phase of Stage 2, we apply the LLM-enhanced edge discriminator fine-tuned in Stage 1 to generate the edge type for each node pair involved in the test set. Subsequently, we can calculate the output representations for the test nodes based on the reweighted edges for node classification.

3.5 LLMs-to-SLMs Distillation

In real-world applications, deploying LLMs as edge discriminators introduces substantial computational challenges, even when only the inference phase is required for predicting the edge types on test graphs. To mitigate the computational burden of the inference phase, we explore knowledge distillation techniques to transfer the heterophily-specific capabilities of the fine-tuned LLM into more lightweight SLMs (Xu et al., 2024).

As shown in Fig. 1(d), after fine-tuning the LLM in Stage 1, we use it as a teacher model to generate the pseudo-labels for additional node pairs sampled from the entire graph. The labeling process follows the inference phase introduced in Stage 1, asking the LLM whether a given node pair is homophilic or heterophilic based on the input template. These pseudo-labels are combined with the ground-truth labels, forming an expanded label set which is subsequently used to fine-tune the SLMs. The fine-tuning of the SLMs follows the same approach as fine-tuning the LLM in Stage 1, using the same template and the LoRA technique. Finally, we replace the fine-tuned LLM with the fine-tuned SLM during inference, which predicts the homophilic or heterophilic relationship between any two given nodes to guide edge reweighting in Stage 2.

Learning objectives. We outline the objectives of knowledge distillation in the two stages. In Stage 1, the fine-tuning of SLMs follows the same approach as fine-tuning the LLM shown in Eq. (2). In Stage 2, we train the edge reweighting using the same cross-entropy objective shown in Eq. (9).

Dataset	Classes	Nodes	Edges	$\mathcal{H}(G)$
Cornell	5	195	304	0.13
Texas	5	187	328	0.12
Wisconsin	5	265	530	0.20
Actor	5	4,416	12,172	0.56
Amazon	5	24,492	93,050	0.38

Table 1: Dataset statistics.

4 Experiment

In this section, we present an empirical study to demonstrate the feasibility of leveraging LLMs for node classification on heterophilic graphs.

4.1 Experimental Setup

Datasets. Given that the datasets commonly employed in heterophily graph tasks lack original textual information, we collect publicly available raw text directly from the original data providers and preprocess these datasets. Consequently, our experiments only include datasets that contain raw text. Specifically, we have prepared five datasets: Cornell, Texas, and Wisconsin (Pei et al., 2020), Actor (Tang et al., 2009) and Amazon (Platonov et al., 2023). The Cornell, Texas, and Wisconsin datasets (Pei et al., 2020) are derived from computer science department websites where webpages serve as nodes and hyperlinks as edges. The Actor dataset (Tang et al., 2009) is an actor co-occurrence network with nodes representing actors and edges denoting their collaborations. The Amazon dataset (Platonov et al., 2023) is a product co-purchasing network, where nodes represent products and edges link products frequently bought together. The statistical details of the datasets are presented in Table 1 with more details provided in Appendix A. Additionally, we include the edge homophily score $\mathcal{H}(G)$ (Yan et al., 2022) for each dataset, which quantifies the level of homophily (1 means perfectly homophily while 0 stands for total heterophily).

Baselines. We compare our method against a set of baseline models that fall into two main categories: classic GNN models and heterophily-specific models. The classic GNN models include GCN (Kipf and Welling, 2016), GraphSAGE (Hamilton et al., 2017) and GAT (Veličković et al., 2018). The heterophily-specific models include H2GCN (Zhu et al., 2020), FAGCN (Bo et al., 2021), JacobiConv (Wang and Zhang, 2022), GBK-GNN (Du et al.,

Methods	Cornell	Texas	Wisconsin	Actor	Amazon
Classic GNNs					
GCN	52.86±1.8	43.64±3.3	41.40±1.8	66.70±1.3	39.33±1.0
GraphSAGE	75.71±1.8	81.82±2.5	80.35±1.3	70.37±0.1	46.63±0.1
GAT	54.28±5.1	51.36±2.3	50.53±1.7	63.74±6.7	35.12±6.4
Heterophily-specific GNNs					
H2GCN	69.76±3.0	79.09±3.5	80.18±1.9	70.73±0.9	47.09±0.3
FAGCN	76.43±3.1	84.55±4.8	83.16±1.4	75.58±0.5	49.83±0.6
JacobiConv	73.57±4.3	81.80±4.1	76.31±11.3	73.81±0.3	49.43±0.5
GBK-GNN	66.19±2.8	80.00±3.0	72.98±3.3	72.49±1.0	44.90±0.3
OGNN	71.91±1.8	85.00±2.3	79.30±2.1	72.08±2.4	47.79±1.6
SEGSL	66.67±4.1	85.00±2.0	79.30±1.8	72.73±0.8	47.38±0.2
DisamGCL	50.48±2.0	65.00±1.2	57.89±0.0	67.78±0.3	43.90±0.4
LLM4HeG (fine-tuned LLM/SLMs and distilled SLMs)					
Vicuna 7B	**77.62**±2.9	**89.09**±3.3	86.14±2.1	**76.82**±0.5	51.53±0.4
Bloom 560M	75.48±2.1	80.00±4.0	86.49±1.9	76.16±0.6	51.52±0.5
Bloom 1B	75.71±1.4	83.86±2.8	83.86±1.7	74.99±0.5	**52.33**±0.6
7B-to-560M	75.00±4.0	88.18±2.2	**87.19**±2.5	75.78±0.2	51.51±0.4
7B-to-1B	77.38±2.7	88.18±4.0	86.14±1.5	75.37±0.9	51.58±0.4

