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

Anonymous ACL submission

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 (GFMs) 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 finest GNN workflow within the text space, facilitating seamless GNN-LLM integration and elegant graph-text alignment; (b) 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 our superiority, especially effective transferability across diverse graphs and tasks and flexible adaptability to out-of-distribution (OOD) pre-training tasks.

1 Introduction

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

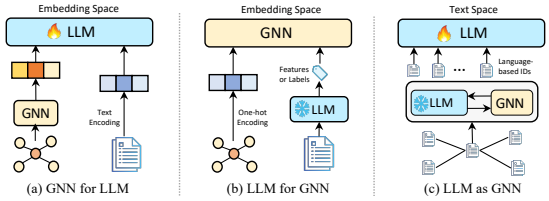
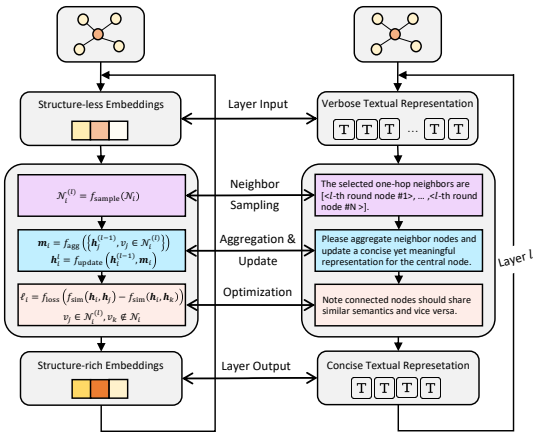


Figure 1: Overview of three GNN-LLM integration paradigms for graph-text alignment: (a) *GNN for LLM* and (b) *LLM for GNN* use decoupled architectures in the embedding space, while (c) our work functions *LLM as GNN* in the text space.

In TAGs, GNNs and LLMs are commonly combined to process structural and textual modalities, as shown 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 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 yielding 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 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

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



(a) Embedding-based GNN (b) Prompt-based GNN

mechanism. As shown in Figure 2(a), a standard GNN layer typically involves neighbor sampling, aggregation-update, and optimization (Kipf and Welling, 2017; Velickovic et al., 2017). Stacking such layers transforms structure-less embeddings into structure-rich representations with higher-order dependencies. This raises a key question: ***Can LLMs faithfully replicate GNNs 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 (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; 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 function 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 connections. 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 OOD pre-training tasks.

2 Preliminaries

Text-Attributed Graphs (TAGs). A TAG is defined as $G = (V, E, X)$, where V is the set of nodes, E is the set of edges, and X is the textual node attributes. Each node $v_i \in V$ is associated with a textual description $X_i = \left(x_i^1, x_i^2, \ldots, x_i^{n_i}\right)$, where each $x_i^k \in \mathcal{X}, k = 1, \ldots, n_i$, 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 neighboring nodes to 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:

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

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

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

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

3 Methodology

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. **prompt-based GNN**. We first initialize node representations by summarizing textual attributes, analogous to the lookup layer in conventional GNNs. To reduce computational overhead, we sample one-hop neighbors as follows:

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

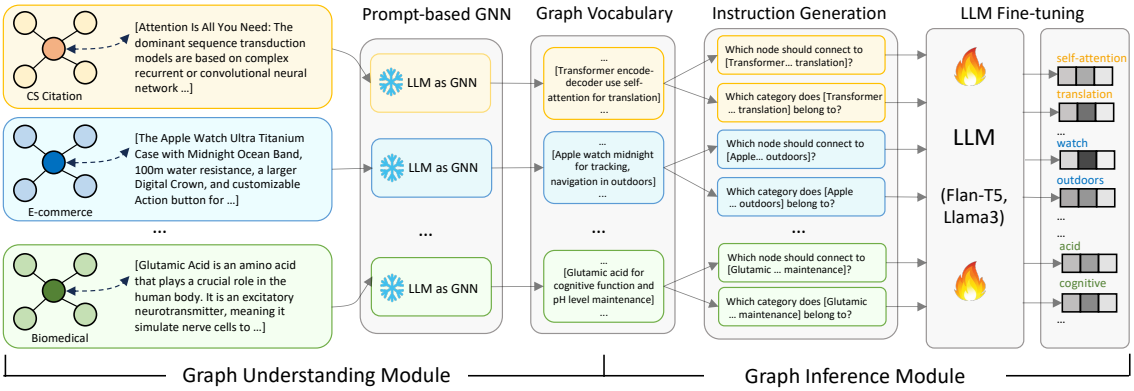
where $X^{(l-1)} = \left\{X_0^{(l-1)}, X_1^{(l-1)}, \ldots, X_{|V|-1}^{(l-1)}\right\}$ are textual representations of all nodes from the previous layer, and $\left\{X_j^{(l-1)}\right\}$ corresponds to the sampled neighbors $\mathcal{N}_i$. Then, we directly prompt LLMs to perform message aggregation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

