

# LLM as GNN: Graph Vocabulary Learning for Text-Attributed Graph Foundation Models

Xi Zhu<sup>1\*</sup> Haochen Xue<sup>2\*</sup> Ziwei Zhao<sup>3</sup> Wujiang Xu<sup>1</sup> Jingyuan Huang<sup>1</sup>  
Minghao Guo<sup>1</sup> Qifan Wang<sup>4</sup> Kaixiong Zhou<sup>5</sup> Imran Razzak<sup>2</sup> Yongfeng Zhang<sup>1</sup>

<sup>1</sup>Rutgers University <sup>2</sup>Mohamed bin Zayed University of Artificial Intelligence (MBZUAI)

<sup>3</sup>University of Science and Technology of China <sup>4</sup>Meta AI <sup>5</sup>North Carolina State University

## Abstract

Text-Attributed Graphs (TAGs), where each node is associated with text descriptions, are ubiquitous in real-world scenarios. They typically exhibit distinctive structure and domain-specific knowledge, motivating the development of Graph Foundation Models (GFM) that generalize across diverse graphs and tasks. Despite large efforts to integrate Large Language Models (LLMs) and Graph Neural Networks (GNNs) for TAGs, most approaches suffer from decoupled architectures with two-stage alignment, limiting their synergistic potential. Even worse, existing methods assign out-of-vocabulary (OOV) tokens to graph nodes, leading to graph-specific semantics, token explosion, and incompatibility with task-oriented prompt templates, which hinders cross-graph and cross-task transferability. To address these challenges, we propose PromptGFM, a versatile GFM for TAGs grounded in graph vocabulary learning. PromptGFM comprises two key components: a graph understanding module, which explicitly prompts LLMs to replicate the fine-grained GNN workflow within the text space, facilitating seamless GNN-LLM integration and elegant graph-text alignment; and a graph inference module, which establishes a language-based graph vocabulary to ensure expressiveness, transferability, and scalability, enabling readable instructions for joint LLM fine-tuning across graphs and tasks. Extensive experiments demonstrate the superiority of PromptGFM, highlighting its strong transferability across diverse graphs and tasks and its adaptability to auxiliary pre-training tasks. The code is available at <https://github.com/agiresearch/PromptGFM>.

## 1 Introduction

Graphs, characterized by their non-Euclidean structures and rich domain-specific knowledge, serve

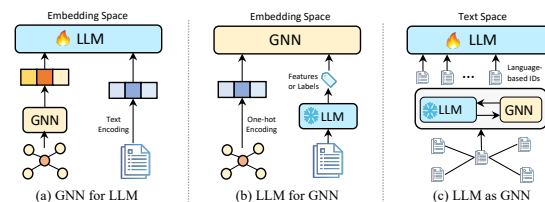


Figure 1: Overview of three GNN-LLM integration paradigms for graph-text alignment: (a) *GNN for LLM* and (b) *LLM for GNN* use decoupled architectures in the embedding space, while (c) our work operates *LLM as GNN* in the text space.

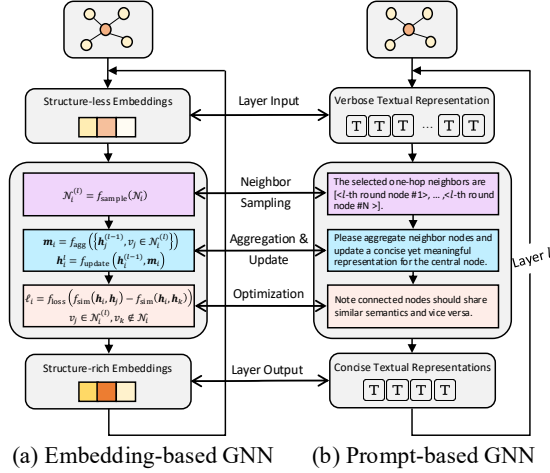
as fundamental representations of complex relational data. Many real-world graphs are further enriched with node-level textual attributes, forming Text-Attributed Graphs (TAGs), such as citation networks (Eto, 2019; Buneman et al., 2021), social networks (Kempe et al., 2003; Myers et al., 2014), and molecular graphs (Wieder et al., 2020; Jin et al., 2024). However, existing methods often depend on task- or dataset-specific training, limiting their transferability. This motivates the development of Graph Foundation Models (GFMs) that generalize across graphs and tasks (Mao et al., 2024; Xia et al., 2024; Sun et al., 2025b).

In TAGs, GNNs and LLMs are commonly combined to process structural and textual modalities, as illustrated in Figure 1. (a) **GNN for LLM**. GNNs produce structure-aware node embeddings that complement textual embeddings and improve LLM inference (Tang et al., 2024; Chai et al., 2023; Liu et al., 2024b). (b) **LLM for GNN**. LLMs extract semantic features or generate pseudo-labels from text to supervise GNN training (Chen et al., 2024c; Liu et al., 2024a; Zhu et al., 2024). However, these two-stage pipelines only loosely couple GNNs and LLMs, and thus struggle to fully exploit their complementary strengths, often resulting in suboptimal graph-text alignment.

Recently, a growing trend explores **LLM as GNN**, where graph verbalizers convert graph data into heuristic or code-like prompts, enabling LLMs

\*Equal contribution. Email: xi.zhu@rutgers.edu

Figure 2: LLM-driven replication of the GNN workflow. We achieve fine-grained alignment between traditional embedding-based GNN and our prompt-based GNN.



to capture graph semantics and structure (Ye et al., 2024; Wang et al., 2024a; Chen et al., 2024a). However, we argue that these methods are not true GNNs, as they omit the core message-passing mechanism. As shown in Figure 2(a), a standard GNN layer typically involves neighbor sampling, aggregation-update, and optimization (Kipf and Welling, 2017; Veličković et al., 2018). Stacking such layers transforms structure-less embeddings into structure-rich representations with high-order dependencies. This raises a key question: *Can LLMs replicate the GNN workflow in text space to jointly capture graph semantics and structure?*

Meanwhile, existing works intuitively treat each graph node as an out-of-vocabulary (OOV) token, resulting in graph-specific semantics and thus uncontrolled vocabulary explosion (Tang et al., 2024; Zhou et al., 2025). Worse still, due to vocabulary mismatches, ID-based node embeddings and language-based token embeddings reside in different feature spaces, causing semantic misalignment during LLM inference. This incompatibility undermines both transferability and scalability of graph-specific knowledge across diverse graphs and tasks. To enable knowledge transfer, another key question emerges: *Can OOV tokens be replaced with compatible and universal node representations to build a versatile and transferable GFM?*

Inspired by prior work (Cai, 2024; Mao et al., 2024; Wang et al., 2024b), we argue that a true GFM should be grounded in **graph vocabulary learning** with the following properties: (1) **Expressiveness**. It encapsulates both semantic and structural information across diverse graphs. (2) **Transferability**. Every node in any graph should

be representable by a composition of fundamental units within the vocabulary. (3) **Scalability**. The vocabulary should be inclusive to accommodate unseen nodes, even beyond existing graphs. Since natural language inherently provides expressive and transferable tokens (Raffel et al., 2020; Radford et al., 2021; Di Palo et al., 2023; Wang et al., 2024b), we establish a universal graph vocabulary in the text space, and then propose **PromptGFM**, a GFM for TAGs with the following components.

**Graph Understanding Module.** To operate LLMs as GNNs, we initialize node features with textual attributes and prompt LLMs to replicate the fine-grained GNN workflow in text space. As Figure 2 shows, we encode local structure via one-hop neighbor descriptions and design prompts for a flexible aggregation-update mechanism. To support optimization, we introduce heuristic prompts inspired by contrastive learning at each layer. Iterative LLM calls then simulate message passing over the graph, progressively refining verbose textual features into concise yet informative textual representations. Unlike traditional GNNs that output numerical embeddings, our prompt-based GNN produces structured textual encodings that preserve node semantics and higher-order dependencies. Consequently, LLMs can operate as GNNs, and GNNs can be interpreted as LLMs, unlocking the full potential of GNN-LLM integration and empowering elegant graph-text alignment.

**Graph Inference Module.** After capturing semantics and structure with prompt-based GNNs, we decouple the resulting textual representations to establish a graph vocabulary, mapping each node to a finite sequence of language-based tokens, essentially as language-based IDs. Because these IDs are drawn from the LLM’s native vocabulary, they preserve semantic fidelity and avoid OOV issues. Thus, this vocabulary is universally interpretable, transferable across graphs, and scalable to unseen nodes, resolving semantic irrelevance and graph-text incompatibility. These language-based IDs are then integrated into readable natural-language prompts. To support generalization, we adopt multi-instruction fine-tuning over diverse graphs and tasks, enabling cross-graph and cross-task knowledge transfer. Overall, this graph vocabulary not only mitigates graph-text incompatibility but also provides a foundation for general GFMs. Our contributions are summarized as follows:

- We present PromptGFM, a graph foundation model for TAGs built on graph vocabulary learning.

- We introduce an LLM-as-GNN paradigm in text space, enabling seamless GNN-LLM integration and effective graph-text alignment.
- We construct a language-based graph vocabulary as the cornerstone of the GFM, ensuring graph-text semantic fidelity and eliminating the OOV issue.
- We conduct extensive experiments to validate the superiority of PromptGFM, especially its cross-graph and cross-task transferability and adaptability to auxiliary pre-training tasks.

## 2 Preliminaries

**Text-Attributed Graphs (TAGs).** A TAG is defined as  $G = (V, E, X)$ , where  $V$  is the node set,  $E$  is the edge set, and  $X$  is the set of textual node attributes. Each node  $v_i \in V$  is associated with a textual description  $X_i = (x_i^1, x_i^2, \dots, x_i^{n_i})$ , where each  $x_i^k \in \mathcal{X}$ ,  $k = 1, \dots, n_i$ , and  $\mathcal{X}$  denotes the natural language dictionary.

**Graph Neural Networks.** GNNs are state-of-the-art models for graph machine learning, built on the message-passing mechanism. In practice, a GNN selects the neighboring nodes of the target node, aggregates their representations to capture local structure, and updates the target node’s representation. Mathematically, for a central node  $v_i$ , the  $l$ -th layer of a general GNN is defined as:

$$\begin{aligned}\mathcal{N}_i^{(l)} &= f_{\text{sample}}(\mathcal{N}_i), \\ \mathbf{m}_i^{(l)} &= f_{\text{agg}}\left(\left\{\mathbf{h}_j^{(l-1)}, v_j \in \mathcal{N}_i^{(l)}\right\}\right), \\ \mathbf{h}_i^{(l)} &= f_{\text{update}}\left(\mathbf{h}_i^{(l-1)}, \mathbf{m}_i^{(l)}\right)\end{aligned}\quad (1)$$

where  $\mathbf{h}_i^{(l)}$  is the node embedding of  $v_i$  at the  $l$ -th layer.  $\mathcal{N}_i$  denotes the full neighbor set and  $\mathcal{N}_i^{(l)}$  the sampled neighbor set at the  $l$ -th layer. To capture higher-order dependencies, we stack  $L$  layers and derive final embeddings as  $\mathbf{h}_i = f_{\text{pooling}}(\mathbf{h}_i^{(1)}, \dots, \mathbf{h}_i^{(L)})$  (Grattarola et al., 2024). For optimization, a contrastive loss with negative sampling is commonly used in unsupervised graph learning (Hamilton et al., 2017; Veličković et al., 2019), expressed as:

$$\ell = f_{\text{loss}}(f_{\text{sim}}(\mathbf{h}_i, \mathbf{h}_j), f_{\text{sim}}(\mathbf{h}_i, \mathbf{h}_k)), \quad (2)$$

where  $v_j \in \mathcal{N}_i^{(l)}$  is a positive sample and  $v_k \notin \mathcal{N}_i^{(l)}$  is a negative one.  $f_{\text{sim}}(\cdot)$  measures the similarity between two nodes.  $f_{\text{loss}}(\cdot)$  enforces contrastive learning by increasing the similarity for connected nodes and reducing it for unconnected ones.

## 3 Methodology

We illustrate the pipeline of our proposed PromptGFM in Figure 3, which consists of a graph understanding module and a graph inference module.

