Graph as New Language: LLM-Based Graph Learning with Node-to-Center Path Sequences as Training Corpus

Anonymous ACL submission

Abstract

Graph learning is widely encountered in the real-world applications. Existing approaches typically combine graph neural networks with NLP methods, recently with large language models (LLMs), to encode node texts. However, this two-stage paradigm suffers from a suboptimal alignment between textual and structural features. Since LLMs are probabilistic models excelling at next-word prediction, not inherently designed for graphs, we propose a new perspective that treats graphs as a new language, enabling language models to predict node sequences by learning from graph structure. Unlike natural language with existing coherent and abundant corpora, graphs fail to provide structured and meaningful node orders inherently, making the corpus construction with high-quality node sequences challenging. To address this problem, we design **PathGLM** (**Path**-based **G**raph **L**anguage **M**odel), which first builds the community-centric corpus that constrains path selection within community scope. Next, we extract structurally node-to-center paths fed into LLMs to learn the graph language grammar, also serving as prefixes in fine-tuning. Experimental results illustrate that PathGLM improves semantic-structure integration and achieves state-of-the-art performance.

1 Introduction

Text-attributed graphs (TAGs) are ubiquitous in real-world applications (Chen et al., 2024b), where nodes are associated with rich textual content and edges encode relationships between entities. A prominent example is the citation network, where each node represents a paper and edges denote citation links between papers.

A common solution pipeline for TAGs combines language models with graph neural networks: node texts are first encoded into embeddings, followed by GNNs for neighbor aggregation of structural information (He et al.; Pan et al.; Zhao et al., 2023).

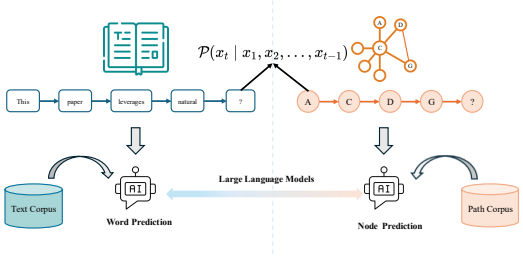


Figure 1: A Case Study about How Graph Structure Mimics Natural Language for LLMs.

Despite their effectiveness, this two-stage design remains suboptimal, since this decouples textual understanding from structural reasoning and fails to jointly capture the complex interplay between node attributes and graph topology. Fundamentally, this limitation arises because large language models are not naturally adapted to graph structures (Guo et al.). Recent works have attempted to utilize LLMs for graph learning by representing graphs as text descriptions in prompts (Tang et al., a; Luo et al.; Li et al.). However, due to the input token limitations of language models, it is infeasible to represent entire graphs within a single paragraph or complete text (Huang et al.), particularly in large-scale graphs with numerous nodes and edges.

Nevertheless, language models excelling at word prediction in natural languages offer a promising inspiration. Since LLMs (Li et al., 2024) are highly effective at modeling word sequences, a natural idea is to treat the graph as a collection of node paths, which could be regarded as meaningful node sequences similar to sentences in natural language. This perspective enables the potential of utilizing LLMs to estimate the probability of the subsequent connected node in the context of graph structure learning, as shown in Figure 1. Compared with natural language, where words inherently form sentences, graphs do not naturally offer meaningful node sequences as a trainable corpus. As a result, constructing a high-quality training corpus that jointly encodes nodes and relations becomes crucial for efficient graph language learning.

Creating a powerful strategy to select coherent and informative node sequences from the graph is critical, as they can form a training corpus that empowers the model to generalize and represent graph structures accurately. However, constructing such a corpus presents two key challenges. First, while a graph contains countless possible node sequences interconnected by edges, only a small portion truly carries structurally valuable signals. Unstructured paths may fail to encode meaningful topological cues, resulting in misleading the model to learn structures. Furthermore, paths without meticulous selection, such as random sampling, may introduce redundant or noisy information, which restrains the graph learning from capturing significant and coherent dependencies. The second challenge in graph language modeling is that nodes play distinct roles across various structural contexts. Relying on limited subgraphs to interpret paths can result in incomplete utilization of topology. Thus, this limitation resembles interpreting a word based on a single scenario, ignoring context-sensitivity, which restricts the model's ability to capture multi-hop, even complex relational patterns.

Therefore, we propose the **Path**-based **G**raph **L**anguage **M**odel (**PathGLM**), including a community-centric corpus that contains structurally meaningful node-to-center paths generated within communities. Similar to the standard training paradigm of large language models, we perform pre-training on the well-designed node sequences to facilitate the large language model to develop a global understanding of graph structure. This stage teaches LLMs to understand the "grammar" of graph language, denoting generalizable structure patterns. In task-specific tuning, we further enhance node representations by incorporating path tokens obtained from the graph language learning step, allowing the model to better adapt to downstream graph tasks such as node classification. The main contributions of PathGLM are as follows: **(1)** We follow the Graph-as-Language paradigm to leverage the nature of language models, as well as sequence prediction, to understand topological patterns on graphs. **(2)** We organize the graph topology as language-interpretable node-to-center paths, allowing LLMs to internalize structural patterns in the form of node sequences. **(3)** We design a task-specific tuning mechanism that integrates structural hints into node texts encoding for effective optimization. **(4)** Experiments demonstrate the effectiveness of PathGLM with community-centric corpus construction and paths as sequence input for graph language learning and downstream tasks.

2 Preliminaries

Given a text-attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$, where each node $v_i \in \mathcal{V}$ is associated with a textual attribute $t_i \in \mathcal{T}$, our objective is to learn node-level embeddings that effectively capture both the semantic content t_i and structural property $\mathcal{E}_i$ of node v_i. In contrast to conventional approaches that leverage message passing on graphs, we inject structural cues directly into the input space of large language models. These path tokens are concatenated with the node's textual content to form the final input:

$$f_{\text{LLM}} : [\mathcal{E}_i; t_i] \longrightarrow \text{Emb}_i, \tag{1}$$

where f_{LLM} represents a large language model that takes the structural prompt $\mathcal{E}_i$ (e.g., path-based context) as a prefix concatenated with the node text t_i. The output Emb_i is a joint representation capturing both semantic and structural features and transferred to graph learning tasks such as node classification and link prediction.