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

where we seek 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 refer to unsupervised graph learning with a prompt-driven contrastive loss, which encourages similarity among neighboring nodes while pushing apart distant ones. As a task-agnostic approach to graph representation learning, contrastive optimization naturally aligns with the generality required by GFMs. While traditional negative sampling becomes redundant in our setting, we prompt LLMs to implicitly guide the contrastive objective. Prompt construction 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

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.



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 high-order dependencies. **(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 are equivalent to an adjacency matrix, collectively representing the entire graph structure while guaranteeing its efficiency (§3.3). We further investigate layer-wise representation discrimination and module performance in Appendix E. In summary, this fine-grained replication 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.

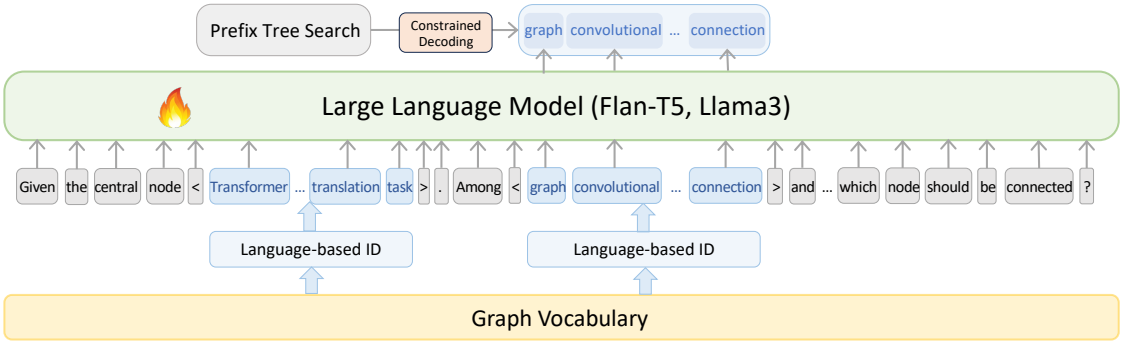
Graph Vocabulary. A transferable graph vocabulary, whose fundamental units can represent each node, is essential for effective GFMs and should satisfy three key criteria: **expressiveness**, **transferability**, and **scalability**. Since each node has been equipped with a textual representation encoding 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:

$$\mathcal{F} : V \to T, \tag{5}$$

where each node v_i is mapped to a sequence $T_i = \left(t_i^1, t_i^2, \ldots, t_i^{m_i}\right)$, with $t_i^k \in \mathcal{X}$ and $\mathcal{X}$ denoting the dictionary of natural language tokens. Collectively, the language-based IDs of all nodes in V are denoted as $T = \left\{T_0, T_1, \ldots, T_{|V|-1}\right\}$. We analyze vocabulary coverage and uniqueness in Appendix F. 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.** Any node, whether previously seen or not, can dynamically generate language-based IDs compatible with existing nodes, 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

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 GFMs.



task-oriented templates to construct instructions $\mathcal{T}$:

$$\mathcal{T} \leftarrow \text{Prompt}_{\text{template}}\left(T, G \mid \mathcal{F}\right), \tag{6}$$

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:

$$\mathcal{L} = -\sum_{j=1}^{|Y|} \log \Pr\left(y_j \mid \mathcal{T}, Y_{<j}\right), \tag{7}$$

where $\Pr\left(y_j \mid \mathcal{T}, Y_{<j}\right)$ denotes the probability of the j-th output token y_j, conditioned on the instruction $\mathcal{T}$ and previous tokens $Y_{<j} = (y_1, y_2, \ldots, y_{j-1})$. To support broad generalization, we fine-tune LLM backbones such as T5 (Raffel et al., 2020), FLAN (Wei et al., 2022), 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, PromptGFM not only facilitates robust knowledge transfer across diverse unseen graphs and tasks (§4.2), but also exhibits adaptability to accommodate OOD pre-training tasks at inference time (§4.3).

