Graph-oriented Instruction Tuning of Large Language Models for Generic Graph Mining

Yanchao Tan *Member, IEEE*, Hang Lv, Pengxiang Zhan, Shiping Wang *Senior Member, IEEE*, Carl Yang *Member, IEEE*

***Abstract*—Graphs with abundant attributes are essential in modeling interconnected entities and enhancing predictions across various real-world applications. Traditional Graph Neural Networks (GNNs) often require re-training for different graph tasks and datasets. Although the emergence of Large Language Models (LLMs) has introduced new paradigms in natural language processing, their potential for generic graph mining—training a single model to simultaneously handle diverse tasks and datasets—remains under-explored. To this end, our novel framework MuseGraph, seamlessly integrates the strengths of GNNs and LLMs into one foundation model for graph mining across tasks and datasets. This framework first features a compact graph description to encapsulate key graph information within language token limitations. Then, we propose a diverse instruction generation mechanism with Chain-of-Thought (CoT)-based instruction packages to distill the reasoning capabilities from advanced LLMs like GPT-4. Finally, we design a graph-aware instruction tuning strategy to facilitate mutual enhancement across multiple tasks and datasets while preventing catastrophic forgetting of LLMs' generative abilities. Our experimental results demonstrate significant improvements in five graph tasks and ten datasets, showcasing the potential of our MuseGraph in enhancing the accuracy of graph-oriented downstream tasks while improving the generation abilities of LLMs.**

***Index Terms*—Generic Graph Mining, Large Language Models, Instruction Tuning**

I. Introduction

GRAPHS with plentiful attributes are widely used to model interconnected real-world entities, and they are pivotal for improving downstream predictions across diverse real-world applications. Recently, Graph Neural Networks (GNNs) have been commonly adopted for modeling attributed graphs [1], [2]. However, they are usually trained on specific tasks and datasets and need to be re-trained whenever applied to different ones. Inspired by the great success of Large Language Models (LLMs), the combination of GNNs and LLMs aims to enhance the processing of text-attributed graphs, which can improve the model's capabilities across various tasks and datasets. Existing studies can be categorized into two main approaches. The first category tries to train GNNs with LLM-enhanced features (e.g., LLM-GNN [3], TAPE [4], OFA [5], ALL-in-One [6], and GHGRL [7]). The second category explores LLMs for various graph applications (e.g., GPT4Graph [8], GraphGPT [9], HiGPT [10], NLGraph [11], and InstructGLM [12]).

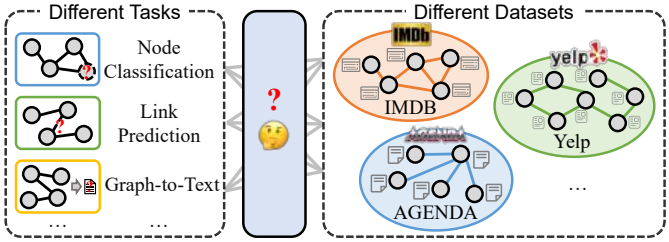


Fig. 1. An illustrative toy example of the need for a generic graph model that can be directly applied to various graph-related tasks and datasets.

Despite the promising direction of integrating LLMs with GNNs, the full exploration of LLMs' power and generation capabilities for graphs has yet to be under-explored [5], [6], [13], where one powerful graph model can be trained on multiple tasks/datasets and generalize well to more tasks/datasets. Fig. 1 demonstrates the necessity for a generic graph framework, showcasing a wide range of task and dataset combinations. Such a generic graph model can capture the semantic and structural information not only for various graph tasks (e.g., node classification, link prediction, and graph-to-text) but also for diverse graph datasets (e.g., from IMDB's movie data to AGENDA's academic texts). This enables one single foundation model to understand graph data by generating contextually rich textual interpretations that are central to LLMs. However, three key challenges hinder the achievement of this goal.

Challenge I: *How to extract informative graph descriptions under the limitation of language tokens?* To harness the full potential of LLMs for generic graph mining, an essential obstacle is translating the graph with abundant semantics and complex structures into a format that LLMs can process effectively, especially under strict language token limitations. Without a compact graph description that accurately encapsulates key information from graphs within the LLMs' token limitations, the model's capabilities to grasp and utilize the graph's semantic and structural richness is severely limited, potentially leading to suboptimal performance in graph applications.