### 3.1 Graph Understanding Module

The graph understanding module generates expressive and generalizable node representations for downstream inference. The key challenge lies in aligning semantic and structural information, as LLMs excel at textual understanding while GNNs specialize in structural modeling (Li et al., 2023b; Ren et al., 2024). To bridge this gap, we propose prompt-based GNNs that prompt LLMs to replicate the general GNN workflow in text space, which requires appropriate prompts guided by three factors: **(a) Graph Structure.** How to simulate message passing to encode structural dependencies? **(b) Node Semantics.** How to distill essential semantics into concise yet informative representations? **(c) Graph Description.** How to convey graph semantics and structure to LLMs simultaneously?

As illustrated in Figure 2, we implement a fine-grained replication of a general GNN workflow, i.e., the **prompt-based GNN**. We first initialize node representations by summarizing textual attributes, analogous to the embedding lookup in conventional GNNs. To reduce computational overhead, we sample one-hop neighbors as follows:

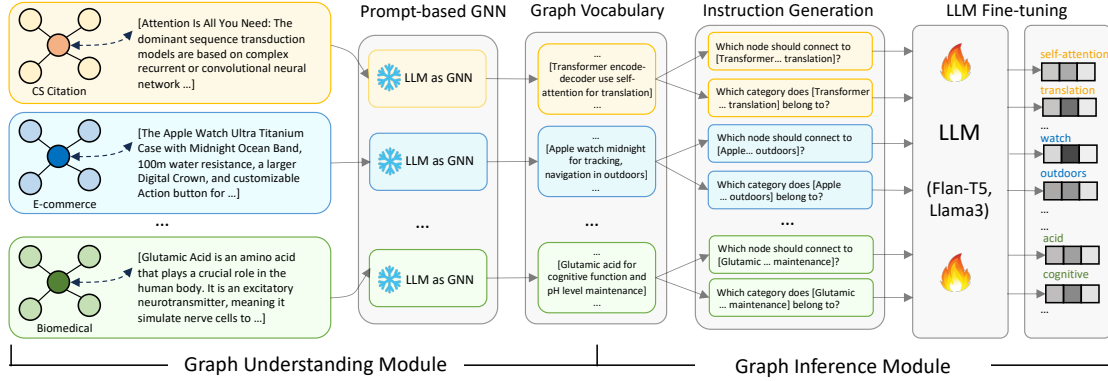
$$\{X_j^{(l-1)}, v_j \in \mathcal{N}_i^{(l)}\} \leftarrow \text{Prompt}_{\text{sample}}(X^{(l-1)}, \mathcal{N}_i) \quad (3)$$

where  $X^{(l-1)} = \{X_0^{(l-1)}, X_1^{(l-1)}, \dots, X_{|V|-1}^{(l-1)}\}$  are the textual representations of all nodes from the previous layer, and  $\mathcal{N}_i^{(l)} \subseteq \mathcal{N}_i$  denotes the neighbors sampled for  $v_i$  at the  $l$ -th layer. Then, we directly prompt LLMs to perform message aggregation and update for each node as follows:

$$X_i^{(l)} \leftarrow \text{Prompt}_{\text{agg-update}}(\{X_j^{(l-1)}, v_j \in \mathcal{N}_i^{(l)}\}, X_i^{(l-1)}), \quad (4)$$

where we seek a flexible message-passing mechanism without relying on predefined operators such as mean and weighted aggregators. We analyze its sensitivity to input order in Appendix E.1 and discuss prompt length constraints in Appendix G. In terms of optimization, we follow unsupervised graph learning and adopt a prompt-driven contrastive objective, which encourages similarity

Figure 3: The pipeline of PromptGFM. (a) Graph Understanding Module: For arbitrary TAGs from different domains, prompt-based GNN replicates traditional embedding-based GNN workflow in the text space, generating compact node representations. (b) Graph Inference Module: We establish a unified graph vocabulary and extract language-based IDs to generate massive pure-language prompts, enabling LLM fine-tuning across graphs and tasks to support knowledge transfer and flexible adaptability.



among neighboring nodes while pushing apart distant ones, serving as a soft structural prior that guides the LLM to filter and integrate informative neighbor signals. As a task-agnostic approach to graph representation learning, contrastive optimization naturally aligns with the generality required by GFM. While traditional negative sampling becomes redundant in our setting, we express this objective as natural-language instructions in the prompt, without introducing an explicit training loss. Prompt details are provided in Appendix D.

After repeating  $L$  rounds, we obtain the final textual representation as  $T_i = X_i^{(L)}$ , which encodes both semantic and structural information and addresses the aforementioned challenges: **(a) Graph Structure.** We iteratively invoke LLMs with shared prompt templates, feeding each round’s output into the next to capture higher-order dependencies, and trace this process for individual nodes in Appendix J. **(b) Node Semantics.** Simultaneously, we explicitly instruct LLMs to produce concise yet expressive node representations, gradually refining them into denser and richer semantic descriptions. **(c) Graph Description.** One-hop neighbor descriptions collectively encode the adjacency structure, representing the entire graph while remaining efficient (§3.3). We further investigate layer-wise representation discrimination, semantic retention, and module performance in Appendix E. This workflow-level replication in text space exemplifies the potential of treating LLMs as GNNs for GNN-LLM integration and graph-text alignment.

### 3.2 Graph Inference Module

The graph inference module seeks to unify diverse graphs and tasks and acquire transferable

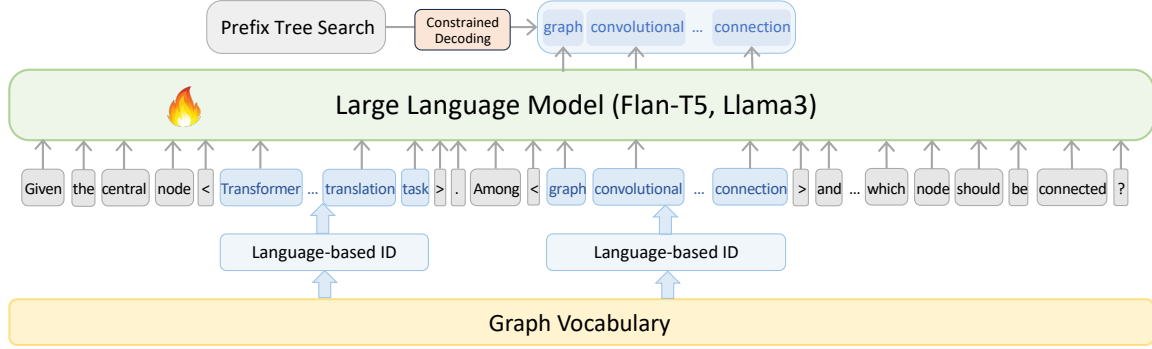
knowledge through multi-instruction LLM fine-tuning. Existing methods, which represent nodes as OOV tokens, create semantic misalignment between ID-based and language-based embeddings in task-oriented prompts due to distinct vocabularies, and substantially limit dataset-level transferability and task-level adaptability (Tang et al., 2024; Xia et al., 2023; Ye et al., 2024). To overcome this challenge, we introduce a language-based graph vocabulary that bridges this incompatibility, enabling the generation of large-scale, coherent, and readable instructions to support effective LLM inference.

**Graph Vocabulary.** A transferable graph vocabulary, whose fundamental units can represent each node, is essential for effective GFMs and should satisfy three key criteria: **expressiveness**, **transferability**, and **scalability**. Since each node has been equipped with a textual representation encoding its core semantics and local structure, we aggregate these representations to build a universal language-based graph vocabulary compatible with the LLM’s native vocabulary. In this vocabulary, each node is then represented by a finite sequence of language tokens, called a **language-based ID**, defined as:

$$\mathcal{F} : V \rightarrow T, \quad (5)$$

where each node  $v_i$  is mapped to a sequence  $T_i = (t_i^1, t_i^2, \dots, t_i^{m_i})$ , with  $t_i^k \in \mathcal{X}$  and  $\mathcal{X}$  denoting the dictionary of natural language tokens. In practice,  $T_i$  is simply the tokenized form of the final textual representation produced by the graph understanding module, requiring no additional summarization, clustering, or remapping. Collectively, the language-based IDs of all nodes in  $V$  are denoted as  $T = \{T_0, T_1, \dots, T_{|V|-1}\}$ . We analyze vocabulary coverage and uniqueness in Appendix F.

Figure 4: An instance of graph inference module in link prediction, where language-based IDs are indexed from the graph vocabulary to generate readable instructions with task-oriented templates. A multi-instruction fine-tuning framework is employed to unify graphs and tasks and learn transferable knowledge for GFM.



Along this line, the structured token sequence of a graph node functions like a word in a standard lexicon, preserving semantic fidelity and eliminating the OOV issue. Overall, our graph vocabulary satisfies the desired criteria: **(a) Expressiveness.** Language-based IDs preserve rich semantic and structural information from open-world settings. **(b) Transferability.** Like human lexicons, the vocabulary is grounded in natural language, providing a shared foundation that ensures cross-graph and cross-task interpretability and transferability. **(c) Scalability.** Language-based IDs can be generated on the fly for any node, whether previously seen or not, and remain compatible with existing ones, avoiding token explosion.

**Multi-Instruction Fine-Tuning.** As shown in Figure 4, we use multi-instruction fine-tuning to integrate diverse graphs and tasks (Chung et al., 2024; Wei et al., 2022). We index nodes from the vocabulary and insert their language-based IDs into task-oriented templates to construct instructions  $\mathcal{T}$ :

$$\mathcal{T} \leftarrow \text{Prompt}_{\text{template}}(T, G), \quad (6)$$

which include the language-based IDs of central nodes and their task-specific local structures from  $G$  (e.g., for node classification or link prediction). These instructions are fully readable and coherent as they are entirely composed of natural language tokens. Thus, graph-centric tasks can be naturally reformulated into a unified text-to-text paradigm for LLM fine-tuning (Mishra et al., 2022). Given a target sequence  $Y$ , the loss is computed as:

$$\mathcal{L} = - \sum_{s=1}^{|Y|} \log \Pr(y_s | \mathcal{T}, Y_{<s}), \quad (7)$$

where  $\Pr(y_s | \mathcal{T}, Y_{<s})$  denotes the probability of the  $s$ -th output token  $y_s$ , conditioned on

the instruction  $\mathcal{T}$  and previous tokens  $Y_{<s} = (y_1, y_2, \dots, y_{s-1})$ . To support broad generalization, we fine-tune LLM backbones such as Flan-T5 (Chung et al., 2024) and Llama (Touvron et al., 2023) to co-train a unified open-world knowledge model with cross-graph and cross-task generalization. As a result, we not only facilitate robust knowledge transfer across diverse unseen graphs and tasks (§4.2), but also exhibit adaptability to accommodate auxiliary pre-training tasks (§4.3).

#### Constrained Decoding via Prefix Tree Search.

To reduce LLM hallucination in generative tasks, we use constrained decoding with prefix tree search (De Cao et al., 2021; Tan et al., 2024). Taking generative link prediction as an example, we build a prefix tree from the language-based IDs of all candidate nodes, where each tree node represents a natural language token and each root-to-leaf path denotes a valid node ID. During autoregressive generation, we restrict token selection to valid prefix-tree paths, ensuring that predictions correspond to actual graph nodes and preventing hallucinated outputs. This effectiveness can be attributed to the discrete and compositional nature of language-based IDs, further highlighting the flexibility of our proposed graph vocabulary.

