PATH: A Dual Perspective for High-quality Text-attributed Graph Learning

Yifan Wang[1] **Yuhang Pei**[2*] **Changhu Wang**[3] **Fanchun Meng**[2] **Tao Ren**[2]
Wei Ju[4] **Chong Chen**[5] **Xian-Sheng Hua**[6] **Xiao Luo**[7]
[1] University of International Business and Economics, China
[2] Northeastern University, China
[3] Fred Hutchinson Cancer Research Center, USA
[4] Peking University, China
[5] Hesicare Technology Co. Ltd, China
[6] Tongji University, China
[7] University of Wisconsin-Madison, USA

Abstract

This paper studies the problem of zero-shot text-attributed graph learning, which aims to generate high-quality node representations in unseen text-attributed graphs. Recent approaches usually utilize large language models (LLMs) instead of graph neural networks (GNNs) to extract semantics due to their strong generalization ability, which could neglect the intrinsic geometric structure. Towards this end, in this paper, we propose a novel approach named Prototypical Mutual Prompting Enhancement (PATH) for zero-shot text-attributed graph learning. The core of our PATH is to generate high-quality prompts using dual prototypical learning to combine the advantages of both language models and graph models. In particular, we first utilize dual graph pre-training from both instance and informativeness perspectives to generate a generalizable GNN. Then, we incorporate the frozen language and graph models into a mutual prompt learning framework. On the one hand, we extract node tokens with geometric relationships using the graph model, which will be sent to multiple prototypical projections to enhance the understanding of the language model. On the other hand, we extract graph information and task descriptions using the language model, which serves as instruction for the graph models. Extensive experiments on node classification and link prediction validate the effectiveness of the proposed PATH against strong baselines. Our code is available at https://github.com/kalae-2000/PATH-code.

1 Introduction

Graph serves as a versatile data structure for effectively capturing intricate relationships and dependencies. A notable variation is the Text-Attributed Graph (TAG), where textual information, such as node or edge descriptions, is incorporated into the graph to enhance its data representation [28]. This integration makes TAGs particularly valuable in various domains, including social network analysis [2] and recommender systems [43]. Graph models, especially Graph Neural Networks (GNNs), have achieved remarkable performance and become a *de facto* approach for graph-based machine learning. Despite success, these graph models are usually trained or fine-tuned for a particular dataset or task, struggling to retain their effectiveness when applied to new datasets or tasks [24, 26].

Fortunately, the emergence of Large Language Models (LLMs) has significantly advanced the zero-shot capabilities of machine learning models. By leveraging vast amounts of encoded pre-existing

*Correspondence to: Yuhang Pei (peiyh@mails.neu.edu.cn)

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

knowledge, LLMs can effectively generalize to new datasets or tasks, making them highly adaptable across various fields. For instance, in the natural language processing (NLP) field, models like GPT-4 [1] and Llama [37] unify all the tasks as a generative paradigm, allowing them to handle tasks they have never seen before. In the computer vision (CV) field, models such as CLIP [31] employ a retrieval-based approach, mapping images and textual descriptions into the same embedding space to enable zero-shot recognition of new images. However, because LLMs are designed for sequential text modeling, applying them to graph-related tasks remains challenging, especially for encoding graph structure and relational dependencies in practice, where both topology and semantics must be preserved.

In recent years, leveraging the strength of LLMs for graph models has sparked growing interest [14, 15, 53]. LLM as Enhancer [51, 8, 27] leverages language models instead of traditional shallow embedding methods such as Bag of Words (BoW) to enrich the graph feature space. These approaches have shown promising performance since their effectiveness in capturing semantic nuances, but they are still constrained by their reliance on GNNs for final predictions. LLM as Aligner [44] further maps both graph and corresponding text modalities into a shared embedding space, focusing on transferring pre-trained models within the same graph. In contrast, LLM as Predictor [18, 12, 55] directly translates graph data into plain texts suitable for LLMs and uses them for specific predictions, leveraging the zero-shot ability of LLMs for graph reasoning tasks in practical transfer settings.

Despite the promising performance of these methods, formalizing a framework for zero-shot graph learning remains challenging since two questions are required to tackle: ❶ *How to leverage the GNN to generate the transferable representation of the intrinsic graph structure?* Graphs inherently contain complex structural dependencies that traditional LLMs may not capture effectively. The GNN model needs to be fine-tuned to generate strong expressive embeddings that can be generalized across tasks and domains. ❷ *How to integrate the transferable representation into LLM that works effectively for zero-shot graph learning?* Unlike structured graph models, LLMs operate on sequential text-based inputs. This presents a challenge in aligning graph-generated embeddings, which capture the structural dependencies of the graph, with the LLM model to ensure that the graph's information is effectively interpreted and utilized by the LLM for zero-shot learning tasks.

Towards this end, in this paper, we propose a novel Prototypical Mutual Prompting Enhancement (`PATH`) approach, which combines the advantages of both GNNs and LLMs to generate high-quality prompts for zero-shot text-attributed graph learning. Specifically, we first perform the dual graph pre-training, which considers two perspectives. The instance view focuses on learning node representations based on immediate neighbors to capture the structural relationships in the graph. The informativeness view emphasizes identifying and leveraging the parts most relevant to the LLM token embeddings for alignment between the two models. Then, we integrate the frozen graph and language models into a mutual prompt learning framework. On the one hand, the graph model extracts node tokens with geometric relationships, which are then passed through prototypical projections to transform the graph into a more comprehensible format to enhance the LLM model. On the other hand, the LLM processes graph-related information and task descriptions, providing high-level instructions as prompts to enhance the graph model. This mutual prompting improves cross-model interaction and strengthens overall zero-shot graph learning performance.

The contribution of the paper can be summarized as follows: (1) *New Connection.* We pioneer a new perspective to utilize dual prototypical learning to combine the advantages of both language models and graph models for zero-shot text-attributed graph learning. (2) *Novel Methodology*. Our `PATH` not only leverages graph models to extract geometric relationships for language model prompting, but also generates text-based prompting using prototypical projections for graph model enhancement. (3) *Extensive Experiments.* Extensive experiments on both node classification and link predictions validate the superiority of our proposed `PATH`.

2 Preliminary

Notations. We represent a graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{A}, \boldsymbol{X})$, where $\mathcal{V}$ comprises N nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ specifies the edge set. The graph topology and node attributes are encoded by the adjacency matrix $\boldsymbol{A} \in \{0,1\}^{N \times N}$, where $\boldsymbol{A}_{uv} = 1$ if $(u, v) \in \mathcal{E}$, otherwise $\boldsymbol{A}_{uv} = 0$. The node feature matrix is denoted by $\boldsymbol{X} \in \mathbb{R}^{N \times F}$, where each row $\boldsymbol{x}_v \in \mathbb{R}^F$ is the F-dimensional vector containing attribute information for node v. For node classification, each node v is assigned a label $y_v \in \mathcal{Y}$.

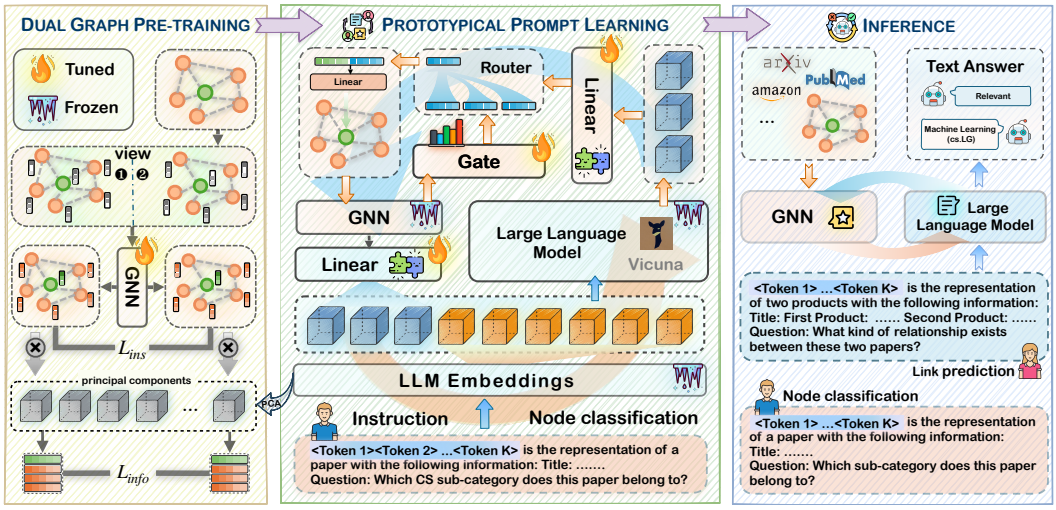

Figure 1: The overall framework of the proposed PATH. The framework consists of two phases: (1) Dual graph pre-training captures instance-level structural relationships and informativeness-aware semantic alignment with LLM token embeddings. (2) Mutual prompt learning enables iterative co-enhancement: GNN tokens guide LLMs via prototypical projections, while LLM instructions strengthen GNNs through specialized prompt experts.

Graph Neural Networks. Graph Neural Networks (GNNs) have become a foundational framework for learning effective representations of graph-structured data. By employing a message-passing paradigm, GNNs iteratively update node representations by embedding both the graph topology and node features. Specifically, at the l-th layer, each node $v \in \mathcal{V}$ aggregates information from its neighbors $\mathcal{N}_v$ and combines it with its previous-layer embedding $\boldsymbol{h}_v^{(l-1)}$ for update:

$$\boldsymbol{z}_v^{(l)} = \mathcal{C}^{(l)}\left(\boldsymbol{z}_v^{(l-1)}, \mathcal{A}^{(l)}\left(\{\boldsymbol{z}_u^{(l-1)}\}_{u\in\mathcal{N}_v}\right)\right), \tag{1}$$

where $\mathcal{A}^{(l)}$ and $\mathcal{C}^{(l)}$ are the two functions that aggregate and combine embedding from the neighborhood. By iteratively stacking L message-passing layers, the node representation can be $Z = \{\mathbf{z}_1^{(L)}, \ldots, \mathbf{z}_N^{(L)}\} \in \mathbb{R}^{N\times F_G}$, where F_G denotes the representation dimension.