Table 2: Accuracy for node classification of different methods. (Best results bolded; runners-up underlined.)

2022), OGNN (Song et al., 2023), SEGSL (Zou et al., 2023), DisamGCL (Zhao et al., 2024) with more details provided in Appendix D.

Implementation Details. We adopt Vicuna-v1.5-7B (Zheng et al., 2024) as the LLM model and the Bloom model (Le Scao et al., 2023) with 560M and 1B parameters as SLMs. Consistent with most existing works (Zhu et al., 2020), we randomly split the nodes into train, validation and test sets with a proportion of 48%/32%/20% for Cornell, Texas, Wisconsin and Actor datasets. We set the proportion of Amazon dataset as 50%/25%/25% following Platonov et al. (2023). In our main experiments, we use FAGCN (Bo et al., 2021) as the GNN backbone, adhering to the parameter settings outlined in FAGCN. We only fine-tune the other hyperparameters based on the performance observed on the validation sets. All experiments are repeated 10 times, and we report the averaged results with standard deviation. To ensure a fair comparison, we use the initial node features derived from the Vicuna 7B model for all methods. More implementation details and hyper-parameters are introduced in Appendix B and C.

4.2 Performance Comparison

Table 2 shows the average accuracy and the standard deviation of the baselines and LLM4HeG with fine-tuned LLM/SLMs and distilled SLMs. For LLM4HeG, we choose the Vicuna 7B as the LLM and the Bloom models with 560M and 1B as SLMs. We conduct experiments with various strategies for the LLM and SLMs to explore their performance. (1) “Vicuna 7B”: We fine-tune the LLM via LoRA technique for edge discrimination. (2) “Bloom 560M” and “Bloom 1B”: We directly fine-tune SLMs via LoRA in the same way as “Vicuna 7B”. (3) “7B-to-560M” and “7B-to-1B”: We distill the Vicuna 7B model to Bloom 560M and 1B models, respectively, as introduced in Sect. 3.5.

Among the baselines, heterophily-specific GNNs generally outperform classic GNNs, while our methods consistently achieve the best performance. These results indicate that LLM4HeG effectively captures the complex relationships among different nodes in heterophilic graphs.

For different strategies of LLM4HeG, directly fine-tuning the SLMs often leads to notable performance decline compared to fine-tuning the LLM. However, the distilled SLMs attain performance comparable to that of the fine-tuned LLM, demonstrating the effectiveness of the distillation process in retaining heterophily-specific knowledge. The only exception is the performance on the Amazon dataset, where the directly fine-tuned SLMs achieve similar performance as the LLM and the distilled ones. This may be due to the semi-structured patterns within the textual descriptions in the Amazon dataset such as product specifications, which are relatively easy to capture by both LLMs and SLMs. We also present experiments that directly utilize LLMs for node classification in Appendix E.

4.3 Model Analysis

Ablation Study. To evaluate the effectiveness of the learnable weight for adaptive message passing and deep node features from LLM, we conducted experiments using different variants of our model: (1) *w/o reweight*: This variant only uses the graph-based weight w_{uv}^{G} (2) *w/o learnable weights*: This variant uses fixed weights instead of learnable ones. Specifically, we manually set the weights for homophilic and heterophilic pairs to 1 and -1, respectively, as these values were found to perform well across most datasets.

As shown in Fig. 2, the performance drops after removing the reweighting mechanism. When reweighting is applied, the *w/o learnable weight* variant outperforms the backbone model and the learnable weight of LLM4HeG shows the best performance. These results indicate the significance of learnable adaptive reweighting guided by LLM for message aggregation in GNNs.

Analysis of edge discrimination by LLM/SLMs.