### 3.3 Computational Complexity

**Graph Understanding Module.** Although PromptGFM incurs an initial API cost during graph understanding, this cost is a one-time overhead and remains manageable with modern infrastructure. Each graph is processed once to generate language-based IDs, which are then cached and reused throughout inference. When new graphs are introduced, existing IDs remain unchanged, and only the new graphs require ID generation. Subsequent LLM fine-tuning over the shared vocabulary

Table 1: Evaluation results (%) on node classification accuracy ( $\uparrow$ ). We highlight the **first** and **second** best results, and report the statistical significance over the strongest baseline using a two-tailed  $t$ -test.

| Method                | Cora         | Citeseer     | PubMed       | Ogbn-arxiv   | History      | Photo        | WikiCS       |
|-----------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| MLP                   | 62.29        | 64.42        | 62.88        | 62.07        | 64.62        | 61.21        | 68.41        |
| GCN                   | 82.47        | 76.11        | 77.36        | 66.15        | 81.93        | 78.58        | 76.33        |
| GAT                   | 82.92        | 77.30        | 74.36        | 65.29        | 82.85        | 82.38        | 78.21        |
| GraphSAGE             | 83.69        | 73.17        | 83.22        | 68.78        | 82.12        | 80.06        | 79.56        |
| RevGNN                | 86.90        | 77.34        | 82.16        | 70.43        | <b>83.04</b> | 83.24        | 81.22        |
| SGFormer              | 82.36        | 73.76        | 78.92        | 63.44        | 78.98        | 80.12        | 76.56        |
| NodeFormer            | 81.55        | 72.98        | 76.49        | 73.21        | 79.60        | 78.51        | 75.47        |
| BERT                  | 79.02        | 72.83        | 76.74        | 71.90        | 72.97        | 68.82        | 77.98        |
| Sentence-BERT         | 78.09        | 72.12        | 75.48        | 77.24        | 74.10        | 69.02        | 77.72        |
| Llama3                | 78.17        | 74.68        | 77.46        | 77.65        | 76.49        | 74.42        | 79.94        |
| OFA                   | 75.72        | 71.58        | 75.26        | 74.68        | 81.43        | 84.46        | 78.02        |
| LLaGA                 | 81.25        | 68.80        | 86.54        | 76.05        | 82.55        | 85.34        | 80.74        |
| ENGINE                | 91.48        | 78.46        | 90.24        | 76.02        | 82.46        | 83.75        | <b>81.56</b> |
| GraphPrompter         | 80.26        | 73.61        | <b>94.80</b> | 75.61        | 79.42        | 80.04        | 80.98        |
| GFT                   | 79.88        | 71.45        | 80.21        | 73.11        | 75.58        | 76.24        | 81.02        |
| GraphText             | 88.23        | 76.64        | 85.19        | 76.80        | 81.47        | 82.26        | 81.49        |
| PromptGFM (Flan-T5)   | <b>91.72</b> | <b>84.49</b> | 92.83        | <b>80.58</b> | 82.33        | <b>85.41</b> | 81.49        |
| PromptGFM (Llama3-8B) | <b>92.42</b> | <b>85.32</b> | <b>94.65</b> | <b>83.78</b> | <b>86.72</b> | <b>86.61</b> | <b>84.66</b> |
| $p$ -value            | 4e-3         | 2.73e-8      | 3.21e-7      | 4.48e-10     | 2.64e-6      | 3.62e-2      | 9.93e-4      |

Table 2: Accuracy (%) on discriminative link prediction.

| Method         | Discriminative: Accuracy ( $\uparrow$ ) |              |              |              |
|----------------|-----------------------------------------|--------------|--------------|--------------|
|                | Cora                                    | Citeseer     | Ogbn-arxiv   | PubMed       |
| GCN            | 77.15                                   | 78.72        | 80.89        | 77.36        |
| GAT            | 70.44                                   | 77.17        | 76.25        | 74.36        |
| GraphSAGE      | 85.31                                   | 87.15        | 80.76        | 83.22        |
| GraphPrompter  | 90.10                                   | 91.67        | 73.21        | 80.49        |
| Ours (Flan-T5) | <b>90.57</b>                            | <b>92.03</b> | <b>81.12</b> | <b>87.64</b> |
| Ours (Llama3)  | <b>91.68</b>                            | <b>93.46</b> | <b>84.27</b> | <b>88.12</b> |

then incorporates the features of these new graphs. **Graph Inference Module.** With IDs cached, fine-tuning and decoding operate entirely over ID tokens. At inference time, each decoding step costs  $O(\bar{m})$ , where  $\bar{m}$  is the average length of a language-based ID. With prefix tree search constraining the candidate space to the  $|V|$  nodes of the graph-specific vocabulary, the overall decoding complexity is bounded by  $O(|V| \times \bar{m})$ . Thus, these properties keep the computational overhead manageable for large-scale graphs. An empirical efficiency study is provided in §4.3.

## 4 Experiments

In this section, we conduct extensive experiments to address these research questions (RQs): **RQ1:** How does PromptGFM perform on node classification and link prediction? **RQ2:** How does PromptGFM transfer across diverse graphs and tasks as a general-purpose GFM? **RQ3:** How does PromptGFM adapt to more auxiliary pre-training tasks? **RQ4:** How does each module contribute

Table 3: HR@1 (%) on generative link prediction.

| Method       | Generative: HR@1 ( $\uparrow$ ) |             |             |
|--------------|---------------------------------|-------------|-------------|
|              | Cora                            | Citeseer    | PubMed      |
| GCN          | 5.95                            | 6.82        | 0.51        |
| GAT          | 2.22                            | 3.59        | 0.28        |
| GraphSAGE    | 6.59                            | 8.73        | 0.45        |
| InstructGLM  | 6.65                            | 7.13        | 0.58        |
| KGT5-context | 8.03                            | 8.41        | 1.17        |
| PromptGFM    | <b>8.21</b>                     | <b>8.90</b> | <b>1.21</b> |

to the overall performance? **RQ5:** What is the computational efficiency in our graph understanding and inference modules? We evaluate PromptGFM on seven TAG datasets from computer science, e-commerce, and biomedical domains. We compare against graph-agnostic, GNN-based, LM-only, GNN-LLM integration, and GFM baselines, and report node classification and link prediction performance under consistent evaluation protocols. Dataset statistics are summarized in Appendix A, baseline details in Appendix B, and reproduction settings in Appendix C.

### 4.1 Main Performance Comparison (RQ1)

**Main Results.** We train each model from scratch on a single graph and report NC and LP accuracy in Tables 1 and 2. PromptGFM substantially outperforms state-of-the-art baselines, leading to the following observations. (a) Graph-based models consistently outperform graph-agnostic ones, confirming the importance of structural information. (b) PromptGFM surpasses OFA and ENGINE on node classification, as well as GraphPrompter on link prediction, suggesting that decoupled GNN-for-

Table 4: Intra-domain cross-graph transferability to Citeseer within the computer science domain.

| Source |       | Pre-training |              | Co-training  |              |
|--------|-------|--------------|--------------|--------------|--------------|
| Cora   | Arxiv | Acc(↑)       | M-F1(↑)      | Acc(↑)       | M-F1(↑)      |
| ✗      | ✗     | 27.64        | 17.10        | 84.49        | 82.31        |
| ✓      | ✗     | 51.63        | 45.10        | 84.96        | 83.22        |
| ✗      | ✓     | <b>60.34</b> | <b>54.81</b> | <b>85.45</b> | <b>83.91</b> |
| ✓      | ✓     | <b>61.25</b> | <b>55.66</b> | <b>86.77</b> | <b>84.24</b> |

Table 5: Inter-domain cross-graph transferability from e-commerce to biomedical domain.

| Source  |       | Pre-training |              | Co-training  |              |
|---------|-------|--------------|--------------|--------------|--------------|
| History | Photo | Acc(↑)       | M-F1(↑)      | Acc(↑)       | M-F1(↑)      |
| ✗       | ✗     | 39.12        | 39.84        | <b>90.67</b> | <b>91.82</b> |
| ✓       | ✗     | <b>46.24</b> | <b>47.41</b> | <b>88.52</b> | <b>89.39</b> |
| ✗       | ✓     | 44.21        | 44.93        | 85.34        | 85.57        |
| ✓       | ✓     | <b>45.18</b> | <b>45.65</b> | 86.22        | 87.03        |

LLM and LLM-for-GNN architectures have limited graph-text alignment. (c) PromptGFM also outperforms LLaGA, indicating that heuristic prompting alone is insufficient to capture higher-order dependencies without a true GNN mechanism, whereas our prompt-based GNN performs explicit message passing. (d) PromptGFM also surpasses the vocabulary-based GFT and the text-space Graph-Text, which respectively lack a language-grounded vocabulary and an iterative aggregation-update process, showing that the two components of our design are complementary. (e) Notably, the gains also hold on History and Photo, whose co-purchase and co-view edges relate items behaviorally rather than lexically, indicating that PromptGFM selectively integrates informative neighbor signals and generalizes beyond lexically homophilous graphs.

**Generative Link Prediction.** Following the transductive setting, we partition the graph by links, construct the input graph from training edges, and predict missing connections for test nodes with existing nodes as candidates. As shown in Table 3, PromptGFM consistently outperforms both traditional GNNs and generative sequence-to-sequence baselines (Ye et al., 2024; Kochsiek et al., 2023) on generative link prediction. Existing GNN-LLM methods fail on this task because their OOV token embeddings cannot be directly mapped from LLM outputs to specific nodes, often causing hallucinated predictions. In contrast, PromptGFM represents nodes as finite token sequences and uses prefix-tree constrained decoding to regulate LLM outputs. This highlights the importance of graph vocabulary and the flexibility of PromptGFM in

Table 6: Inter-domain cross-graph transferability from computer science to biomedical domain.

| Source |          |       | Pre-training |              | Co-training  |              |
|--------|----------|-------|--------------|--------------|--------------|--------------|
| Cora   | Citeseer | Arxiv | Acc(↑)       | M-F1(↑)      | Acc(↑)       | M-F1(↑)      |
| ✗      | ✗        | ✗     | 39.12        | 39.84        | <b>90.67</b> | <b>91.82</b> |
| ✓      | ✗        | ✗     | 51.76        | 52.84        | <b>86.34</b> | <b>87.29</b> |
| ✗      | ✓        | ✗     | 40.12        | 42.38        | 85.21        | 86.12        |
| ✗      | ✗        | ✓     | <b>60.21</b> | <b>62.02</b> | 82.17        | 86.01        |
| ✓      | ✓        | ✗     | 50.17        | 51.74        | 84.39        | 81.94        |
| ✓      | ✗        | ✓     | <b>57.28</b> | <b>59.71</b> | 81.32        | 83.14        |
| ✗      | ✓        | ✓     | 55.34        | 57.11        | 82.13        | 83.10        |
| ✓      | ✓        | ✓     | 53.07        | 54.90        | 80.43        | 80.79        |

Table 7: Cross-task generalization between node classification (NC) and link prediction LP. *ST-X* refers to supervised training on the task X from scratch.

| Task | Setting          | Cora         | Citeseer     | PubMed       |
|------|------------------|--------------|--------------|--------------|
| NC   | <i>zero-shot</i> | 18.54        | 27.64        | 39.12        |
|      | <i>LP→NC</i>     | 60.74        | 50.12        | 57.42        |
|      | <i>ST-NC</i>     | <b>91.72</b> | <b>84.49</b> | <b>92.83</b> |
| LP   | <i>zero-shot</i> | 67.28        | 69.22        | 73.59        |
|      | <i>NC→LP</i>     | 78.16        | 79.23        | 84.11        |
|      | <i>ST-LP</i>     | <b>91.68</b> | <b>93.46</b> | <b>88.12</b> |

generative scenarios.

## 4.2 Cross-graph and Cross-task Transfer (RQ2)

We evaluate cross-graph transferability under both **intra-domain** and **inter-domain** scenarios. Following prior work (Chen et al., 2024b), we consider two settings: (a) **Pre-training**, where models are evaluated on unseen datasets not used during training, and (b) **Co-training**, where part of the target dataset is jointly trained with auxiliary datasets. These two dimensions define four transfer paradigms for a comprehensive assessment of cross-graph transferability.

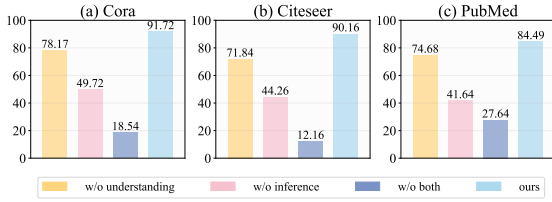
**Intra-domain Cross-graph Transferability.** As shown in Table 4, all settings outperform direct inference with an off-the-shelf LLM. Incorporating Arxiv brings larger gains than Cora, suggesting that larger and richer datasets improve transferability more effectively. In the *pre-training* setting, the substantial gains over direct inference indicate strong in-domain zero-shot transferability. The *co-training* setting shows a similar but smaller trend, further validating the effectiveness of intra-domain knowledge transfer. These results highlight the value of collecting more relevant graph data to build more knowledgeable and transferable GFMs.