Constrained Decoding via Prefix Tree Search. To reduce LLM hallucination in generative tasks, we use constrained decoding with prefix tree search (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 is a natural language token and each root-to-leaf path denotes a valid node ID. During autoregressive generation, token selection is restricted to valid prefix-tree paths, ensuring predictions correspond to actual graph nodes and preventing hallucinated outputs. This effectiveness can be attributed to the discrete 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 need not be regenerated. Instead, we expect subsequent LLM fine-tuning over vocabulary-based natural language tokens can absorb and reflect updated graph features.

Graph Inference Module. With IDs cached, fine-tuning and decoding operate entirely over ID tokens. At inference time, each decoding step costs $O(L)$, where L is the average ID length. With prefix tree search constraining the candidate space to the graph-specific vocabulary of size N, the overall decoding complexity is bounded by $O(N \times L)$. 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 to transfer across diverse graphs and tasks as a general-purpose GFM? **RQ3:** How does PromptGFM adapt to more OOD pre-training tasks? **RQ4:** How does each module contribute to the overall performance? **RQ5:** What is the computational efficiency in graph understanding and inference modules?

We evaluate PromptGFM on seven TAG datasets

Table 1: Evaluation results (%) on node classification accuracy (↑). We highlight the methods with the **first** and **second** best performances, 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
SAGE	83.69	73.17	83.22	68.78	82.12	80.06	79.56
RevGNN	86.90	77.34	82.16	70.43	83.04	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	**81.56**
GraphPrompter	80.26	73.61	**94.80**	75.61	79.42	80.04	80.98
PromptGFM (Flan-T5)	**91.72**	**84.49**	92.83	**80.58**	82.33	**85.41**	81.49
PromptGFM (Llama3-8B)	**92.42**	**85.32**	**94.65**	**83.78**	**86.72**	**86.61**	**84.66**
p-value	4e-3	2.73e-8	3.21e-7	4.48e-10	4.85e-7	3.62e-2	9.93e-4

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

Method	Discriminative: Accuracy (↑)			
	Cora	Citeseer	Ogbn-arxiv	PubMed
GCN	77.15	78.72	80.89	77.36
GAT	70.44	77.17	76.25	74.36
SAGE	85.31	87.15	80.76	83.22
GraphPrompter	90.10	91.67	73.21	80.49
Ours (Flan-T5)	**90.57**	**92.03**	**81.12**	**87.64**
Ours (Llama3)	**91.68**	**93.46**	**84.27**	**88.12**

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

Method	Generative: HR@1 (↑)		
	Cora	Citeseer	PubMed
GCN	5.95	6.82	0.51
GAT	2.22	3.59	0.28
SAGE	6.59	8.73	0.45
PromptGFM	**8.21**	**8.90**	**1.21**

from computer science, e-commerce, and biomedical domains. We compare against graph-agnostic, GNN-based, LM-only, and GNN-LLM integration baselines, and report node classification and link prediction performance under consistent evaluation protocols. Dataset statistics, baseline details, and reproduction settings are provided in Appendix A.

4.1 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 three observations. (a) Graph-based models consistently outperform graph-agnostic ones, confirming the importance of structural information. (b) PromptGFM surpasses OFA, ENGINE, and GraphPrompter on node classification, suggesting that decoupled GNN-for-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 high-order dependencies without a true GNN mechanism. In contrast, by replicating message passing through prompt-based GNNs, PromptGFM demonstrates the feasibility of using LLMs as GNNs for graph-text alignment. Further analysis is provided in Appendix E.

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 traditional GNNs 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 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 (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** trans-

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
✗	✓	**60.34**	**54.81**	**85.45**	**83.91**
✓	✓	**61.25**	**55.66**	**86.77**	**84.24**

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	**90.67**	**91.82**
✓	✗	**46.24**	**47.41**	**88.52**	**89.39**
✗	✓	44.21	44.93	85.34	85.57
✓	✓	**45.18**	**45.65**	86.22	87.03

fer 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 *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 could 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). In contrast, 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→NC* generally surpasses *zero-shot*,

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	**90.67**	**91.82**
✓	✗	✗	51.76	52.84	**86.34**	**87.29**
✗	✓	✗	40.12	42.38	85.21	86.12
✗	✗	✓	**60.21**	**62.02**	82.17	86.01
✓	✓	✗	50.17	51.74	84.39	81.94
✓	✗	✓	**57.28**	**59.71**	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	*zero-shot*	18.54	27.64	39.12
	LP→NC	60.74	50.12	57.42
	ST-NC	**91.72**	**84.49**	**92.83**
LP	*zero-shot*	67.28	69.22	73.59
	NC→LP	78.16	79.23	84.11
	ST-LP	**91.68**	**93.46**	**88.12**

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→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 it as a task-adaptive GFM.

