LLaGA: Large Language and Graph Assistant

Runjin Chen [1]   **Tong Zhao** [2]   **Ajay Jaiswal** [1]   **Neil Shah** [2]   **Zhangyang Wang** [1]

Abstract

Graph Neural Networks (GNNs) have empowered the advance in graph-structured data analysis. Recently, the rise of Large Language Models (LLMs) like GPT-4 has heralded a new era in deep learning. However, their application to graph data poses distinct challenges due to the inherent difficulty of translating graph structures to language. To this end, we introduce the **L**arge **L**anguage **a**nd **G**raph **A**ssistant (**LLaGA**), an innovative model that effectively integrates LLM capabilities to handle the complexities of graph-structured data. LLaGA retains the general-purpose nature of LLMs while adapting graph data into a format compatible with LLM input. LLaGA achieves this by reorganizing graph nodes to structure-aware sequences and then mapping these into the token embedding space through a versatile projector. LLaGA excels in versatility, generalizability and interpretability, allowing it to perform consistently well across different datasets and tasks, extend its ability to unseen datasets or tasks, and provide explanations for graphs. Our extensive experiments across popular graph benchmarks show that LLaGA delivers outstanding performance across four datasets and three tasks using one single model, surpassing state-of-the-art graph models in both supervised and zero-shot scenarios. Our code is available at `https://github.com/VITA-Group/LLaGA`

1. Introduction

Graphs are omnipresent, representing a myriad of real-world data from social networks, biological networks and recommendation systems, etc. Graph neural networks (GNNs) (Kipf & Welling, 2017; Defferrard et al., 2016; Veličković

et al., 2017), embedded with message passing and aggregation techniques, are powerful algorithmic tools on handling complex graph structures.

Recently, the advent of large language models (LLMs) that have massive context-aware knowledge and semantic comprehension capabilities (*e.g.,* LLaMa (Touvron et al., 2023), GPTs (Achiam et al., 2023), Claude (Perez et al., 2022)) marks a significant advancement in AI research. A key advantage of LLMs is their ability to solve various tasks with a single model, showcasing strong language skills and the capacity to explain provided answers. These models have demonstrated remarkable proficiency not only in language-related tasks but also in understanding and generating visual content (Liu et al., 2023b; Wang et al., 2023). However, the direct application of such models presents challenges over graph-structured data, which inherently contains rich relational and structural information.

Researchers (Fatemi et al., 2023; Chen et al., 2023a) explored ways to translate graph structures into natural language suitable for consumption by language models. Yet, describing graphs in plain texts tends to be verbose and fails to directly represent the intrinsic characteristics of graphs, often leading to repetitive and unintuitive descriptions of nodes and edge relationships. Consequently, LLMs would perform poorly on basic graph tasks without specific adaptations (Chen et al., 2023a). InstructGLM (Ye et al., 2023) describes graphs in language and attempts to enhance LLMs' graph-task performance by task-specific fine-tuning. However, this specialization constrains the model's versatility, potentially limiting its effectiveness in other graph tasks or non-graph-related domains. GraphGPT (Tang et al., 2023) combines text descriptions with a self-supervised graph transformer to incorporate graph data into LLMs. However, the pre-trained graph transformer might not distill all relevant structural information for specific downstream tasks, leading to less satisfactory performances. Motivated by these issues, this work poses an important **question**: *How to develop a framework that effectively encodes structural information for graphs across various tasks and domains, enabling its comprehension by LLMs, while maintaining LLMs' general-purpose ?*

In response, we introduce the **L**arge **L**anguage **a**nd **G**raph **A**ssistant (**LLaGA**), a novel framework that seamlessly in-

[1] The University of Texas at Austin [2] Snap Inc. Correspondence to: Zhangyang Wang <atlaswang@utexas.edu>, Runjin Chen <chenrunjin@utexas.edu>.

tegrates rich graph-structured data with the massive context-awareness skills and comprehension capabilities of Large Language Models. LLaGA has three impressive characteristics that distinguish LLaGA with prior works as follows:

- **Versatility:** LLaGA adopts a simple but universally applicable method for encoding structural details in graphs, and achieves a general alignment between graph and token spaces using a single, versatile projector. This projector efficiently handles various graph tasks across multiple datasets, eliminating the need for task-specific adjustments. Significantly, the performance of our versatile LLaGA framework can even exceed that of specialized task-focused graph models.

- **Generalizability:** Given the comprehensive alignment between graph and token spaces, LLaGA not only excels in those datasets and tasks encountered during training but also demonstrates robust generalization to previously unseen datasets and tasks without additional tuning.

- **Interpretability:** A key feature of LLaGA is its ability to provide detailed interpretations of node embeddings, greatly enhancing the understanding of its decision-making processes.

To achieve this, LLaGA uniquely reorganizes graph data into *node sequences*, without converting structural information into potentially ambiguous natural language descriptions. These sequences are formatted with the help of novel **node-level templates**, to reflect the structural information surrounding each central node while preserving the graph's node features. Note that this transformation is parameter-free, ensuring the preservation of the original structural integrity without necessitating further distillation. Subsequently, LLaGA translates node representations into LLMs' comprehensible token embedding space through a **versatile projector**, which can help in mitigating the expensive computational cost of fine-tuning LLMs as well as keeping LLMs' general purpose. The projector is generally trained on multiple graph datasets across various tasks, such as node classification, link prediction, and node description. This ensures it can interpret graph data from diverse perspectives and ingest an inherent ability to handle multiple tasks (all at once), bolstering its practical utility, and potentially augmenting LLaGA's generalization capabilities across various unseen datasets and tasks.

Notably, unlike traditional multi-task learning methodologies used in GNNs, LLaGA trained all tasks in a uniform Question-Answer format, eschewing the need for task-specific loss functions or heads. Our extensive experiments illustrate that LLaGA achieves a robust alignment between the graphs and token space of LLMs, facilitating the model's application to multiple tasks, unseen test set, and interestingly out-of-distribution datasets.

2. Methodology

In this section, we introduce the details of **LLaGA** framework. We start with the notation setup, followed by a detailed explanation of the method employed for translating graphs into token embedding space. Subsequently, we delve into the training process, encompassing both the design of prompts and tasks as well as the training objectives.

2.1. Notation

A graph is a structure that encapsulates a set of entities and the interrelationships among them. Formally, a graph is denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{X})$. Here, $\mathcal{V}$ denotes the set of nodes (entities). The set of edges, $\mathcal{E}$, represents the connections between the nodes in $\mathcal{V}$. $\mathcal{X}$ is the attribute information corresponding to the nodes. Each node $v_i \in \mathcal{V}$ is associated with an attribute feature $x_i \in \mathcal{X}$. In this paper, our primary focus is on text-attributed graphs, implying that the attributes $x_i \in \mathcal{X}$ of each node are expressed in a textual format. Additionally, we introduce $\mathcal{N}_v^k$ to denote the kth hop neighborhood set surrounding the node v.