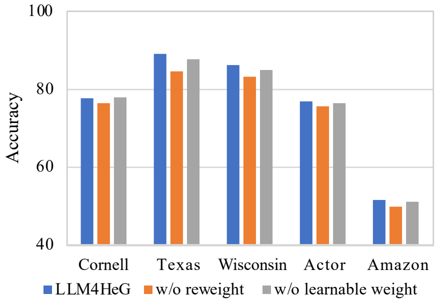


Figure 2: The effectiveness of learnable weight.

Model	Cornell	Texas	Wisconsin	Actor	Amazon	Average
Vicuna 7B	65.71	64.00	92.66	81.50	44.68	69.71
Bloom 560M	47.62	26.51	71.62	79.02	56.26	56.21
Bloom 1B	40.86	23.91	79.76	79.52	59.89	56.78
7B-to-560M	50.85	64.86	80.75	81.03	50.77	65.65
7B-to-1B	51.72	80.00	75.95	80.47	51.48	67.92

Table 3: F1 scores for edge discrimination of fine-tuned LLM/SLMs and distilled SLMs.

As the edge reweighting in Stage 2 depends on the effectiveness of edge discrimination in Stage 1, we further evaluate the edge classification performance (F1 score) of different models on the node pairs used for Stage 2. As shown in Table 3, compared with the fine-tuned LLM, directly fine-tuning the SLMs generally gives worse edge discrimination performance, which is reasonable considering the model capacity. However, the distilled SLM manages to maintain a comparable performance with only a marginal drop. This indicates the effectiveness of the distillation process, which allows the distilled SLM to retain heterophily-specific knowledge of the fine-tuned LLM. The results are consistent with the node classification performance reported in Table 2.

Performance on inductive setting. Our method is inherently adaptable to inductive node classification, where test nodes remain entirely unseen during training. We conduct additional experiments on inductive test sets, requiring the model to classify them based on their raw textual attributes and structural connections. As shown in Fig. 3, our approach consistently outperforms baseline methods in this setting. These results indicate that our method can effectively benefit from the semantic understanding of LLMs in both transductive and inductive settings.

Efficiency study. Fig. 4 illustrates the efficiency of our framework using an LLM or distilled SLMs in terms of training and inference times, measured in minutes. For the LLM approach, the training time denotes the fine-tuning time in Stage 1 on the selected node pairs (see Section 3.3). For the distilled SLMs, as discussed in Section 3.5, we employ the fine-tuned LLM to generate pseudo-labels for further fine-tuning the SLM. Thus, its training time includes fine-tuning the LLM, generating the pseudo-labels, and fine-tuning the SLM. Thus, the total training time for model distillation is slightly higher than the LLM. On the other hand, inference time includes predicting edge types using either the LLM or distilled SLM, as required by the edge reweighting module in Stage 2. It is worth noting that the inference times of SLMs are significantly lower than LLMs, especially for larger datasets such as Amazon. Hence, the distilled SLMs can be more easily deployed given their smaller size and faster inference time, while maintaining competitive performance as shown in Table 2.

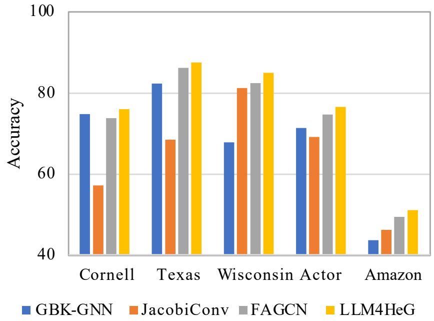


Figure 3: The accuracy of inductive node classification.

	Cornell	Texas	Wisconsin	Actor	Amazon
GCN	52.86±1.8	43.64±3.3	41.40±1.8	66.70±1.3	39.33±1.0
+LLM4HeG	66.19±1.0	68.18±2.0	76.84±2.6	71.68±1.0	40.98±0.7
GAT	54.28±5.1	51.36±2.3	50.53±1.7	63.74±6.7	35.12±6.4
+LLM4HeG	58.57±4.9	58.18±2.3	57.54±6.1	70.78±0.7	36.01±5.8
H2GCN	69.76±3.0	79.09±3.5	80.18±1.9	70.73±0.9	47.09±0.3
+LLM4HeG	76.43±3.6	84.77±1.0	86.49±1.1	74.51±0.6	52.14±0.4
FAGCN	76.43±3.1	84.55±4.8	83.16±1.4	75.58±0.5	49.83±0.6
+LLM4HeG	77.62±2.9	89.09±3.3	86.14±2.1	76.82±0.5	51.53±0.4
GBK-GNN	66.19±2.8	80.00±3.0	72.98±3.3	72.49±1.0	44.90±0.3
+LLM4HeG	68.57±2.6	81.82±2.0	76.14±1.4	73.39±0.6	48.25±0.3

Table 4: The accuracy for node classification of LLM4HeG with different backbones.