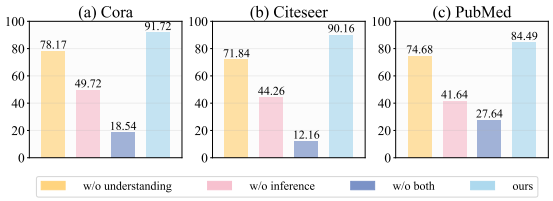
4.3 Exploration Studies

Adding OOD Pre-training Tasks (RQ3). A robust foundation model should support diverse pre-training tasks to improve downstream generalization. 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 *[x]* denotes a node's language-based ID. 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 OOD 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**.

Table 8: The impact of incorporating Node Discrimination (+ND) and Link Discrimination (+LD) as pretraining 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)
	✓	✓	**86.02** (↑1.53)	**83.77** (↑1.46)
PubMed	✗	✗	90.67	91.82
	✓	✗	92.53 (↑1.86)	92.84 (↑1.02)
	✗	✓	91.12 (↑0.45)	92.03 (↑0.21)
	✓	✓	**93.06** (↑2.39)	**93.27** (↑1.45)

Figure 5: Ablation studies on node classification.



Ablation Studies (RQ4). We construct three ablated variants: **(a) w/o understanding**, which removes the prompt-based GNN; **(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 consistently 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.

Efficiency Analysis (RQ5). We evaluate the computational and monetary cost of PromptGFM, as summarized in Table 9. In the graph understanding module, the API or token cost grows 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"). 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); 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). All three rely on decoupled two-stage alignment, motivating our unified LLM-as-GNN paradigm.

Table 9: 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

Graph-Text Alignment in Embedding Space. Existing methods align graph and text representations through graph-encoder prefixes (Huang et al., 2023, 2024), contrastive learning (Tang et al., 2024; Zhu et al., 2025), iterative training (Zhu et al., 2024), or knowledge distillation (Mavromatis et al., 2023). 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.

Graph Foundation Models. GFMs aim for transferability across diverse datasets and tasks, where the central challenge is identifying a graph vocabulary of transferable units (Sun et al., 2025a; Fang et al., 2025). Prior work relies on domain-specific vocabularies: GraphGPT (Tang et al., 2024) assigns unique IDs per dataset, while MoleBERT (Xia et al., 2023) encodes molecular structure. These lack cross-domain transferability; a language-based graph vocabulary, which we propose, remains unexplored. A full discussion appears in Appendix J.

6 Conclusion

We present PromptGFM, a GFM for TAGs built on graph vocabulary learning. By replicating GNNs in text space, PromptGFM decouples refined textual node representations into a unified graph vocabulary. This vocabulary improves compatibility and scalability, enabling effective LLM fine-tuning with readable instructions and stronger transferability. Experiments show superior performance and effective cross-graph and cross-task generalization, revealing the potential of using LLMs as GNNs and opening new directions for GFMs on TAGs.

7 Limitations

We discuss several aspects of the current design that may benefit from further exploration. **(1) Closed-source LLM for node initialization.** Our graph understanding module leverages GPT-4o mini to produce initial node summaries. Since the resulting language-based IDs are cached and reused, the associated API cost is bounded and amortized in practice. Nonetheless, exploring open-source alternatives is a natural direction that could further broaden accessibility for the community. **(2) Prompt length for dense graphs.** For graphs with extreme degree distributions, the one-hop prompt of very high-degree nodes can become lengthy. Our neighbor sampling and 20-neighbor cap (Appendix G) effectively bound this on the benchmarks we study, and additional strategies such as hierarchical summarization may complement our framework when scaling to web-scale graphs. **(3) Cross-domain transferability.** While our language-based vocabulary inherently supports cross-domain reuse, our adaptive prompting study (Appendix H) suggests that highly heterogeneous source-target pairs remain challenging 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.

8 Ethical Considerations

Data. All datasets used in this work are publicly released academic benchmarks (Cora, Citeseer, PubMed, Ogbn-arxiv, WikiCS, Amazon Photo/History). They contain no personally identifiable information and have been widely used in prior graph learning literature. 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 9. 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.

Reproducibility. Anonymized code, prompt templates, and dataset processing scripts are provided at `https://anonymous.4open.science/r/PromptGFM` for review.

References

William Brannon, Suyash Fulay, and Hang Jiang. 2023. Congrat: Self-supervised contrastive pretraining for joint graph and text embeddings. *CoRR*.