3 Methodology

We propose a novel LLM-based framework that considers graph structure as a new format language to enable language models to process structural information. To avoid understanding nodes from a single point, we partition nodes into different communities to capture context-sensitive interpretations. Moreover, we design the coherent corpus including node sequences connected with complex relational patterns, i.e., node-to-center paths, to pretrain LLMs to learn graph grammar by sequence prediction. This design addresses the challenge of limited structural information input for language models and enables the model to better align nodes in communities with diverse structural contexts by well-structured paths. Our model includes three key components, including **Community-Centric Corpus Design, Graph Grammar Acquisition, Topology-Aware Task Tuning** in Figure 2. The rationale of these modules is detailed as follows.

3.1 Community-Centric Corpus Design

To transform a graph into a language-like format, we conceptualize graphs as linguistic systems by defining nodes as "words" and multi-hop relations as "sentences". One challenge is that nodes usually exhibit different roles depending on contexts in the

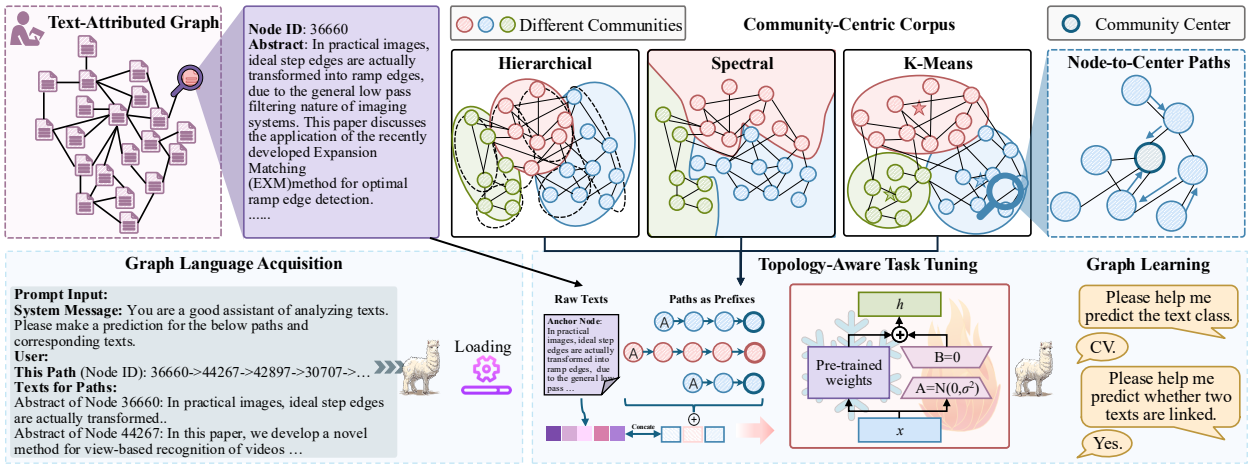


Figure 2: The illustration of **PathGLM**, graph language modeling on a toy text-attributed graph. This framework groups nodes with long semantic texts into communities, representing distinct topological contexts, and then node-to-center paths are sampled to construct the structurally coherent corpus. Next, these texts of nodes in paths are concatenated as input sequences, which are encoded to learn the graph language grammar. The path sequences, also serving as prefixes, guide LLMs to learn text-structure representations for graph tasks.

graph topology. Similar to how words have diverse meanings in diverse communication scenarios, the context-dependent contributions of nodes mainly arise from the fact that relations exhibit many-to-many mappings and intricate combinations, and long-range dependencies. Therefore, the contexts not only capture their roles in the topology broadly but also assist in designing high-quality sentences composed of the training corpus.

Next, we group nodes based on their semantic similarity into concept-related communities based on shallow embeddings. Furthermore, we employ three clustering algorithms rather than relying on only one single approach to avoid a narrow understanding of graph language paradigm: k-means, spectral, and hierarchical clustering. Each method focuses on distinct aspects such as node semantic similarity and structural properties, enabling us to assemble various and multifaceted communities.

3.1.1 K-means Clustering

Specifically, the k-means algorithm partitions nodes based on their feature vectors by minimizing intra-cluster variance. Formally, given node embeddings of $\mathcal{V}$, $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ with $\mathbf{x}_i \in \mathbb{R}^d$, the objective is to find K clusters $\{\mathcal{C}_1, \ldots, \mathcal{C}_K\}$ that minimize the within-cluster sum of squares:

$$\min_{\{\mathcal{C}_1,\ldots,\mathcal{C}_K\}} \sum_{k=1}^{K} \sum_{\mathbf{x}_i \in \mathcal{C}_k} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2, \qquad (2)$$

where $\boldsymbol{\mu}_k$ denotes the centroid of cluster $\mathcal{C}_k$. By grouping nodes that share similar semantic embeddings, this method constructs compact semantic communities for our graph-as-language framework, which reflect the nuanced understanding of nodes. These communities are formed based on embedding proximity in a joint semantic-structural feature space. Consequently, the communities $\{\mathcal{C}_1, \ldots, \mathcal{C}_K\}$ exhibit high textual and topological affinity, enabling the identification of nodes with closely aligned meanings and structural roles.

3.1.2 Hierarchical Clustering

We employ hierarchical clustering in a bottom-up manner by starting from singleton clusters where each node forms an individual community. Pairs of clusters are merged according to the Ward linkage criterion, which minimizes the increase in total within-cluster variance. This method adaptively uncovers structurally cohesive groups based on the distinct granularity of nodes. Formally, given clusters $\{\mathcal{C}_1, \ldots, \mathcal{C}_m, m > K\}$ and their centroids $\{\mu_1, \ldots, \mu_m\}$, the distance between two clusters $\mathcal{C}_i$ and $\mathcal{C}_j$ is defined as:

$$d(\mathcal{C}_i, \mathcal{C}_j) = \frac{|\mathcal{C}_i||\mathcal{C}_j|}{|\mathcal{C}_i| + |\mathcal{C}_j|} \|\mu_i - \mu_j\|_2^2, \qquad (3)$$

where $|\mathcal{C}_i|, |\mathcal{C}_j|$ denote community cardinalities and μ_i, μ_j represent centroids. Optimizing the above geometrical criterion and penalizing intra-cluster variance growth promotes compact communities. This optimization seeks the optimal K clusters based on shallow embeddings, without incorporating structural features like the adjacency matrix. Consequently, K communities reflect semantic features, representing topic-related similarity.