Zero-Shot Text-attributed Graph Learning. Recently, zero-shot learning has been developed in areas like image and text data, enabling models to generalize to new classes or tasks without relying on labeled data from the target domain. In this work, we aim to study zero-shot learning for graph data, with a particular emphasis on *cross-dataset* and *cross-task* scenarios. For *cross-dataset zero-shot learning*, we aim to train a classifier on a fully labeled source graph $\mathcal{G}^s$ and test it on a different target graph $\mathcal{G}^t$, where $\mathcal{G}^s \cap \mathcal{G}^t = \emptyset$ and $\mathcal{Y}_s \cap \mathcal{Y}_t = \emptyset$. For *cross-task zero-shot learning*, we directly evaluate link prediction on previously unseen target graphs using a model trained for node classification, without fine-tuning it for the target task.

3 The Proposed PATH

3.1 Framework Overview

The overview of our proposed zero-shot graph learning framework is illustrated in Figure 1. By pre-training the GNN model and aligning it with the LLM through mutual prompting, the PATH is capable of exhibiting substantial zero-shot learning abilities in both cross-dataset and cross-task scenarios. Our PATH framework consists of two phases. The GNN model is first pre-trained with the LLM's token embeddings from both instance and informativeness perspectives to capture graph structural relationships while identifying and leveraging the parts relevant to the LLM token embeddings for alignment between the two models (see Section 3.2). Then, we align the GNN with the LLM in an iterative mutual prompt learning manner to effectively transfer knowledge between the two models. On the one hand, we derive node tokens that encode geometric relations and pass these tokens through prototypical projections, which transform the graph into a more comprehensible format to enhance

the LLM. On the other hand, the LLM processes graph-related information and task descriptions, generating high-level instructions as prompts to refine the graph model (see Section 3.3).

3.2 Dual Graph Pre-training for Generalizable GNNs

In this part, we introduce a graph pre-training strategy to capture the transferable node representations suitable for alignment with LLMs. In general, prior research on GNN pre-training has explored various formulations of self-supervised pretext tasks, especially contrastive methods [57], which offer broader applicability and overlapping task sub-spaces for better knowledge transfer.

Instance-aware Pre-training. Given the training graph, we derive two augmented graph views by applying edge removal (RE) and node feature masking (MF) to perturb graph topology and node attributes respectively. For the RE strategy, we generate a binary matrix $\widetilde{\boldsymbol{R}} \in \{0,1\}^{N\times N}$, whose entries are sampled from a Bernoulli distribution $\widetilde{\boldsymbol{R}}_{ij} \sim \mathcal{B}(1-p_r)$ to mask edges with probability p_r, which are computed as:

$$\widetilde{\boldsymbol{A}} = \boldsymbol{A} \circ \widetilde{\boldsymbol{R}}, \tag{2}$$

where $\circ$ here denotes the Hadamard product. Similarly, for the MF strategy, we generate a random masking vector $\widetilde{\boldsymbol{m}} \in \mathbb{R}^F$ from another Bernoulli distribution $\widetilde{\boldsymbol{m}}_i \sim \mathcal{B}(1-p_m)$ with probability p_m. The masked node feature $\widetilde{\boldsymbol{X}}$ can be:

$$\widetilde{\boldsymbol{X}} = [\boldsymbol{x}_1 \circ \widetilde{\boldsymbol{m}}; \cdots ; \boldsymbol{x}_N \circ \widetilde{\boldsymbol{m}}]^{\mathrm{T}}. \tag{3}$$

The two views of the graph can be generated as $\widetilde{\mathcal{G}}_1 = (\widetilde{\boldsymbol{A}}_1, \widetilde{\boldsymbol{X}}_1)$ and $\widetilde{\mathcal{G}}_2 = (\widetilde{\boldsymbol{A}}_2, \widetilde{\boldsymbol{X}}_2)$. Then, we encode the two graph views to get the node embeddings, denoted as:

$$\boldsymbol{Z}^* = f_{\mathrm{GNN}}(\widetilde{\boldsymbol{A}}^*, \widetilde{\boldsymbol{X}}^*), * \in \{1,2\}, \tag{4}$$

where $\boldsymbol{Z}^* = \{\boldsymbol{z}_1^*, \ldots, \boldsymbol{z}_N^*\} \in \mathbb{R}^{N\times F_G}$. We further formulate a contrastive objective that promotes alignment between representations of the same node across views while enforcing separation between representations of distinct nodes:

$$\ell(\boldsymbol{z}_v^1, \boldsymbol{z}_v^2) = \quad \log\Big(\frac{\phi(\boldsymbol{z}_v^1, \boldsymbol{z}_v^2)}{\sum_{u=1}^{N}\phi(\boldsymbol{z}_v^1, \boldsymbol{z}_u^1) + \sum_{u\neq v}\phi(\boldsymbol{z}_v^1, \boldsymbol{z}_u^2)}\Big), \tag{5}$$

The corresponding instance-level objective, using temperature parameter τ, is given as follows:

$$\mathcal{L}_{ins} = \frac{1}{2N}\sum_{v=1}^{N}[\ell(\boldsymbol{z}_v^1, \boldsymbol{z}_v^2) + \ell(\boldsymbol{z}_v^2, \boldsymbol{z}_v^1)]. \tag{6}$$

Informativeness-aware Pre-training. Since a notable discrepancy exists between the node representations and the semantic space of LLMs, we introduce informativeness-aware contrastive learning with token embeddings to bridge this gap. Specifically, we employ principal component analysis (PCA) to extract the top P principal components $\boldsymbol{C} \in \mathbb{R}^{P\times F_L}$ from the token embeddings of LLMs, where F_L is the token embedding dimension. These components represent the directions that maximize variance in token embeddings and serve as coordinate axes for aligning node representations with the textual embedding space. We then map the node representations to the space as:

$$\boldsymbol{Z}^* = \boldsymbol{Z}^* \times \boldsymbol{C}^{\mathrm{T}}. \tag{7}$$

In practice, we set $F_G = F_L$ to facilitate mapping. And we break the independence between nodes to conduct the informativeness-aware contrastive learning:

$$\mathcal{L}_{info} = \frac{1}{F_L}\sum_{i=1}^{F_L}\frac{\phi(\boldsymbol{u}_i^1, \boldsymbol{u}_i^2)}{\sum_{j=1}^{F_L}[\phi(\boldsymbol{u}_i^1, \boldsymbol{u}_j^1) + \phi(\boldsymbol{u}_i^1, \boldsymbol{u}_j^2)]}, \tag{8}$$

where $(\boldsymbol{Z}^*)^{\mathrm{T}} = \{\boldsymbol{u}_1^*, \ldots, \boldsymbol{u}_P^*\} \in \mathbb{R}^{P\times N}$. Final objective for token-aligned graph pre-training is:

$$\mathcal{L} = \frac{1}{2}(\mathcal{L}_{ins} + \mathcal{L}_{info}). \tag{9}$$

3.3 Prototypical Prompt Learning for Mutual Enhancement

The advent of LLMs provides a new approach for graph learning. However, existing research [22] suggests that LLMs alone are insufficient for fully comprehending the graph data. Thus, given the pre-trained GNN, we aim to learn transferable prompts [9, 34, 50] that enable effective mutual alignment between the GNN and LLM throughout adaptation.

Geometric Prompting for Language Model. To enable the LLM to capture graph data more effectively and enhance its performance in zero-shot graph learning tasks, we introduce a new graph-guided prompt tuning that includes specially designed instructions. Here, the instruction can be divided into two parts. We first provide the context to describe the graph information and then introduce the goal of the task. For the graph information, the encoded graph representations from the pre-trained GNN are utilized to construct the soft prompt. We further add the node's text attribute to enhance the LLMs' understanding. The graph information in the instructions can be presented as follows: $\langle\mathsf{graph}\rangle$ *is/are the representation(s) of a paper/two papers/a paper set with the following information: Title: First paper:*$\{\mathsf{title}_1\}\ldots\backslash n$, where $\langle\mathsf{graph}\rangle$ and $\{\mathsf{title}_1\}$ denote the placeholders for both graph representation and text description inputs. Given the intricate nature of graphs and their diverse semantics, relying on a single prompt instruction may fail to cover the entire prompt space, thereby limiting the model's ability to capture the full spectrum of information on the targeted task. To address this, we introduce a set of GNN prototypes to characterize the entire prompt space, dividing it into several homogeneous regions, with each region being handled by a specialized prototype. Given the dual graph pre-training, the linear projector is sufficient to capture the mapping relationship:

$$\boldsymbol{H}_v = \{\boldsymbol{h}_v^1, \ldots, \boldsymbol{h}_v^K\}, \quad \boldsymbol{h}_v^k = f_{\text{Linear}}^{G,k}(\boldsymbol{z}_v), \tag{10}$$

where $\boldsymbol{H}_v \in \mathbb{R}^{K\times F_L}$ denotes K distinct spaces, $\boldsymbol{h}_v^k \in \mathbb{R}^{F_L}$ is the projected k-th node embedding for the LLM, and $f_{\text{Linear}}^{G,k}(\cdot)$ denotes the k-th linear project function. In this way, we replace $\langle\mathsf{graph}\rangle$ with K token embeddings as the soft prompt for the LLM, and the output token can be seen as diverse experts for the prompt of the GNN model. For the task descriptions, we directly add the question and alternative answers for the task to construct the instruction. Take the node classification task as an example. The instruction can be formulated as follows: *Which category does this paper belong to? Please directly choose the most likely answer from the following categories:* $\{\mathsf{ans}\}$, where $\{\mathsf{ans}\}$ denotes the set of alternative answers for each target dataset and evaluation setting [40].