2.2. Structure-Aware Graph Translation

The key step of LLaGA is to translate graph inputs into a token embedding space that is comprehensible to LLMs. This translation enables the utilization of LLMs' inherent reasoning capabilities for graph-related tasks, without necessitating any LLM parameter modifications. LLaGA accomplishes this by initially reorganizing nodes in graphs into node embedding sequences. These sequences are structured according to our proposed templates and are then converted into a sequence of token embeddings using a projector.

The first step involves converting graphs into node embedding sequences. Recognizing that the fundamental unit for graph analysis is the node, we developed two node-level templates for analysis on graphs. These templates are versatile, applicable not only to node-level tasks but also to other tasks like link prediction. Both templates are designed to encode structural information surrounding a node, offering different perspectives for analysis. The first, the **Neighborhood Detail Template**, provides an in-depth view of the central node and its immediate surroundings. The second, the **Hop-Field Overview Template**, offers a summarized view of a node's neighborhood, extendable to larger fields.

Neighborhood Detail Template is designed to elaborate on the detailed information of a node and its surrounding neighborhood. Given a node v, we first construct a fixed-shape, sampled computational tree centered around v. For every hop of neighbors, we define a neighbor sample size, denoted as $n_1, n_2, ...$, where n_i indicates the sample size for the ith hop. The computational tree is built with the root node being the central node v. From the 1-hop neighbor set
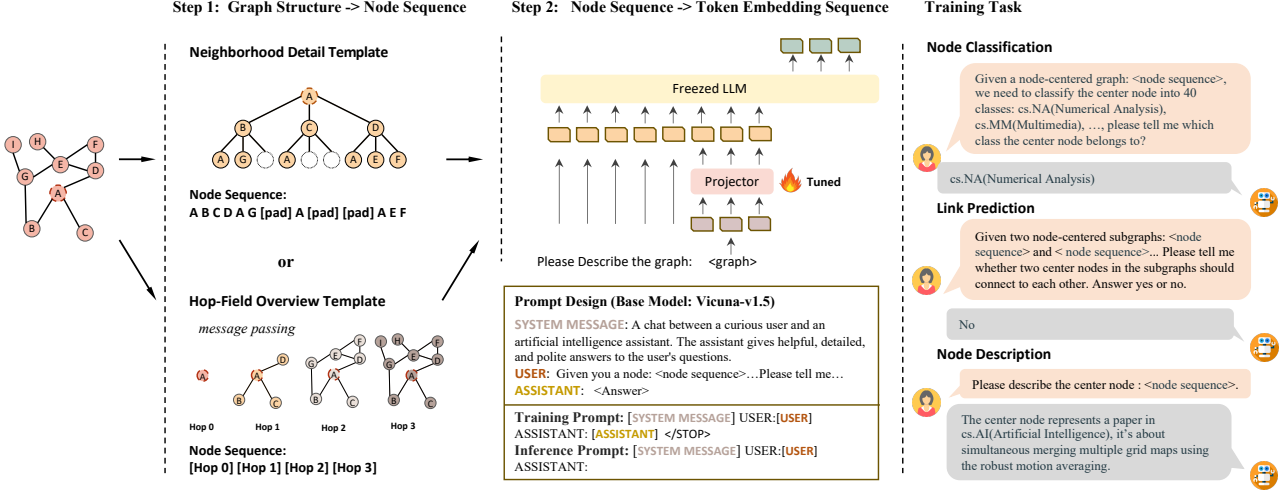
Figure 1. Illustration of LLaGA framework and its prompt design paradigm.

of v, denoted as $\mathcal{N}_v^1$, we randomly select n_1 nodes to form a new neighbor set $\widetilde{\mathcal{N}_v^1}$. If the size of $\mathcal{N}_v^1$ is smaller than n_1, i.e., $|\mathcal{N}_v^1| < n_1$, we supplement the set with placeholder nodes to reach a size of n_1. Therefore, the size of $\widetilde{\mathcal{N}_v^1}$ is consistently n_1, i.e., $|\widetilde{\mathcal{N}_v^1}| = n_1$. The nodes in $\widetilde{\mathcal{N}_v^1}$ are treated as children of the root node. Subsequently, for each node in $\widetilde{\mathcal{N}_v^1}$, we recursively sample n_2 neighbors as its children. Any sets with insufficient nodes are filled with placeholder nodes. For any placeholder node, its children are exclusively placeholder nodes. As illustrated in upper-left of Figure 1, with the root node being A, we display a 2-hop neighbor structure of A, with the sample size of 3 for both hops. The first-order neighbors of A are $\{B, C, D\}$, so they are shown in the second layer of the computational graph. Since B has 2 neighbors $\{A, G\}$, we expand this set to $\{A, G, [pad]\}$, where $[pad]$ represents the placeholder node. And similarly for nodes C and D. Ultimately, this process yields a perfect 3-ary computational tree centered around node A. We then perform a level-order traversal on the computational tree, transforming the comprehensive details of the central node and its neighborhood into a fixed-length node sequence. For instance, in Figure 1, the sequence representing node A and its neighborhood is $A\ B\ C\ D\ A\ G\ [pad]\ A\ [pad]\ [pad]\ A\ E\ F$, where each sequence position uniquely corresponds to a relative structural position within the original graph.

Post conversion of the center node and its structural information into a node sequence, we shift to mapping them into the node embedding space. In the context of text-attributed graphs, we can utilize various off-the-shelf text encoding models ϕ, such as SBERT (Reimers & Gurevych, 2019), RoBERTa (Liu et al., 2019), and SimTeG (Duan et al., 2023), to encode text features. Placeholder nodes are represented by a zero vectors of the same size. We further integrate a Laplacian Embedding (Dwivedi & Bresson, 2020) at each sequence position, enhancing the representation of struc-

tural information. Denoting the adjacency matrix of the computational tree by $\mathcal{A}_{tree}$, the Laplacian Embedding is defined as the eigenvectors of the Laplacian matrix of $\mathcal{A}_{tree}$:

$$L = I - \mathcal{D}^{-\frac{1}{2}} \mathcal{A}_{tree} \mathcal{D}^{-\frac{1}{2}} = U^T \Lambda U \qquad (1)$$

where $\mathcal{D}$ represents the degree matrix of $\mathcal{A}_{tree}$ and U symbolizes the Laplacian Embedding of the template. Notably, with a fixed sample size, the computational tree's shape remains unchanged, so the Laplacian Embedding is computed *only once* for all graphs using this template. This Embedding is then appended to the encoded node feature to form the final node embedding. The process is outlined as follows: Let $v_1, v_2, ..., v_n$ represent the encoded node sequence. The final node embedding h_{v_i} for v_i is given by

$$h_{v_i} = \begin{cases} \mathbf{0}\ ||\ U_i, & \text{if } v_i = [pad]; \\ \phi(x_{v_i})\ ||\ U_i, & \text{otherwise,} \end{cases} \qquad (2)$$

where $||$ denotes concatenation. Subsequently, the central node and its structural information are transformed into the node embedding sequence $h_{v_1}, h_{v_2}, ..., h_{v_n}$.