3.1.3 Spectral Clustering

This algorithm extracts communities by analyzing the graph's global structure in a low-dimensional hidden space. The key target is to approximate the graph minimum cut problem by employing spectral decomposition and continuous relaxation technique to effectively obtain balanced graph partitions:

$$N_{\text{cut}}(\mathcal{C}_i, \mathcal{C}_j) = \frac{\text{cut}(\mathcal{C}_i, \mathcal{C}_j)}{\text{vol}(\mathcal{C}_i)} + \frac{\text{cut}(\mathcal{C}_i, \mathcal{C}_j)}{\text{vol}(\mathcal{C}_j)}, \quad (4)$$

where $\text{cut}(\mathcal{C}_i, \mathcal{C}_j)$ measures inter-cluster edge weights, and $\text{vol}(*)$ represents cluster density computation. Formally, we compute the normalized graph Laplacian $L_{\text{sym}} = I - D^{-1/2}AD^{-1/2}$ from adjacency matrix A and degree matrix D, and extract the first q eigenvectors, $\mathbf{U} \in \mathbb{R}^{n \times q}$, to form a structural node embeddings. Applying k-means clustering on $\mathbf{U}$ yields clusters $\{\mathcal{C}_1, \ldots, \mathcal{C}_K\}$, which correspond to the spectral communities as in Eq.2. Consequently, the identified K structurally cohesive groups reflect globally structural co-usage, emphasizing intrinsic connectivity to capture the latent topological "grammar" of graph language.

3.1.4 Node-to-Center Paths

After graph partition, we obtain communities that provide expressive and abundant interpretations of nodes from different perspectives. Moreover, considering structural and sequential connections in graphs for node roles is crucial because nodes exhibit extensive links, even within communities. Considering graphs as language-like formats, we have a "vocabulary" with contextual groups, but we lack the notion of "sentences" of graph language, as well as structured compositions encompassing informative dependencies and avoiding redundant or distracting information for language learning.

To encode multi-relational structure into language-like sequences, we build structural paths upon the previous communities since they narrow the graph scope for formulating the complicated relations. However, LLMs are inherently designed for sequence prediction, not for graph structure. Drawing inspiration from how dictionaries often include exemplary sentences to illustrate the use of words, we aim to generate high-quality structural sequences that demonstrate both semantic roles and topological dependencies. Further, they serve as expressive and representative "sentences" in our graph language, enabling the model to learn grammar rules within and across communities.

Therefore, we construct node-to-center paths for each node as communities offer a valuable anchor, i.e., centroids. This design reduces noise from other irrelevant nodes and organizes the graph structure into concise and interpretable paths. The links among nodes in paths reflect meaningful topology, such as adjacency frequency and distance, which carry on both local and global structure, ensuring a high-quality training corpus. Formally, for each community $\mathcal{C}_k^{(l)}$ from the l methods, we define the centroid node $c_k^{(l)}$. For any node $v_i \in \mathcal{C}_k^{(l)}$, we compute the shortest path $\mathcal{P}_{i \to c_k^{(l)}}$ from v_i to the community center $c_k^{(l)}$ over the graph $\mathcal{G}$. This can be expressed as:

$$\mathcal{P}_{i \to c_k^{(l)}} = \arg\min_{\mathcal{P} \subseteq G} \text{Length}(\mathcal{P}) \quad (5)$$

$$\text{s.t.} \quad \mathcal{P} \text{ connects } v_i \text{ and } c_k^{(l)}, \quad (6)$$

where $\mathcal{P}_{i \to c_k^{(l)}} = [v_i, \ldots, c_k^{(l)}]$ denotes the specific expression of node-to-center paths. This shortest path guarantees efficient sequences from individual nodes to their community centers.

3.2 Graph Grammar Acquisition

To address LLMs' input length limits and the need for topology-aware semantic understanding, we have obtained nodes as topic-related "vocabulary" and paths as "grammar." Our community-based node-to-center paths capture both semantic roles and structural dependencies more effectively than random walks or neighborhood sampling. These paths, centered on community hubs that represent key concepts, provide coherent and representative sequences for learning graph grammars. Unlike prior methods that focus mainly on text relevance and overlook explicit structure (Huang et al.), our approach better integrates topology. However, how to leverage these structured paths as training data for LLMs to learn and generalize graph grammar remains a significant challenge. Furthermore, we provide experimental evidence and theoretical justification in Appendix A to illustrate long-range dependencies preservation.

By representing node-to-center paths as sentence-like sequences, we train large language models to learn graph topology rules. Following the standard LLM paradigm, we usually pre-train the model on a high-quality corpus to learn linguistic patterns. Therefore, we model the graph structure by treating each path as a sequence to

mimic word sequence prediction. However, to improve efficiency and holistic understanding, we constrain predictions to only the terminal node label rather than every next node to improve efficiency and promote topological learning.

Directly exploiting truth labels for pre-training supervision risks data leakage since test data are included in paths. Instead, we leverage the model to predict the terminal node (i.e., the community center) based on unsupervised clustering, which prevents data leakage and encourages the model to learn structural patterns shared across diverse path constructions from communities. Specifically, we encode the node-to-center path of node v_i in community $\mathcal{C}_k^{(l)}$, $\mathcal{P}_{v_i \to c_k^{(l)}} = [v_i, \ldots, c_k^{(l)}]$, into sequence by concatenating texts along paths:

$$X_i = \text{Concat}(\text{Text}(v) \mid v \in \mathcal{P}_{v_i \to c_k^{(l)}}), \quad (7)$$

where $\text{Text}(v)$ denotes textual content. To explicitly incorporate structural information, the prompt prepends node IDs to their corresponding texts, reflecting the traversal order. An illustrative prompt example is provided in Appendix D.