**Inter-domain Cross-graph Transferability.** We report transfer results from multiple domains to the biomedical domain in Tables 5 and 6. In the

Table 8: The impact of incorporating Node Discrimination (+ND) and Link Discrimination (+LD) as pre-training tasks on NC performance.

| Dataset  | +ND | +LD | Accuracy (%)         | Macro-F1 (%)         |
|----------|-----|-----|----------------------|----------------------|
| Citeseer | ✗   | ✗   | 84.49                | 82.31                |
|          | ✓   | ✗   | 85.67 (↑1.18)        | 83.22 (↑0.91)        |
|          | ✗   | ✓   | 84.73 (↑0.24)        | 82.65 (↑0.34)        |
|          | ✓   | ✓   | <b>86.02 (↑1.53)</b> | <b>83.77 (↑1.46)</b> |
| PubMed   | ✗   | ✗   | 90.67                | 91.82                |
|          | ✓   | ✗   | 92.53 (↑1.86)        | 92.84 (↑1.02)        |
|          | ✗   | ✓   | 91.12 (↑0.45)        | 92.03 (↑0.21)        |
|          | ✓   | ✓   | <b>93.06 (↑2.39)</b> | <b>93.27 (↑1.45)</b> |

Figure 5: Ablation studies on node classification.



*pre-training* setting, adding a single external graph consistently improves performance, whereas multiple source graphs may slightly degrade target-domain performance. More importantly, in the supervised *co-training* setting, performance can deteriorate as more cross-domain data is introduced. This is mainly due to domain shifts, including structural differences, semantic drift, and mismatched label spaces (Chen et al., 2024b; Zhao et al., 2024). Nonetheless, PromptGFM adopts a unified language-based vocabulary and pure language interface, which in principle enables natural cross-domain transferability and avoids token collision. We discuss this community-wide challenge and potential solutions in Appendix H.

**Cross-task Transferability.** We present transfer performance between LP and NC in Table 7. As expected,  $LP \rightarrow NC$  generally surpasses *zero-shot*, showing successful transfer to unseen tasks, but remains far below  $ST-NC$ , indicating that cross-task transfer is harder than cross-graph generalization. A similar trend appears for LP:  $NC \rightarrow LP$  outperforms *zero-shot* but still trails  $ST-LP$ , suggesting that PromptGFM transfers task-agnostic structural knowledge while task-specific supervision remains important. Overall, these results demonstrate our cross-graph and cross-task generalization capability, establishing a task-adaptive GFM.

### 4.3 Exploration Studies (RQ3~5)

**Adding Auxiliary Pre-training Tasks (RQ3).** A robust foundation model should support diverse pre-training tasks to improve downstream gener-

alization. Beyond NC and LP, we introduce two additional tasks: **(a) Node Discrimination (ND):** Given three nodes  $[a]$ ,  $[b]$ ,  $[c]$ , exactly one belongs to a different category. Which one is it? and **(b) Link Discrimination (LD):** Given a central node  $[X]$ , which candidate node  $[a]$ ,  $[b]$ ,  $[c]$  is not connected to  $[X]$ ?, where bracketed symbols denote nodes’ language-based IDs. We incorporate ND and LD individually or jointly into multi-task fine-tuning and evaluate their effects on NC under identical settings. As shown in Table 8, adding either ND or LD improves both accuracy and M-F1, while combining them achieves the best performance. These results show that auxiliary tasks enrich node representations and improve target-task performance. Thus, PromptGFM goes beyond memorizing task patterns and leverages its unified language interface to exhibit stronger cross-task generalization.

**Ablation Studies (RQ4).** We construct three ablated variants: **(a) w/o understanding**, which removes the prompt-based GNN and instead uses the summarized central-node text from the initialization prompt as the language-based ID; **(b) w/o inference**, which removes multi-instruction fine-tuning; and **(c) w/o both**, which directly prompts LLMs with raw textual attributes. Figure 5 shows NC accuracy, where the full model outperforms all variants. The drop in *w/o understanding* highlights the importance of replicating the GNN workflow to preserve semantic and structural signals. The larger decline in *w/o inference* emphasizes the necessity of LLM fine-tuning for incorporating domain-specific knowledge. Finally, *w/o both* performs worst, confirming the complementarity between graph understanding and inference.

**Controlled Neighbor Study.** To further isolate the contribution of the graph understanding module, we construct three controlled variants that replace the sampled neighbors while keeping the same backbone, prompt template, token budget, and padding strategy: **(a) central text only**, which removes neighbor context; **(b) semantic non-neighbors**, which uses Sentence-BERT-nearest disconnected nodes; and **(c) degree-preserving rewired**, which samples neighbors from a degree-preserving rewired graph. Table 9 reports NC accuracy and M-F1, where true neighbors consistently outperform all variants. The drop in *central text only* shows the benefit of neighbor context beyond the central node. The remaining gap in *semantic non-neighbors* indicates that semantic related-

Table 9: Controlled neighbor study on node classification, reporting accuracy and Macro-F1 (%) results.

| Variant                    | Cora         |              | Citeseer     |              |
|----------------------------|--------------|--------------|--------------|--------------|
|                            | Acc          | M-F1         | Acc          | M-F1         |
| Central text only          | 78.17        | 75.44        | 74.68        | 71.19        |
| Semantic non-neighbors     | 82.54        | 81.27        | 75.24        | 72.36        |
| Degree-preserving rewired  | 61.13        | 58.71        | 62.48        | 60.04        |
| PromptGFM (true neighbors) | <b>91.72</b> | <b>90.06</b> | <b>84.49</b> | <b>80.13</b> |

ness cannot substitute for graph topology. Finally, *degree-preserving rewired* performs worst, even below *central text only* despite the same amount of neighbor text, confirming that our prompt-based GNN relies on the given topology.

**Efficiency Analysis (RQ5).** We evaluate the computational and monetary cost of PromptGFM, as summarized in Table 10. In the graph understanding module, the API and token cost grow with the graph size, but this constitutes a one-time, graph-specific overhead that can be cached and reused throughout the inference pipeline (denoted as “n/a”). On the largest graph, Ogbn-arxiv, this one-time graph understanding stage costs \$2.69 and 17.91M tokens (about 176 minutes and 1.41 kWh), which is amortized by ID caching. During inference, as more graphs are introduced and the overall vocabulary expands, we observe that time and energy consumption increase in an approximately linear manner. These results align with our discussion in §3.3 and demonstrate PromptGFM’s efficiency as a scalable GFM.

## 5 Related Works

**GNN-LLM Integration.** Motivated by the potential of LLMs for graph learning, existing GNN-LLM integration methods for TAGs fall into three paradigms: **(a) GNN for LLM**, where GNNs or graph transformers act as structure tokenizers to enhance LLM understanding (Tang et al., 2024; Chai et al., 2023; Liu et al., 2024b); **(b) LLM for GNN**, where LLMs generate labels or features to supervise GNN training (Chen et al., 2024c; Liu et al., 2024a; Zhu et al., 2024; He et al., 2024; Guo et al., 2024); and **(c) LLM as GNN**, where LLMs consume structured prompts but lack an intrinsic GNN mechanism (Chen et al., 2024a; Ye et al., 2024; Wang et al., 2024a; Zhao et al., 2023b). All paradigms rely on decoupled two-stage alignment and share a common co-training challenge, motivating our unified LLM-as-GNN paradigm.

**Graph-Text Alignment.** Existing methods align graph and text representations through graph-

Table 10: Efficiency analysis in terms of API cost (USD), number of tokens (M), inference time (min), and power consumption (kWh).

| Source |          |       | Understanding |          | Inference |       |
|--------|----------|-------|---------------|----------|-----------|-------|
| Cora   | Citeseer | Photo | API Cost      | # Tokens | Time      | Power |
| ✓      | ✗        | ✗     | 0.188         | 1.25     | 16.47     | 0.07  |
| ✗      | ✓        | ✗     | 0.189         | 1.29     | 21.41     | 0.08  |
| ✗      | ✗        | ✓     | 1.213         | 8.08     | 83.67     | 0.34  |
| ✓      | ✓        | ✗     | n/a           | n/a      | 35.31     | 0.16  |
| ✓      | ✓        | ✓     | n/a           | n/a      | 117.56    | 1.04  |

encoder prefixes (Huang et al., 2023, 2024), contrastive learning (Tang et al., 2024; Zhu et al., 2025; Li et al., 2023a; Brannon et al., 2023), iterative training (Zhu et al., 2024; Zhao et al., 2023a), knowledge distillation (Mavromatis et al., 2023), or structure-aware self-supervised learning (Zhang et al., 2025; Liu et al., 2025; Yin et al., 2025). However, they still suffer from a modality gap that limits transferability. Since TAGs already contain textual information, we shift graph-text alignment entirely into text space in this work.

**Graph Foundation Models.** GFMs aim to transfer across diverse datasets and tasks, with a central challenge being the identification of a graph vocabulary of transferable units (Sun et al., 2025a; Fang et al., 2025; Liu et al., 2023). Existing approaches rely on domain-specific vocabularies: GraphGPT (Tang et al., 2024) assigns unique IDs to each dataset, MoleBERT (Xia et al., 2023) encodes molecular structures, and GFT (Wang et al., 2024b) quantizes computation trees into a discrete tree vocabulary. However, such vocabularies do not readily transfer across domains. While prior work has represented and reasoned over graphs in natural language (Fatemi et al., 2024; Wang et al., 2023), natural language has yet to be explored as a transferable graph vocabulary.

## 6 Conclusion

We present PromptGFM, a graph foundation model for text-attributed graphs using graph vocabulary learning. By replicating the GNN workflow in text space, PromptGFM decouples refined textual node representations into a unified vocabulary, preserving graph-text semantic fidelity while avoiding out-of-vocabulary issues. This vocabulary improves compatibility and scalability, enabling effective LLM fine-tuning with readable instructions and stronger transfer. Experiments show superior performance, cross-graph and cross-task generalization, and adaptability to auxiliary pre-training tasks, highlighting the potential of LLMs as GNNs and new directions for graph foundation models.

## Limitations

We discuss several aspects of the current design that warrant further exploration.

### (1) Closed-source LLM for Node Initialization.

Our graph understanding module uses GPT-4o mini to generate initial node summaries. Because the resulting language-based IDs are cached and reused, the associated API cost is bounded and amortized over subsequent use. Nevertheless, replacing the closed-source model with capable open-source alternatives could further improve the accessibility and reproducibility of our framework.

**(2) Prompt Length for Dense Graphs.** For graphs with highly skewed degree distributions, one-hop prompts for high-degree nodes can become lengthy. In our experiments, neighbor sampling and a fixed neighbor cap, as described in Appendix G, effectively bound the prompt length. For substantially larger or denser graphs, additional techniques such as hierarchical neighborhood summarization could further improve scalability.

**(3) Dynamic Graphs.** Our current evaluation mainly focuses on static graphs, where language-based IDs are generated once and then reused. For dynamic graphs, we introduce a localized update strategy that regenerates representations only for affected nodes, with preliminary results reported in Appendix I. A more systematic investigation of large-scale streaming graphs and frequent updates remains future work.

**(4) Cross-domain Transferability.** Our language-based vocabulary naturally supports knowledge sharing across domains. However, the adaptive prompting experiments in Appendix H show that transfer between highly different source and target domains remains challenging, especially under co-training. We view robust cross-domain alignment as an open research direction for graph foundation models more broadly, and adaptive prompting offers a promising step toward it.

## Ethical Considerations

**Data.** All datasets are publicly released academic benchmarks (Cora, Citeseer, PubMed, Ogbn-arxiv, WikiCS, Amazon Photo/History). They contain no personally identifiable information. We follow the licenses and intended uses of each dataset.