Hop-Field Overview Template provides a summarized view of the central node and its neighborhood. This template employs hop embeddings to characterize the node features across various neighborhood hops. These hop embeddings are obtained through *parameter-free* message passing on encoded text features. For each central node v, the ith-hop embedding h_v^i is calculated as follows:

$$h_v^i = \frac{1}{|\mathcal{N}_v^1|} \sum_{v' \in \mathcal{N}_v^1} h_{v'}^{i-1}, \qquad (3)$$

where $h_x^0 = \phi(x_v)$. Through this calculation, h_v^i potentially contains information from all neighbors in the ith-hop neighborhood set $\mathcal{N}_v^i$. A sequence of hop embeddings h_v^0, h_v^1, h_v^2,

. . . can represent the central node and its structural information. Unlike the Neighborhood Detail Template, which utilizes individual embeddings for each neighbor, the Hop-Field Overview Template summarizes each hop's neighbors with a single embedding. This approach may sacrifice some detail for the sake of a broader respective field. The choice between these templates should be based on the nature of the input data and the required level of detail.

To enhance the natural comprehension of graph inputs by Large Language Models (LLMs), it is essential to align the node embedding space with the input token space. This alignment is realized by mapping each node embedding into the token embedding space, utilizing a specifically calibrated projector, denoted as f_θ. Mathematically, this process can be represented for a given node embedding h_i as:

$$e_i = f_\theta(h_i). \tag{4}$$

Consequently, a sequence of node embeddings, h_1, h_2, ..., h_n, is transformed into a corresponding sequence of token embeddings, e_1, e_2, ..., e_n. In our framework, this transformation is facilitated by a simple MLP serving as the projector. It is important to note that the parameters θ of the projector are the only parameters subject to tuning during the training process of LLaGA.

2.3. Alignment Tuning

In LLaGA, we employ three key tasks on graphs – node classification, link prediction, and node description – to meticulously tune the projector. The first two tasks, node classification and link prediction, are well-established and widely recognized in the field of graph ML. Contrastingly, the node description task, which is somewhat less common in conventional graph analysis, is designed to align node embeddings with specific descriptive texts. This innovative task enables the provision of rich semantic interpretations of the graphs, providing a deeper insight of the logic behind graph-based predictions. The questions and answers to this task can be articulated as follows: **Questions:** Please describe the center node: <node sequence>. **Answers:** The center node represents a [paper / products /...], it's about [node description]. For textual-attributed graphs, the node description can be obtained from node features. By integrating these three diverse tasks into the training process, our projector develops a comprehensive and nuanced understanding of graphs and can serve as a versatile translator between node embedding and token embedding space for all those tasks. Moreover, it can explicitly generate explanations for node embeddings, enhancing interpretability.

During training, we organize our questions and answers in a chat format. In our experiments, Vicuna (Chiang et al., 2023) serves as the primary foundational Large Language Model (LLM), so we follow the implementation strategy of Vicuna and set the system message accordingly. For details regarding the question-answer template and the training or inference input sequences, please refer to the illustrations in Figure 1. In the input processing phase, we tokenize all words in the prompt and convert them into their respective token embeddings. For the <node sequence>, we substitute this part with the projected node embeddings e_1, e_2, ..., e_n, maintaining their original positions. The training objective is to maximize the probability of generating the correct answer, formulated as

$$\underset{\theta}{\text{maximize}}\; p(X_{answer}|X_{graph}, X_{question}, X_{system}). \tag{5}$$

3. Experimental Results

We conduct comprehensive experiments to validate the effectiveness of our framework across various settings, aiming to address several key research questions:

- **RQ1:** How does LLaGA perform in comparison to baseline models in standard graph tasks, such as node classification and link prediction?

- **RQ2:** How good are the interpretations generated by LLaGA for node embeddings?

- **RQ3:** How effectively does the model transfer knowledge when adapting to new datasets or tasks in zero-shot?

- **RQ4:** What is the contribution of our encoding templates to the overall performance?

3.1. Setup

Datasets. We train and evaluate our model on four widely-recognized graph datasets: ogbn-Arxiv (Hu et al., 2020), ogbn-Products (Hu et al., 2020), Pubmed, and Cora (Yang et al., 2016). These datasets span domains of citation networks and e-commerce, varying in terms of sparsity and size, ranging from small to large scales. Detailed statistics and data splitting methods are presented in Appendix A.

Tasks. Our model utilizes LLaGA for 3 tasks: node classification, link prediction, and graph-based node description. The targets of ***node classification*** are to categorize nodes based on research topics or product characteristics. In the ***link prediction*** task, we predict the existence of edges between node pairs. The ***node description*** task involves generating node descriptions based on encoded node embeddings. The training ground truth is derived from classification labels and text features, structured as: *The center node represents a paper/product in the [label] domain, it's about [text feature].*

Evaluation Metrics. For evaluation metrics, we employ ***Accuracy*** for both node classification and link prediction tasks, Sbert score and Description Label Accuracy for the

node description task. The ***Sbert score*** measures the similarity between embeddings of the generated descriptions and the ground truth descriptions encoded by Sbert. ***Description Label Accuracy*** represents the Accuracy of labels inferred from node descriptions. For LLaGA framework, a sample is considered accurate only if it precisely identifies the category's full name in its response. Besides, we also employ AUC and Hits@k metrics for evaluating link prediction tasks, with details provided in AppendixB.

Implementation Details. In our model's implementation, we primarily employ Vicuna-7B-v1.5-16K (Chiang et al., 2023) as the foundational base models, and SimTeg (Duan et al., 2023) as default text-encoding model. Additionally, we conduct a comparative analysis of various base LLMs and embeddings in Appendix D and E. The learning rate is consistently set to 2e-5, and the batch size is maintained at 16 for all models. We train our model for one epoch. However, to compensate for the limited data size, we replicate the training samples from the smallest dataset, Cora, three times. For the Neighborhood Detail Template, we sample two-hop neighbors around each node, setting the sample size to 10 for each hop. In the Hop-Field Overview Template, 4 hop embeddings are employed to encapsulate the structural information surrounding the central node. We denote LLaGA implementations with the Neighborhood Detail Template and Hop-Field Overview Template as **LLaGA-ND-7B** and **LLaGA-HO-7B**, respectively.

Baselines. In our comparative analysis, we benchmark our framework against three categories of state-of-the-art models to ensure a thorough evaluation. The first category comprises Graph Neural Networks, including GCN (Kipf & Welling, 2016), GraphSage (Hamilton et al., 2017), GAT (Veličković et al., 2018), SGC (Wu et al., 2019), and SAGN (Sun et al., 2021). The second category encompasses transformer-based graph models, NodeFormer (Wu et al., 2022). The final category is represented by GPT-3.5, a leading general LLM. For the first two categories, identical text-encoding methods are employed to encode text features, ensuring a fair comparison. For GPT-3.5, we utilized node classification results from the survey by Chen et al. (Chen et al., 2023a) and extended this approach to the link prediction task by employing a consistent graph-description prompt format. In addition, we also compare with the concurrent work, GraphGPT (Tang et al., 2023).