4.3.1 Plug-and-play with various backbones

Our method is designed to be highly flexible, allowing it to be integrated with various GNN backbones. To demonstrate the flexibility, we tested our approach with several well-known GNN architectures, including GCN, GAT, H2GCN, FAGCN and GBK-GNN. We provide the implementation details of the integration process in Appendix B. Table 4 indicates that our method consistently enhances the performance of these backbones. This capability highlights the versatility of our approach, making it valuable for improving multiple backbones without the need for significant modifications.

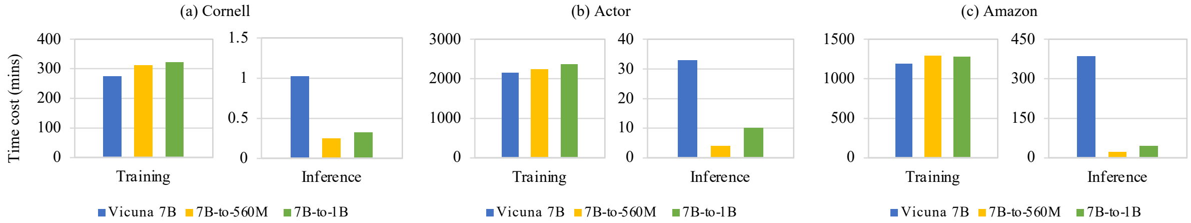


Figure 4: Analysis on the efficiency of the fine-tuned LLM and distilled SLMs.

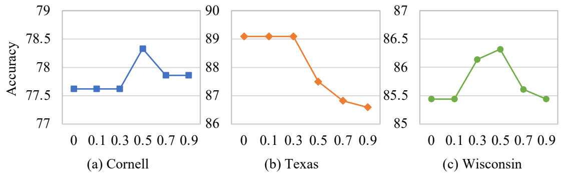


Figure 5: Effect of the edge weight margin α.

4.3.2 Hyper-parameter study.

Finally, we analyze the impact of a key hyper-parameter in our work, namely the weight margin α in Eq. (9), the training loss of Stage 2. We vary it over $\{0, 0.1, 0.3, 0.5, 0.7, 0.9\}$, and report the results for three datasets in Fig. 5, with the results for the remaining datasets in Appendix C. Generally speaking, when α is too low or too high, the performance tends to drop, suggesting that a balance is required to appropriately consider the difference between the learned weights. In particular, $[0.3, 0.5]$ appears to be a good range for α on these datasets.

We also conduct experiments on the influence of the initial values of w_{Ho} and w_{He} in Eq. (4). The results in Table 5 show that the classification performance is relatively stable across different initial values of w_{Ho} and w_{He}, with minor variations observed for specific datasets. This suggests that the model is robust to different initializations, and can generally find the optimal values for w_{Ho} and w_{He} despite different initializations.

w_{Ho}	w_{He}	Cornell	Texas	Wisconsin	Actor	Amazon
0.5	0.5	$78.57_{\pm 2.6}$	$89.55_{\pm 2.9}$	$85.44_{\pm 2.5}$	$76.83_{\pm 0.6}$	$51.47_{\pm 0.4}$
1	0	$77.86_{\pm 2.4}$	$89.55_{\pm 2.5}$	$85.44_{\pm 2.1}$	$77.00_{\pm 0.4}$	$51.42_{\pm 0.6}$
1.5	-0.5	$77.62_{\pm 2.9}$	$89.09_{\pm 3.3}$	$86.14_{\pm 2.1}$	$76.82_{\pm 0.5}$	$51.53_{\pm 0.4}$
2	-1	$78.57_{\pm 1.8}$	$87.73_{\pm 3.6}$	$85.44_{\pm 1.8}$	$76.88_{\pm 0.5}$	$51.40_{\pm 0.4}$
2.5	-1.5	$78.57_{\pm 1.5}$	$87.95_{\pm 2.3}$	$86.14_{\pm 1.8}$	$76.99_{\pm 0.5}$	$51.30_{\pm 0.3}$

Table 5: Effect of the initial weights of w_{Ho} and w_{He}.

5 Conclusion

In this study, we explored the potential of LLMs to enhance the performance of GNNs for node classification on heterophilic graphs. We introduced a novel two-stage framework LLM4HeG, integrating LLMs into the GNN learning process through an LLM-enhanced edge discriminator and an LLM-guided edge reweighting module. LLM4HeG allows more precise identification of heterophilic edges and finer-grained context aggregation, leveraging the rich semantics in nodes' textual data. Additionally, to address the computational challenges of deploying LLMs, we implemented model distillation techniques to create smaller models that achieve much faster inference while maintaining competitive performance. Our extensive experiments demonstrate that LLM4HeG significantly improves node classification on heterophilic graphs, underscoring the potential of LLMs for advancing complex graph learning.