By training the model based on node-to-center paths during pre-training, it gains a deeper understanding of the graph structure. Given the complete path sequence composed of node texts along the path X_i, the LLM is trained to predict center node $y_i = k$ of the community $\mathcal{C}_k^{(l)}$ associated with the terminal node $c_k^{(l)}$. Consequently, the model can interpret these paths not as plain texts but as signals containing graph structural information in multiple clustering methods. The pre-training objective is defined as:

$$\mathcal{L}_{\text{pre}} = -\log \mathbb{P}(y_i = k \mid X_i; \theta_{pt}), \quad (8)$$

where θ_{pt} denotes LLM parameters. This objective encourages the model to align semantic representations with the topological information embedded in node-to-center paths, effectively learning the structured graph language.

3.3 Topology-Aware Task Tuning

While pre-training on node-to-center paths allows the LLM to acquire a structural understanding of graph connections, downstream tasks such as node classification demand that the model apply these structural patterns to specific tasks. Leveraging the pre-trained topological grammar to complement semantic information could perform more accurate node classification. Fine-tuning aims to ensure that the LLM fully leverages acquired structural knowledge by integrating path-aware cues into the task-specific learning process.

Therefore, we reuse node-to-center paths as soft prompts during fine-tuning. Preserving consistent paths ensures alignment between pre-trained structural patterns and task-specific tuning. This prefix-based method integrates graph topology directly into the LLM's language modeling, allowing joint utilization of structural and textual information. Specifically, we construct prefixes for all nodes by tokenizing the corresponding node-to-center paths. This design not only reduces the dimensionality of path embeddings but also reinforces the activation of structural grammar learned during pre-training. For each node v_i, we define the prefix:

$$P_i^{(l)} = \text{Concat}\big(t_v \mid v \in \mathcal{P}_{v_i \to c_k^{(l)}}\big). \quad (9)$$

We form the fine-tuning input by concatenation:

$$X_i^{\text{ft}} = \text{Concat}\big(P_i^{(l)} || \ \text{Text}(v_i)\big), \quad (10)$$

where includes a hyperparameter, the length of $P_i^{(l)}$. The model is fine-tuned with LoRA adapters with input of X_i^{ft} and outputs class probabilities $\hat{y}_i$. We formulate the standard cross-entropy loss over the ground true labels y_i as follows:

$$\mathcal{L}_{\text{ft}} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{|\mathcal{Y}|} y_i^j \log p\big(y_i = j \mid X_i^{\text{ft}};\ \theta_{\text{pt}},\ \theta_{\text{ft}}\big), \quad (11)$$

where only θ_{ft} is updated during fine-tuning while θ_{pt} remains fixed in this step.

4 Experiments

In this section, we conducted extensive experiments to evaluate the effectiveness of PathGLM, which leverages LLaMA3 as the backbone. We compared our model against state-of-the-art baselines under the same settings of LLaMA3. Moreover, they are designed to answer the following key research questions (RQs): **RQ1:** Does our PathGLM consistently outperform competitive baselines on node classification? **RQ2:** How do different strategies for constructing the structural communities affect path informativeness in the training corpus? **RQ3:** How do node-to-center paths contribute to learning graph language grammar by pre-training LLMs? Then, we assessed the efficiency of pre-training and task tuning and analyzed hyperparameters.

Table 1: Comparison of classification *Accuracy* and *Macro-F1* score among baselines and the PathGLM on three benchmark datasets (averaged over 3 runs).

NLP Models	Graph Models	ACM		Wikipedia		Amazon	
		Test-Acc.	Macro-F1.	Test-Acc.	Macro-F1.	Test-Acc.	Macro-F1.
Fine-tuned Language Models +/- GNNs							
BERT	-	73.2	69.0	68.8	58.4	90.5	86.9
	GCN	77.1	74.6	68.4	58.7	93.3	90.1
	GAT	78.0	74.1	69.8	60.8	93.6	90.6
	GraphSAGE	76.8	74.1	72.7	59.5	92.9	90.0
RoBERTa	-	76.6	70.7	68.1	57.6	85.9	83.9
	GCN	79.4	74.1	68.0	56.1	92.5	90.7
	GAT	78.9	74.2	71.0	61.2	92.4	90.5
	GraphSAGE	78.3	74.1	72.1	57.5	92.1	90.4
Fine-tuned Large Language Models +/- GNNs							
Llama3_8b	-	80.6	73.8	71.2	59.0	91.6	88.3
Llama3_8b	GraphSAGE	<u>81.3</u>	<u>76.4</u>	<u>73.0</u>	<u>60.6</u>	92.8	89.6
Pre-trained Large Language Models							
GPT-3.5		54.3	51.8	61.8	59.1	49.1	46.5
GPT-4		67.5	64.7	60.9	58.3	40.3	38.6
Deepseek_v2		64.8	62.3	15.4	10.5	23.4	26.0
Tailored Frameworks							
MPAD		78.9	71.6	68.0	53.9	92.8	88.6
GLEM		79.8	73.9	71.2	58.3	**94.3**	<u>90.9</u>
LLAGA		77.5	72.1	72.0	60.5	90.8	88.6
GraphFormers		75.1	65.4	67.5	51.2	86.4	82.2
InstructGLM		74.5	68.9	70.6	58.1	<u>94.2</u>	89.6
PathGLM		**85.1**	**79.2**	**74.8**	**63.1**	93.6	**91.7**

4.1 Experimental Setup

4.1.1 Datasets

We evaluate the performance of PathGLM on three datasets: **ACM**, **Wikipedia**, and **Amazon**, which are manually constructed from raw corpora with associated textual descriptions and categorical labels. All datasets are split into training, validation, and test sets with a ratio of 8:1:1. Detailed dataset statistics are provided in Appendix B.

4.1.2 Baselines

We evaluate PathGLM performance against baselines in four categories: (1) pretrained language models (e.g., BERT, RoBERTa) combined with GNNs like GCN; (2) large language models combined with neural networks, where we use LLaMA3 and GraphSAGE for strong performance; (3) pretrained large language models used directly via APIs; and (4) tailored frameworks integrating structure and text through prompt design or co-training, including MPAD (Nikolentzos et al., 2020), GLEM (Zhao et al., 2023), LLAGA (Chen et al., 2024a), GraphFormers (Yang et al.), and InstructGLM (Ye et al.), referring to Appendix C.