3.2. Overall Performance Comparison (RQ1)

We compare our LLaGA model with various baselines across four distinct settings: Single Focus, Task Expert, Classification Expert, and General Model. The ***Single Focus*** setting involves models trained on a single dataset for a specific task, thereby concentrating exclusively on that task. ***Task Expert*** refers to models trained across all datasets

but focused on a single task, enabling them to perform as specialists in that area. In the ***Classification Expert*** setting, models are trained on all datasets for both node classification and link prediction tasks. The ***General Model*** is trained for node classification, link prediction, and node description across all datasets, equipping the model to handle not just classification tasks but also semantic tasks like node description. The comparative results are presented in Table 1. Notably, when implementing the GNN-based or Transformer-based baselines in the Task Expert or Classification Expert settings, they were trained using a multi-task learning approach, which incorporates a shared backbone with task-specific classification heads for different datasets or tasks. In contrast, our LLaGA framework employs a single projector to handle all tasks.

Comparision with Baselines: Our analysis reveals three key observations: *Observation 1: LLaGA framework demonstrates superior performance compared to baseline models across all settings, particularly in multi-task learning scenarios.* This highlights LLaGA's versatility and robust capability in addressing various graph tasks.

Observation 2: While many baseline models experience significant performance degradation in multi-task learning scenarios, LLaGA stands out by exhibiting minimal decline or even improvements in performance. This reflects LLaGA's proficiency in extracting common patterns across different datasets and tasks. Such a trait hints at the potential for developing a powerful multi-model LLM equipped with simple projectors.

Observation 3: Both the Neighborhood Detail Template and the Hop-Field Overview Template exhibit distinct advantages. The Neighborhood Detail Template excels in tasks requiring detailed neighbor information, whereas the Hop-Field Overview Template is more effective in tasks that depend on a broader overview of neighbor information with a larger receptive field. For instance, in identifying product categories, it is illogical to classify a product as 'Video Games' based solely on many of its neighbors being 'Electronics'. A more detailed analysis, revealing numerous 'Nintendo Switch' neighbors, makes classification more accurate, as seen in the case of the ogbn-Products dataset. Conversely, for some citation graphs, an overview of a paper's neighboring categories can be more informative, making the Hop-Field Overview Template the preferable choice.

Comparison with Concurrent Work: We conduct a comparative analysis with our concurrent work, GraphGPT (Tang et al., 2023). GraphGPT is a generalizable model designed for solving graph tasks using LLM. It employs a text-encoding model to extract node features and utilizes a pre-trained graph transformer for encoding structural information. In our comparison, we focus on our most robust and generalizable models, with the results detailed

Table 1. Performance comparison with baseline models on both node classification and link prediction under 4 settings. ***Single Focus*** denotes models trained on a single task and dataset. ***Task Expert*** refers to models trained exclusively on one task across all datasets, specializing in that task.***Classification Expert*** indicates models trained in both node classification and link prediction on all datasets, becoming proficient in classification tasks. ***General Model*** are capable of handling classification tasks across datasets and excel in semantic tasks, such as generating interpretable descriptions for node embeddings. (**bold** signifies the **best result across all methods**, while underline highlights the best baseline result under this setting)

MODEL TYPE	MODEL	NODE CLASSIFICATION ACCURACY(%)				LINK PREDICTION ACCURACY(%)			
		ARXIV	PRODUCTS	PUBMED	CORA	ARXIV	PRODUCTS	PUBMED	CORA
SINGLE FOCUS	GCN	73.72	80.75	92.96	88.93	91.43	93.95	90.91	81.59
	GRAPHSAGE	76.29	82.87	94.87	88.89	91.64	94.96	90.64	79.15
	GAT	74.06	83.06	92.33	88.97	85.99	93.85	83.96	80.06
	SGC	71.77	75.47	87.35	87.97	87.99	88.51	83.60	80.94
	SAGN	75.70	82.58	**95.17**	89.19	90.62	94.85	90.48	79.88
	NODEFORMER	74.85	83.72	94.90	88.23	91.84	90.93	77.69	77.26
	LLAGA-ND-7B	75.98	84.60	95.03	88.86	91.24	**97.36**	**91.41**	83.79
	LLAGA-HO-7B	**76.66**	**84.67**	95.03	**89.22**	**94.15**	95.56	89.18	**86.82**
TASK EXPERT	GCN	71.45	80.88	89.25	81.62	88.51	93.54	81.01	78.88
	GRAPHSAGE	72.56	82.50	94.15	81.99	87.76	93.49	76.14	80.74
	GAT	72.19	82.61	87.97	83.58	82.58	92.03	76.85	79.76
	NODEFORMER	72.35	82.99	94.41	83.27	84.11	93.42	80.40	81.03
	LLAGA-ND-7B	**76.41**	**84.60**	94.78	88.19	91.20	**97.38**	**93.27**	**89.41**
	LLAGA-HO-7B	76.40	84.18	**95.06**	**89.85**	**94.36**	95.85	88.88	87.50
CLASSIFICATION EXPERT	GCN	70.95	80.02	89.00	82.77	89.67	93.02	78.79	79.82
	GRAPHSAGE	71.91	81.62	91.81	82.44	89.23	92.22	75.36	82.09
	GAT	70.90	81.83	87.72	82.07	85.18	92.11	75.00	80.35
	NODEFORMER	63.20	75.55	89.50	69.19	82.33	75.42	78.22	81.47
	LLAGA-ND-7B	75.85	**83.58**	**95.06**	87.64	90.81	**96.56**	**92.36**	87.35
	LLAGA-HO-7B	**75.99**	83.32	94.80	**89.30**	**94.30**	96.05	88.64	**88.53**
GENERAL MODEL	GPT3.5-TURBO	55.00	75.25	88.00	71.75	63.80	60.30	68.70	65.74
	LLAGA-ND-7B	74.29	**82.21**	92.42	**87.82**	90.53	**96.82**	86.31	81.91
	LLAGA-HO-7B	75.01	82.07	94.45	**87.82**	92.04	86.80	**89.81**	84.41

Table 2. Compare with Concurrent Work.

MODEL	ARXIV NC	PUBMED NC	PUBMED LP
GRAPHGPT-MIX-7B	64.76	74.16	58.86
GRAPHGPT-STD-7B	63.90	–	80.26
LLAGA-ND-7B(GENERAL)	74.29	92.42	86.31
LLAGA-HO-7B(GENERAL)	**75.01**	**94.45**	**89.81**

in Table 2, GraphGPT's results are referenced directly from its original paper. 'Mix' and 'Std' represent two categories of prompts used in GraphGPT's training process. LLaGA's most general model is trained across 12 tasks, including node classification, link prediction, and node description on datasets such as Arxiv, Products, Pubmed, and Cora. In contrast, GraphGPT's most general model is trained on just three tasks: node classification on Arxiv and Pubmed, and link prediction on Pubmed. But our model still demonstrates superior performance on these three tasks, highlighting the efficacy of our LLaGA framework.

3.3. Interpretation Ability Investigation (RQ2)

As previously stated, our LLaGA framework excels in providing comprehensive interpretations of node embeddings. We initially assess LLaGA's performance in the node description task using several quantitative metrics, with results presented in Table 4. The ***Sbert Score*** indicates the semantic similarity between the ground truth and LLaGA-generated text, measured using Sbert. We also include a ***Base value*** for your reference, representing the average similarity across two randomly chosen samples. Notably, LLaGA's Sbert score significantly exceeds this base value, demonstrating its effectiveness in generating meaningful and relevant descriptions for node embeddings. Furthermore, the high accuracy in extracting labels from these descriptions corroborates the precision of the generated content.