Peter Buneman, Dennis Dosso, and Matteo Lissandrini. 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*.

Nicola De Cao, Gautier Izacard, and Sebastian Riedel. 2021. Autoregressive entity retrieval. In *ICLR 2021*.

Ziwei Chai, Tianjie Zhang, and Liang Wu. 2023. Graphllm: Boosting graph reasoning ability of large language model. *CoRR*.

Runjin Chen, Tong Zhao, and Ajay Kumar Jaiswal. 2024a. Llaga: Large language and graph assistant. In *ICML 2024*.

Zhikai Chen, Haitao Mao, and Jingzhe Liu. 2024b. Text-space graph foundation models: Comprehensive benchmarks and new insights. In *NeurIPS 2024*.

Zhikai Chen, Haitao Mao, and Hongzhi Wen. 2024c. Label-free node classification on graphs with large language models (llms). In *ICLR 2024*.

Hyung Won Chung, Le Hou, and Shayne Longpre. 2024. Scaling instruction-finetuned language models. *J. Mach. Learn. Res.*, 25.

Jacob Devlin, Ming-Wei Chang, and Kenton Lee. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019*, pages 4171–4186.

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 *Proceedings of the 3rd ACM International Conference on Digital Libraries, June 23-26, 1998, Pittsburgh, PA, USA*.

Daniele Grattarola, Daniele Zambon, and Filippo Maria Bianchi. 2024. Understanding pooling in graph neural networks. *IEEE Trans. Neural Networks Learn. Syst.*, 35(2).

Zirui Guo, Lianghao Xia, and Yanhua Yu. 2024. Graphedit: Large language models for graph structure learning. *CoRR*.

William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *NeurIPS 2017*.

Xiaoxin He, Xavier Bresson, and Thomas Laurent. 2024. Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning. In *ICLR 2024*.

Weihua Hu, Matthias Fey, and Marinka Zitnik. 2020. Open graph benchmark: Datasets for machine learning on graphs. In *NeurIPS 2020*.

Xuanwen Huang, Kaiqiao Han, and Dezheng Bao. 2023. Prompt-based node feature extractor for few-shot learning on text-attributed graphs. *CoRR*.

Xuanwen Huang, Kaiqiao Han, and Yang Yang. 2024. Can GNN be good adapter for llms? In *WWW 2024*.

Mingyu Jin, Haochen Xue, and Zhenting Wang. 2024. Prollm: Protein chain-of-thoughts enhanced LLM for protein-protein interaction prediction. *CoRR*.

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*.

Guohao Li, Matthias Müller, and Bernard Ghanem. 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, and Peisong Wang. 2023b. A survey of graph meets large language model: Progress and future directions. *CoRR*.

Hao Liu, Jiarui Feng, and Lecheng Kong. 2024a. One for all: Towards training one graph model for all classification tasks. In *ICLR 2024*.

Jiawei Liu, Cheng Yang, and Zhiyuan Lu. 2023. Towards graph foundation models: A survey and beyond. *CoRR*.

Zheyuan Liu, Xiaoxin He, and Yijun Tian. 2024b. Can we soft prompt llms for graph learning tasks? In *WWW 2024*.

Haitao Mao, Zhikai Chen, and Wenzhuo Tang. 2024. Position: Graph foundation models are already here. In *ICML 2024*.

Costas Mavromatis, Vassilis N. Ioannidis, and Shen Wang. 2023. Train your own GNN teacher: Graph-aware distillation on textual graphs. In *ECML PKDD 2023*.

Andrew Kachites McCallum, Kamal Nigam, and Jason Rennie. 2000. Automating the construction of internet portals with machine learning. *Infomration Retrieval.*, 3(2).

Péter Mernyei and Catalina Cangea. 2020. Wiki-cs: A wikipedia-based benchmark for graph neural networks. *CoRR*.

Swaroop Mishra, Daniel Khashabi, and Chitta Baral. 2022. Cross-task generalization via natural language crowdsourcing instructions. In *ACL 2022*.

Seth A. Myers, Aneesh Sharma, and Pankaj Gupta. 2014. Information network or social network?: the structure of the twitter follow graph. In *WWW 2014*.

Norman Di Palo, Arunkumar Byravan, and Leonard Hasenclever. 2023. Towards A unified agent with foundation models. *CoRR*.

Alec Radford, Jong Wook Kim, and Chris Hallacy. 2021. Learning transferable visual models from natural language supervision. In *ICML 2021*, volume 139.