Language Prompting for Graph Model. Since the LLM encodes both graph information and task descriptions, its semantic richness can be utilized to guide the generation of graph prompts that emphasize class-specific representations. Specifically, let $\boldsymbol{h}_k \in \mathbb{R}^{F_L}$ represent the k-th output token corresponding to $\langle\mathsf{graph}\rangle$ in the LLM. We apply an additional projector to map $\boldsymbol{h}_k$ to the k-th soft graph prompt $\boldsymbol{p}^k$, which is formally defined as follows:

$$\boldsymbol{P} = \{\boldsymbol{p}^1, \ldots, \boldsymbol{p}^K\}, \quad \boldsymbol{p}^k = f_{\text{Linear}}^{L_1,k}(\boldsymbol{h}_k), \tag{11}$$

where $\boldsymbol{P} \in \mathbb{R}^{K\times F_L}$, $f_{\text{Linear}}^{L_1}(\cdot)$ denotes the k-th linear projectors. The output tokens can be seen as diverse experts for the prompt of the GNN model. We then introduce a gating mechanism with a router model that decides how the input should be directed to the appropriate soft prompt, depending on the relevant semantic context. The prototypical weight of the k-th prompt can be:

$$w_k(\boldsymbol{z}_v) = \left[\text{Softmax}(f_{\text{Linear}}^{R}(\boldsymbol{z}_v) \circ (1+\delta))\right]_k, \tag{12}$$

where $\delta \sim \mathcal{N}(0,1)$ denotes the scaled Gaussian noise to encourage exploration of inputs over diverse prompts. The final soft prompt is given by the following weighted combination of the prompt set:

$$\boldsymbol{x}_v' = [\boldsymbol{x}_v; \boldsymbol{w}^{\mathrm{T}}\boldsymbol{P}], \boldsymbol{x}_{v,p} = f_{\text{Linear}}^{L_2}(\boldsymbol{x}_v'), \tag{13}$$

where $\boldsymbol{w} \in \mathbb{R}^K = \{w_1, \ldots, w_K\}$, $[\boldsymbol{x}; \boldsymbol{y}]$ here denotes the concatenation operation between $\boldsymbol{x}$ and $\boldsymbol{y}$. Note that through K prompt experts, we can allow each expert to focus on and specialize in a specific region to enable the model's generalization [47]. $\boldsymbol{x}_{v,p}$ denote the prompted node feature.

Regularization to Mitigate Collapse. To prevent a trivial solution where only one group of experts is consistently selected, we introduce two additional regularizations. For the geometric prompting in the LLM, we enforce that the projected K token embeddings are independent of each other by introducing a constraint of orthogonality between each token. The independent loss can be:

$$\mathcal{L}_{ind} = \frac{1}{N}\sum_{v\in\mathcal{V}} |\boldsymbol{H}_v\boldsymbol{H}_v^{\mathrm{T}} - I|, \tag{14}$$

where $|\cdot|$ is the L_1 norm and I denotes the identity matrix. Inspired by the previous work [41], for language prompting in the GNN model, the importance loss of each expert can be:

$$\text{Imp}(w)_k = \sum_{v \in \mathcal{V}} (w_k(\boldsymbol{z}_v)), \mathcal{L}_{imp} = \text{CV}(\text{Imp}(w))^2, \tag{15}$$

where $\text{CV}(\cdot)$ represents the coefficient of variation. Here the importance loss measures the variation of routing probabilities and enforces each expert to be similarly important.

3.4 Model Training and Evaluation

To facilitate the iterative mutual prompt tuning between the GNN and LLM models, we first utilize the embedding layer of Vicuna-7B-v1.5 [56] to encode raw text as node features, followed by dual graph contrastive learning from both instance and informativeness perspectives to pre-train the GNN model. During each time of mutual alignment tuning, we freeze the GNN model, leverage the prompted node feature matrix $\boldsymbol{X}_p$, and train the corresponding linear projectors, along with a prompt router, using the dual graph pre-training loss (Equation 9) together with the importance regularization (Equation 15) for language prompting in the GNN model. Then, we in turn freeze the LLM and directly train the linear projector on the downstream-specific task within the same dataset, as well as the orthogonality regularization (Equation 14) for the geometric prompting in the LLM model. We denote the number of iterative mutual alignment tuning times as I. Finally, we evaluate the model performance on unseen datasets and tasks.

3.5 Theoretical Analysis of PATH

Here, we provide an intuitive analysis for the mixture-of-experts prompting component in PATH, showing that omitting mixed prompting can limit the model's ability to capture task complexity and can introduce bias into the promoted node feature matrix $\boldsymbol{X}_p$ during downstream transfer.

For clarity, let $\boldsymbol{Y}_p$ denote the promoted node feature matrix derived from a single prompt instruction, and $\boldsymbol{Y}_p^M$ represent the matrix obtained using mixed prompt instructions. The true node feature matrices are denoted as:

$$\boldsymbol{X}_p = (\boldsymbol{X}, \boldsymbol{p}) \quad \text{and} \quad \boldsymbol{X}_p^M = (\boldsymbol{X}, \boldsymbol{P}),$$

where $\boldsymbol{p}$ is one of K prompt instructions in the set $\boldsymbol{P}$, and $\boldsymbol{P}_{-1}$ is the set of prompt instructions excluding $\boldsymbol{p}$. We assume the true promoted node feature matrix $\boldsymbol{Y}_p^M$ is linearly related to $\boldsymbol{X}_p^M$:

$$\boldsymbol{Y}_p^M = \boldsymbol{X}_p^M \boldsymbol{W} = \boldsymbol{X}_p \boldsymbol{W}_1 + \boldsymbol{P}_{-1} \boldsymbol{W}_2,$$

where $\boldsymbol{W}$ is the weight matrix learned by the neural network, and $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are sub-matrices of $\boldsymbol{W}$. This linear relationship reflects that the mixed prompt instructions are computed as the inner product of routed prompts and probability values, i.e., $\boldsymbol{w}^\top \boldsymbol{P}$, and a linear projector $f_{\text{Linear}}^{L_2}(\cdot)$.

The following theorem highlights the potential bias introduced by a single prompt instruction.

Theorem 3.1. *Under the MSE loss, and assuming that $\boldsymbol{X}_p^\top \boldsymbol{X}_p$ is invertible, using a single prompt instruction introduces bias in the predicted promoted node feature matrix $\boldsymbol{Y}_p$ relative to the true promoted matrix $\boldsymbol{Y}_p^M$:*

$$\boldsymbol{Y}_p^M - \boldsymbol{Y}_p = (\boldsymbol{I} - \boldsymbol{X}_p (\boldsymbol{X}_p^\top \boldsymbol{X}_p)^{-1} \boldsymbol{X}_p^\top) \boldsymbol{P}_{-1} \boldsymbol{W}_2,$$

where $\boldsymbol{I} - \boldsymbol{X}_p (\boldsymbol{X}_p^\top \boldsymbol{X}_p)^{-1} \boldsymbol{X}_p^\top$ is a projection matrix.

The proof of our Theorem 3.1 can be found in Appendix D. Theorem 3.1 shows that without mixed prompting, the bias in $\boldsymbol{Y}_p$ arises from the projection matrix and the weight matrix $\boldsymbol{W}_2$. This bias occurs because a single prompt instruction fails to capture the full complexity of the task. By leveraging diverse mixed prompt strategies, this bias can be significantly reduced, enabling the model to more accurately approximate the true promoted node feature matrix.

4 Experiments

4.1 Experimental Setup

Datasets. Our evaluation covers eight commonly used text-attributed graph datasets drawn from two distinct domains. The citation domain includes Arxiv [21], Pubmed [20], and the extended

Table 1: Zero-shot node classification accuracy under **cross-dataset** transfer in the citation and e-commerce domains (Bold and underline indicate the best and second-best results, respectively).

	Model	Pubmed	Cora	Children	History	Photo	Sports
	MLP	$0.323_{\pm 0.027}$	$0.021_{\pm 0.006}$	$0.029_{\pm 0.037}$	$0.080_{\pm 0.041}$	$0.110_{\pm 0.070}$	$0.042_{\pm 0.021}$
GNN as Predictor	GCN	$0.288_{\pm 0.092}$	$0.017_{\pm 0.004}$	$0.030_{\pm 0.018}$	$0.063_{\pm 0.042}$	$0.103_{\pm 0.047}$	$0.042_{\pm 0.025}$
	GraphSAGE	$0.316_{\pm 0.058}$	$0.014_{\pm 0.007}$	$0.008_{\pm 0.007}$	$0.195_{\pm 0.206}$	$0.056_{\pm 0.055}$	$0.051_{\pm 0.015}$
	GAT	$0.343_{\pm 0.064}$	$0.016_{\pm 0.004}$	$0.086_{\pm 0.084}$	$0.172_{\pm 0.098}$	$0.050_{\pm 0.027}$	$0.142_{\pm 0.138}$
	DGI	$0.329_{\pm 0.103}$	$0.026_{\pm 0.009}$	$0.082_{\pm 0.035}$	$0.218_{\pm 0.168}$	$0.224_{\pm 0.127}$	$0.049_{\pm 0.017}$
	GKD	$0.399_{\pm 0.033}$	$0.042_{\pm 0.008}$	$0.202_{\pm 0.064}$	$0.339_{\pm 0.138}$	$0.166_{\pm 0.086}$	$0.208_{\pm 0.077}$
	GLNN	$0.390_{\pm 0.011}$	$0.031_{\pm 0.006}$	$0.187_{\pm 0.012}$	$0.283_{\pm 0.102}$	$0.140_{\pm 0.019}$	$0.317_{\pm 0.048}$
	NodeFormer	$0.308_{\pm 0.093}$	$0.018_{\pm 0.007}$	$0.048_{\pm 0.022}$	$0.168_{\pm 0.127}$	$0.073_{\pm 0.015}$	$0.165_{\pm 0.057}$
	DIFFormer	$0.361_{\pm 0.071}$	$0.029_{\pm 0.014}$	$0.129_{\pm 0.030}$	$0.275_{\pm 0.171}$	$0.311_{\pm 0.025}$	$0.306_{\pm 0.131}$
	OFA	$0.314_{\pm 0.059}$	$0.130_{\pm 0.019}$	$0.064_{\pm 0.086}$	$0.052_{\pm 0.049}$	$0.340_{\pm 0.026}$	$0.101_{\pm 0.071}$
LLM as Predictor	Vicuna-7B-v1.5	$0.719_{\pm 0.010}$	$0.156_{\pm 0.001}$	$0.270_{\pm 0.011}$	$0.363_{\pm 0.001}$	$0.378_{\pm 0.004}$	$0.370_{\pm 0.001}$
	Vicuna-7B-SPT	$0.768_{\pm 0.036}$	$0.168_{\pm 0.018}$	$0.227_{\pm 0.015}$	$0.281_{\pm 0.088}$	$0.350_{\pm 0.061}$	$0.230_{\pm 0.018}$
	GraphGPT-std	0.701	0.126	-	-	-	-
	GraphGPT-cot	0.521	**0.181**	-	-	-	-
	LLaGA	$0.793_{\pm 0.036}$	$0.168_{\pm 0.032}$	$0.199_{\pm 0.007}$	$0.146_{\pm 0.067}$	$0.276_{\pm 0.069}$	$0.352_{\pm 0.033}$
	TEA-GLM	$\underline{0.839}_{\pm 0.013}$	$0.164_{\pm 0.010}$	$\underline{0.271}_{\pm 0.010}$	$\underline{0.528}_{\pm 0.058}$	$\underline{0.497}_{\pm 0.027}$	$\underline{0.404}_{\pm 0.010}$
	PATH **(Ours)**	$\mathbf{0.845}_{\pm \mathbf{0.010}}$	$\underline{0.173}_{\pm 0.008}$	$\mathbf{0.275}_{\pm \mathbf{0.015}}$	$\mathbf{0.594}_{\pm \mathbf{0.050}}$	$\mathbf{0.520}_{\pm \mathbf{0.015}}$	$\mathbf{0.436}_{\pm \mathbf{0.024}}$