To further illustrate this, Table 3 showcases sample descriptions. These examples indicate the high quality of text produced by LLaGA. Even in some instances where LLaGA's

Table 3. Examples Demonstrating the Interpretability of the LLaGA Framework.

Title: Real time hand gesture detection and classification using convolutional neural networks.
Abstract: Real-time recognition of dynamic hand gestures from video streams is a challenging task since (i) there is no indication when a gesture starts and ends in the video, (ii) performed gestures should only be recognized once, and (iii) the entire architecture should be designed considering the memory and power budget. In this work, ...
Label: cs.CV(Computer Vision and Pattern Recognition)

USER: Please describe the center node: <node embedding sequence>.
LLaGA: This node represents a paper in **cs.CV(Computer Vision and Pattern Recognition)** domain, it's about learning to detect and recognize hand gestures using convolutional neural networks.

Title: An efficient graph convolutional network technique for the travelling salesman problem.
Abstract: This paper introduces a new learning-based approach for approximately solving the Travelling Salesman Problem on 2D Euclidean graphs. We use deep Graph Convolutional Networks to build efficient TSP graph representations and output tours in a non-autoregressive manner via highly parallelized beam search. Our approach ...
Label: cs.LG(Machine Learning)

USER: Please describe the center node: <node embedding sequence>.
LLaGA: This node represents a paper in **cs.LG(Machine Learning)** domain, it's about deep graph convolutional networks for solving the traveling salesman problem.

Title: Inverse boosting pruning trees for depression detection on twitter
Abstract: Depression is one of the most common mental health disorders, and a large number of depression people commit suicide each year... We argue that it is feasible to identify depression at an early stage by mining online social behaviours. Our approach, ...
Label: cs.LG(Machine Learning)

USER: Please describe the center node: <node embedding sequence>.
LLaGA: This node represents a paper in **cs.SI(Social and Information Networks)** domain, it's about predicting suicide risk using social media data. (Label is different from ground truth, but also reasonable)

Table 4. Quantitative evaluation of the node description task using Sbert Score and Description Label Accuracy. The term ***Base value*** refers to the mean Sbert similarity calculated between the ground truth descriptions of two randomly selected samples.

DATASET	MODEL	BASE VALUE	SBERT SCORE	ACC
ARXIV	LLaGA-ND-7B	0.2231	0.6023	74.64
	LLaGA-HO-7B		0.6228	75.49
PRODUCTS	LLaGA-ND-7B	0.1513	0.4952	83.18
	LLaGA-HO-7B		0.5193	84.60
PUBMED	LLaGA-ND-7B	0.4869	0.6847	92.27
	LLaGA-HO-7B		0.6934	94.27
CORA	LLaGA-ND-7B	0.3221	0.6465	86.72
	LLaGA-HO-7B		0.6545	86.90

label predictions diverge from the ground truth, its results are found to be reasonable and LLaGA effectively utilizes its generated text to substantiate these plausible interpretations.

3.4. Zero-Shot Ability Investigation (RQ3)

In this section, we illustrate the generalization capabilities of LLaGA, concentrating on the task of link prediction and node classification within a zero-shot setting. Zero-shot learning entails training a model on certain datasets and subsequently evaluating it on unseen datasets or tasks. This approach is instrumental in assessing a model's proficiency in transferring knowledge. In our study, we examine LLaGA's zero-shot performance in both in-domain and out-of-domain transfer scenarios. For in-domain transfer, the model is

Table 5. Zero-Shot on Link Prediction

TRAIN → TEST	MODEL	ACCURACY
ARXIV+PUBMED ↓ CORA	GCN	58.97
	GRAPHSAGE	67.68
	GRAPHGPT-7B	50.74
	LLaGA-ND-7B	**86.47**
	LLaGA-HO-7B	**87.35**
ARXIV+PUBMED+CORA ↓ PRODUCTS	GCN	56.73
	GRAPHSAGE	58.92
	GRAPHGPT-7B	50.74
	LLaGA-ND-7B	**92.65**
	LLaGA-HO-7B	**92.99**

trained on the Arxiv and Pubmed datasets and evaluated on the Cora dataset. All three datasets comprise citation graphs. Conversely, for out-of-domain transfer, training is conducted on the Arxiv, Pubmed, and Cora datasets, with the evaluation on the Products dataset. Here, while the training datasets are citation graphs, the test set consists of e-commerce graphs. For the link prediction task, we train and test the model both on link prediction tasks. The node classification task, however, presents unique challenges in applying zero-shot learning due to distinct label sets and varied knowledge requirements across tasks. A universal aspect potentially transferable across all node classification tasks is the alignment between the graph structure and the semantic token space. To this end, we trained models on node description tasks from training datasets to establish a generalized alignment between the graph structure and the token space, and then tested this alignment on node classifi-

Table 6. Zero-Shot on Node Classification

TRAIN → TEST	PROMPT TYPE	MODEL	ACCURACY(%)
ARXIV+PUBMED → CORA (TEST TASK: 7 CATEGORIES)	ONLY NODE EMBEDDING	GRAPHGPT-7B	8.30
		LLaGA-7B	**34.69**
	NODE EMBEDDING+TEXT ATTRIBUTES	GRAPHGPT-7B	44.65
		LLaGA-7B	**59.59**
ARXIV+PUBMED+CORA → PRODUCTS (TEST TASK: 47 CATEGORIES)	ONLY NODE EMBEDDING	GRAPHGPT-7B	1.40
		LLaGA-7B	**13.89**
	NODE EMBEDDING+TEXT ATTRIBUTES	GRAPHGPT-7B	18.84
		LLaGA-7B	**43.79**

cation tasks using testing datasets. Since traditional GNNs rely on task-specific classification heads and new classification tasks may have different label sets, they are unsuitable for zero-shot learning on node classification tasks. Our node classification comparison was limited by the use of llm-based baselines, specifically GraphGPT. We conducted the tests using two different prompts to evaluate node classification capabilities. In the first prompt, the model is only supplied with node embedding sequences, containing both attribute and structural information of the central node. The second prompt enhances this by also incorporating the textual attributes of the central node to assist the model.

We present the link prediction results in Table 5 and node classification results in Table 6. The results indicate that our model exhibits robust zero-shot capabilities across all scenarios. This superiority is attributed to LLaGA's comprehensive alignment between the graph space and the token space, enabling the model to effectively discern and leverage similar patterns across datasets, adeptly transferring knowledge not only to analogous data but also to datasets that significantly differ in domain. Moreover, the evaluation of the node classification task shows that including the central node's textual attributes appears to offer some advantages in zero-shot scenarios. However, prompts based solely on node sequence embeddings show potential for application to graphs whose node attributes are challenging to describe textually, such as non-textual graphs.