Limitations

Despite the promising results obtained by our approach LLM4HeG, it is important to acknowledge several limitations. (1) LLM4HeG follows a two-stage pipeline, which may lead to error accumulation between stages, as compared to end-to-end approaches that jointly optimize the entire process. (2) The effectiveness of LLM4HeG depends on the availability and quality of textual data associated with nodes, particularly the alignment between the semantics of textual attributes and the class labels. The model may struggle to deliver optimal results when textual data is sparse or irrelevant. (3) LLM4HeG operates in a supervised learning paradigm, requiring labeled data for training, which can limit its scalability and applicability in domains where labeled data is scarce or costly to obtain.

Acknowledgments

This research / project is supported by the Ministry of Education, Singapore, under its Academic Research Fund Tier 2 (Proposal ID: T2EP20122-0041). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of the Ministry of Education, Singapore.

References

Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. 2019. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *international conference on machine learning*, pages 21–29. PMLR.

Wendong Bi, Lun Du, Qiang Fu, Yanlin Wang, Shi Han, and Dongmei Zhang. 2024. Make heterophilic graphs better fit gnn: A graph rewiring approach. *IEEE Transactions on Knowledge and Data Engineering*, 36(12):8744–8757.

Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. 2021. Beyond low-frequency information in graph convolutional networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 3950–3957.

Cristian Bodnar, Francesco Di Giovanni, Benjamin Chamberlain, Pietro Lio, and Michael Bronstein. 2022. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns. *Advances in Neural Information Processing Systems*, 35:18527–18541.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *NeurIPS*.

Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, et al. 2024. Exploring the potential of large language models (llms) in learning on graphs. *ACM SIGKDD Explorations Newsletter*, 25(2):42–61.

Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. 2021. Adaptive universal generalized pagerank graph neural network. In *International Conference on Learning Representations*.

Lun Du, Xiaozhou Shi, Qiang Fu, Xiaojun Ma, Hengyu Liu, Shi Han, and Dongmei Zhang. 2022. Gbk-gnn: Gated bi-kernel graph neural networks for modeling both homophily and heterophily. In *Proceedings of the ACM Web Conference 2022*, pages 1550–1558.

Chenghua Gong, Yao Cheng, Jianxiang Yu, Can Xu, Caihua Shan, Siqiang Luo, and Xiang Li. 2024. A survey on learning from graphs with heterophily: Recent advances and future directions. *arXiv preprint arXiv:2401.09769*.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. Minillm: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*.

Jiayan Guo, Lun Du, Hengyu Liu, Mengyu Zhou, Xinyi He, and Shi Han. 2023. Gpt4graph: Can large language models understand graph structured data? an empirical evaluation and benchmarking. *arXiv preprint arXiv:2305.15066*.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Xiaoxin He, Xavier Bresson, Thomas Laurent, Adam Perold, Yann LeCun, and Bryan Hooi. 2024. Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning. In *The Twelfth International Conference on Learning Representations*.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2022. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Wei Jin, Tyler Derr, Yiqi Wang, Yao Ma, Zitao Liu, and Jiliang Tang. 2021. Node similarity preserving graph convolutional networks. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 148–156.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2023. Bloom: A 176b-parameter open-access multilingual language model.

Liunian Harold Li, Jack Hessel, Youngjae Yu, Xiang Ren, Kai-Wei Chang, and Yejin Choi. 2023a. Symbolic chain-of-thought distillation: Small models can also "think" step-by-step. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2665–2679.

Xiang Li, Renyu Zhu, Yao Cheng, Caihua Shan, Siqiang Luo, Dongsheng Li, and Weining Qian. 2022. Finding global homophily in graph neural networks when meeting heterophily. In *International Conference on Machine Learning*, pages 13242–13256. PMLR.

Yuhan Li, Zhixun Li, Peisong Wang, Jia Li, Xiangguo Sun, Hong Cheng, and Jeffrey Xu Yu. 2023b. A survey of graph meets large language model: Progress and future directions. *arXiv preprint arXiv:2311.12399*.

Langzhang Liang, Xiangjing Hu, Zenglin Xu, Zixing Song, and Irwin King. 2024. Predicting global label relationship matrix for graph neural networks under heterophily. *Advances in Neural Information Processing Systems*, 36.

Chen Ling, Zhuofeng Li, Yuntong Hu, Zheng Zhang, Zhongyuan Liu, Shuang Zheng, and Liang Zhao. 2024. Link prediction on textual edge graphs. *arXiv preprint arXiv:2405.16606*.

Chang Liu and Bo Wu. 2023. Evaluating large language models on graphs: Performance insights and comparative analysis. *arXiv preprint arXiv:2308.11224*.