Colin Raffel, Noam Shazeer, and Adam Roberts. 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, and Dawei Yin. 2024. A survey of large language models for graphs. In *KDD 2024*.

Prithviraj Sen, Galileo Namata, and Mustafa Bilgic. 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 *Proceedings of the ACM on Web Conference 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 the Association for Computational Linguistics: NAACL 2025*, pages 2440–2459.

Juntao Tan, Shuyuan Xu, and Wenyue Hua. 2024. Idgenrec: Llm-recsys alignment with textual ID learning. In *SIGIR 2024*.

Jiabin Tang, Yuhao Yang, and Wei Wei. 2024. Graphgpt: Graph instruction tuning for large language models. In *SIGIR 2024*.

Hugo Touvron, Thibaut Lavril, and Gautier Izacard. 2023. Llama: Open and efficient foundation language models. *CoRR*.

Petar Velickovic, Guillem Cucurull, and Arantxa Casanova. 2017. Graph attention networks. *CoRR*.

Petar Velickovic, William Fedus, and William L. Hamilton. 2019. Deep graph infomax. In *ICLR 2019*.

Heng Wang, Shangbin Feng, and Tianxing He. 2023. Can language models solve graph problems in natural language? In *NeurIPS 2023*.

Jianing Wang, Junda Wu, and Yupeng Hou. 2024a. Instructgraph: Boosting large language models via graph-centric instruction tuning and preference alignment. In *Findings of the ACL 2024*.

Zehong Wang, Zheyuan Zhang, and Nitesh V. Chawla. 2024b. GFT: graph foundation model with transferable tree vocabulary. In *NeurIPS 2024*.

Jason Wei, Maarten Bosma, and Vincent Y. Zhao. 2022. Finetuned language models are zero-shot learners. In *ICLR 2022*.

Oliver Wieder, Stefan Kohlbacher, and Mélaine Kuenemann. 2020. A compact review of molecular property prediction with graph neural networks. *Drug Discovery Today: Technologies*, 37.

Qitian Wu, Wentao Zhao, and Zenan Li. 2022. Nodeformer: A scalable graph structure learning transformer for node classification. In *NeurIPS 2022*.

Qitian Wu, Wentao Zhao, and Chenxiao Yang. 2023. Simplifying and empowering transformers for large-graph representations. In *NeurIPS 2023*.

Jun Xia, Chengshuai Zhao, and Bozhen Hu. 2023. Mole-bert: Rethinking pre-training graph neural networks for molecules. In *ICLR 2023*.

Lianghao Xia, Ben Kao, and Chao Huang. 2024. Opengraph: Towards open graph foundation models. In *Findings of the EMNLP 2024*.

Hao Yan, Chaozhuo Li, and Ruosong Long. 2023. A comprehensive study on text-attributed graphs: Benchmarking and rethinking. In *NeurIPS 2023*.

Ruosong Ye, Caiqi Zhang, and Runhui Wang. 2024. Language is all a graph needs. In *Findings of the ACL: EACL 2024*.

Haihong Zhao, Aochuan Chen, and Xiangguo Sun. 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, and Chaozhuo Li. 2023a. Learning on large-scale text-attributed graphs via variational inference. In *ICLR 2023*.

Jianan Zhao, Le Zhuo, and Yikang Shen. 2023b. Graphtext: Graph reasoning in text space. *CoRR*.

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 the Association for Computational Linguistics: 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 *Proceedings of the ACM on Web Conference 2025*, pages 2183–2197.

Yun Zhu, Yaoke Wang, and Haizhou Shi. 2024. Efficient tuning and inference for large language models on textual graphs. In *IJCAI 2024*.

A Datasets

We use seven public benchmarking datasets from three domains to evaluate PromptGFM, with statistics summarized in Table 10. 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 introduces a citation network comprising 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 four categories.

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** (Velickovic et al., 2017). 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

Table 10: 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

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** (Touvron et al., 2023). 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

• **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.

• **OFA** (Liu et al., 2024a). This framework handles various graph classification tasks across different domains using a single model.

• **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.

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 hyper-parameters of the compared methods, we follow the original papers and carefully tune them to each dataset.

D Prompt Design

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 Workflow

We illustrate the prompt-based GNN workflow in Figure 6 and provide three analyses below: permutation sensitivity, representation discrimination across layers, 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 evaluating 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 11, 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.

Table 11: Permutation sensitivity of prompt-based GNNs.