3.5. Templates Ablation Study (RQ4)

We conduct an ablation study to examine the individual contributions of our encoding templates. For this purpose, we train a new model under a classification expert setting, omitting parts of our template design.

We use 'None' to denote the model that does not utilize templates and relies solely on the embedding of the center node for prediction, instead of a node embedding sequence that captures structural information surrounding the center node. 'ND w/o Order' refers to using the ND template but shuffling the order of all neighbors, thereby disregarding

Table 7. Templates Ablation Study.

TASK	TEMPLATE	ARXIV	PRODUCTS	PUBMED	CORA
NC	NONE	73.92	80.45	94.60	84.50
	ND W/O ORDER	74.35	82.87	94.93	86.16
	ND W/O LAP	75.53	82.77	94.70	86.35
	ND	75.85	83.58	95.06	87.64
	HO	75.99	83.32	94.80	89.30
LP	NONE	89.98	91.73	78.19	83.97
	ND W/O ORDER	90.16	96.31	81.32	80.00
	ND W/O LAP	90.59	96.26	84.48	85.88
	ND	90.81	96.56	92.36	87.35
	HO	94.30	96.05	88.64	88.53

the structural information encoded into the sequence order. 'ND w/o Lap' indicates that we do not use the Laplacian embedding. The results are presented in Table 7. It is clear that both the Neighborhood Detail Template and the Hop-Field Overview Template significantly enhance performance compared to the model without a template, particularly in the link prediction task, which heavily depends on structural information. Additionally, the sequence order and Laplacian embedding play crucial roles in providing structural information. These findings highlight the effectiveness of our templates in capturing the structural information of nodes.

4. Related Work

4.1. Graph Neural Networks

GNNs have long been at the forefront of graph machine learning. They are designed to transform input nodes into compact vector representations, suitable for subsequent classification tasks when paired with a classification head. A common strategy among many GNNs (Kipf & Welling, 2016; Veličković et al., 2018; Xu et al., 2018; Gao et al., 2018; Chiang et al., 2019; You et al., 2020; Chen et al., 2018; Thekumparampil et al., 2018), involves a layer-wise message-passing mechanism. This approach enables nodes to progressively aggregate and process information from their immediate neighbors, thereby embedding the nodes into lower-dimensional spaces. Concurrently, a growing body of research (Yun et al., 2019; Ying et al., 2021; Wu

et al., 2022; Chen et al., 2022; Dwivedi et al., 2023), has been exploring the integration of transformer-based encoders for graph data analysis, opening new avenues for enhancing GNN capabilities. Some studies (Zhao et al.) also propose using learning techniques beyond message passing. However, a significant limitation of traditional graph models is their poor task generalization capability. GNNs are usually trained on a single classification task. When applied to a variety of datasets or downstream tasks, these models often fail to perform consistently well across all tasks with one single model (Ju et al., 2023).

4.2. Self-Supervised Learning for GNNs

Recent advancements have employed self-supervised learning strategies on GNNs to bolster their generalization performance. These methods encompass developing specialized pretext tasks for graph structures, such as mutual information maximization (Veličković et al., 2019; Hassani & Khasahmadi, 2020), whitening decorrelation (Zhang et al., 2021), and generative reconstruction (Hou et al., 2022). Moreover, investigations into integrating multi-task learning with self-supervised learning paradigms have been conducted, offering novel insights into enhancing model generalization ability (Ju et al., 2023). However, these methods still require task-specific classification heads and tuning for every downstream task, after obtaining a general embedding from the graph encoder.

4.3. Large Language Models for Graphs

Recent studies have explored integrating Large Language Models (LLMs) with GNNs, leveraging LLMs' extensive knowledge for graph data enhancement. Research has focused on augmenting GNNs with LLMs to enrich graph textual attributes or provide high-quality node representations (Ye et al., 2023; Chen et al., 2023b; Tang et al., 2023; Guo et al., 2023; He et al., 2023; Huang et al., 2023; Jin et al., 2023). Most recently, OFA (Liu et al., 2023a) introduces a graph prompting paradigm that incorporates task-specific information directly into the input graph, enabling varied task handling. However, these methods still rely heavily on GNNs for predictions.

A complementary line of efforts aim to use LLMs as the backbone for performing graph tasks, allowing for direct interactions and reasoning over graphs with language, positioning LLMs as potential foundational models for graph analysis (Mao et al., 2024; Liu et al., 2023c). Efforts to linguistically represent graphs for direct LLM processing have encountered challenges in effectively translating structures into natural language, often yielding suboptimal results (Huang et al., 2023; Guo et al., 2023). Graphllm (Chai et al., 2023) focuses on customizing a task-specific, graph-enhanced prefix for the keys (K) and values (V) in each layer,

with the goal of improving LLMs' ability to reason about graphs. GraphGPT (Tang et al., 2023) sought to find a pretrained graph transformer for encoding graph structures for LLMs, though finding a universally applicable graph model proved difficult. Our contribution differs from prior arts by introducing a novel encoding method that translates graph data into sequences directly compatible with LLMs, avoiding the need for intermediary models. This method shows superior versatility and generalizability across a range of tasks, even in zero-shot scenarios, outperforming traditional graph models.

5. Conclusion

This paper introduces LLaGA, an innovative framework that effectively integrates LLMs into the graph domain while preserving their proficiency in other tasks. Instead of using complex language for describing structure information, LLaGA employs templates to transform graph structure into sequences, and then maps node embeddings to token embedding spaces using a tuned projector. This projector establishes a comprehensive alignment between texts and graphs, enabling the use of LLMs for fundamental graph tasks like node classification and link prediction across various datasets. And it can be further generalized to unseen datasets or tasks without any adaption. Additionally, it facilitates the generation of textual explanations for node embeddings. Through extensive evaluations in different settings, our method has demonstrated its effectiveness in both supervised and zero-shot graph learning scenarios.

Impact Statement

The broader impact of LLaGA extends to numerous fields where graph data is pivotal, including but not limited to, bioinformatics, social network analysis, and knowledge graphs. As we push the boundaries of Machine Learning and AI, we recognize the importance of monitoring for unintended consequences, such as the perpetuation of biases or misuse of predictive insights. To this end, we encourage continued ethical evaluation and the development of guidelines to ensure that the applications of LLaGA contribute constructively to society. This work aspires to be a stepping stone towards more sophisticated, equitable, and transparent AI systems that respect the intricate structure of data across various domains.

Acknowledgement

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Chai, Z., Zhang, T., Wu, L., Han, K., Hu, X., Huang, X., and Yang, Y. Graphllm: Boosting graph reasoning ability of large language model. *arXiv preprint arXiv:2310.05845*, 2023.

Chen, J., Ma, T., and Xiao, C. Fastgcn: fast learning with graph convolutional networks via importance sampling. *arXiv preprint arXiv:1801.10247*, 2018.

Chen, J., Gao, K., Li, G., and He, K. Nagphormer: A tokenized graph transformer for node classification in large graphs. In *The Eleventh International Conference on Learning Representations*, 2022.

Chen, Z., Mao, H., Li, H., Jin, W., Wen, H., Wei, X., Wang, S., Yin, D., Fan, W., Liu, H., et al. Exploring the potential of large language models (llms) in learning on graphs. *arXiv preprint arXiv:2307.03393*, 2023a.