Jiawei Liu, Cheng Yang, Zhiyuan Lu, Junze Chen, Yibo Li, Mengmei Zhang, Ting Bai, Yuan Fang, Lichao Sun, Philip S Yu, et al. 2023. Towards graph foundation models: A survey and beyond. *arXiv preprint arXiv:2310.11829*.

Haitao Mao, Zhikai Chen, Wenzhuo Tang, Jianan Zhao, Yao Ma, Tong Zhao, Neil Shah, Mikhail Galkin, and Jiliang Tang. 2024. Position: Graph foundation models are already here. In *Forty-first International Conference on Machine Learning*.

Bo Pan, Zheng Zhang, Yifei Zhang, Yuntong Hu, and Liang Zhao. 2024. Distilling large language models for text-attributed graph learning. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 1836–1845.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. 2020. Geom-gcn: Geometric graph convolutional networks. In *International Conference on Learning Representations*.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. 2023. A critical look at the evaluation of gnns under heterophily: Are we really making progress? In *The Eleventh International Conference on Learning Representations*.

Timo Schick and Hinrich Schütze. 2021. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352.

Yunchong Song, Chenghu Zhou, Xinbing Wang, and Zhouhan Lin. 2023. Ordered gnn: Ordering message passing to deal with heterophily and over-smoothing. In *The Eleventh International Conference on Learning Representations*.

Shengyin Sun, Yuxiang Ren, Chen Ma, and Xuecang Zhang. 2023. Large language models as topological structure enhancers for text-attributed graphs. *arXiv preprint arXiv:2311.14324*.

Yifei Sun, Haoran Deng, Yang Yang, Chunping Wang, Jiarong Xu, Renhong Huang, Linfeng Cao, Yang Wang, and Lei Chen. 2022. Beyond homophily: Structure-aware path aggregation graph neural network. In *IJCAI*, pages 2233–2240.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. 2024. Graphgpt: Graph instruction tuning for large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 491–500.

Jie Tang, Jimeng Sun, Chi Wang, and Zi Yang. 2009. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 807–816.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. 2024a. Can language models solve graph problems in natural language? *Advances in Neural Information Processing Systems*, 36.

Tao Wang, Di Jin, Rui Wang, Dongxiao He, and Yuxiao Huang. 2022. Powerful graph convolutional networks with adaptive propagation mechanism for homophily and heterophily. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 4210–4218.

Wenhai Wang, Zhe Chen, Xiaokang Chen, Jiannan Wu, Xizhou Zhu, Gang Zeng, Ping Luo, Tong Lu, Jie Zhou, Yu Qiao, et al. 2024b. Visionllm: Large language model is also an open-ended decoder for vision-centric tasks. *Advances in Neural Information Processing Systems*, 36.

Xiyuan Wang and Muhan Zhang. 2022. How powerful are spectral graph neural networks. In *International conference on machine learning*, pages 23341–23362. PMLR.

Han Xie, Da Zheng, Jun Ma, Houyu Zhang, Vassilis N Ioannidis, Xiang Song, Qing Ping, Sheng Wang, Carl Yang, Yi Xu, et al. 2023. Graph-aware language model pre-training on a large graph corpus can help multiple graph applications. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5270–5281.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, pages 5453–5462. PMLR.

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. 2024. A survey on knowledge distillation of large language models. *arXiv preprint arXiv:2402.13116*.

Yujun Yan, Milad Hashemi, Kevin Swersky, Yaoqing Yang, and Danai Koutra. 2022. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. In *2022 IEEE International Conference on Data Mining (ICDM)*, pages 1287–1292. IEEE.

Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. 2024. Language is all a graph needs. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 1955–1973.

Xingtong Yu, Yuan Fang, Zemin Liu, Yuxia Wu, Zhihao Wen, Jianyuan Bo, Xinming Zhang, and Steven CH Hoi. 2024a. Few-shot learning on graphs: from meta-learning to pre-training and prompting. *arXiv preprint arXiv:2402.01440*.

Xingtong Yu, Jie Zhang, Yuan Fang, and Renhe Jiang. 2024b. Non-homophilic graph pre-training and prompt learning. *arXiv preprint arXiv:2408.12594*.

Mengmei Zhang, Mingwei Sun, Peng Wang, Shen Fan, Yanhu Mo, Xiaoxiao Xu, Hong Liu, Cheng Yang, and Chuan Shi. 2024. Graphtranslator: Aligning graph model to large language model for open-ended tasks. In *Proceedings of the ACM on Web Conference 2024*, pages 1003–1014.

Haiteng Zhao, Shengchao Liu, Ma Chang, Hannan Xu, Jie Fu, Zhihong Deng, Lingpeng Kong, and Qi Liu. 2023a. Gimlet: A unified graph-text model for instruction-based molecule zero-shot learning. *Advances in Neural Information Processing Systems*, 36:5850–5887.