Cora benchmark [44], in which nodes denote papers, edges represent citation links, and class labels correspond to research areas or clinical topics. For the e-commerce domain, we use five datasets from the TAG benchmark [48].Products form the nodes, co-viewing or co-purchasing relationships define the edges, and product taxonomies provide the labels. We adopt GraphGPT's split methodology for citation-domain datasets [35] and TAG benchmark scripts for e-commerce datasets [48]. Appendix E summarizes the detailed dataset statistics.

Baselines. We evaluate PATH against a diverse collection of representative methods. Specifically, we include a non-graph MLP baseline; supervised GNNs: GCN [25], GraphSAGE [19], and GAT [38]; the self-supervised method DGI [39]; graph knowledge distillation methods GKD [49] and GLNN [54]; graph transformers NodeFormer [46] and DIFFormer [45]; the general-purpose LLM Vicuna-7B-v1.5 [56]; and recent state-of-the-art graph foundation or graph-LLM models, including OFA [27], GraphGPT [35], LLaGA [6], and TEA-GLM [40]. This comparison set allows us to assess PATH against supervised, self-supervised, distillation-based, transformer-based, and zero-shot transfer-oriented alternatives. Further implementation details appear in Appendix F and G.

4.2 Cross-Dataset Zero-Shot Performance

The model is trained on two source datasets, Arxiv and Computer, representing the citation and e-commerce domains, respectively, and directly evaluate performance on other datasets within the same domain without any target-dataset fine-tuning. Table 1 presents the comparison results and we have three observations. *Firstly*, PATH consistently outperforms competing GNN and LLM predictors across multiple datasets, demonstrating its strong transferability without additional training. *Secondly*, GNN-based methods struggle in zero-shot settings due to their dependence on source graph structures, limiting generalization to domains with different graph properties. Despite its strong performance on citation datasets, OFA struggles to generalize to e-commerce data, where the heterogeneity of product categories poses challenges to effective adaptation. In contrast, LLMs excel by leveraging rich pretrained semantic knowledge, enabling them to handle unseen domains better. *Finally*, PATH outperforms both Vicuna-v1.5 and Vicuna-SPT due to its two innovative strategies: (1) Iterative Mutual Alignment Tuning, which facilitates iterative optimization through mutual alignment between GNNs and LLMs, and (2) Mixture of Prompt Expert, which leverages specialized LLM prompts to enhance the performance of GNNs.

4.3 Cross-Task Zero-Shot Performance

We test the model's performance on the link prediction task across all e-commerce datasets after training on the Computer dataset for node classification. The Area Under Curve (AUC) is used as

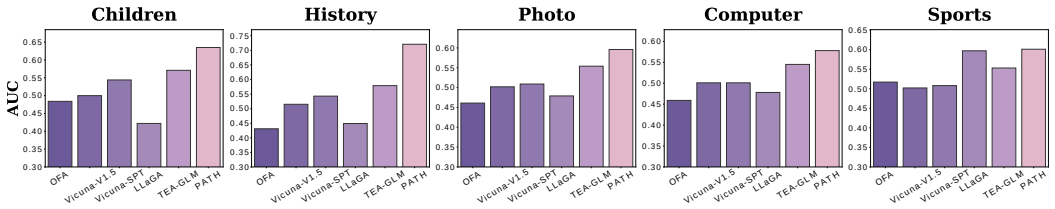

Figure 2: **Cross-task link prediction AUC** across e-commerce datasets.

the evaluation metric. Figure 2 summarizes the evaluation results. Our proposed PATH achieves the highest AUC among the compared methods on all five evaluated e-commerce datasets, demonstrating strong generalization capabilities for cross-task zero-shot learning. The results reveal three key observations: (1) Pure language model variants (Vicuna) and graph-agnostic approaches (OFA) exhibit fundamental limitations, either ignoring graph topology or suffering from negative transfer, highlighting the necessity of unified modality interaction; (2) Existing graph-text alignment methods show limited capacity to model complex cross-modal dependencies, which often fail to capture the intricate interplay between structural and semantic features, particularly in tasks requiring fine-grained reasoning; (3) While baseline methods show inconsistent performance across domains, PATH maintains robust accuracy by dynamically balancing semantic and structural signals. The results validate that explicit modeling of modality interplay, rather than simple alignment or isolated adaptation, drives effective cross-task transfer in graph-language learning.

4.4 Further Analysis

Ablation Study. To validate the contribution of core components in PATH, we design four variants:(1) *PATH w/o* $f^G_{Linear}(\cdot)$: Removes the linear projection layer between GNN and LLM, directly feeding raw GNN outputs to LLM; (2) *PATH w/o* $\mathcal{L}_{info}$: Removes feature-wise graph pre-training loss for prototypical prompt learning; (3) *PATH w/o* $\mathcal{L}_{ins}$: Disables instance loss; (4) *PATH w/o Iteration*: Performs single-step inference without multi-round iterative refinement during mutual prompting. The ablation results presented in Table 2 reveal four key observations: (1) Removing $f^G_{\text{Linear}}(\cdot)$ causes the largest drop on most datasets, highlighting the importance of embedding-space alignment for effective GNN–LLM interaction; this variant performs especially poorly on *History* and *Photo*, suggesting that raw graph features cannot be directly transferred into the language space without a dedicated projection step. (2) Disabling $\mathcal{L}_{info}$ severely degrades semantic-rich tasks (e.g., *History*), indicating its role in capturing fine-grained semantics; at the same time, its weaker effect on *Pubmed* and *Cora* implies that citation datasets rely relatively more on coarse structural regularities. (3) The full model consistently surpasses the single-pass variant, demonstrating the benefit of iterative refinement; the clearest gains on *History*, *Photo*, and *Sports* further indicate that iteration is most useful when structural and textual cues need repeated coordination. (4) Removing $\mathcal{L}_{ins}$ yields unstable performance, highlighting its importance in balancing structural and semantic signals; although this variant remains competitive on a few datasets, its inconsistency across domains shows that instance-level supervision is important for robust transfer.

Table 2: **Ablation study** on all datasets.

Variants	Pubmed	Cora	Children	History	Photo	Sports
PATH w/o $f^G_{\text{Linear}}(\cdot)$	0.711	0.162	0.243	0.373	0.257	0.289
PATH w/o $\mathcal{L}_{info}$	0.807	0.151	0.263	0.497	0.465	0.405
PATH w/o $\mathcal{L}_{ins}$	0.819	0.152	0.268	0.368	0.519	0.382
PATH w/o Iteration	0.835	0.163	0.270	0.538	0.508	0.403
PATH	**0.845**	**0.173**	**0.275**	**0.594**	**0.520**	**0.436**

Parameter Sensitivity. We examine how performance varies with the token count and the number of iterations. We vary the number of tokens from 1 to 5 and the number of iterations from 0 to 3, with the other parameter fixed, respectively. As shown in Figure 3, PATH achieves the best performance with 3 tokens on most datasets, indicating that a moderate number of tokens is crucial for effective mutual alignment tuning. One iteration

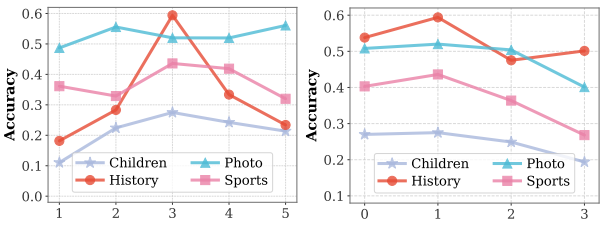

Figure 3: **Sensitivity analysis** to parameters. **Left**: Token count; **Right**: Iteration count.

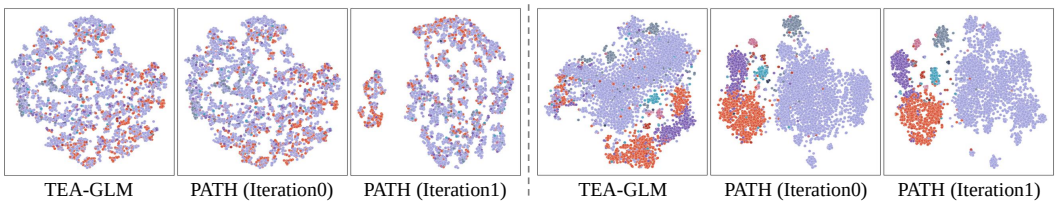

Figure 4: **t-SNE visualization** of the representations on the History dataset. The left panel displays GNN representations, and the right panel displays LLM representations.

achieves the best performance on the evaluated datasets, while further iterations generally reduce performance, potentially because errors accumulate during repeated prompting rounds.

Visualization. To better understand the learned representations, we visualize the node embeddings of GNN and LLM features for the History dataset using t-SNE, as shown in Figure 4. Without classification constraints, the GNN focuses on structural encoding, leading to highly uniform feature distributions. In the initial iteration, the GNN features closely resemble those of TEA-GLM, indicating minimal differentiation. However, after iterative mutual alignment tuning, the GNN features exhibit noticeable clustering patterns, suggesting that mutual prompting effectively integrates text-related semantics into the GNN representations for downstream transfer.