Chen, Z., Mao, H., Wen, H., Han, H., Jin, W., Zhang, H., Liu, H., and Tang, J. Label-free node classification on graphs with large language models (llms). *arXiv preprint arXiv:2310.04668*, 2023b.

Chiang, W.-L., Liu, X., Si, S., Li, Y., Bengio, S., and Hsieh, C.-J. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 257–266, 2019.

Chiang, W.-L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J. E., Stoica, I., and Xing, E. P. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.

Duan, K., Liu, Q., Chua, T.-S., Yan, S., Ooi, W. T., Xie, Q., and He, J. Simteg: A frustratingly simple approach improves textual graph learning. *arXiv preprint arXiv:2308.02565*, 2023.

Dwivedi, V. P. and Bresson, X. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020.

Dwivedi, V. P., Liu, Y., Luu, A. T., Bresson, X., Shah, N., and Zhao, T. Graph transformers for large graphs. *arXiv preprint arXiv:2312.11109*, 2023.

Fatemi, B., Halcrow, J., and Perozzi, B. Talk like a graph: Encoding graphs for large language models. *arXiv preprint arXiv:2310.04560*, 2023.

Gao, H., Wang, Z., and Ji, S. Large-scale learnable graph convolutional networks. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018.

Guo, J., Du, L., and Liu, H. Gpt4graph: Can large language models understand graph structured data? an empirical evaluation and benchmarking. *arXiv preprint arXiv:2305.15066*, 2023.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Hassani, K. and Khasahmadi, A. H. Contrastive multiview representation learning on graphs. In *International conference on machine learning*, pp. 4116–4126. PMLR, 2020.

He, X., Bresson, X., Laurent, T., Perold, A., LeCun, Y., and Hooi, B. Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning. *arXiv preprint arXiv:2305.19523*, 2023.

Hou, Z., Liu, X., Cen, Y., Dong, Y., Yang, H., Wang, C., and Tang, J. Graphmae: Self-supervised masked graph autoencoders. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 594–604, 2022.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

Huang, J., Zhang, X., Mei, Q., and Ma, J. Can llms effectively leverage graph structural information: when and why. *arXiv preprint arXiv:2309.16595*, 2023.

Jin, B., Zhang, W., Zhang, Y., Meng, Y., Zhang, X., Zhu, Q., and Han, J. Patton: Language model pretraining on text-rich networks. *arXiv preprint arXiv:2305.12268*, 2023.

Ju, M., Zhao, T., Wen, Q., Yu, W., Shah, N., Ye, Y., and Zhang, C. Multi-task self-supervised graph neural networks enable stronger task generalization. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1tHAZRqftM.

Kipf, T. and Welling, M. Semi-supervised classification with graph convolutional networks. *ArXiv*, abs/1609.02907, 2017.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2016.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Liu, H., Feng, J., Kong, L., Liang, N., Tao, D., Chen, Y., and Zhang, M. One for all: Towards training one graph model for all classification tasks. *arXiv preprint arXiv:2310.00149*, 2023a.

Liu, H., Li, C., Wu, Q., and Lee, Y. J. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023b.

Liu, J., Yang, C., Lu, Z., Chen, J., Li, Y., Zhang, M., Bai, T., Fang, Y., Sun, L., Yu, P. S., et al. Towards graph foundation models: A survey and beyond. *arXiv preprint arXiv:2310.11829*, 2023c.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Mao, H., Chen, Z., Tang, W., Zhao, J., Ma, Y., Zhao, T., Shah, N., Galkin, M., and Tang, J. Graph foundation models, 2024.

Perez, E., Ringer, S., Lukošiūtė, K., Nguyen, K., et al. Discovering language model behaviors with model-written evaluations, 2022. URL https://arxiv.org/abs/2212.09251.

Reimers, N. and Gurevych, I. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992, 2019.

Rozemberczki, B., Allen, C., and Sarkar, R. Multi-scale attributed node embedding, 2019.

Sun, C., Gu, H., and Hu, J. Scalable and adaptive graph neural networks with self-label-enhanced training. *arXiv preprint arXiv:2104.09376*, 2021.

Tang, J., Yang, Y., Wei, W., Shi, L., Su, L., Cheng, S., Yin, D., and Huang, C. Graphgpt: Graph instruction tuning for large language models. *arXiv preprint arXiv:2310.13023*, 2023.

Thekumparampil, K. K., Wang, C., Oh, S., and Li, L.-J. Attention-based graph neural network for semi-supervised learning. *arXiv preprint arXiv:1803.03735*, 2018.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Veličković, P., Fedus, W., Hamilton, W. L., Liò, P., Bengio, Y., and Hjelm, R. D. Deep graph infomax. 2019.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Wang, W., Chen, Z., Chen, X., Wu, J., Zhu, X., Zeng, G., Luo, P., Lu, T., Zhou, J., Qiao, Y., et al. Visionllm: Large language model is also an open-ended decoder for vision-centric tasks. *arXiv preprint arXiv:2305.11175*, 2023.

Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., and Weinberger, K. Simplifying graph convolutional networks. In *International conference on machine learning*, pp. 6861–6871. PMLR, 2019.

Wu, Q., Zhao, W., Li, Z., Wipf, D. P., and Yan, J. Node-former: A scalable graph structure learning transformer for node classification. *Advances in Neural Information Processing Systems*, 35:27387–27401, 2022.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2018.

Yang, Z., Cohen, W., and Salakhudinov, R. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pp. 40–48. PMLR, 2016.

Ye, R., Zhang, C., Wang, R., Xu, S., and Zhang, Y. Natural language is all a graph needs. *arXiv preprint arXiv:2308.07134*, 2023.

Ying, C., Cai, T., Luo, S., Zheng, S., Ke, G., He, D., Shen, Y., and Liu, T.-Y. Do transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34:28877–28888, 2021.

You, Y., Chen, T., Wang, Z., and Shen, Y. L2-gcn: Layer-wise and learned efficient training of graph convolutional

networks. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2124–2132, 2020.

Yun, S., Jeong, M., Kim, R., Kang, J., and Kim, H. J. Graph transformer networks. *Advances in neural information processing systems*, 32, 2019.

Zhang, H., Wu, Q., Yan, J., Wipf, D., and Yu, P. S. From canonical correlation analysis to self-supervised graph neural networks. *Advances in Neural Information Processing Systems*, 34:76–89, 2021.

Zhao, T., Shah, N., and Ghazizadeh, E. Learning from graphs beyond message passing. In *The Second Tiny Papers Track at ICLR 2024*.

A. Dataset Statistics

Table 8. Dataset Statistics

Dataset	Domain	#Node	#Edge	Sparsity(‰₀)
Cora	citation	2708	5429	14.8065
Pubmed	citation	19717	44338	2.2810
Arxiv	citation	169343	1166243	0.8134
Products	e-commerce	2449029	61859140	0.2063

In citation graphs (ogbn-Arxiv, Pubmed, Cora), each node represents a paper, where the title and abstract serve as node features, and edges denote co-citations. For ogbn-Products, nodes represent Amazon products, featuring item descriptions as node features, with edges indicating co-purchases.