**Models.** We build on public LLM backbones (Flan-T5, Llama3) and call OpenAI’s GPT-4o mini for the one-time graph understanding step. Outputs of LLMs may reflect biases inherited from their pre-

training corpora, and the language-based IDs we derive inherit this risk. For sensitive deployments (e.g., recommender systems, biomedical decision support), downstream calibration and human review are warranted.

**Compute and Energy.** We report API cost, token count, inference time, and power consumption in Table 10. Because language-based IDs are cached and reused, the dominant cost is one-time per graph, which limits the marginal carbon footprint of running PromptGFM at deployment.

**AI Use Disclosure.** AI assistants were used solely for language polishing, including grammar correction and readability improvement. All research ideas, methods, experiments, analyses, and conclusions are the authors’ own, and the authors take full responsibility for the content.

## References

- William Brannon, Wonjune Kang, Suyash Fulay, Hang Jiang, Brandon Roy, Deb Roy, and Jad Kabbara. 2023. ConGraT: Self-supervised contrastive pre-training for joint graph and text embeddings. *CoRR*, abs/2305.14321.
- Peter Buneman, Dennis Dosso, Matteo Lissandrini, and Gianmaria Silvello. 2021. Data citation and the citation graph. *Quant. Sci. Stud.*, 2(4).
- Yongqiang Cai. 2024. Vocabulary for universal approximation: A linguistic perspective of mapping compositions. In *ICML 2024*.
- Ziwei Chai, Tianjie Zhang, Liang Wu, Kaiqiang Han, Xiaohai Hu, Xuanwen Huang, and Yang Yang. 2023. GraphLLM: Boosting graph reasoning ability of large language model. *CoRR*, abs/2310.05845.
- Runjin Chen, Tong Zhao, Ajay Jaiswal, Neil Shah, and Zhangyang Wang. 2024a. LLaGA: Large language and graph assistant. In *ICML 2024*.
- Zhikai Chen, Haitao Mao, Jingzhe Liu, Yu Song, Bingheng Li, Wei Jin, Bahare Fatemi, Anton Tsitsulin, Bryan Perozzi, Hui Liu, and Jiliang Tang. 2024b. Text-space graph foundation models: Comprehensive benchmarks and new insights. In *NeurIPS 2024*.
- Zhikai Chen, Haitao Mao, Hongzhi Wen, Haoyu Han, Wei Jin, Haiyang Zhang, Hui Liu, and Jiliang Tang. 2024c. Label-free node classification on graphs with large language models (LLMs). In *ICLR 2024*.
- Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex

- Castro-Ros, Marie Pellat, Kevin Robinson, and 16 others. 2024. Scaling instruction-finetuned language models. *J. Mach. Learn. Res.*, 25.
- Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *ICLR 2021*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019*, pages 4171–4186.
- Norman Di Palo, Arunkumar Byravan, Leonard Hasenclever, Markus Wulfmeier, Nicolas Heess, and Martin Riedmiller. 2023. Towards A unified agent with foundation models. *CoRR*, abs/2307.09668.
- Masaki Eto. 2019. Extended co-citation search: Graph-based document retrieval on a co-citation network containing citation context information. *Inf. Process. Manag.*, 56(6).
- Yi Fang, Dongzhe Fan, Sirui Ding, Ninghao Liu, and Qiaoyu Tan. 2025. UniGLM: Training one unified language model for text-attributed graphs embedding. In *WSDM 2025*.
- Bahare Fatemi, Jonathan Halcrow, and Bryan Perozzi. 2024. Talk like a graph: Encoding graphs for large language models. In *ICLR 2024*.
- C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. 1998. CiteSeer: An automatic citation indexing system. In *ACM DL 1998*.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, and 1 others. 2024. The Llama 3 herd of models. *CoRR*, abs/2407.21783.
- Daniele Grattarola, Daniele Zambon, Filippo Maria Bianchi, and Cesare Alippi. 2024. Understanding pooling in graph neural networks. *IEEE Trans. Neural Networks Learn. Syst.*, 35(2).
- Zirui Guo, Lianghao Xia, Yanhua Yu, Yuling Wang, Zixuan Yang, Wei Wei, Liang Pang, Tat-Seng Chua, and Chao Huang. 2024. GraphEdit: Large language models for graph structure learning. *CoRR*, abs/2402.15183.
- William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NeurIPS 2017*.
- Xiaoxin He, Xavier Bresson, Thomas Laurent, Adam Perold, Yann LeCun, and Bryan Hooi. 2024. Harnessing explanations: LLM-to-LM interpreter for enhanced text-attributed graph representation learning. In *ICLR 2024*.
- Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. In *NeurIPS 2020*.
- Xuanwen Huang, Kaiqiao Han, Dezheng Bao, Quanjin Tao, Zhisheng Zhang, Yang Yang, and Qi Zhu. 2023. Prompt-based node feature extractor for few-shot learning on text-attributed graphs. *CoRR*, abs/2309.02848.
- Xuanwen Huang, Kaiqiao Han, Yang Yang, Dezheng Bao, Quanjin Tao, Ziwei Chai, and Qi Zhu. 2024. Can GNN be good adapter for LLMs? In *WWW 2024*.
- Mingyu Jin, Haochen Xue, Zhenting Wang, Boming Kang, Ruosong Ye, Kaixiong Zhou, Mengnan Du, and Yongfeng Zhang. 2024. ProLLM: Protein chain-of-thoughts enhanced LLM for protein-protein interaction prediction. *CoRR*, abs/2405.06649.
- David Kempe, Jon M. Kleinberg, and Éva Tardos. 2003. Maximizing the spread of influence through a social network. In *KDD 2003*.
- Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR 2017*.
- Adrian Kochsiek, Apoorv Saxena, Inderjeet Nair, and Rainer Gemulla. 2023. Friendly neighbors: Contextualized sequence-to-sequence link prediction. In *RepL4NLP 2023*, pages 131–138.
- Guohao Li, Matthias Müller, Bernard Ghanem, and Vladlen Koltun. 2021. Training graph neural networks with 1000 layers. In *ICML 2021*.
- Yichuan Li, Kaize Ding, and Kyumin Lee. 2023a. GRENADE: graph-centric language model for self-supervised representation learning on text-attributed graphs. In *Findings of EMNLP 2023*.
- Yuhan Li, Zhixun Li, Peisong Wang, Jia Li, Xiangguo Sun, Hong Cheng, and Jeffrey Xu Yu. 2023b. A survey of graph meets large language model: Progress and future directions. *CoRR*, abs/2311.12399.
- Hao Liu, Jiarui Feng, Lecheng Kong, Ningyue Liang, Dacheng Tao, Yixin Chen, and Muhan Zhang. 2024a. One for all: Towards training one graph model for all classification tasks. In *ICLR 2024*.
- Jiawei Liu, Cheng Yang, Zhiyuan Lu, Junze Chen, Yibo Li, Mengmei Zhang, Ting Bai, Yuan Fang, Lichao Sun, Philip S. Yu, and Chuan Shi. 2023. Towards graph foundation models: A survey and beyond. *CoRR*, abs/2310.11829.
- Ruyue Liu, Rong Yin, Xiangzhen Bo, Xiaoshuai Hao, Yong Liu, Jinwen Zhong, Can Ma, and Weiping Wang. 2025. SSTAG: Structure-aware self-supervised learning method for text-attributed graphs. In *NeurIPS 2025*.
- Zheyuan Liu, Xiaoxin He, Yijun Tian, and Nitesh V Chawla. 2024b. Can we soft prompt LLMs for graph learning tasks? In *WWW 2024*.

- Haitao Mao, Zhikai Chen, Wenzhuo Tang, Jianan Zhao, Yao Ma, Tong Zhao, Neil Shah, Mikhail Galkin, and Jiliang Tang. 2024. Position: Graph foundation models are already here. In *ICML 2024*.
- Costas Mavromatis, Vassilis N Ioannidis, Shen Wang, Da Zheng, Soji Adeshina, Jun Ma, Han Zhao, Christos Faloutsos, and George Karypis. 2023. Train your own GNN teacher: Graph-aware distillation on textual graphs. In *ECML PKDD 2023*.
- Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. 2000. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2).
- Péter Mernyei and Catalina Cangea. 2020. Wiki-CS: A wikipedia-based benchmark for graph neural networks. *CoRR*, abs/2007.02901.
- Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022. Cross-task generalization via natural language crowdsourcing instructions. In *ACL 2022*.
- Seth A Myers, Aneesh Sharma, Pankaj Gupta, and Jimmy Lin. 2014. Information network or social network?: the structure of the twitter follow graph. In *WWW 2014*.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *ICML 2021*, volume 139.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21.
- Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *EMNLP-IJCNLP 2019*, pages 3980–3990.
- Xubin Ren, Jiabin Tang, Dawei Yin, Nitesh Chawla, and Chao Huang. 2024. A survey of large language models for graphs. In *KDD 2024*.
- Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI Mag.*, 29(3).
- Li Sun, Zhenhao Huang, Suyang Zhou, Qiqi Wan, Hao Peng, and Philip S. Yu. 2025a. RiemannGFM: Learning a graph foundation model from riemannian geometry. In *WWW 2025*, pages 1154–1165.
- Yuanfu Sun, Zhengnan Ma, Yi Fang, Jing Ma, and Qiaoyu Tan. 2025b. GraphICL: Unlocking graph learning potential in LLMs through structured prompt design. In *Findings of NAACL 2025*, pages 2440–2459.
- Juntao Tan, Shuyuan Xu, Wenyue Hua, Yingqiang Ge, Zelong Li, and Yongfeng Zhang. 2024. IDGenRec: LLM-RecSys alignment with textual ID learning. In *SIGIR 2024*.
- Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. 2024. GraphGPT: Graph instruction tuning for large language models. In *SIGIR 2024*.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and efficient foundation language models. *CoRR*, abs/2302.13971.
- Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2018. Graph attention networks. In *ICLR 2018*.
- Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. 2019. Deep graph infomax. In *ICLR 2019*.
- Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. 2023. Can language models solve graph problems in natural language? In *NeurIPS 2023*.
- Jianing Wang, Junda Wu, Yupeng Hou, Yao Liu, Ming Gao, and Julian McAuley. 2024a. InstructGraph: Boosting large language models via graph-centric instruction tuning and preference alignment. In *Findings of ACL 2024*.
- Zehong Wang, Zheyuan Zhang, Nitesh V Chawla, Chuxu Zhang, and Yanfang Ye. 2024b. GFT: graph foundation model with transferable tree vocabulary. In *NeurIPS 2024*.
- Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *ICLR 2022*.
- Oliver Wieder, Stefan Kohlbacher, Méline Kuenemann, Arthur Garon, Pierre Ducrot, Thomas Seidel, and Thierry Langer. 2020. A compact review of molecular property prediction with graph neural networks. *Drug Discov. Today Technol.*, 37.
- Qitian Wu, Wentao Zhao, Zenan Li, David P Wipf, and Junchi Yan. 2022. NodeFormer: A scalable graph structure learning transformer for node classification. In *NeurIPS 2022*.
- Qitian Wu, Wentao Zhao, Chenxiao Yang, Hengrui Zhang, Fan Nie, Haitian Jiang, Yatao Bian, and Junchi Yan. 2023. Simplifying and empowering transformers for large-graph representations. In *NeurIPS 2023*.