4.1.3 Implementation Details

Many details and parameter settings of experiment design, including hyperparameters, are provided in the appendix E. To assess classification performance, we adopt two metrics: *Accuracy* measures the proportion of correctly predicted node labels over all test instances. *Macro-F1* computes the averaged F1-score independently for each class.

4.2 Overall Performance (RQ1)

We evaluate the performance of PathGLM with four categories of baselines across multiple TAG benchmarks. As shown in Table 1, our model consistently achieves state-of-the-art performance, highlighting the advantages of graph language modeling. PathGLM outperforms both LM-GNN pipelines and strong baselines such as LLaMA3 combined with GraphSAGE, despite their ability to handle longer inputs with more parameters. Unlike sequential pipelines that struggle with semantic-structural learning, our model leverages node-to-center paths and graph language modeling for more effective structural understanding. While larger pretrained models exist, their lack of task-specific adaptation limits effectiveness. Hybrid methods

by joint training or concatenating neighborhood embeddings, still underutilize graph relations.

4.3 Effect of Communities on Path Informativeness (RQ2)

To investigate how multiple communities influence the informativeness of generated node-to-center paths, we select several paths by random walks and compare their semantic and structural information with that of node-to-center paths. We hypothesize that paths derived from unsupervised clustering approaches contain more semantic and structural information. To verify this, we conduct experiments comparing the path informativeness generated by clustering with that of the random walk algorithm. Specifically, we evaluate these sequences using two metrics: (1) average semantic similarity between the beginning and remaining nodes along paths (measured by BERT-based embeddings), and (2) average closeness centrality of all nodes in paths.

Table 2: Semantic and Structural Evaluation of Paths Rooted at Many Randomly Selected Anchors.

Path Sequences	Similarity	Centrality
Random walks	0.1324	0.1454
K-means	0.2576	0.1467
Spectral	0.3565	0.1527
Hierarchical	0.2138	0.1359

As shown in Table 2, we compare path sequences generated by random walk and three clustering methods (k-means, spectral, and hierarchical). The results demonstrate that clustering-based sequences can achieve higher semantic similarity than random walks, indicating more coherent topic-centric information. Although structural centrality varies in three types of paths, node-to-center paths generally preserve meaningful topological context, which supports the necessity of leveraging different algorithms to construct "sentences" for the corpus.

4.4 Analysis of Node-to-Center Paths (RQ3)

Our method employs node-to-center paths from different clustering techniques for graph language acquisition. Unlike random walks, which sample neighbors in an unordered, purely structural way, these paths reflect semantic categories and topological roles. To clearly visualize their impacts on PathGLM, we evaluate paths from these clustering methods separately, with random walks as a baseline. This comparison shows that our paths provide higher-quality topological context than paths from random combinations.

As shown in Table 3, even a single node-to-center path consistently outperforms the path from random walks, demonstrating their effectiveness in capturing meaningful graph language patterns. While individual clustering may not yield node-to-center paths for every node, the combination of structured paths mitigates that by compensating for missing paths, contributing to robust performance. Moreover, combining paths from three clustering strategies achieves the highest accuracy overall, suggesting that different clustering methods encode complementary semantic and topological information. Notably, when combined paths are used only during pretraining, performance significantly drops, and this highlights the importance of fine-tuning in fully exploiting structural cues. These findings confirm that community-centric constructed paths offer richer and more informative supervision than randomly sampled neighbors.

4.5 Hyperparameter and Efficiency Analysis

We analyze the impact of a key hyperparameter, the prefix length of node-to-center paths concatenated to node texts in the task tuning stage, on different datasets. The chosen lengths were based on the average path lengths: for ACM, we tested {3, 6, 12, 18}, while for Amazon and Wiki, the lengths were {5, 11, 16, 22}. Results show that prefix lengths close to the average path length sufficiently carry the informative content, as shown in Figure 3.

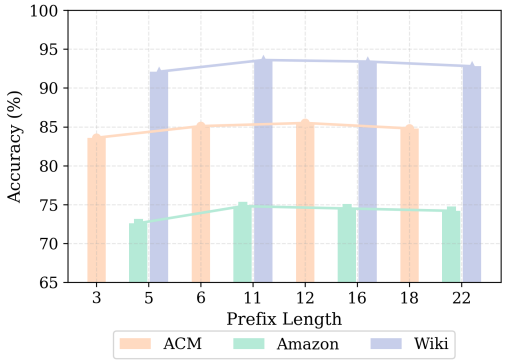


Figure 3: Performance Impacts of Prefix Length.

Our framework demonstrates practical training efficiency except for outperformance in experiments, with pre-training and fine-tuning completed within an acceptable implementation time span (about 40h and 20h per dataset, respectively).

Table 3: Performance Comparison on Three Datasets Using Different Path Sampling Strategies.

Path Strategy	ACM		Wiki		Amazon	
	Accuracy	Macro-F1	Accuracy	Macro-F1	Accuracy	Macro-F1
No path	80.6	73.8	71.2	59.0	91.6	88.3
Random walks	83.2	78.5	72.7	60.5	92.9	89.8
K-means	84.5	79.1	73.2	61.4	93.5	90.9
Spectral	83.6	78.9	73.9	61.7	93.3	90.6
Hierarchical	83.7	78.5	73.0	61.0	92.8	90.1
Pre-train only	82.7	77.1	71.7	59.9	92.7	88.5
Multi-paths	**85.1**	**79.2**	**74.8**	**63.1**	**93.6**	**91.2**

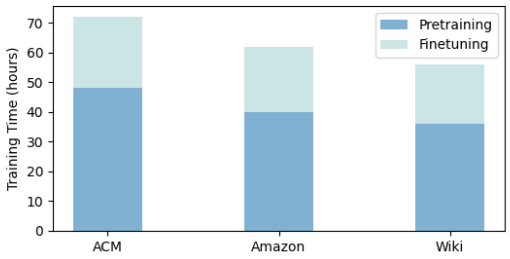


Figure 4: Time Cost Analysis.