Data Split. For node-level tasks, we adhere to the standard train/validation/test splits (Hu et al., 2020) for each dataset: 6:2:3 for Arxiv, 8:2:90 for Products, and 6:2:2 for both Pubmed and Cora. For link prediction, we randomly select node pairs from the node-level training set for training and from the node-level test set for testing, ensuring the edge-level training sets are equal in size to the node-level training sets.

B. More Metrics for Link Prediction Task

Table 9. AUC and Hits@k metrics on Link Prediction Task. Evaluated in classification expert and general model settings.

MODEL TYPE	MODEL	ARIXV		PRODUCTS		PUBMED		CORA	
		AUC	HITS@100	AUC	HITS@100	AUC	HITS@50	AUC	HITS@50
CLASSIFICATION EXPERT	GCN	97.19	16.74	98.23	30.58	88.54	37.08	89.20	73.89
	GRAPHSAGE	95.84	14.69	97.61	25.56	81.48	11.12	90.41	73.69
	GAT	94.53	7.77	97.55	23.65	83.07	14.20	88.40	67.64
	NODEFORMER	97.09	19.96	98.91	45.66	87.57	22.01	88.96	70.76
	LLAGA-ND-7B	97.36	29.63	99.54	79.09	97.16	64.72	95.53	93.36
	LLAGA-HO-7B	98.53	45.56	94.80	78.22	96.06	60.63	94.87	92.69
GENERAL MODEL	**LLAGA-ND-7B**	97.43	36.85	99.39	79.93	86.75	63.44	94.69	89.37
	LLAGA-HO-7B	97.80	38.05	98.95	43.52	96.07	51.48	91.06	82.72

We also employ the AUC and Hits@K metrics to evaluate the link prediction task. Since these metrics require ranking all positive and negative links, we use the next token prediction logits for 'yes' and 'no' following the query, "Please tell me whether two center nodes should connect to each other," as the scores for positive and negative links, respectively. Then, we apply SoftMax to these scores and rank all links using the 'positive' score post-SoftMax. AUC provides an unbiased assessment of a predictive model's ability to rank potential links, while Hits@K measures whether the true or relevant items appear in the top 'K' positions of the model's predictions.

C. Extending to Non-TAGs

In this paper, we followed the settings of some recent works (Tang et al., 2023; Chen et al., 2023a) and conducted experiments on four text-attributed graph datasets. However, it is important to note that unlike these methods, which predominantly leverage textual node attributes as LLM input, the design of LLaGA is inherently independent of text attributes, allowing for a wider range of applications. To evaluate LLaGA in a non-TAG setting, we tested it on the Cora and Facebook (Rozemberczki et al., 2019), omitting all textual attributes. Specifically, in this setting, LLaGA's input was simplified to solely one-hot encoding of nodes, thereby eliminating reliance on any textual attributes. The results, shown in Table 10, demonstrate that LLaGA still exhibits great capacity wwith non-TAGs.

Table 10. Evaluation on non-TAGs

MODEL	CORA	FACEBOOK
GCN	84.46	93.89
GRAPHSAGE	84.58	93.98
LLaGA-ND	85.42	94.13
LLaGA-HO	85.98	94.77

D. Flexibility with Text Encoding Methods

Table 11. LLaGA Trained with SBert and Roberta Embedding.

EMBEDDING	MODEL	NODE CLASSIFICATION ACCURACY				LINK PREDICTION ACCURACY			
		ARXIV	PRODUCTS	PUBMED	CORA	ARXIV	PRODUCTS	PUBMED	CORA
SBERT	GCN	66.00	77.41	82.04	79.70	91.38	94.91	84.31	83.15
	GRAPHSAGE	66.79	76.00	82.74	80.66	88.18	94.23	78.38	83.62
	LLaGA	**74.46**	**80.70**	**90.04**	**88.56**	**93.68**	**96.84**	**91.39**	**87.79**
ROBERTA	GCN	66.51	77.74	80.04	79.30	91.01	94.66	80.94	81.03
	GRAPHSAGE	68.14	76.73	81.27	82.29	88.80	94.11	74.31	82.88
	LLaGA	**74.19**	**81.13**	**89.78**	**88.19**	**93.52**	**96.79**	**89.96**	**85.15**

LLaGA demonstrates flexibility in its text encoding methods for node attributes. In our initial experiments, we employed SimTeG (Duan et al., 2023) as the primary encoding model. This section also explores the use of SBERT (Reimers & Gurevych, 2019) and RoBERTa (Liu et al., 2019) as alternative encoding methods. The outcomes of these trials are shown in Table 11. All models, including baselines, underwent training in a classification expert setting. For LLaGA, we utilized the Hop-Field Overview Template for structure encoding. Notably, LLaGA consistently surpassed other leading GNNs in performance, regardless of the chosen encoding model.

E. Integration with Various LLMs

Table 12. Integration with Various LLMs

BASE MODEL	NODE CLASSIFICATION ACCURACY				LINK PREDICTION ACCURACY			
	ARXIV	PRODUCTS	PUBMED	CORA	ARXIV	PRODUCTS	PUBMED	CORA
VICUNA-7B	75.99	83.32	94.80	89.30	94.30	96.05	88.64	88.53
LLAMA2-7B	76.26	84.21	94.83	86.53	94.15	96.03	89.39	85.44
OPT-2.7B	75.66	83.01	95.01	88.38	93.36	92.83	86.92	89.41

LLaGA also demonstrates flexibility with various Base Large Language Models (LLMs). In our primary experiments, Vicuna-7B served as the foundational model. This section details the substitution of LLaGA's base LLM with alternative models, including LLaMA2-7B and OPT-2.7B. The outcomes of these replacements are presented in Table 12. For structural encoding, we employ the Hop-Field Overview Template. And models are trained in classification setting. It is evident that LLaGA consistently yields favorable results irrespective of the base LLM, showcasing its effectiveness even with comparatively lighter models such as OPT-2.7B.

F. Compared with Baseline Targets for TAGs

Here we also broaden our comparison by including Patton (Jin et al., 2023) that concentrates on text-rich graphs. We test both methods on node classification task and the results are as follows:

Table 13. Compared with Baseline Targets for TAGs

MODEL	ARXIV	PRODUCTS	PUBMED	CORA
PATTON	71.62	80.43	90.82	86.53
LLAGA-ND	75.98	84.60	95.03	88.86
LLAGA-HO	76.66	84.67	95.03	89.22

G. Experiment Variance

Table 14. Variance Information on Cora and Pubmed Dataset

SETTING	DATASET	MODEL	NC(%)	LP(%)
SINGLE FOCUS	CORA	LLAGA-ND-7B	88.86±0.78	83.79±1.26
		LLAGA-HO-7B	89.22±0.46	86.82±0.88
	PUBMED	LLAGA-ND-7B	95.03±0.12	91.41±0.21
		LLAGA-HO-7B	95.03±0.07	89.18±0.34

We perform training and inference five times on relatively small datasets, with the variance information detailed in Table 14.