Challenge II: *How to automatically generate diverse instructions?* Creating a diverse set of high-quality instructions for fine-tuning LLMs is fundamental for generic graph tasks. However, these instructions are often difficult for LLMs to comprehend when encountering unfamiliar graph structures or new tasks, and are costly to produce manually. While advanced LLMs like GPT-4 possess the capabilities to understand and reason

Y. Tan, H. Lv, P. Zhan, and S. Wang are with the Engineering Research Center of Big Data Intelligence, Ministry of Education; the Fujian Key Laboratory of Network Computing and Intelligent Information Processing; and the College of Computer and Data Science, Fuzhou University, Fuzhou 350116, China, Email: yctan@fzu.edu.cn, lvhangkenn@gmail.com, yyyzhanpengxiang@163.com, shipingwangphd@163.com.

C. Yang (Corresponding Author) is with the Department of Computer Science, Emory University, Atlanta 30322, United States, E-mail: j.carlyang@emory.edu.

diverse instructions, it remains unknown how to effectively and efficiently leverage the reasoning abilities of advanced LLMs to produce relevant and task-specific instructions.

Challenge III: *How to properly allocate instructions for graph-oriented instruction tuning?* The effectiveness of instruction tuning for LLMs largely depends on how the instructions are structured, enabling LLMs to accurately understand and execute graph-related tasks. Balancing a variety of tasks and datasets presents a significant challenge in facilitating mutual enhancement across these graph applications while preventing catastrophic forgetting of LLMs' generative abilities.

To tackle these challenges, we propose Graph-oriented Instruction Tuning of Large Language Models for Generic Graph Mining (MuseGraph), which consists of three pivotal steps: (i) *Development of Compact Graph Descriptions*, where we introduce a novel "node energy" metric to textualize graphs with essential semantic and structural details under limited language tokens; (ii) *Generation of Diverse Instructions*, which distills the reasoning abilities of advanced LLMs like GPT-4 to create Chain-of-Thought (CoT)-based instruction packages tailored for various graph tasks, thus enriching LLMs' capabilities in understanding and analyzing graph data without the expense of manual instruction crafting; (iii) *Graph-aware Instruction Tuning*, which introduces a dynamic instruction package allocation strategy based on the specific needs of each graph task, ensuring comprehensive and effective LLM tuning.

Our overall contributions are summarized as follows:

- *Formulation of generic graph mining.* We establish a generic graph framework that effectively and efficiently transforms graph semantics and structures into LLM-friendly formats while enhancing the generation capabilities necessary for diverse graph-related tasks.
- *Effective model designs.* We design and implement a set of models and mechanisms, including the development of compact graph descriptions, automatically generating diverse task-specific Chain-of-Thought (CoT)-based instruction packages, and graph-aware instruction tuning, targeting a unified graph model across tasks and datasets.
- *Extensive experiments across graph tasks and datasets.* We conduct thorough experiments to validate our approach with five tasks and ten datasets, demonstrating its superiority over existing state-of-the-art methods and highlighting its effective, generative, and interpretable abilities in enhancing generic graph mining.

II. Related Work

A. Semantic-rich Graph Representation Learning

Graph representation learning has emerged as a key technique for the complex structures of networks with abundant attributes [2], [14]–[16]. Many existing node embedding approaches have explored and harnessed the significant potential of Random Walks (RWs) in preserving the graph topological structures [17]–[20]. However, these approaches overlook the rich attribute information of nodes and edges on the graph [16]. Recently, Graph Neural Networks (GNNs) learn node representations through aggregating content information from neighbor nodes while preserving the surrounding structure of graphs [1], [21], [22]. However, most current GNNs are trained within a supervised learning setting, which demands a large amount of task-specific labeled data and may not always be available in real-world scenarios [23], [24]. Moreover, the learned embeddings often lack adaptability across different downstream tasks [20], which have to be re-trained whenever applied to various graph applications.

Despite the efforts to reduce reliance on labeled data through pre-training expressive GNNs through self-supervised methods (e.g., contrastive learning [23], [25], [26]), their effectiveness in specific downstream tasks still significantly depends on the appropriate choice of suitable self-supervision tasks and attribute encoders [27], [28]. Therefore, there is still a lack of a uniform framework for generic graph mining across different tasks and datasets.

TABLE I
A comparison between MuseGraph and related methods, where "GNN" and "LLM" refer to methods that train GNNs and LLMs as predictors, respectively.