5 Related Work

5.1 Graph Descriptions for LLMs

Text-attributed graph learning relies on extracting rich features via language models like BERT (Devlin et al.) and RoBERTa (Liu et al., b), but the rise of large language models (LLMs) has greatly boosted performance and efficiency compared to LM and GNN combos with fine-tuning techniques like LoRA (Hu et al., a). However, topology in the graph remains underexploited. Recent work adapts LLMs for graphs by encoding structural signals as natural language prompts, with methods like OFA (Liu et al., a) and SimCSE (Li et al.). In addition, WalkLM (Tan et al.) and InstructGLM (Ye et al.) augment node texts through neighborhood texts or relation summaries. However, these face information loss and input length limits. Therefore, approaches such as GraphGPT (Tang et al., a) and LLAGA (Chen et al., 2024a) encode structure as graph embeddings aligned with text embeddings, reducing redundant texts in subgraphs but relying on local neighborhoods, with limited compatibility.

5.2 Integration of GNN and LLM

Integrating GNN and LLM directly can leverage both graph structure and text semantics, two modalities, and offer complementary interaction. However, simple cascaded pipelines suffer from embedding space mismatch and limitations of interplay between graph topology and textual content. Modality alignment-based approaches such as ConGraT (Brannon et al.), LinguGKD (Hu et al., b), and TEAGLM (Wang et al.) align embeddings from networks and language models by contrastive learning or projection modules. Distillation of structural knowledge (Pan et al.) has been used to transfer rich structure-text patterns from teacher models. More deeply integrated architectures such as TAPE (He et al.) and Dr.E (Liu et al., c) stack GNN and LLM layers to inject aggregated node embeddings into language models or describe structural information for subsequent graph aggregation. Despite the above improvements, some important challenges, including optimization difficulties and high computational cost, remain at the same time. Besides, GraphFormers (Yang et al.) and HASHCODE (Zhang et al.) tackle these issues by designing graph neural network layers specifically compatible with language models and adopting coordinated training. These approaches generally rely on intermediate embedding exchanges but lack deep semantic-structural fusion.

6 Conclusion

In this paper, we propose PathGLM, a novel model that encodes graphs as a new language to enable LLMs learning on text-attributed graphs. By clustering nodes into contextual communities, our model constructs node-to-center paths as high-quality sequences, employed for pre-training large language models to understand graph language grammar. Topology-aware task tuning leverages acquired structure patterns by concatenating paths as prefixes instead of relying on graph neural networks for node classification. We present a Graph-as-Language framework that integrates structure and semantic modeling, yielding outperformance across benchmarks. Future work will explore extensions to large-scale and heterogeneous graphs.

7 Limitations

Our method offers a new perspective by modeling graphs as a new language for semantic and structural fusion, but that presents limitations. Though effective on moderate-sized graphs, our method faces computational challenges on large graphs with millions or billions of nodes, where community detection and path design become significantly more expensive. Scaling to such settings remains an open direction for future exploration.

Further, our current framework is designed and evaluated primarily on text-attributed graphs with node classification as the main downstream task. Its applicability to alternative tasks like link prediction remains unexplored and open a promising direction. Additionally, experiments focuses on widely-used benchmarks composed of homogeneous graphs, but could be extended to complex settings such as heterogeneous graphs and dynamic graphs.

While this model relies on LLaMA3 as the backbone within the limitation of computation cost, large language models with more parameters such as 13B could perform better in graph learning.

Ethics Statement

We all comply with the ACL Ethics Policy[1] during our study. All datasets used contain anonymized consumer data, ensuring strict privacy protections.

References

William Brannon, Wonjune Kang, Suyash Fulay, Hang Jiang, Brandon Roy, Deb Roy, and Jad Kabbara. ConGraT: Self-supervised contrastive pretraining for joint graph and text embeddings. Contrastive.

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

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arxiv:1810.04805 [cs].

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. Prompts-GDL.

Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*, pages 507–517. International World Wide Web Conferences Steering Committee.

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. GNN-LLM.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. a. Lora: Low-rank adaptation of large language models. *Preprint*, arxiv:2106.09685 [cs].

Shengxiang Hu, Guobing Zou, Song Yang, Yanglan Gan, Bofeng Zhang, and Yixin Chen. b. Large language model meets graph neural network in knowledge distillation. GNN-LLM.

Jin Huang, Xingjian Zhang, Qiaozhu Mei, and Jiaqi Ma. Can LLMs effectively leverage graph structural information through prompts, and why? Prompts.

Rui Li, Jiwei Li, Jiawei Han, and Guoyin Wang. Similarity-based neighbor selection for graph LLMs. Similar neighbor.

Xin Li, Weize Chen, Qizhi Chu, Haopeng Li, Zhaojun Sun, Ran Li, Chen Qian, Yiwei Wei, Zhiyuan Liu, Chuan Shi, Maosong Sun, and Cheng Yang. 2024. Can Large Language Models Analyze Graphs like Professionals? A Benchmark, Datasets and Models. In *Advances in Neural Information Processing Systems*, volume 37, pages 141045–141070. Curran Associates, Inc.

Hao Liu, Jiarui Feng, Lecheng Kong, Ningyue Liang, Dacheng Tao, Yixin Chen, and Muhan Zhang. a. One for all: Towards training one graph model for all classification tasks. All tasks.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. b. RoBERTa: A robustly optimized BERT pretraining approach. *Preprint*, arxiv:1907.11692 [cs].

Zipeng Liu, Likang Wu, Ming He, Zhong Guan, Hongke Zhao, and Nan Feng. c. Dr.e bridges graphs with large language models through words. GNN-LLM.

Zihan Luo, Xiran Song, Hong Huang, Jianxun Lian, Chenhao Zhang, Jinqi Jiang, and Xing Xie. GraphInstruct: Empowering large language models with graph understanding and reasoning capability. Reasoning.

[1] https://www.aclweb.org/portal/content/acl-code-ethics

Giannis Nikolentzos, Antoine Tixier, and Michalis Vazirgiannis. 2020. Message Passing Attention Networks for Document Understanding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8544–8551. Number: 05.