5 Related Work

5.1 Prompt Learning for GNNs

Prompt learning for GNNs has evolved from simple feature augmentation to increasingly sophisticated designs [52]. Early works introduced learnable prompt tokens to node features for pre-training alignment [10, 32], later extended with multiple prompt tokens for greater flexibility [11]. Subsequent approaches diversified the prompt space: view-specific prompts [17], subgraph-based or task-specific prompts [33, 23], and edge-level prompt tuning such as EdgePrompt [13]. Recent efforts also include benchmarking platforms like ProG [58], theoretical analyses of prompting's approximation capabilities [42], and specialized frameworks for heterogeneous graphs [30], dual-task pre-training [5], and self-adaptive components [16]. However, the intrinsic parameter limitations of GNNs often restrict their prompt-learning effectiveness. `PATH` addresses this limitation by leveraging LLM capacity to construct GNN prompts through mutual interaction.

5.2 Graph Alignment with LLMs

The integration of LLMs with graph-structured data has gained significant attention due to the potential to combine LLMs' generalization capabilities with graphs' relational reasoning. A common approach involves converting graph structures into textual representations for LLM input [18, 7, 27], translating nodes, edges, and attributes into natural language descriptions or linearized sequences. However, this often struggles to preserve structural and hierarchical properties. Recent works address this by leveraging GNNs as structural encoders to align graph data with LLMs. Node-level alignment methods, such as GraphGPT [35] and HiGPT [36], project node embeddings from GNNs to LLMs and incorporate heterogeneous graph tokenizers for relation-aware alignment. Graph-level alignment techniques [4] use graph transformers to generate global embeddings for structural reasoning, while several works [29, 3] bridge molecular graphs with text through contrastive and multi-task pretraining. However, these methods often rely on unidirectional alignment, limiting bidirectional feature integration and joint optimization. Our work addresses this issue through a mutual prompting framework that enables bidirectional information exchange between GNNs and LLMs for stronger alignment and generalization across settings.

6 Conclusion

In this paper, we propose `PATH`, a novel framework for zero-shot text-attributed graph learning that synergizes GNNs and LLMs through dual graph pre-training and mutual prompt learning. Dual

pre-training captures both instance-level structural relationships and informativeness-aware semantic alignment with LLM token embeddings, enabling transferable graph representations. The mutual prompt learning framework enables iterative enhancement: GNN-derived geometric tokens guide LLM via prototypical projections, while LLM-generated instructions boost GNN performance via prompt experts. Extensive experiments demonstrate PATH's superior performance in zero-shot transfer across datasets and tasks, attaining leading performance in both node classification and link prediction.

Acknowledgements

Tao Ren is supported by the Fundamental Research Funds for the Central Universities (N25GFZ011). Yifan Wang is supported by the Fundamental Research Funds for the Central Universities in UIBE (Grant No. 23QN02), the Humanities and Social Sciences Research Fund of the Ministry of Education of China under Grant 25YJCZH275 and Open Research Fund of The State Key Laboratory of Multimodal Artificial Intelligence Systems.

Competing Interests. The authors declare no competing interests.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Lars Backstrom and Jure Leskovec. Supervised random walks: predicting and recommending links in social networks. In *Proceedings of the International ACM Conference on Web Search & Data Mining*, pages 635–644, 2011.

[3] He Cao, Zijing Liu, Xingyu Lu, Yuan Yao, and Yu Li. Instructmol: Multi-modal integration for building a versatile and reliable molecular assistant in drug discovery. *arXiv preprint arXiv:2311.16208*, 2023.

[4] Ziwei Chai, Tianjie Zhang, Liang Wu, Kaiqiao Han, Xiaohai Hu, Xuanwen Huang, and Yang Yang. Graphllm: Boosting graph reasoning ability of large language model. *arXiv preprint arXiv:2310.05845*, 2023.

[5] Mouxiang Chen, Zemin Liu, Chenghao Liu, Jundong Li, Qiheng Mao, and Jianling Sun. Ultra-dp: Unifying graph pre-training with multi-task graph dual prompt. *arXiv preprint arXiv:2310.14845*, 2023.

[6] Runjin Chen, Tong Zhao, Ajay Jaiswal, Neil Shah, and Zhangyang Wang. Llaga: Large language and graph assistant. *arXiv preprint arXiv:2402.08170*, 2024.

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

[8] Zhikai Chen, Haitao Mao, Hongzhi Wen, Haoyu Han, Wei Jin, Haiyang Zhang, Hui Liu, and Jiliang Tang. Label-free node classification on graphs with large language models (llms). In *Proceedings of the International Conference on Learning Representations*, 2024.

[9] Zhekai Du, Xinyao Li, Fengling Li, Ke Lu, Lei Zhu, and Jingjing Li. Domain-agnostic mutual prompting for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23375–23384, 2024.

[10] Taoran Fang, Yunchao Zhang, Yang Yang, and Chunping Wang. Prompt tuning for graph neural networks. *CoRR*, Sep 2022.

[11] Taoran Fang, Yunchao Zhang, Yang Yang, Chunping Wang, and Lei Chen. Universal prompt tuning for graph neural networks. *Advances in Neural Information Processing Systems*, 36, 2024.

[12] Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. Talk like a graph: Encoding graphs for large language models. *arXiv preprint arXiv:2310.04560*, 2023.

[13] Xingbo Fu, Yinhan He, and Jundong Li. Edge prompt tuning for graph neural networks. *arXiv preprint arXiv:2503.00750*, 2025.

[14] Yiwei Fu, Chunchun Chen, Yuxing Zhang, Weizhi Zhang, Guanjie Zheng, Wei Ye, Xiao Luo, Philip S Yu, and Minghua Deng. Mask: Minority-aware semantic alignment via knowledge enhancement for long-tailed graph clustering. In *Proceedings of the 35th ACM International Conference on Information and Knowledge Management*, 2026.

[15] Yiwei Fu, Yuxing Zhang, Chunchun Chen, Jianwen Ma, Quan Yuan, Rong-Cheng Tu, Xinli Huang, Wei Ye, Xiao Luo, and Minghua Deng. Mark: Multi-agent collaboration with ranking guidance for text-attributed graph clustering. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 6057–6072, 2025.

[16] Chenghua Gong, Xiang Li, Jianxiang Yu, Yao Cheng, Jiaqi Tan, and Chengcheng Yu. Self-pro: A self-prompt and tuning framework for graph neural networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 197–215. Springer, 2024.

[17] Chenghua Gong, Xiang Li, Jianxiang Yu, Cheng Yao, Jiaqi Tan, Chengcheng Yu, and Dawei Yin. Prompt tuning for multi-view graph contrastive learning. *arXiv preprint arXiv:2310.10362*, 2023.

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

[19] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 1024–1034, 2017.

[20] Xiaoxin He, Xavier Bresson, Thomas Laurent, Adam Perold, Yann LeCun, and Bryan Hooi. Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning. *arXiv preprint arXiv:2305.19523*, 2023.

[21] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

[22] Jin Huang, Xingjian Zhang, Qiaozhu Mei, and Jiaqi Ma. Can llms effectively leverage graph structural information: when and why. *arXiv preprint arXiv:2309.16595*, 2023.

[23] Qian Huang, Hongyu Ren, Peng Chen, Gregor Kržmanc, Daniel Zeng, Percy S Liang, and Jure Leskovec. Prodigy: Enabling in-context learning over graphs. *Advances in Neural Information Processing Systems*, 36, 2024.

[24] Mingxuan Ju, Tong Zhao, Qianlong Wen, Wenhao Yu, Neil Shah, Yanfang Ye, and Chuxu Zhang. Multi-task self-supervised graph neural networks enable stronger task generalization. 2023.

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

[26] Yuhan Li, Peisong Wang, Zhixun Li, Jeffrey Xu Yu, and Jia Li. Zerog: Investigating cross-dataset zero-shot transferability in graphs. In *Proceedings of the International ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1725–1735, 2024.

[27] Hao Liu, Jiarui Feng, Lecheng Kong, Ningyue Liang, Dacheng Tao, Yixin Chen, and Muhan Zhang. One for all: Towards training one graph model for all classification tasks. In *Proceedings of the International Conference on Learning Representations*, 2024.

[28] Yunhui Liu, Xudong Jin, Qizhuo Xie, Chunhui Zhao, Xiao Luo, Tao Zheng, Bin Chong, and Tieke He. Uncovering semantic hierarchies in text-attributed graphs via variational em-based llm-gnn synergy. *arXiv preprint arXiv:2610.24624*, 2026.

[29] Zhiyuan Liu, Sihang Li, Yanchen Luo, Hao Fei, Yixin Cao, Kenji Kawaguchi, Xiang Wang, and Tat-Seng Chua. Molca: Molecular graph-language modeling with cross-modal projector and uni-modal adapter. *arXiv preprint arXiv:2310.12798*, 2023.

[30] Yihong Ma, Ning Yan, Jiayu Li, Masood Mortazavi, and Nitesh V Chawla. Hetgpt: Harnessing the power of prompt tuning in pre-trained heterogeneous graph neural networks. In *Proceedings of the ACM Web Conference 2024*, pages 1015–1023, 2024.

[31] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning*, pages 8748–8763, 2021.

[32] Reza Shirkavand and Heng Huang. Deep prompt tuning for graph transformers. *arXiv preprint arXiv:2309.10131*, 2023.

[33] Xiangguo Sun, Hong Cheng, Jia Li, Bo Liu, and Jihong Guan. All in one: Multi-task prompting for graph neural networks. In *Proceedings of the International ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2120–2131, 2023.

[34] Xiangguo Sun, Jiawen Zhang, Xixi Wu, Hong Cheng, Yun Xiong, and Jia Li. Graph prompt learning: A comprehensive survey and beyond. *arXiv:2311.16534*, 2023.

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

[36] Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Long Xia, Dawei Yin, and Chao Huang. Higpt: Heterogeneous graph language model. In *Proceedings of the International ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2842–2853, 2024.

[37] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[38] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *Proceedings of the International Conference on Learning Representations*, 2018.

[39] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *arXiv preprint arXiv:1809.10341*, 2018.