- Jun Xia, Chengshuai Zhao, Bozhen Hu, Zhangyang Gao, Cheng Tan, Yue Liu, Siyuan Li, and Stan Z Li. 2023. Mole-BERT: Rethinking pre-training graph neural networks for molecules. In *ICLR 2023*.
- Lianghao Xia, Ben Kao, and Chao Huang. 2024. Open-Graph: Towards open graph foundation models. In *Findings of EMNLP 2024*.
- Hao Yan, Chaozhuo Li, Ruosong Long, Chao Yan, Jianan Zhao, Wenwen Zhuang, Jun Yin, Peiyan Zhang, Weihao Han, Hao Sun, Weiwei Deng, Qi Zhang, Lichao Sun, Xing Xie, and Senzhang Wang. 2023. A comprehensive study on text-attributed graphs: Benchmarking and rethinking. In *NeurIPS 2023*.
- Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. 2024. Language is all a graph needs. In *Findings of EACL 2024*.
- Rong Yin, Ruyue Liu, Xiaoshuai Hao, Xingrui Zhou, Yong Liu, Can Ma, and Weiping Wang. 2025. Multi-modal molecular representation learning via structure awareness. *IEEE Trans. Image Process.*, 34:3225–3238.
- Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *KDD 2018*, pages 974–983.
- Zheng Zhang, Yuntong Hu, Bo Pan, Chen Ling, and Liang Zhao. 2025. TAGA: Text-attributed graph self-supervised learning by synergizing graph and text mutual transformations. In *CIKM 2025*, pages 4263–4272.
- Haihong Zhao, Aochuan Chen, Xiangguo Sun, Hong Cheng, and Jia Li. 2024. All in one and one for all: A simple yet effective method towards cross-domain graph pretraining. In *KDD 2024*, pages 4443–4454.
- Jianan Zhao, Meng Qu, Chaozhuo Li, Hao Yan, Qian Liu, Rui Li, Xing Xie, and Jian Tang. 2023a. Learning on large-scale text-attributed graphs via variational inference. In *ICLR 2023*.
- Jianan Zhao, Le Zhuo, Yikang Shen, Meng Qu, Kai Liu, Michael Bronstein, Zhaocheng Zhu, and Jian Tang. 2023b. GraphText: Graph reasoning in text space. *CoRR*, abs/2310.01089.
- Huachi Zhou, Jiahe Du, Chuang Zhou, Chang Yang, Yilin Xiao, Yuxuan Xie, and Xiao Huang. 2025. Each graph is a new language: Graph learning with LLMs. In *Findings of ACL 2025*, pages 17548–17559.
- Yun Zhu, Haizhou Shi, Xiaotang Wang, Yongchao Liu, Yaoke Wang, Boci Peng, Chuntao Hong, and Siliang Tang. 2025. GraphCLIP: Enhancing transferability in graph foundation models for text-attributed graphs. In *WWW 2025*, pages 2183–2197.
- Yun Zhu, Yaoke Wang, Haizhou Shi, and Siliang Tang. 2024. Efficient tuning and inference for large language models on textual graphs. In *IJCAI 2024*.

# Contents

|          |                                                               |           |
|----------|---------------------------------------------------------------|-----------|
| <b>1</b> | <b>Introduction</b>                                           | <b>1</b>  |
| <b>2</b> | <b>Preliminaries</b>                                          | <b>3</b>  |
| <b>3</b> | <b>Methodology</b>                                            | <b>3</b>  |
| 3.1      | Graph Understanding Module . . . . .                          | 3         |
| 3.2      | Graph Inference Module . . . . .                              | 4         |
| 3.3      | Computational Complexity . . . . .                            | 5         |
| <b>4</b> | <b>Experiments</b>                                            | <b>6</b>  |
| 4.1      | Main Performance Comparison (RQ1) . . . . .                   | 6         |
| 4.2      | Cross-graph and Cross-task Transfer (RQ2) . . . . .           | 7         |
| 4.3      | Exploration Studies (RQ3~5) . . . . .                         | 8         |
| <b>5</b> | <b>Related Works</b>                                          | <b>9</b>  |
| <b>6</b> | <b>Conclusion</b>                                             | <b>9</b>  |
|          | <b>Limitations</b>                                            | <b>10</b> |
|          | <b>Ethical Considerations</b>                                 | <b>10</b> |
| <b>A</b> | <b>Datasets</b>                                               | <b>15</b> |
| A.1      | Computer Science Datasets . . . . .                           | 15        |
| A.2      | E-commerce Datasets . . . . .                                 | 15        |
| A.3      | Biomedical Datasets . . . . .                                 | 15        |
| <b>B</b> | <b>Baselines</b>                                              | <b>15</b> |
| B.1      | Graph-agnostic Methods . . . . .                              | 16        |
| B.2      | GNN-based Methods . . . . .                                   | 16        |
| B.3      | LM-only Methods . . . . .                                     | 16        |
| B.4      | GNN-LLM Integration Methods . . . . .                         | 16        |
| B.5      | Graph Foundation Models . . . . .                             | 16        |
| B.6      | Additional Baselines for Generative Link Prediction . . . . . | 16        |
| <b>C</b> | <b>Implementation Details</b>                                 | <b>17</b> |
| <b>D</b> | <b>Prompt Templates</b>                                       | <b>17</b> |
| <b>E</b> | <b>Prompt-based GNN Analysis</b>                              | <b>17</b> |
| E.1      | Permutation Sensitivity . . . . .                             | 17        |
| E.2      | Layer-wise Behavior . . . . .                                 | 17        |
| <b>F</b> | <b>Graph Vocabulary Analysis</b>                              | <b>19</b> |
| <b>G</b> | <b>Prompt Length and Scalability</b>                          | <b>19</b> |
| G.1      | Neighbor Sampling Strategy . . . . .                          | 19        |
| G.2      | Token Capacity of LLMs . . . . .                              | 19        |
| G.3      | Bounded Prompt Size . . . . .                                 | 19        |
| <b>H</b> | <b>Adaptive Prompting for Cross-Domain Transfer</b>           | <b>20</b> |
| <b>I</b> | <b>Dynamic Graph Adaptation</b>                               | <b>20</b> |
| <b>J</b> | <b>Case Studies</b>                                           | <b>21</b> |
| J.1      | Cora Citation Network . . . . .                               | 21        |
| J.2      | PubMed Citation Network . . . . .                             | 22        |

## Appendix

The appendix complements the main text as follows. Appendices A, B, and C detail the experimental setup, covering dataset statistics, the six families of baselines, and reproduction settings. Appendix D lists all prompt templates used by the prompt-based GNN and the task-oriented instructions. Appendix E analyzes the prompt-based GNN through permutation sensitivity and layer-wise behavior, Appendix F quantifies the coverage and uniqueness of the graph vocabulary, and Appendix G examines prompt length and scalability on dense graphs. Appendix H explores adaptive prompting for cross-domain transfer, Appendix I extends PromptGFM to evolving graphs through localized updates, and Appendix J traces the layer-wise refinement of node representations on Cora and PubMed.

### A Datasets

We use seven public benchmarking datasets from three domains to evaluate PromptGFM, with statistics summarized in Table 11. Detailed descriptions of each dataset are provided below.

#### A.1 Computer Science Datasets

- **Cora** (McCallum et al., 2000). The dataset includes a citation network consisting of 2,708 scientific publications in the field of machine learning, categorized into 7 classes based on their research topics. Nodes represent individual papers, and edges denote citation links between them, totaling 10,858 connections. Each paper is associated with its title and abstract.
- **Citeseer** (Giles et al., 1998). This dataset comprises a citation network of 3,327 scientific publications, categorized into 6 classes, including Agents, Artificial Intelligence, Database, Information Retrieval, Machine Learning, and Human-Computer Interaction. Nodes correspond to documents, and edges represent citation relationships between them, amounting to 9,464 links.
- **Ogbn-arxiv** (Hu et al., 2020). The ogbn-arxiv dataset is part of the Open Graph Benchmark and consists of a directed citation graph of 169,343 arXiv papers, categorized into 40 subject areas. Nodes represent individual papers, and edges indicate citation relationships, totaling over 2.3 million connections. Each paper includes textual data from its title and abstract.

- **WikiCS** (Mernyei and Cangea, 2020). The WikiCS dataset is a benchmark dataset derived from Wikipedia, designed for evaluating GNNs. It comprises a citation network with 11,701 nodes representing computer science articles and 431,726 edges corresponding to hyperlinks between them. The dataset features 10 classes corresponding to branches of computer science, with very high connectivity.

#### A.2 E-commerce Datasets

- **History** (Yan et al., 2023). The History dataset, extracted from the Amazon-Books dataset, includes items categorized under the second-level label “History”. It comprises 41,551 nodes and 503,180 edges, where each node represents a book, and edges indicate frequent co-purchases or co-views between books. This dataset incorporates the title and description of each book as text attributes of the node. The classification task involves assigning books to 12 distinct categories.
- **Photo** (Yan et al., 2023). The Photo dataset, derived from the Amazon-Electronics dataset, consists of 48,362 nodes and 873,782 edges, forming a dense network that reflects user purchasing behavior. Each node represents an electronic product, while edges indicate frequent co-purchases or co-views between items. For textual attributes, user reviews are incorporated, prioritizing the most upvoted review when available; otherwise, a randomly selected review is used. The classification task involves categorizing electronic products into 12 distinct categories.

#### A.3 Biomedical Datasets

- **PubMed** (Sen et al., 2008). PubMed is a citation network of 19,717 scientific publications from the PubMed database pertaining to diabetes, classified into 3 classes: experimental induced diabetes, type 1 diabetes, and type 2 diabetes. Nodes are research papers, and edges signify citation links, amounting to 88,648 connections. This dataset is used for large-scale graph representation learning and evaluating algorithms in the biomedical domain.

## B Baselines

We compare PromptGFM against a comprehensive set of baselines, organized into six categories.

Table 11: Statistics of seven benchmarking datasets from three domains.

| Dataset    | Domain           | #Nodes | #Edges  | #Classes |
|------------|------------------|--------|---------|----------|
| Cora       | Computer Science | 2708   | 10858   | 7        |
| Citeseer   | Computer Science | 3327   | 9464    | 6        |
| Ogbn-arxiv | Computer Science | 169343 | 2332486 | 40       |
| WikiCS     | Computer Science | 11701  | 431726  | 10       |
| Photo      | E-commerce       | 48362  | 873782  | 12       |
| History    | E-commerce       | 41551  | 503180  | 12       |
| PubMed     | Biomedical       | 19717  | 88648   | 3        |

### B.1 Graph-agnostic Methods

- **MLP**. This method adopts a multi-layer perceptron to learn low-dimensional embeddings for each node without using graph structure.

### B.2 GNN-based Methods

- **GCN** (Kipf and Welling, 2017). This model generalizes convolutional operations to graphs for semi-supervised learning.
- **GAT** (Veličković et al., 2018). This method integrates attention mechanisms to weigh neighbor features during aggregation.
- **GraphSAGE** (Hamilton et al., 2017). GraphSAGE is an inductive framework that enables generalization to unseen data via neighborhood sampling and aggregation.
- **RevGNN** (Li et al., 2021). This method scales GNNs to over 1000 layers using reversible residual connections that bound memory consumption.
- **SGFormer** (Wu et al., 2023). This work introduces a simplified graph transformer that captures both local and global graph structure with a single-layer global attention.
- **NodeFormer** (Wu et al., 2022). This framework presents a scalable graph transformer that utilizes a kernelized message-passing scheme, enabling efficient training on large-scale graphs.

### B.3 LM-only Methods

- **BERT** (Devlin et al., 2019). This language model learns bidirectional context through pre-training with masked language modeling and next sentence prediction.
- **Sentence-BERT** (Reimers and Gurevych, 2019). This model is a modification of BERT that uses a siamese network to generate fixed-size sentence embeddings.
- **Llama3-8B** (Grattafiori et al., 2024). This is an open-weight, decoder-only Transformer with 8 billion parameters that achieves strong performance on a wide range of benchmarks, particularly

in reasoning, multilingual tasks, and instruction following.

### B.4 GNN-LLM Integration Methods

- **OFA** (Liu et al., 2024a). This framework handles various graph classification tasks across different domains using a single model.
- **LLaGA** (Chen et al., 2024a). This model integrates LLMs for graph data by transforming nodes into structure-aware sequences and mapping them into the token embedding space with a projector.
- **GraphPrompter** (Liu et al., 2024b). This framework aligns graphs with LLMs via soft prompts, including a GNN to capture structure and an LLM to interpret node text, combining semantic and structural insights through prompt tuning.
- **ENGINE** (Zhu et al., 2024). This method presents a parameter- and memory-efficient fine-tuning approach that combines a tunable GNN-based side structure alongside each LLM layer.
- **GraphText** (Zhao et al., 2023b). This framework converts graphs into hierarchical graph-syntax trees and performs graph reasoning with LLMs entirely in text space.

### B.5 Graph Foundation Models

- **GFT** (Wang et al., 2024b). This model treats computation trees as transferable patterns and builds a discrete tree vocabulary shared across graphs and tasks.