Bo Pan, Zheng Zhang, Yifei Zhang, Yuntong Hu, and Liang Zhao. Distilling large language models for text-attributed graph learning. Student-interpre.

Yanchao Tan, Zihao Zhou, Hang Lv, Weiming Liu, and Carl Yang. WalkLM: A uniform language model fine-tuning framework for attributed graph embedding. In *Proceedings of the 37th Conference on Neural Information Processing Systems*. GSCC: 0000031.

Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. a. GraphGPT: Graph instruction tuning for large language models. ****instruction Tuning.

Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. b. ArnetMiner: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 990–998. ACM.

Duo Wang, Yuan Zuo, Fengzhi Li, and Junjie Wu. LLMs as zero-shot graph learners: Alignment of GNN representations with LLM token embeddings. *Preprint*, arxiv:2408.14512 [cs].

Junhan Yang, Zheng Liu, Shitao Xiao, Chaozhuo Li, Defu Lian, Sanjay Agrawal, Amit Singh, Guangzhong Sun, and Xing Xie. GraphFormers: GNN-nested transformers for representation learning on textual graph. *Preprint*, arxiv:2105.02605 [cs].

Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. Language is all a graph needs. *Preprint*, arxiv:2308.07134.

Peiyan Zhang, Chaozhuo Li, Liying Kang, Feiran Huang, Senzhang Wang, Xing Xie, and Sunghun Kim. High-frequency-aware hierarchical contrastive selective coding for representation learning on text-attributed graphs. Contrastive.

Jianan Zhao, Meng Qu, Chaozhuo Li, Hao Yan, Qian Liu, Rui Li, Xing Xie, and Jian Tang. 2023. Learning on large-scale text-attributed graphs via variational inference. *ICLR*.

A Theoretical Foundation

We provide an information-theoretic perspective to justify the effectiveness of node-to-center paths. In graphs, closer nodes (e.g., one-hop or two-hop neighbors) typically share stronger semantic or structural similarities, while long-range dependencies may introduce noise or diluted signals. Our goal is to sample high-quality sequences that maximize the structural and semantic information available to language models. We quantify the informational value of a k-hop neighborhood by the mutual information between a target node v and its neighbors N_k:

$$IG(v, N_k) = I(v; N_k) = H(v) - H(v \mid N_k), \tag{12}$$

where $H(v)$ is the entropy of node v, and $H(v \mid N_k)$ is the conditional entropy given its k-hop neighbors. A lower conditional entropy implies that the neighborhood contains more relevant information about the node.

We compare the information gain from two neighborhoods with hop distances $i < j$:

$$IG(v, N_i) > IG(v, N_j) \Leftrightarrow H(v \mid N_i) < H(v \mid N_j), \tag{13}$$

where we assume that closer neighbors are more predictive of the node itself. To express this relationship, we model the conditional entropy as proportional to that of the 1-hop neighborhood:

$$H(v \mid N_i) = \lambda H(v \mid N_1) \tag{14}$$

$$H(v \mid N_j) = \gamma H(v \mid N_1), \tag{15}$$

with constants $0 < \lambda < \gamma < 1$ reflecting the diminishing predictability as neighborhood radius increases. Thus, the difference in entropy becomes:

$$H(v \mid N_j) - H(v \mid N_i) = (\gamma - \lambda) H(v \mid N_1) > 0, \tag{16}$$

leading to:

$$IG(v, N_i) > IG(v, N_j). \tag{17}$$

This theoretical insight supports our node-to-center path design: by prioritizing paths through semantically central nodes within a cluster (i.e., a lexicon), we form sequences that carry more informative and coherent structural language content than those relying solely on local neighborhoods.

B Datasets

The statistics of datasets are shown in Table 4, and details are described below:

Wikipedia. This dataset is constructed from Wikipedia articles, where each node represents an article and edges are formed by hyperlink references [2]. The category labels are assigned based on the taxonomy provided in Wikipedia's reference lists.

ACM. We collect papers from the ACM digital library (Tang et al., b), where each node corresponds to a paper and citation links form directed edges. Each paper is labeled according to its research field, covering areas such as Artificial Intelligence, Data Mining, and Machine Learning.

Amazon. This dataset is built from Amazon product metadata (He and McAuley), where nodes represent products and edges are formed by co-view relationships in user browsing history. Each product is categorized into a department-level class as its label.

Table 4: Statistics of datasets in our experiment.

Datasets	#nodes	#edges	#classes
ACM	48,579	193,034	9
Wiki	36,501	1,190,369	10
Amazon	50,000	632,802	7

C Baselines

To comprehensively evaluate our model, we compared PathGLM with current representative baselines, which can be broadly categorized into four groups:

Language models combined with graph neural networks: These models adopt a cascaded approach where they train LM-based encoders like BERT and RoBERTa and feed embeddings into common GNNs, i.e., GCN, GAT, and GraphSAGE.

Large language models combined with graph neural networks: We select LLaMA3, serving as the backbone of PathGLM. And GraphSAGE is integrated to aggregate neighborhood features due to its best performance.

Pretrained LLMs only: These approaches directly utilize APIs of large language models for the inference stage.

Tailored frameworks: These models integrate structural and textual information by proposing either prompt design or co-training: **MPAD** (Nikolentzos et al., 2020) constructs word co-occurrence graphs from the corpus and applies a message-passing framework to propagate information across the graph. **GLEM** (Zhao et al., 2023) iteratively updates a pretrained language model and a GNN using co-training loops, aligning semantic and structural signals. **LLAGA** (Chen et al., 2024a) bridges LLMs and GNNs via adaptive graph construction and task-specific LLM prompting, enabling multi-modal interaction across structure and text. **GraphFormers** (Yang et al.) unifies transformers and GNNs in a joint encoder, leveraging graph-aware self-attention. **InstructGLM** (Ye et al.) encodes structural features into natural language prompts and fine-tunes a large language model to perform graph-specific tasks.

[2] http://www.mattmahoney.net/dc/textdata

D Pre-training Prompt

We have designed a detailed path prompt for pre-training to help the LLM better understand our training objectives. This template includes general ID information for each node in the path and concatenates the text of each node. The task is to perform label prediction for the last node in the path.