Variant	Cora	Citeseer
shuffle nodes	90.64 ± 1.12	83.79 ± 0.89
shuffle tokens	90.55 ± 1.03	84.28 ± 1.06

Table 12: 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

E.2 Representation Discrimination over Layers

To validate that our language-based graph vocabulary effectively preserves structural proximity, we conduct an analysis 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 12. 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 demonstrating its theoretical equivalence in representational power to traditional embedding-based GNNs.

E.3 Performance over GNN Layers

We investigate the impact of layer depth in our prompt-based GNN. As shown in Figure 7, increasing the number of layers initially improves results by capturing broader context and higher-order dependencies. However, beyond a certain 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

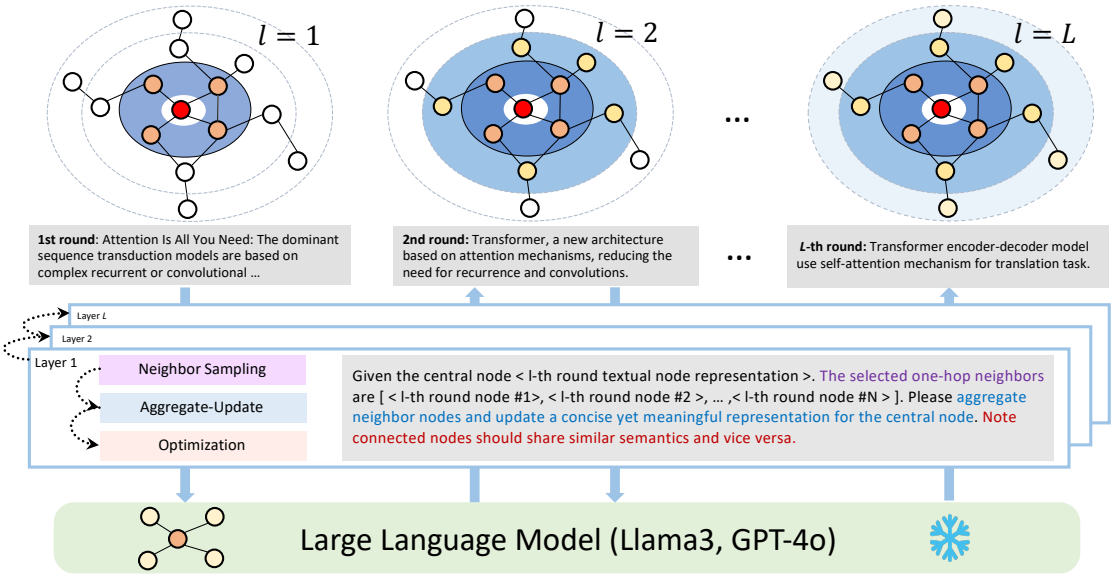


Figure 6: Graph understanding module via prompt-based GNNs. We prompt LLMs to achieve fine-grained reproduction of traditional GNN workflow, refining verbose textual representations into concise yet meaningful ones. In the prompt, neighbor sampling (see Equation 3) is in blue, the aggregation-update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

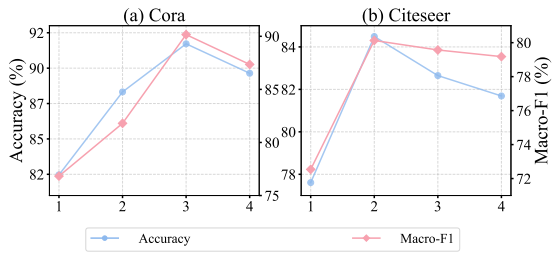


Figure 7: Impact of layer numbers.

same way as numerical embeddings, effectively encoding both semantic and structural information.

F Vocabulary Analysis

We evaluate the quality of our graph vocabulary using two metrics in Table 13. **(a) Coverage.** We compute the proportion of nodes successfully matched to IDs. Since the graph vocabulary is grounded in LLM's native language, any node's semantics can, in principle, be expressed through a combination of natural language tokens. Thus, every node is assigned a valid ID, confirming full coverage. **(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 GFMs.

G Prompt Length Discussion

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 in Scalable GNNs

Following standard practices, we adopt a neighbor sampling strategy in each GNN layer to reduce prompt length and enhance robustness. As shown in Table 15, 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 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

Table 13: Quantitative graph vocabulary analysis.

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

Table 14: 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 15: The performance with varying neighbor sampling ratios across three datasets.

Sampling Ratio	100%	60%	30%
Cora	81.36	**89.35**	88.36
Citeseer	79.34	**82.23**	81.13
PubMed	90.23	**91.12**	89.34

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 14, 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 without information loss.