Tianxiang Zhao, Xiang Zhang, and Suhang Wang. 2024. Disambiguated node classification with graph neural networks. In *Proceedings of the ACM on Web Conference 2024*, pages 914–923.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023b. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.

Xin Zheng, Yi Wang, Yixin Liu, Ming Li, Miao Zhang, Di Jin, Philip S Yu, and Shirui Pan. 2022. Graph neural networks for graphs with heterophily: A survey. *arXiv preprint arXiv:2202.07082*.

Jiong Zhu, Ryan A Rossi, Anup Rao, Tung Mai, Nedim Lipka, Nesreen K Ahmed, and Danai Koutra. 2021. Graph neural networks with heterophily. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11168–11176.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. 2020. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in neural information processing systems*, 33:7793–7804.

Dongcheng Zou, Hao Peng, Xiang Huang, Renyu Yang, Jianxin Li, Jia Wu, Chunyang Liu, and Philip S Yu. 2023. Se-gsl: A general and effective graph structure learning framework through structural entropy optimization. In *Proceedings of the ACM Web Conference 2023*, pages 499–510.

Appendices

A More Details for Datasets

- Cornell, Texas, and Wisconsin (Pei et al., 2020) are collected from computer science departments at various universities. In these datasets, each node corresponds to a web page, while edges represent hyperlinks connecting these pages. In our experiments, we use the original webpage data as the textual information for each node.

- Actor (Pei et al., 2020; Tang et al., 2009) is an actor-only induced subgraph of the film-director-actor-writer network. In this graph, nodes represent actors, and an edge between two nodes indicates their co-occurrence on the same Wikipedia page. The task involves categorizing actors into five distinct classes based on their roles. We selected the actors based on category information provided in the metadata, focusing on those with high occurrence frequencies. The category keywords of the selected actors include “American film actors", “American film and television actors", “American stage and television actors", “English" and “Canadian". Afterward, we construct the graph based on the edges and remove the isolated nodes from the graph.

- Amazon (Platonov et al., 2023) is constructed from the Amazon product co-purchasing network metadata. In this dataset, nodes represent products such as books, music CDs, DVDs, and VHS video tapes. Edges link products that are frequently bought together. The goal is to predict the average rating a product receives from reviewers, with ratings grouped into five classes. To manage the graph’s complexity, only the largest connected component of the 5-core of the graph is considered.

B More Implementation Details

Node Pairs. In Stage 1, we train the edge discriminator by sampling node pairs from the graph,

Dataset	Cornell	Texas	Wisconsin	Actor	Amazon
Training	4,186	3,741	7,626	36,248	23,210
Distillation⋆	916	991	1,299	1,781	11,422

⋆: the number of additional samples for distillation.

Table 6: The number of node pairs in Stage 1 and the distillation process.

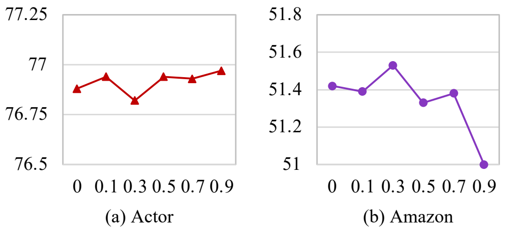
with the selection process tailored to the characteristics of each dataset. For small graphs like Cornell, Texas, and Wisconsin, we select all node pairs within the training set, including those without direct edges. For the Actor dataset, node pairs are selected based on the 1-hop and 2-hop neighbor relationship, while for the Amazon dataset, we focus on node pairs with a 1-hop neighbor relationship, taking into account the graph's size.

In the distillation process, we use node pairs from the validation and testing data as additional samples, allowing the fine-tuned LLM to generate pseudo-labels. For small graphs like Cornell, Texas, and Wisconsin, we select node pairs with 1-hop and 2-hop neighbor relationships, while for the Actor and Amazon datasets, we choose node pairs with 1-hop neighbor relationships. The number of node pairs used for training in Stage 1 and distillation is shown in Table 6.

Backbones. We provide the implementation details for integrating the LLM4HeG with other backbones. As discussed in Sec. 3.4, the edge reweighting in Stage 2 combines the LLM-based weight with graph-based information. For the backbones that don't contain an additional edge weight learning module (*e.g.*, GCN (Kipf and Welling, 2016) and H2GCN (Zhu et al., 2020)), we only use the edge weight obtained from LLM. For the backbones with specific designs of the edge weight (*e.g.*, GAT (Veličković et al., 2018), FAGCN (Bo et al., 2021) and GBK-GNN (Du et al., 2022)), we follow the Eq. (6) to combine the LLM-based weight and graph-based weight to perform fine-grained context aggregation of GNN.