### B.6 Additional Baselines for Generative Link Prediction

- **InstructGLM** (Ye et al., 2024). This framework verbalizes a node’s multi-hop neighborhood into natural-language instructions and instruction-tunes generative LLMs to directly generate predictions for node classification and link prediction without GNNs.
- **KGT5-context** (Kochsiek et al., 2023). This model extends the sequence-to-sequence knowl-

edge graph completion model KGT5 by appending the query entity’s one-hop neighborhood to the input sequence, and decodes the target entity directly as text.

## C Implementation Details

We conduct all experiments in PyTorch on four NVIDIA RTX A6000 GPUs. In the graph understanding module, we use OpenAI’s GPT-4o mini to facilitate prompt-based GNNs. The number of GNN layers is selected from  $\{1, 2, 3, 4\}$ , and the first-order neighbor sampling ratio from  $\{0.3, 0.6, 1.0\}$ . In the graph inference module, we fine-tune either Flan-T5 or Llama3-8B with a learning rate of  $3e-4$  and a batch size of 4. To mitigate potential biases introduced by task-specific prompts, we design a prompt pool for each task and randomly sample prompts during instruction construction to enhance robustness. We adopt a standard early-stopping strategy during training: if the validation performance does not improve for a fixed number of consecutive epochs (determined per dataset), training is halted to prevent overfitting. All results are reported using 10-fold cross-validation, with mean performance and standard deviations. For evaluation, we use accuracy and Macro-F1 (M-F1) for node classification, and accuracy and HR@1 for link prediction. To ensure fair comparison, BERT is used for node initialization across all applicable baselines. For other hyperparameters of the compared methods, we follow the original papers and carefully tune them to each dataset.

## D Prompt Templates

This section provides templates of prompts, including prompt-based GNNs and task-oriented prompts.

### Node Initialization Prompt

The title of the paper is `<the title of the paper>`, the abstract of the paper is `<the abstract of the paper>`. Please summarize the paper.  
 /\*\*\* This prompt varies depending on the dataset. The present instance is for citation datasets only. \*\*\*/

### Layer-wise GNN Replication Prompt

Given the central node `<l-th round textual representation of the central node>`. The selected one-hop neighbors are `[<l-th round of node #1>, ..., <l-th round node #N>]`.

Please aggregate neighbor nodes and update a concise yet meaningful representation for the central node. Note connected nodes should share similar semantics and vice versa.

### Node Classification Task Template

Given `<central node language-based ID>` has 1-hop connections with [..., `<1-hop neighbor language-based IDs>`, ...], and it also has 2-hop connections with [..., `<2-hop neighbor language-based IDs>`, ...]. Which category should `<central node language-based ID>` be classified as?

### Discriminative Link Prediction Task Template

Given `<central node language-based ID>` has 1-hop connections with [..., `<1-hop neighbor language-based IDs>`, ...], and it also has 2-hop connections with [..., `<2-hop neighbor language-based IDs>`, ...]. Among `<central node language-based ID>` and `<negative sampling node language-based ID>`, which node will be connected to `<central node language-based ID>`?

### Generative Link Prediction Task Template

Given `<central node language-based ID>` has 1-hop connections with [..., `<1-hop neighbor language-based IDs>`, ...], and it also has 2-hop connections with [..., `<2-hop neighbor language-based IDs>`, ...]. Which node should be connected to `<central node language-based ID>`?

## E Prompt-based GNN Analysis

We illustrate the prompt-based GNN workflow in Figure 6 and validate it from two complementary angles: permutation sensitivity (§E.1) and the layer-wise behavior of the prompt-based GNN (§E.2), which covers structural discrimination, semantic retention, and the impact of GNN depth.

### E.1 Permutation Sensitivity

We evaluate the sensitivity of our prompt-based GNNs (using Flan-T5) to input order by comparing two variants: **(a) shuffle nodes**, which randomizes the order of neighbors, and **(b) shuffle tokens**, which injects position-specific tokens to nodes and shuffles them instead. As illustrated in Table 12, regardless of the shuffle method, both variants result in only minor fluctuations in node classification accuracy, indicating that prompt-based GNNs with the subsequent fine-tuning are robust to input permutation and maintain stable performance.

### E.2 Layer-wise Behavior

**Structural Discrimination.** To validate that our language-based graph vocabulary effectively preserves structural proximity, we conduct an analysis

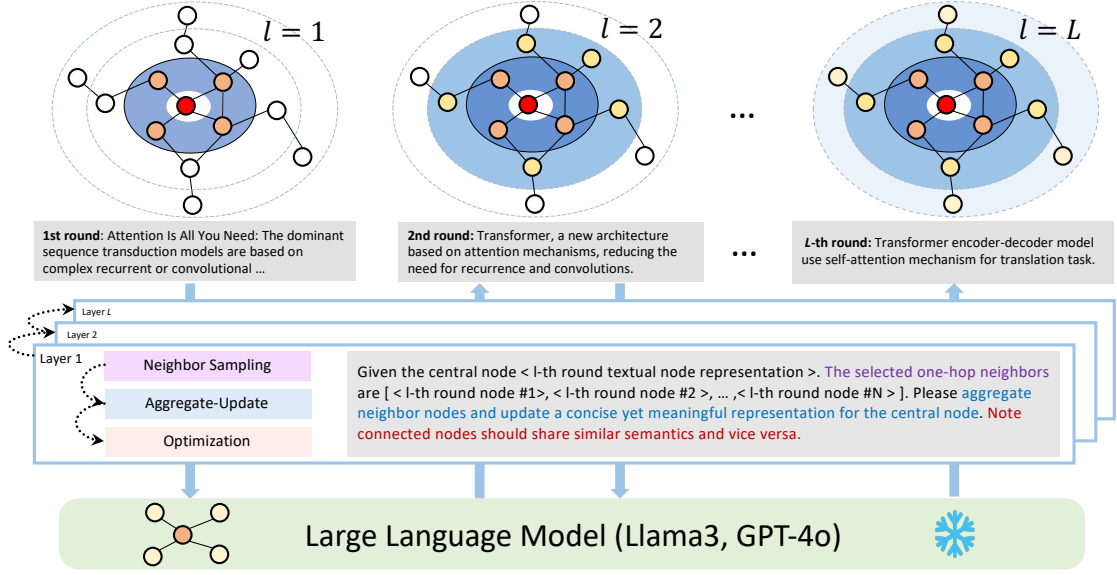


Figure 6: Graph understanding module via prompt-based GNNs. We prompt LLMs to achieve fine-grained reproduction of traditional GNN workflow, refining verbose textual representations into concise yet meaningful ones. In the prompt, neighbor sampling (see Equation 3) is in purple, the aggregation-update mechanism (see Equation 4) in blue, and the optimization in red.

Table 12: Permutation sensitivity of prompt-based GNNs.

| Variant        | Cora             | Citeseer         |
|----------------|------------------|------------------|
| shuffle nodes  | 90.64 $\pm$ 1.12 | 83.79 $\pm$ 0.89 |
| shuffle tokens | 90.55 $\pm$ 1.03 | 84.28 $\pm$ 1.06 |

by encoding the final language-based IDs with a pretrained Sentence-BERT encoder and computing the cosine similarity between node pairs. The results on the Cora dataset are presented in Table 13. We observe that connected nodes exhibit a significantly higher mean similarity than disconnected nodes ( $p < 10^{-5}$ ). Furthermore, the margin between the two groups grows steadily as the number of GNN layers increases, indicating stronger structural discrimination with deeper propagation. This trend highlights that our prompt-based GNN captures meaningful semantics with stacking aggregation-update layers, thereby suggesting that its representational behavior parallels that of traditional embedding-based GNNs. Notably, disconnected pairs remain around 0.03 at every depth, indicating that deeper propagation sharpens structural discrimination without collapsing all nodes into a common representation.

**Semantic Retention and Integration.** We next examine how each node balances its own semantics against the injected neighbor context. We en-

Table 13: Layer-wise mean cosine similarity of node embeddings on Cora, with margin reflecting the gap between connected and disconnected pairs.

|         | Connected | Disconnected | Margin |
|---------|-----------|--------------|--------|
| Layer 1 | 0.171     | 0.036        | 0.135  |
| Layer 2 | 0.223     | 0.032        | 0.191  |
| Layer 3 | 0.308     | 0.034        | 0.274  |

code the initial center-only summary and the output of each graph understanding layer with the same frozen encoder, and report the similarity to the initial summary (self-retention) and to the sampled neighbors before and after each layer-wise update. As shown in Table 14, neighbor similarity consistently increases after every update, showing that structural context is progressively injected. Meanwhile, the representation stays closer to its own initial summary than to its neighbors at every depth, so central-node semantics are compressed and refined rather than erased. The two similarities nevertheless converge as layers are stacked, with the margin on Cora narrowing from 0.673 at the first layer to 0.295 at the fourth, indicating that the central node progressively loses its dominance.

**Impact of GNN Depth.** This trade-off is directly reflected in task performance. As shown in Figure 7, increasing the number of layers initially improves results by capturing broader context and higher-order dependencies. However, beyond a cer-

Table 14: Layer-wise self-retention and neighbor similarity of textual representations. Neighbor similarity is reported before and after each layer-wise update.

| Metric                       | $l = 1$ | $l = 2$ | $l = 3$ | $l = 4$ |
|------------------------------|---------|---------|---------|---------|
| <i>Cora</i>                  |         |         |         |         |
| Self-retention               | 0.837   | 0.758   | 0.687   | 0.632   |
| Neighbor similarity (before) | 0.118   | 0.173   | 0.226   | 0.306   |
| Neighbor similarity (after)  | 0.164   | 0.214   | 0.279   | 0.337   |
| <i>Citeseer</i>              |         |         |         |         |
| Self-retention               | 0.807   | 0.724   | 0.643   | 0.584   |
| Neighbor similarity (before) | 0.103   | 0.147   | 0.188   | 0.248   |
| Neighbor similarity (after)  | 0.144   | 0.181   | 0.234   | 0.269   |

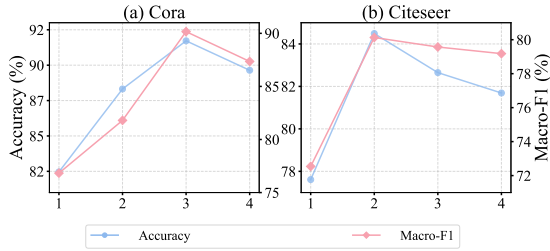


Figure 7: Impact of layer numbers.

tain depth, performance plateaus or declines due to over-smoothing, a phenomenon also observed in traditional GNNs. The optimal depth is achieved with three layers on Cora and two layers on Citeseer. These results suggest that textual representations can be propagated across the graph in much the same way as numerical embeddings, effectively encoding both semantic and structural information.

## F Graph Vocabulary Analysis

We evaluate the quality of our graph vocabulary using two metrics in Table 15. **(a) Coverage.** We compute the proportion of nodes successfully assigned IDs. Since the graph vocabulary is grounded in the LLM’s native vocabulary, any node’s semantics can be expressed through a combination of natural language tokens. In practice, every node receives a valid ID, yielding full coverage across all datasets. **(b) Uniqueness.** We measure the entropy of token distributions across nodes within both individual and merged graphs. Higher entropy indicates more diverse and fine-grained token usage, indicating stronger uniqueness and representational capacity. Notably, the merged vocabulary (“ALL”) achieves the highest entropy (12.2761), showing that integrating multiple graphs improves vocabulary expressiveness and discriminability, which reinforces the necessity of building GFM.

## G Prompt Length and Scalability

In dense TAGs, high-degree nodes present a challenge for prompt-based GNNs, as their long prompts may exceed LLM input limits and hinder efficiency. To address this issue, we examine strategies that balance efficiency, scalability, and representational fidelity.

### G.1 Neighbor Sampling Strategy

Following standard practices, we adopt a neighbor sampling strategy in each GNN layer to reduce prompt length and enhance robustness. As shown in Table 17, sampling 60% of neighbors yields the best overall performance across datasets. Meanwhile, limiting the maximum sampled neighbors is a common practice in scalable GNNs. For example, GraphSAGE samples 25 and 10 neighbors at the first and second hops, while PinSAGE (Ying et al., 2018) selects from {10, 20, 50}. Following this rationale, we explore {10, 20, 30, 40} and empirically set 20 as the cap, balancing LLM input constraints with sufficient structural coverage.