[40] Duo Wang, Yuan Zuo, Fengzhi Li, and Junjie Wu. Llms as zero-shot graph learners: Alignment of gnn representations with llm token embeddings. *arXiv preprint arXiv:2408.14512*, 2024.

[41] Haotao Wang, Ziyu Jiang, Yuning You, Yan Han, Gaowen Liu, Jayanth Srinivasa, Ramana Kompella, Zhangyang Wang, et al. Graph mixture of experts: Learning on large-scale graphs with explicit diversity modeling. *Advances in Neural Information Processing Systems*, 36:50825–50837, 2023.

[42] Qunzhong Wang, Xiangguo Sun, and Hong Cheng. Does graph prompt work? a data operation perspective with theoretical analysis. *arXiv preprint arXiv:2410.01635*, 2024.

[43] Yifan Wang, Suyao Tang, Yuntong Lei, Weiping Song, Sheng Wang, and Ming Zhang. Disenhan: Disentangled heterogeneous graph attention network for recommendation. In *Proceedings of the International Conference on Information and Knowledge Management*, pages 1605–1614, 2020.

[44] Zhihao Wen and Yuan Fang. Augmenting low-resource text classification with graph-grounded pre-training and prompting. In *Proceedings of the International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 506–516, 2023.

[45] Qitian Wu, Chenxiao Yang, Wentao Zhao, Yixuan He, David Wipf, and Junchi Yan. Difformer: Scalable (graph) transformers induced by energy constrained diffusion. *arXiv preprint arXiv:2301.09474*, 2023.

[46] Qitian Wu, Wentao Zhao, Zenan Li, David P Wipf, and Junchi Yan. Nodeformer: A scalable graph structure learning transformer for node classification. *Advances in Neural Information Processing Systems*, 35:27387–27401, 2022.

[47] Xiongxiao Xu, Canyu Chen, Yueqing Liang, Baixiang Huang, Guangji Bai, Liang Zhao, and Kai Shu. Sst: Multi-scale hybrid mamba-transformer experts for time series forecasting. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 3655–3665, 2025.

[48] Hao Yan, Chaozhuo Li, Ruosong Long, Chao Yan, Jianan Zhao, Wenwen Zhuang, Jun Yin, Peiyan Zhang, Weihao Han, Hao Sun, et al. A comprehensive study on text-attributed graphs: Benchmarking and rethinking. *Advances in Neural Information Processing Systems*, 36:17238–17264, 2023.

[49] Chenxiao Yang, Qitian Wu, and Junchi Yan. Geometric knowledge distillation: Topology compression for graph neural networks. *Advances in Neural Information Processing Systems*, 35:29761–29775, 2022.

[50] Jiexia Ye, Weiqi Zhang, Ziyue Li, Jia Li, Meng Zhao, and Fugee Tsung. Medualtime: A dual-adapter language model for medical time series-text multimodal learning. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 7913–7921, 2024.

[51] Jianxiang Yu, Yuxiang Ren, Chenghua Gong, Jiaqi Tan, Xiang Li, and Xuecang Zhang. Empower text-attributed graphs learning with large language models (llms). *arXiv preprint arXiv:2310.09872*, 2023.

[52] Jiayi Zeng, Tao Ren, Changhu Wang, Yifan Wang, Wei Ju, Zhipeng Sun, and Xiao Luo. Date: Dual prompt learning with information bottleneck for graph out-of-distribution generalization. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 8920–8929, 2025.

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

[54] Shichang Zhang, Yozen Liu, Yizhou Sun, and Neil Shah. Graph-less neural networks: Teaching old mlps new tricks via distillation. *arXiv preprint arXiv:2110.08727*, 2021.

[55] Yusheng Zhao, Qixin Zhang, Xiao Luo, Weizhi Zhang, Zhiping Xiao, Wei Ju, Philip S Yu, and Ming Zhang. Dynamic bundling with large language models for zero-shot inference on text-attributed graphs. *Advances in Neural Information Processing Systems*, 38:155296–155326, 2026.

[56] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.

[57] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131*, 2020.

[58] Chenyi Zi, Haihong Zhao, Xiangguo Sun, Yiqing Lin, Hong Cheng, and Jia Li. Prog: A graph prompt learning benchmark. *Proceedings of the Conference on Neural Information Processing Systems*, 37:95406–95437, 2024.

A Limitation

PATH addresses zero-shot learning on text-attributed graphs by bringing together GNNs and LLMs through prototypical mutual prompting. This implies that the overall performance could fluctuate depending on the specific pre-trained knowledge and reasoning capacity of the underlying LLM, which is a common challenge in LLM-based applications. However, to mitigate this concern and ensure the robustness of our conclusions, we have empirically validated PATH by substituting various backbones. Our results consistently demonstrate that the effectiveness of the mutual prompting enhancement remains significant across different architectures. Future work will explore more diverse backbone models to further establish the broad applicability of the framework across graph-learning settings.

B Broader Impact

By combining GNNs with LLMs, PATH supports zero-shot learning on text-attributed graphs. By leveraging dual graph pre-training and mutual prompting, PATH enhances the extraction of both structural and semantic information, enabling effective generalization across unseen graphs and tasks. This work has potential applications in domains requiring robust graph analysis, such as social networks, recommender systems, knowledge graphs, and biomedical networks. By lowering the requirements of task-specific fine-tuning, PATH enables more efficient and scalable graph-based machine learning.

C Large Language Model (LLM) Usage Statement

Our method uses a frozen LLM backbone as part of the core model. In the main experiments, this backbone is Vicuna-7B-v1.5, and we also study a smaller LLaMA-3.2-3B variant in the appendix. The LLM is used to encode text semantics, interact with graph prompts, and produce prompt-aware representations inside the mutual prompting framework described in Sec. 3. Therefore, the LLM directly affects the model design and the reported results. Any use of LLM tools for writing support is limited to minor language editing rather than research design or experimental analysis.

D Proof of Theorem 3.1

Using the mean squared error (MSE) loss, the estimated parameter $\hat{\boldsymbol{W}}$ is obtained by minimizing:

$$\mathcal{L} = \|\boldsymbol{Y}_p^M - \boldsymbol{X}_p\boldsymbol{W}\|^2,$$

which leads to the optimal solution:

$$\hat{\boldsymbol{W}} = (\boldsymbol{X}_p^\top \boldsymbol{X}_p)^{-1}\boldsymbol{X}_p^\top \boldsymbol{Y}_p^M.$$

Substituting this into the expression for the predicted promoted node feature matrix $\boldsymbol{Y}_p$, we obtain:

$$\begin{aligned}
\boldsymbol{Y}_p &= \boldsymbol{X}_p\hat{\boldsymbol{W}} \\
&= \boldsymbol{X}_p(\boldsymbol{X}_p^\top \boldsymbol{X}_p)^{-1}\boldsymbol{X}_p^\top \boldsymbol{Y}_p^M \\
&= \boldsymbol{X}_p(\boldsymbol{X}_p^\top \boldsymbol{X}_p)^{-1}\boldsymbol{X}_p^\top (\boldsymbol{X}_p\boldsymbol{W}_1 + \boldsymbol{P}_{-1}\boldsymbol{W}_2) \\
&= \boldsymbol{X}_p\boldsymbol{W}_1 + \boldsymbol{X}_p(\boldsymbol{X}_p^\top \boldsymbol{X}_p)^{-1}\boldsymbol{X}_p^\top \boldsymbol{P}_{-1}\boldsymbol{W}_2. \qquad (16)
\end{aligned}$$

From Equation (16), the difference between the predicted and true promoted node feature matrices is:

$$\begin{aligned}
\boldsymbol{Y}_p - \boldsymbol{Y}_p^M &= \boldsymbol{X}_p\boldsymbol{W}_1 \\
&\quad + \boldsymbol{X}_p(\boldsymbol{X}_p^\top \boldsymbol{X}_p)^{-1}\boldsymbol{X}_p^\top \boldsymbol{P}_{-1}\boldsymbol{W}_2 \\
&\quad - (\boldsymbol{X}_p\boldsymbol{W}_1 + \boldsymbol{P}_{-1}\boldsymbol{W}_2) \\
&= \boldsymbol{X}_p(\boldsymbol{X}_p^\top \boldsymbol{X}_p)^{-1}\boldsymbol{X}_p^\top \boldsymbol{P}_{-1}\boldsymbol{W}_2 \\
&\quad - \boldsymbol{P}_{-1}\boldsymbol{W}_2 \\
&= (\boldsymbol{X}_p(\boldsymbol{X}_p^\top \boldsymbol{X}_p)^{-1}\boldsymbol{X}_p^\top - \boldsymbol{I}) \\
&\quad \times \boldsymbol{P}_{-1}\boldsymbol{W}_2. \qquad (17)
\end{aligned}$$

Thus, we conclude the proof of Theorem 3.1.

Table 3: Summary of dataset characteristics.

Domain	Dataset	#Nodes	#Edges	#Classes
Citation	Arxiv	169,343	1,166,243	40
	Pubmed	19,717	44,338	3
	Cora	25,120	91,140	70
E-commerce	Ele-Computer	87,229	721,081	10
	Ele-Photo	48,362	500,928	12
	Book-Children	76,875	1,554,578	24
	Book-History	41,551	358,574	12
	Sports-Fitness	173,055	1,773,500	13

E Detail of datasets

Table 3 summarizes the key statistics of our evaluation datasets. All datasets are utilized within the scope permitted by their licenses. We provide detailed descriptions:

Citation Networks focus on academic paper analysis. The **Arxiv** [21] dataset contains 169,343 computer science papers from arXiv, where nodes represent publications connected by citations, and labels correspond to 40 subfields. **Pubmed** [20] includes 19,717 diabetes-related papers categorized into three clinical types (Type 1/2 Diabetes and Experimentally Induced Diabetes), with edges reflecting citation relationships. The extended **Cora** [44] dataset expands the classic version to 25,120 machine learning papers and 70 fine-grained research topics, covering a broader research taxonomy for machine learning.