G.3 Prompt Design Considerations

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 ∼400 to ∼10 tokens as semantic and structural information are 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.

H Cross-Domain Transfer Discussion

H.1 Why does performance drop?

Cross-domain transfer mainly stems from domain shifts, including structure heterogeneity, semantic shift in text, and label space mismatch (especially for node classification), rather than hard vocabulary conflicts. In contrast, PromptGFM adopts a unified language-based vocabulary and pure language interface, which enables natural cross-domain transferability and avoids token collision in principle.

H.2 Adaptive Prompting Exploration

While inter-domain transfer remains a community-wide challenge rather than a limitation of our method, the domain shift assumption inspires us to investigate adaptive prompting. 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/link prediction over a graph representing biomedical research citations, rather than co-purchase relations in e-commerce."* As summarized in Table 16, adaptive prompting (*w/ AP*) consistently improves results across all evaluation settings. These findings highlight adaptive prompting as an effective means of bridging domain gaps and a promising direction toward more generalizable GFMs.

I Case Study

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,

Table 16: The effectiveness of adaptive prompting that alleviates inter-domain transfer challenges.

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

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., "nonlinear 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.

I.1 Case A: Cora

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 Clearning*. ***/

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.

I.2 Case B: PubMed

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.

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 population* 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.

J Full Related Works

GNN-LLM Integration. LLMs have unlocked unprecedented potential for graph machine learning, driving efforts to integrate GNNs and LLMs for modeling TAGs. **(a) GNN for LLM.** As structure tokenizers, GNNs and graph transformers embed graph topology into textual representations, enhancing semantical understanding in LLMs (Tang et al., 2024; Chai et al., 2023; Liu et al., 2024b). However, effectively coordinating GNN-LLM architectures and co-training remains a challenge. **(b) LLM for GNN.** LLMs assist GNNs by generating node-level and edge-level labels to address data sparsity issues (Chen et al., 2024c; Guo et al., 2024; Xia et al., 2024). Meanwhile, the other line directs LLMs to produce additional features or explanations to overcome semantic deficiencies (Liu et al., 2024a; Zhu et al., 2024; He et al., 2024). However, their reliance on LLM-generated content inevitably introduces noise, impacting performance. **(c) LLM as GNN.** This paradigm directly operates LLMs as GNNs by designing structure verbalizers to convert graph data into heuristic or code-like prompts for LLM inference (Chen et al., 2024a; Ye et al., 2024; Wang et al., 2024a). Nevertheless, they fail to capture high-order connections due to the lack of an intrinsic GNN mechanism. Overall, the current decoupled approaches rely on two-stage alignment, failing to fully exploit the strengths of both models. This limitation motivates us to propose a new paradigm where LLMs inherently function as GNNs, maximizing their synergistic potential.

Graph-Text Alignment in Embedding Space. Current approaches to graph-text alignment primarily operate in the embedding space. One approach uses graph encoders as prefixes, mapping graph-aware embeddings to language-based embeddings for LLM fine-tuning in a shared space (Huang et al., 2023, 2024). Other works adopt a two-tower architecture, leveraging contrastive learning (Li et al., 2023a; Brannon et al., 2023; Tang et al., 2024), iterative training (Zhao et al., 2023a; Zhu et al., 2024), or knowledge distillation (Mavromatis et al., 2023) to align distinct representations. However, these methods encounter a persistent modality gap, limiting transferability and scalability across graphs. Since TAGs contain textual information, we advocate for shifting graph-text alignment to the text space.

Graph Foundation Models. A GFM aims to achieve transferability across different datasets and tasks, where the key challenge lies in finding a graph vocabulary that identifies transferable units to encode invariance on graphs (Liu et al., 2023; Mao et al., 2024; Fang et al., 2025). Pre-

vious works rely on domain-specific vocabularies: GraphGPT (Tang et al., 2024) assumes unique IDs to nodes and creates a dataset-specific vocabulary, while MoleBERT (Xia et al., 2023) defines a molecular graph vocabulary by converting atomic properties into chemically meaningful codes. Despite their success, they lack in-context learning and cross-domain transferability. Recently, while some studies have investigated understanding and inferring graphs in natural language (Fatemi et al., 2024; Wang et al., 2023; Zhao et al., 2023b; Liu et al., 2024a), none of them have attempted to establish a language-based graph vocabulary. To address this, we introduce an expressive graph vocabulary to exploit the inherent transferability of native natural language, advancing the development of a more versatile GFM.