### G.2 Token Capacity of LLMs

Our implementation leverages GPT-4o mini, which supports up to 128k input tokens per call. This capacity allows a single prompt to encode information from over 300 neighbors, given that each neighbor contributes about 400 tokens in the first propagation round. As shown in Table 16, only a small fraction of nodes (<0.25%) in a few datasets surpass this capacity. For these rare high-degree nodes, our neighbor sampling and truncation mechanism can bound prompt length, ensuring broad scalability to dense graphs.

### G.3 Bounded Prompt Size

Prompt-based GNNs guide the LLM to aggregate and update neighbor information through structured prompts, producing increasingly compact yet expressive representations. Across layers, the average length shrinks from  $\sim 400$  to  $\sim 10$  tokens as semantic and structural information is progressively distilled. Since each node is processed independently and depends only on its local degree, prompt size remains bounded regardless of graph scale, ensuring both efficiency and scalability to large graphs.

Table 15: Quantitative graph vocabulary analysis.

| Metric              | Cora    | Citeseer | PubMed | Ogbn-arxiv |
|---------------------|---------|----------|--------|------------|
| <i>Coverage (%)</i> | 100.00  | 100.00   | 100.00 | 100.00     |
| <i>Entropy</i>      | 9.1116  | 9.4561   | 8.4643 | 10.7464    |
| Metric              | History | Photo    | WikiCS | ALL        |
| <i>Coverage (%)</i> | 100.00  | 100.00   | 100.00 | 100.00     |
| <i>Entropy</i>      | 10.3752 | 11.3272  | 9.3825 | 12.2761    |

Table 16: Analysis of high-degree nodes and average connectivity.

| Metric                   | Cora | Citeseer | PubMed | Ogbn-arxiv | History | Photo | WikiCS |
|--------------------------|------|----------|--------|------------|---------|-------|--------|
| Avg. Degree              | 4.01 | 2.84     | 13.77  | 36.90      | 18.07   | 12.11 | 4.50   |
| High-degree (>300) Nodes | 0    | 0        | 0      | 0.20%      | 0.25%   | 0.19% | 0      |

Table 17: The performance with varying neighbor sampling ratios across three datasets.

| Sampling Ratio | 100%  | 60%          | 30%   |
|----------------|-------|--------------|-------|
| Cora           | 81.36 | <b>89.35</b> | 88.36 |
| Citeseer       | 79.34 | <b>82.23</b> | 81.13 |
| PubMed         | 90.23 | <b>91.12</b> | 89.34 |

## H Adaptive Prompting for Cross-Domain Transfer

While inter-domain transfer remains a community-wide challenge rather than a limitation of our method, the observed domain shift motivates us to investigate adaptive prompting (AP). We introduce a lightweight domain-adaptive strategy by prepending domain-specific instructions at inference time. For instance, during E-commerce→Biomedical transfer, we add: “*Now perform node classification (or link prediction) over a graph representing biomedical research citations, rather than co-purchase relations in e-commerce.*” When no source graph is involved, the instruction reduces to stating the target domain. The evaluation mirrors the E-commerce→Biomedical transfer setting of Table 5, where we perform node classification on PubMed with (w/) and without (w/o) adaptive prompting. As summarized in Table 18, adaptive prompting (w/ AP) consistently improves results across all evaluation settings. The gains are most pronounced when cross-domain sources are introduced, indicating that they stem from mitigating domain shift. These findings highlight adaptive prompting as an effective means of bridging domain gaps and a promising direction toward more generalizable GFM.

## I Dynamic Graph Adaptation

PromptGFM adapts to evolving graphs through localized updates, thereby avoiding the need to reprocess the entire graph whenever the graph structure or node content changes. The main update rules can be described as follows.

- **New node.** We generate an initial textual representation for the new node and aggregate information from its current neighbors.
- **New edge.** We update the affected endpoint nodes and, optionally, their  $L$ -hop neighborhoods, balancing representation freshness against computational cost.
- **Changed node text.** We regenerate the representations of the changed node and update only locally affected representations, avoiding recomputation for the rest of the graph.

To assess this protocol, we conduct node classification on a dynamic DBLP co-authorship graph. The graph comprises 26,247 nodes and 415,558 edges with four research-domain labels, and it evolves over yearly snapshots. Following a temporal split, all methods are trained on the first eight snapshots, validated on the middle snapshot, and evaluated on the last two. We compare PromptGFM-dyn, a dynamic variant of PromptGFM equipped with the localized updates above, against representative GNN and GNN-LLM baselines. As shown in Table 19, PromptGFM-dyn consistently outperforms all baselines, suggesting that cached language-based IDs can be locally updated and remain effective in evolving graph settings.

Table 18: The effectiveness of adaptive prompting (w/) on inter-domain transfer from e-commerce to biomedical domain.

| Source       |              | Pre-training       |       |                     |       | Co-training        |       |                     |       |
|--------------|--------------|--------------------|-------|---------------------|-------|--------------------|-------|---------------------|-------|
|              |              | Acc ( $\uparrow$ ) |       | M-F1 ( $\uparrow$ ) |       | Acc ( $\uparrow$ ) |       | M-F1 ( $\uparrow$ ) |       |
| History      | Photo        | w/o                | w/    | w/o                 | w/    | w/o                | w/    | w/o                 | w/    |
| $\times$     | $\times$     | 39.12              | 40.23 | 39.84               | 40.13 | 90.67              | 90.82 | 91.82               | 91.93 |
| $\checkmark$ | $\times$     | 46.24              | 47.44 | 47.41               | 47.98 | 88.52              | 90.27 | 89.39               | 90.22 |
| $\times$     | $\checkmark$ | 44.21              | 45.11 | 44.93               | 46.02 | 85.34              | 88.13 | 85.57               | 88.47 |
| $\checkmark$ | $\checkmark$ | 45.18              | 47.90 | 45.65               | 47.26 | 86.22              | 89.02 | 87.03               | 89.16 |

Table 19: Node classification on a dynamic DBLP co-authorship graph. We report balanced accuracy (Bal Acc) and Macro-F1.

| Method        | Bal Acc      | Macro-F1     |
|---------------|--------------|--------------|
| GCN           | 50.83        | 45.23        |
| GAT           | 50.28        | 45.39        |
| GraphSAGE     | 50.49        | 45.37        |
| GraphPrompter | 52.37        | 47.10        |
| OFA           | 55.17        | 52.54        |
| PromptGFM-dyn | <b>59.46</b> | <b>56.23</b> |

## J Case Studies

We visualize the layer-by-layer refinement of node representations and identify which neighbor nodes contribute to each stage. Textual representations evolve from verbose descriptions into compact, high-density token sequences (8–10 tokens). After the first layer, nodes absorb neighbor semantics (e.g., “error estimation”); after the second round, they integrate broader two-hop context (e.g., “non-linear regression analysis”). These observations demonstrate the explicit message-passing behavior of the prompt-based GNN in text space, effectively capturing both graph semantics and structural dependencies. We provide two representative cases below, from the Cora citation network and the PubMed biomedical citation network respectively.

### J.1 Cora Citation Network

#### Raw Text Attributes

Paper Title: Evaluating Neural Network Predictors by Bootstrapping. Abstract: We present a new method, inspired by the bootstrap, whose goal it is to determine the quality and reliability of a neural network predictor. Our method leads to more robust forecasting along with a large amount of statistical information on forecast performance that we exploit. We exhibit the method in the context of multi-variate time series prediction on financial data from the New York Stock Exchange. It turns out that the variation due

to different resamplings (i.e., splits between training, cross-validation, and test sets) is significantly larger than the variation due to different network conditions (such as architecture and initial weights). Furthermore, this method allows us to forecast a probability distribution, as opposed to the traditional case of just a single value at each time step. We demonstrate this on a strictly held-out test set that includes the 1987 stock market crash. We also compare the performance of the class of neural networks to identically bootstrapped linear models.

#### Round 0: Textual Feature Initialization

This method evaluates neural network predictors using a bootstrap-inspired approach, improving robustness in NYSE financial data forecasting. It predicts probability distributions rather than single values, surpassing resampling variations. Compared to bootstrapped linear models, it enhances accuracy, as demonstrated during the 1987 stock market crash.

#### Round 1: 1st Layer Representations

This bootstrap method improves neural network evaluations in NYSE forecasting, **comparable to traditional error estimation methods [A]. It clears and simplifies data structures [B]**, enhancing accuracy and predicting probability distribution.

/\*\* [A] comes from the paper: *A comparison of some error estimates for neural network models*. \*\*\*/

/\*\* [B] comes from the paper: *The Observer-Observation Dilemma in Neuro-Forecasting: Reliable Models From Unreliable Data Through Cleaning*. \*\*\*/

#### Round 2: 2nd Layer Representations

This bootstrap-inspired method refines neural network evaluations for NYSE forecasting, highlighting **local error bars from nonlinear regression analysis [C]**, improving accuracy with data simplification techniques.

/\*\* [C] comes from the paper: *Learning Local Error Bars for Nonlinear Regression*. \*\*\*/

#### Round 3: 3rd Layer Representations

Bootstrap method refines neural forecasts, integrates error bars, enhances accuracy.

## J.2 PubMed Citation Network

### Raw Text Attributes

Paper Title: Insulin dependent diabetes in Asians. Abstract: Type 1 diabetes is said to be extremely rare in children in India, where diabetes treated with insulin may be due to chronic pancreatic disease or malnutrition. To see whether typical type 1 diabetes occurred in Asian children in the United Kingdom, all known Asian children with diabetes in industrial West Yorkshire were ascertained. A total of 17 such children were studied; of these, seven were from three multiplex families and two fathers from these families had diabetes. All children were ketosis prone and developed diabetes while resident in the UK. There were significant increases in HLA-B8 and HLA-DR3 and increases in HLA-DR4 and HLA-DR3/DR4, while HLA-B15 was absent. Islet cell antibodies, either IgG or complement fixing, were present in four of 18 subjects tested, all of whom had disease of short duration. The prevalence of type 1 diabetes in Asian children aged 15 years or less in West Yorkshire was 36/100,000, assuming complete ascertainment. It is concluded that typical type 1 diabetes may occur in Asian children and this condition may be more common in families who have migrated to the UK.

tion and *HLA, complement C2, C4, properdin factor B and glyoxalase types in South Indian diabetics.* \*\*\*/

### Round 3: 3rd Layer Representations

Type 1 diabetes. Migration and HLA markers linked to increased diabetes incidence.

### Round 0: Textual Feature Initialization

The study examined type 1 diabetes prevalence in 17 Asian children in West Yorkshire, UK, focusing on family history. Seven were from multiplex families, and all were ketosis-prone. Significant increases in HLA-B8, HLA-DR3, and HLA-DR4 were found, with no HLA-B15. Islet cell antibodies were present in four of 18 subjects. The diabetes prevalence was 36/100,000, suggesting it may be more common in migrant families.

### Round 1: 1st Layer Representations

The study examined type 1 diabetes in Asian children in West Yorkshire, UK. **Environmental factors and migration may raise incidence [A]**, unlike the **low incidence of insulin-dependent diabetes in Karachi [B]**. Increases in HLA-B8, HLA-DR3, and HLA-DR4 were found.

/\*\*\* [A] comes from the paper: *Evidence for an environmental effect in the aetiology of insulin dependent diabetes in a transmigratory population.* \*\*\*/

/\*\*\* [B] comes from the paper: *Incidence of insulin dependent diabetes mellitus in Karachi, Pakistan.* \*\*\*/

### Round 2: 2nd Layer Representations

The study examined type 1 diabetes in Asian children in West Yorkshire, UK. **Migration may raise incidence [C]**, unlike the low incidence in Karachi. Increases in HLA markers were found, similar to **North and South Indian diabetics [D]**.

/\*\*\* [C] comes from the paper: *HLA-DR antigen frequencies in a North Indian type I diabetic population.* \*\*\*/

/\*\*\* [D] comes from two papers: *HLA-DR antigen frequencies in a North Indian type I diabetic popula-*