	GNN	LLM	Cross-Task	Cross-Dataset
LLM-GNN [3]	●	○	○	○
TAPE [4]	●	○	○	○
OFA [5]	●	○	●	●
All-in-One [6]	●	○	●	●
GHGRL [7]	●	○	○	○
GPT4Graph [8]	○	○	○	○
GraphGPT [9]	○	●	○	●
HiGPT [10]	○	●	○	●
NLGraph [11]	○	○	○	○
InstructGLM [12]	○	●	○	○
MuseGraph	○	●	●	●

B. Leveraging LLMs for Graph Mining

Inspired by the remarkable advancements in Large Language Models (LLMs), integrating Graph Neural Networks (GNNs) with LLMs is creating substantial progress in handling complex text-attributed graphs [3], [29]. Existing approaches that leverage this integration can be broadly divided into the following two categories (shown in Table I).

The first category is training GNNs as predictors, augmented with LLM-enhanced features, labeled "GNN" in Table I. For example, the model LLM-GNN [3] used LLMs as annotators to generate the pseudo labels of nodes used to train GNNs. TAPE [4] prompted LLMs to obtain the explanations of predictions to enhance node initial features. PRODIGY [13] utilized LLMs to encode the textual information associated with nodes on the graph. OFA [5] described all nodes and edges using human-understandable prompt texts and converted them into embedded features by LLMs. All-in-One [6] reformulated different graph tasks with the unified graph prompts and language prompts. GHGRL [7] employed LLMs to automatically summarize and classify various data formats and types, aligning node features. These approaches primarily leveraged LLMs to process textual content within graphs and are often optimized for specific domains. However, they generally struggle with cross-task or cross-dataset applications, limiting the generation capabilities of LLMs across various graph tasks and datasets.

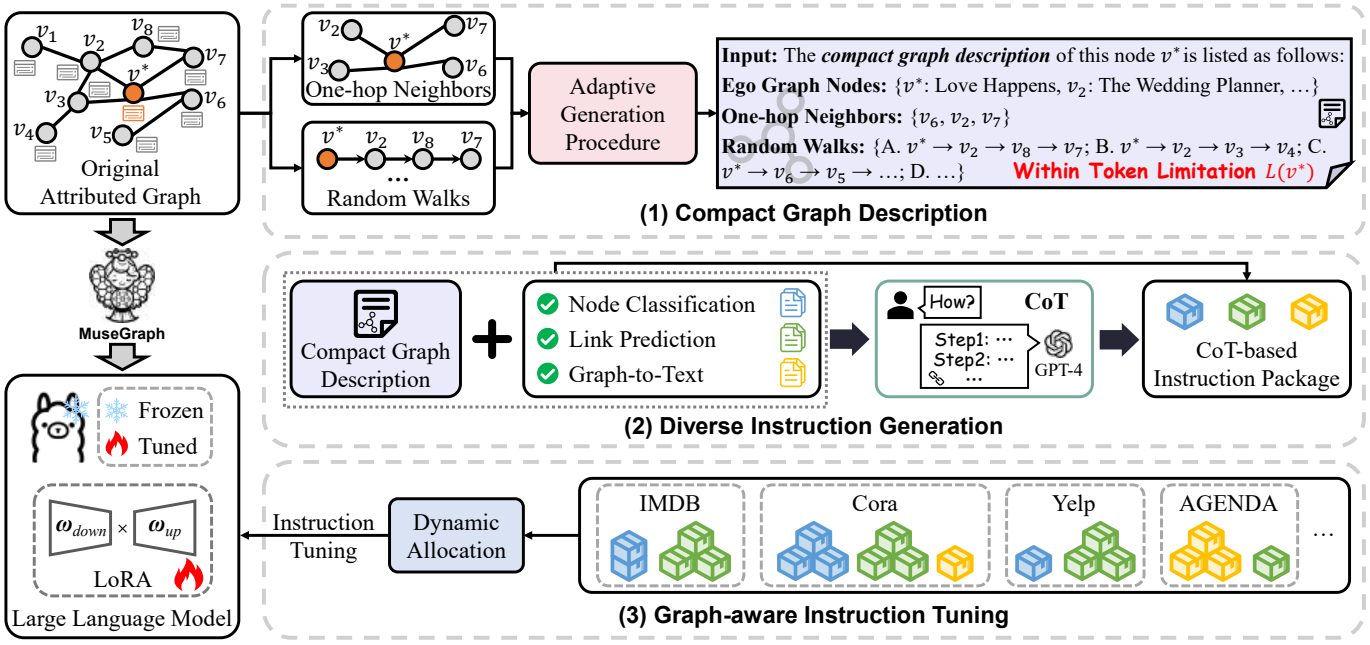


Fig. 2. The overall of MuseGraph, which consists of Compact Graph Description, Diverse Instruction Generation, and Graph-aware Instruction Tuning.