E-commerce Datasets from the TAG benchmark model product relationships [48]. **Book-Children** (76,875 nodes) and **Book-History** (41,551 nodes) represent Amazon book subcategories with three-level hierarchical labels. **Ele-Computer** (87,229 nodes) and **Ele-Photo** (48,362 nodes) cover electronics products with functional categorizations. **Sports-Fitness** (173,055 nodes) is the largest dataset, where edges encode co-purchasing patterns between fitness-related items. All e-commerce edges are derived from co-viewing or co-buying behaviors, and their labels reflect product taxonomies across categories in the benchmark.

F Detail of Baselines

We compare `PATH` with 14 baseline methods across six technical categories:

Traditional Non-GNN: Multi-Layer Perceptron baseline without graph structural awareness, serving as a fundamental reference for non-relational learning.

Supervised GNNs:

- GCN [25]: Spectral graph convolution operator with layer-wise neighborhood aggregation through low-pass frequency filtering.
- GraphSAGE [19]: Inductive framework employing stochastic neighborhood sampling and parameterized aggregation functions.
- GAT [38]: Attention-based architecture with learnable edge importance weights via multi-head attention mechanisms.

Self-supervised Learning:

- DGI [39]: Learns node representations by maximizing their mutual information with a graph-level representation.

Knowledge Distillation:

- GKD [49]: Graph-to-graph distillation framework transferring topological knowledge via adaptive structure matching.

- GLNN [54]: Structure-agnostic neural network trained with GNN-generated soft labels for graph-less inference.

Graph Transformers:

- NodeFormer [46]: Kernelized transformer architecture enabling efficient all-pair message passing with random feature approximation.
- DIFFormer [45]: Spectral diffusion-enhanced transformer with adaptive propagation based on eigenbasis decomposition.

Large Language Models:

- Vicuna-7B-v1.5 [56]: Instruction-following LLM with 7 billion parameters fine-tuned from LLama2.

State-of-the-Art Models:

- OFA [27]: Unified graph foundation model with cross-domain text-graph unification and in-context learning via prompt substructures.
- GraphGPT [35]: Graph-text alignment framework with dual-stage instruction tuning and structural-aware projection modules.
- LLaGA [6]: Language-graph assistant with topology-preserving sequence reorganization and parameter-efficient graph token projection.
- TEA-GLM [40]: GNN-LLM alignment method featuring pretrained representation mapping and unified instruction templates for cross-task generalization.

For GNN-based methods, we follow the evaluation protocol used by TEA-GLM [40]. For GraphGPT [35], we report the results of citation datasets provided by the paper.

Algorithm 1: Prototypical Prompt Learning with Mutual Alignment

Inputs: f_{GNN}, frozen LLM, prototypes K, iterations I;
// Initial GNN processing
$\boldsymbol{Z} = f_{\text{GNN}}(\boldsymbol{A}, \boldsymbol{X})$; // Get initial node embeddings
// Initial LLM processing
For node v with $\boldsymbol{z}_v$, compute $\boldsymbol{h}_v^k = f_{\text{Linear}}^{G,k}(\boldsymbol{z}_v), k = 1..K$; // Eq. 10
Form $\boldsymbol{H}_v = \{\boldsymbol{h}_v^k\}_{k=1}^K$ and use as soft tokens in LLM instruction;
for $i = 1, \cdots, I$ **do**
  // LLM to GNN: Language prompting for GNN
  From LLM outputs $\{\boldsymbol{h}_k\}_{k=1}^K$, set $\boldsymbol{p}^k = f_{\text{Linear}}^{L_1,k}(\boldsymbol{h}_k)$; // Eq. 11
  $w_k(\boldsymbol{z}_v) = [\text{Softmax}(f_{\text{Linear}}^R(\boldsymbol{z}_v) \circ (1+\delta))]_k$; // Eq. 12
  $\boldsymbol{x}'_v = [\boldsymbol{x}_v; \boldsymbol{w}^{\text{T}}\boldsymbol{P}]$, $\boldsymbol{x}_{v,p} = f_{\text{Linear}}^{L_2}(\boldsymbol{x}'_v)$; // Eq. 13
  Freeze GNN and LLM; train $\{f_{\text{Linear}}^{L_1,k}\}$, $f_{\text{Linear}}^{L_2}$, f_{Linear}^R with $\mathcal{L}$ and $\mathcal{L}_{imp}$; // Eq. 9, 15
  // GNN to LLM: Geometric prompting for LLM
  Update $\boldsymbol{Z} = f_{\text{GNN}}(\boldsymbol{A}, \boldsymbol{X}_p)$ with prompted features;
  For node v with updated $\boldsymbol{z}_v$, compute $\boldsymbol{h}_v^k = f_{\text{Linear}}^{G,k}(\boldsymbol{z}_v), k = 1..K$; // Eq. 10
  Form $\boldsymbol{H}_v = \{\boldsymbol{h}_v^k\}_{k=1}^K$ and add $\mathcal{L}_{ind}$; // Eq. 14
  Use $\boldsymbol{H}_v$ as soft tokens in LLM instruction; update $\{f_{\text{Linear}}^{G,k}\}$ on task loss $+\mathcal{L}_{ind}$;
// Final prediction with LLM
Generate final predictions using LLM with aligned prompts;
Return: projectors $\{f_{\text{Linear}}^{G,k}\}$, $\{f_{\text{Linear}}^{L_1,k}\}$, $f_{\text{Linear}}^{L_2}$, router f_{Linear}^R;

G Implementation Details

The framework operates in two phases. During **Dual Graph Pre-training**, we initialize a 2-layer GraphSAGE backbone with mean aggregation and ReLU activation, setting the hidden dimension to

4,096 to align with Vicuna-7B's token embeddings. This phase uses AdamW optimization with a batch size of 512 for 60 epochs, learning structural patterns from raw graph data.

In the **Prototypical Prompt Learning** phase, we reduce the batch size to 2 to accommodate memory constraints when integrating the LLM. The alignment process employs three trainable tokens to bridge GNN and LLM representations, optimized with a learning rate of 1×10^{-3} for one full iteration over the dataset. Node classification is evaluated using accuracy and macro-F1, whereas link prediction is assessed with AUC-ROC. Unless otherwise described, experiments are performed with five random seeds. GraphGPT [35] results are quoted from its original paper.

To clarify the compute requirement, our main experiments were executed on four NVIDIA A100 GPUs (80GB memory each) for throughput, while a complete training run of PATH is feasible on a single NVIDIA A100 GPU with 80GB memory. This is possible because the LLM backbone remains frozen and the trainable parameters are limited to lightweight linear projectors and a prompt router.

For loss computation, the total loss in PATH combines step-wise components for each direction of the iterative mutual prompting process. In the forward step (GNN $\rightarrow$ LLM), we train the linear projectors using the downstream task loss and orthogonality regularization, as described in Algorithm 1. In the backward step (LLM $\rightarrow$ GNN), we optimize

$$\mathcal{L}_{\text{total}} = \mathcal{L} + \lambda \mathcal{L}_{imp}, \tag{18}$$

where $\mathcal{L} = \frac{1}{2}(\mathcal{L}_{ins} + \mathcal{L}_{info})$ is the contrastive loss, and $\mathcal{L}_{imp}$ is the importance regularization for the prompt experts. We set $\lambda = 0.2$ to control its weight. The orthogonality regularization $\mathcal{L}_{ind}$ is applied in the forward step.

Table 4: Legality rates of models with LLM backbones.

Model	Seen		Unseen					
	Arxiv	**Computer**	**Pubmed**	**Cora**	**Children**	**History**	**Photo**	**Sports**
Vicuna-7B-v1.5	99.3	96.7	100.0	95.8	99.2	98.9	94.1	99.6
LLaGA	100.0	100.0	98.9	79.9	93.1	92.4	77.8	94.3
TEA-GLM	100.0	100.0	100.0	92.6	97.0	99.6	99.2	98.5
PATH	100.0	100.0	100.0	75.7	98.2	99.6	99.5	99.1

Table 5: Macro-F1 results for node classification. Bold and underlined entries denote the best and second-best results, respectively.

	Model	Pubmed	Cora	Children	History	Photo	Sports
	MLP	$0.246_{\pm 0.042}$	$0.009_{\pm 0.004}$	$0.007_{\pm 0.007}$	$0.023_{\pm 0.008}$	$0.041_{\pm 0.023}$	$0.019_{\pm 0.005}$
GNN as Predictor	GCN	$0.187_{\pm 0.021}$	$0.007_{\pm 0.001}$	$0.006_{\pm 0.004}$	$0.024_{\pm 0.013}$	$0.034_{\pm 0.007}$	$0.017_{\pm 0.009}$
	GraphSAGE	$0.257_{\pm 0.084}$	$0.007_{\pm 0.003}$	$0.005_{\pm 0.003}$	$0.029_{\pm 0.024}$	$0.020_{\pm 0.011}$	$0.021_{\pm 0.004}$
	GAT	$0.259_{\pm 0.065}$	$0.006_{\pm 0.001}$	$0.063_{\pm 0.067}$	$0.159_{\pm 0.117}$	$0.036_{\pm 0.035}$	$0.091_{\pm 0.090}$
	DGI	$0.213_{\pm 0.127}$	$0.004_{\pm 0.002}$	$0.012_{\pm 0.004}$	$0.038_{\pm 0.015}$	$0.045_{\pm 0.015}$	$0.018_{\pm 0.005}$
	GKD	$0.247_{\pm 0.039}$	$0.004_{\pm 0.001}$	$0.028_{\pm 0.003}$	$0.060_{\pm 0.008}$	$0.049_{\pm 0.015}$	$0.050_{\pm 0.008}$
	GLNN	$0.221_{\pm 0.033}$	$0.006_{\pm 0.001}$	$0.021_{\pm 0.003}$	$0.064_{\pm 0.007}$	$0.057_{\pm 0.002}$	$0.052_{\pm 0.003}$
	NodeFormer	$0.232_{\pm 0.089}$	$0.008_{\pm 0.003}$	$0.019_{\pm 0.008}$	$0.046_{\pm 0.031}$	$0.055_{\pm 0.006}$	$0.049_{\pm 0.009}$
	DIFFormer	$0.187_{\pm 0.007}$	$0.007_{\pm 0.002}$	$0.002_{\pm 0.002}$	$0.050_{\pm 0.019}$	$0.069_{\pm 0.010}$	$0.045_{\pm 0.007}$
	OFA	$0.287_{\pm 0.059}$	$0.091_{\pm 0.013}$	$0.017_{\pm 0.010}$	$0.026_{\pm 0.007}$	$0.103_{\pm 0.007}$	$0.043_{\pm 0.021}$
LLM as Predictor	Vicuna-7B-v1.5	$0.629_{\pm 0.024}$	$0.109_{\pm 0.002}$	$\mathbf{0.279_{\pm 0.002}}$	$0.349_{\pm 0.003}$	$0.383_{\pm 0.001}$	$0.410_{\pm 0.002}$
	GraphGPT-std	0.649	0.082	-	-	-	-
	GraphGPT-cot	0.482	0.127	-	-	-	-
	LLaGA	$0.778_{\pm 0.056}$	$0.108_{\pm 0.014}$	$0.163_{\pm 0.029}$	$0.144_{\pm 0.025}$	$0.362_{\pm 0.039}$	$\underline{0.446_{\pm 0.035}}$
	TEA-GLM	$\underline{0.839_{\pm 0.012}}$	$\underline{0.148_{\pm 0.015}}$	$0.252_{\pm 0.005}$	$0.365_{\pm 0.011}$	$\mathbf{0.421_{\pm 0.032}}$	$0.430_{\pm 0.009}$
	PATH **(Ours)**	$\mathbf{0.841_{\pm 0.010}}$	$\mathbf{0.165_{\pm 0.017}}$	$\underline{0.264_{\pm 0.004}}$	$\mathbf{0.374_{\pm 0.012}}$	$\underline{0.417_{\pm 0.008}}$	$\mathbf{0.457_{\pm 0.013}}$