E Detailed Settings

Our model first employs three clustering methods: K-means, spectral clustering, and hierarchical clustering. Based on the identified communities, shortest paths from each node to its community center are computed using Dijkstra's algorithm. The model utilizes LLaMA3 as the backbone, taking node-to-center paths as input for language acquisition, followed by fine-tuning with pretrained parameters for node classification. All experiments are conducted on a server equipped with three NVIDIA RTX 3090 GPUs (24GB memory each).

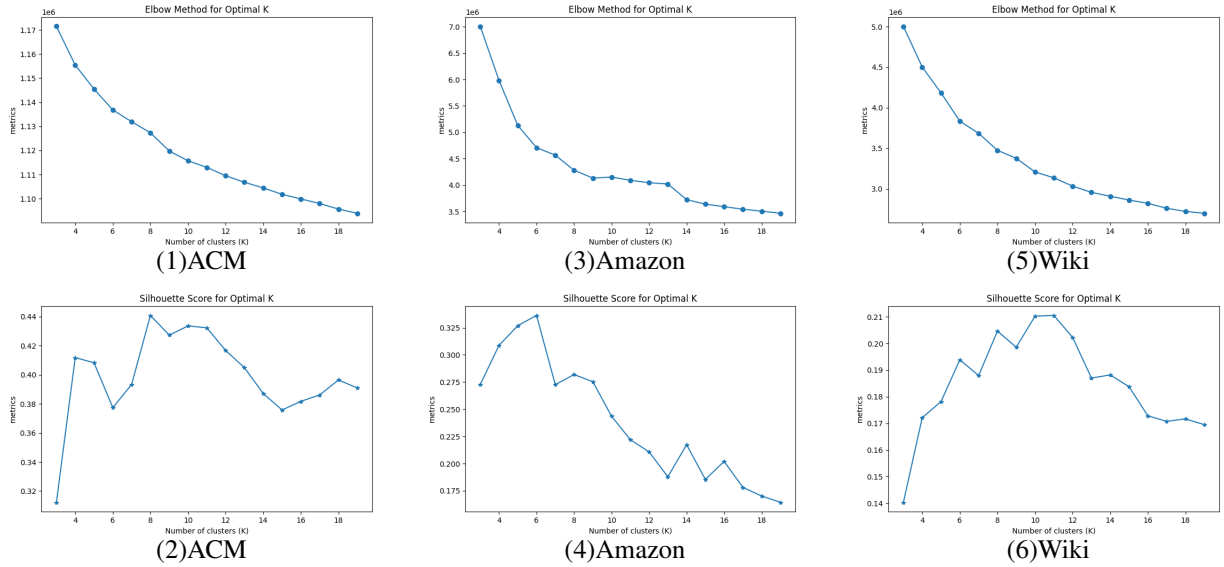
Clustering is implemented primarily using the *fassi* and *scipy* packages. For all three methods, the number of communities is predetermined by jointly considering the elbow criterion and silhouette scores. The candidate community number ranges for all three datasets is set between 2 and 20, as shown in Figure 6. In the pre-training stage, we use the Adam optimizer with a learning rate in $[4e-5, 1e-4]$. Tokenization uses a maximum sequence length of 2048. The number of training epochs is set to 2, in which the LLM achieves rapid convergence and delivers highly stable results. We

Pre-training Prompt Template:

```
System: You are a good assistant in analyzing texts. Please make a prediction for the
paths below and the corresponding texts.
User: Classify the last academic paper into 9 categories. The Path (Node ID): 36660 ->
3577 -> 35782 -> 42722 -> 21232 -> 37688 -> 11229. The texts of this path are as
follows.
Abstract of Node ID 36660: In practical images, ideal step edges are actually
transformed into ramp edges, due to the general low pass filtering nature of imaging
systems. This paper discusses the application of the recently developed Expansion
Matching (EXM) method ...
Abstract of Node ID 3577: This paper presents a novel interactive system for guiding
artists to paint using traditional media and tools. The enabling technology is a
multi-projector display capable of controlling the appearance of an artist's canvas.
This display-on-canvas ...
Abstract of Node ID 35782: Boundary detection is essential for a variety of computer
vision tasks such as segmentation and recognition. We propose a unified formulation
for boundary detection, with closed-form solution, which is applicable to the
localization ...
Abstract of Node ID 42722: We propose a novel approach for solving the perceptual
grouping problem in vision. Rather than focusing on local features and their
consistencies in the image data, our approach aims at extracting the global impression
of an image. We treat image segmentation ...
Abstract of Node ID 21232: Constrained clustering has been well-studied for algorithms
like K-means and hierarchical agglomerative clustering. However, how to encode
constraints into spectral clustering remains a developing area. In this paper, we
propose a flexible and generalized framework ...
Abstract of Node ID 37688: We pose the problem of network discovery which involves
simplifying spatio-temporal data into cohesive regions (nodes) and relationships
between those regions (edges). Such problems naturally exist in fMRI scans of human
subjects ...
Abstract of Node ID 11229: Effective diagnosis of Alzheimer's disease (AD), the most
common type of dementia in elderly patients, is of primary importance in biomedical
research. Recent studies have demonstrated that AD is closely related to the structure
change of the brain network ...
```

Figure 5: Prompt used for type prediction in PathGLM.

adopt LoRA layers to flexibly and efficiently incorporate pretrained parameters. The configuration of this layer follows standard settings: $rank = 8$, $\alpha = 32$, and $dropout = 0.05$, also applied to the task tuning. During fine-tuning, the path embeddings are concatenated with text embeddings using the same hidden dimension as the LLaMA3 (4096). The best path sequence length is aligned with the average node-to-center path length of each dataset: 6 (Wikipedia), 6 (ACM), and 11 (Amazon). Unless otherwise specified, all baseline models are reproduced according to the official settings reported in their original papers to ensure fair comparison.



(1)ACM (3)Amazon (5)Wiki

(2)ACM (4)Amazon (6)Wiki

Figure 6: Elbow Criterion and Silhouette Scores Analysis for Three Datasets.