C Hyper-parameters

We run all the experiments on an NVIDIA A800 GPU. For LLM4HeG, the edge weight margin $\alpha = 0.3$ and the regularization coefficient $\lambda = 0.1$. For the baselines, we use the hyper-parameters as listed in previous literature. For the number of hops in the i-hop neighborhood of each node, we use a 1-hop and 2-hop neighborhood for H2GCN (Zhu et al., 2020) and a 1-hop neighborhood for the other backbones. The hidden unit for GCN and GAT is 16. The number of heads in GAT is 3 for Amazon and 8 for other datasets. For H2GCN, we adopt the H2GCN-1 variant using one embedding round ($K = 1$). The parameter setting of FAGCN is: the hidden unit = 32, layers = 2, $\epsilon = 0.4$. For JacobiConv, the parameter setting is $\gamma = 2$ for Polynomial Coefficient Decomposition (PCD), $a = 0.5$ and $b = 0.25$ for Jacobi Basis. For OGNN, the number of MLP layers is 1 for Cornell, Texas and Wisconsin and 2 for the Actor and Amazon dataset. For SEGSL, the height of the encoding tree $K = 2$. The subtree sampling parameter θ is 2 for the Amazon dataset and 3 for other datasets. The GNN encoder model for the reconstructed graph is GraphSAGE. For DisamGCL, the weight of historical memory $\mu = 0.6$, the controlling variables for the node similarity $\epsilon_1 = 0.74$ and ϵ_2 = 0.4, the threshold of the node similarity $\mathcal{T} = 0.8$, the number of augment instances $K = 8$, the weight of contrastive loss $\lambda = 1$.



(a) Actor

(b) Amazon

Figure 6: Effect of the edge weight margin α.

The effect of edge weight margin α of LLM4HeG for other datasets is shown in Fig. 6. Generally speaking, the results in Fig. 5 and Fig. 6 show that $[0.3, 0.5]$ appears to be a good range for α on all datasets.

D More Details for Baselines

- H2GCN (Zhu et al., 2020) considers higher-order neighbors, ego-neighbor embedding separation and intermediate layer representations for heterophilic graph.

- FAGCN (Bo et al., 2021) employs a self-gating mechanism to adaptively integrate low- and high-frequency signals during message passing.

- JacobiConv (Wang and Zhang, 2022) deserts non-linearity and approximates filter functions with Jacobi polynomial bases.

- GBK-GNN (Du et al., 2022) introduces a learnable kernel selection gate to discriminate node

	Cornell	Texas	Wisconsin	Actor	Amazon
LLM-nofinetune	26.16	25.00	26.32	24.75	35.44
LLM-finetune	61.90	40.91	71.93	59.96	36.52
LLM4HeG	**75.95**	**87.50**	**84.91**	**76.54**	**51.53**

Table 7: Performance comparison with LLMs.

pairs and apply two different kernels for homophily and heterophily node pairs.

- OGNN (Song et al., 2023) introduces an ordered gating mechanism for message passing, effectively handling heterophily and mitigating the over-smoothing problem.
- SEGSL (Zou et al., 2023) is a graph structure learning framework leveraging structural entropy and the encoding tree to improve both the effectiveness and robustness.
- DisamGCL (Zhao et al., 2024) automatically identifies ambiguous nodes and dynamically augments the learning objective through a contrastive learning framework.

E Comparison to Direct LLM Predictions

Given the rich textual data associated with the nodes, it is possible to directly feed this information into the LLM for classification, bypassing the use of a GNN. We conduct experiments using the original and fine-tuned LLMs.

(1) “LLM-nofinetune” employs the original Vicuna 7B model to make the prediction, with the following prompt template for the Cornell, Texas, and Wisconsin datasets. A similar prompt template with different background and task descriptions is used for the Actor and Amazon datasets.

> *Background:* I have a dataset containing web page information collected from computer science department websites of various universities. These web pages have been manually categorized into five categories, including student, staff, faculty, course, and project.
>
> *Task:* I will provide you with the text information of a web page, and I would like you to classify it into one of the following categories: student, staff, course, faculty, or project.
>
> The web page content: <text>
>
> You may only output the category name, and do not discuss anything else!

(2) “LLM-finetune” employs the fine-tuned Vicuna 7B model using the edge discriminator in Stage 1 to make the prediction, using the same prompt template as above.

As shown in Table 7, the performance of the original LLM without fine-tuning performs poorly when directly provided with textual information. This is likely because while LLMs are powerful for natural language processing tasks, they struggle to infer node categories without adaptation to the specific task. After fine-tuning with the LLM-enhanced edge discriminator in Stage 1, the performance improves significantly, but is still worse than LLM4HeG. This highlights the importance of combining node semantic features, structural contexts, and LLM-inferred edge characteristics for effective node classification.