H Complexity Analysis

We break down PATH into its main computational components to analyze their respective costs.

Table 6: Node classification results on the training datasets, measured by accuracy and macro-F1. Bold and underlining identify the best and second-best results, respectively.

	Model	Arxiv		Computer	
		Acc	F1	Acc	F1
	MLP	$0.546_{\pm 0.004}$	$0.295_{\pm 0.007}$	$0.420_{\pm 0.006}$	$0.267_{\pm 0.005}$
GNN as Predictor	GCN	$0.545_{\pm 0.005}$	$0.317_{\pm 0.006}$	$0.424_{\pm 0.012}$	$0.386_{\pm 0.014}$
	GraphSAGE	$0.556_{\pm 0.006}$	$0.315_{\pm 0.008}$	$0.534_{\pm 0.037}$	$0.347_{\pm 0.036}$
	GAT	$0.561_{\pm 0.003}$	$0.339_{\pm 0.005}$	$0.609_{\pm 0.035}$	$\underline{0.598_{\pm 0.039}}$
	DGI	$0.342_{\pm 0.024}$	$0.336_{\pm 0.011}$	$0.594_{\pm 0.004}$	$0.452_{\pm 0.008}$
	GKD	$0.393_{\pm 0.085}$	$0.164_{\pm 0.029}$	$0.351_{\pm 0.031}$	$0.155_{\pm 0.016}$
	GLNN	$0.602_{\pm 0.004}$	$0.362_{\pm 0.008}$	$0.393_{\pm 0.005}$	$0.243_{\pm 0.007}$
	NodeFormer	$0.544_{\pm 0.016}$	$0.297_{\pm 0.029}$	$0.434_{\pm 0.012}$	$0.288_{\pm 0.012}$
	DIFFormer	$0.616_{\pm 0.025}$	$0.356_{\pm 0.024}$	$0.629_{\pm 0.012}$	$0.467_{\pm 0.022}$
	OFA	$\underline{0.682_{\pm 0.006}}$	$\underline{0.495_{\pm 0.006}}$	$\mathbf{0.753_{\pm 0.004}}$	$\mathbf{0.687_{\pm 0.006}}$
LLM as Predictor	Vicuna-7B-v1.5	$0.347_{\pm 0.000}$	$0.164_{\pm 0.001}$	$0.372_{\pm 0.010}$	$0.304_{\pm 0.002}$
	GraphGPT-std	0.626	0.262	-	-
	GraphGPT-cot	0.576	0.228	-	-
	LLaGA	$\mathbf{0.749_{\pm 0.001}}$	$\mathbf{0.575_{\pm 0.003}}$	$\underline{0.642_{\pm 0.004}}$	$0.562_{\pm 0.001}$
	TEA-GLM	$0.655_{\pm 0.001}$	$0.445_{\pm 0.002}$	$0.578_{\pm 0.002}$	$0.496_{\pm 0.010}$
	PATH **(Ours)**	$0.631_{\pm 0.008}$	$0.412_{\pm 0.007}$	$0.580_{\pm 0.002}$	$0.510_{\pm 0.008}$

PCA Pre-computation. Before training, we perform a single offline principal component analysis (PCA) of the LLM token embeddings [40]. This one-time pre-computation cost is negligible during subsequent model training and inference.

Graph Model Pre-training. For each forward pass of the GNN encoder, the time complexity is $O(|E|d)$, in which $|E|$ represents the number of edges in the graph and d is the hidden dimension. This phase is executed once to obtain transferable node representations.

Mutual Prompting. During the mutual prompting stage, the GNN outputs are projected through linear layers with a complexity of $O(d^2)$. On the LLM side, linear projections and prompt routing with K experts introduce an additional complexity of $O(Kd^2)$. For I rounds of mutual prompting, the alignment stage has the following total time complexity:

$$O\big((I+1)(|E|d+d^2)+IKd^2\big).$$

Overall, the cost of PCA pre-computation and graph pre-training is incurred once, while the mutual prompting cost scales with the number of iterations I and the number of prompt experts K. This analysis shows that PATH remains practical for large-scale text-attributed graphs, with linear complexity in the number of edges and quadratic complexity in the hidden dimension d.

H.1 Runtime and Memory Overhead

To further quantify the practical overhead of iterative mutual prompting, we compare PATH against the simpler TEA-GLM baseline in terms of wall-clock training time per step and representative zero-shot node classification accuracy. Because the LLM backbone is frozen throughout training and PATH only updates lightweight linear projectors together with a prompt router, the additional overhead mainly comes from repeated forward passes under multiple prompting iterations rather than from full LLM fine-tuning.

Table 7 shows that one iteration of PATH increases the per-step time from 1.243 seconds to 2.028 seconds, while improving zero-shot accuracy on both the Photo and History datasets. Increasing the number of iterations beyond one leads to higher latency without consistent gains, which supports the one-iteration design used in our main experiments. These results indicate that PATH offers a favorable cost-performance trade-off at one iteration: the runtime remains comparable to prior LLM-graph alignment baselines, while the zero-shot transfer performance is stronger.

Table 7: Runtime and cost-performance trade-off under different numbers of mutual prompting iterations.

Method	Iterations	Time / Step (s)	Photo (Acc)	History (Acc)
TEA-GLM	–	1.243	0.497	0.528
PATH	1	2.028	0.520	0.594
PATH	2	2.975	0.505	0.476
PATH	3	3.868	0.402	0.502

Table 8: Zero-shot node classification performance of PATH across different language model sizes.

Model	Children	History	Photo	Sports
LLaMA-3B	0.267	0.350	0.448	0.328
Vicuna-7B-v1.5	0.270	0.363	0.378	0.370
TEA-GLM	0.271	0.528	0.497	0.404
PATH (LLaMA-3B)	0.274	0.318	0.501	0.396
PATH **(Vicuna-7B)**	**0.275**	**0.594**	**0.520**	**0.436**

I More Experimental Results

I.1 Legitimacy experiment

To assess the model's capability of generating valid responses under open-ended scenarios, we conduct legality evaluation following the methodology in [53]. This experiment measures the model's ability to produce answers strictly conforming to predefined formats and semantic constraints, particularly when handling unseen domains. The legality rate is calculated as the proportion of responses that satisfy content constraints (e.g., valid label candidates).

As shown in Table 4, our preliminary results on seen datasets (Arxiv and Computer domains) demonstrate that PATH achieves perfect legality rates (100%), indicating strong alignment with format specifications through our instruction tuning strategy. On unseen datasets, our model achieves high legality rates except on Cora, where its rate is 75.7%, lower than those of the reported baselines. These results indicate that response validity remains challenging on some unseen datasets.

I.2 F1 score on node classification task

Table 5 reports the node classification results in terms of F1 score..

I.3 Supervised results

The accuracy and macro-F1 results for the training datasets are reported in Table 6. Due to the lack of supervised loss during the GNN pre-training phase, PATH does not achieve the best results on seen domains. However, it still outperforms most baseline methods, demonstrating the effectiveness of our approach.

I.4 Scalability of PATH to Different LLM Sizes

The main experiments use Vicuna-7B [56] as the language backbone. However, PATH does not depend on a particular language model and can also be used with smaller backbones. To measure the impact of model size, we replace Vicuna-7B with LLaMA-3.2-3B[2] and perform zero-shot node classification on four benchmark datasets. The results are reported in Table 8.

As shown in Table 8, although the base performance of the smaller LLaMA-3B model is lower than that of Vicuna-7B on several datasets, applying PATH yields performance gains on most datasets, though the improvement is not uniform across all settings. These results suggest that PATH can also improve zero-shot performance with a smaller LLM backbone.

[2] https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction match what we actually study and what we report in Secs. 3.5 and 4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the main limitations in Sec. A and also comment on efficiency in Appendix Sec. H.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The assumptions are stated in Sec. 3.5, and the full proof is given in Appendix Sec. D.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We describe the model, objectives, datasets, baselines, and training/evaluation pipeline in Secs. 3, 3.5, 4, and the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide an anonymized code link in Sec. 1.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: The appendix includes the main training details, hyperparameters, splits, metrics, and benchmark setup.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We clarify in Appendix Sec. G that results reported with standard deviation over 5 runs with different random seeds.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the hardware setup in Appendix Sec. G and runtime information in Appendix Secs. H and H.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We use public benchmarks and cited pretrained models, preserve anonymity, and do not involve human subjects.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We add a short broader impact discussion in Appendix Sec. B, including possible benefits and possible risks.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We do not release a high-risk generative model or scraped dataset.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the datasets and pretrained models used in this work.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide an anonymized code release for the implementation.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing or human participants.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This work does not involve human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs are part of the core method, and we describe their role in Sec. 3 and Appendix Sec. C.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.