The second category, denoted as "LLM" in Table I, mainly focuses on utilizing LLMs to directly perform graph tasks. For example, GPT4Graph [8] and NLGraph [11] proposed graph-based benchmarks, evaluating the understanding and analytical capabilities of LLMs on graph data with designed natural language prompts. Notably, although they do not fine-tune LLMs on graph structures, their benchmarks effectively uncover current limitations and future directions for improving LLMs' capabilities in graph comprehension and reasoning. Both GraphGPT [9] and HiGPT [10] effectively combined task-specific GNNs with one projector on top of LLMs, enhancing performance across different datasets. InstructGLM [12] tuned LLMs with scalable graph prompts based on natural language instructions, where it can integrate node features from trained GNNs to improve the accuracy of node classification tasks. Compared with GNN-based methods, LLM-based methods have the potential to perform generic graph mining with powerful cross-task and cross-dataset abilities. However, such potential remains largely under-explored, as many existing LLM-based approaches still rely on node representations generated by GNNs tailored to specific tasks/datasets. According to the characteristics of different models (shown in Table I), there is a pressing need for a generic graph framework that can accurately understand graph data while enhancing generation capabilities across diverse tasks and datasets.

III. The MuseGraph Framework

A. Overview of Our Framework

Objective: In this paper, we aim to develop a unified framework that can seamlessly integrate the strengths of Graph Neural Networks (GNNs) and Large Language Models (LLMs) into one foundation model via graph-oriented instruction tuning of LLMs. Through this, we seek to enable a more effective and generic approach for graph mining across various downstream tasks and datasets.

Overview: To achieve this goal, we propose MuseGraph framework, which comprises three major components. Firstly, we develop a compact graph description mechanism that captures critical semantic and structural details within the constraints of language token limitations. Secondly, we generate a diverse range of instructions via the reasoning capabilities of advanced LLMs like GPT-4, thus facilitating task-specific Chain-of-Thought (CoT)-based instruction packages for graph tasks. Thirdly, we adopt a graph-aware instruction tuning, utilizing a dynamic allocation strategy for instruction packages tailored to the unique requirements of each graph task and dataset. The overall model architecture is shown in Fig. 2, and we elaborate on the three main components.

B. Compact Graph Description

Leveraging the capabilities of LLMs for graphs presents a unique set of challenges, primarily due to LLMs' inherent limitations in directly processing textual interpretations within language token constraints. Therefore, it is non-trivial to automatically encapsulate key information from graphs under the token limitations, which requires a compact description including complex node and edge attributes along with structural details to accurately describe the graph.

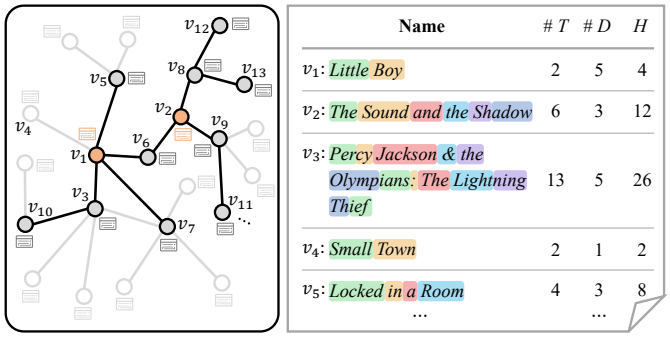
Inspired by common graph analysis techniques such as neighbors and walks, we propose a novel method of textualization to describe graphs via these concepts. In this way, neighbors are helpful to understand local connectivity and feature distribution [1], [21], [22], providing a granular view of node attributes; while walks offer a dynamic method to explore the graph's structure and the high-order relationships between nodes, highlighting the diversity of connectivity and paths [19], [20]. The integration of neighbors and walks can

Algorithm 1: Procedure for generating the compact graph description tailored to a specified node.

Input: Attributed graph $\mathcal{G}$ with N nodes, token count set $\mathcal{T}$, node energy set $\mathcal{H}$, target node v^*, token limitation $L(v^*)$

Output: Key neighbor set $\mathcal{N}(v^*)$, key walk set $\mathcal{W}(v^*)$

1: Initialize $\mathcal{N}(v^*)$, $\mathcal{W}(v^*)$ as empty sets;
2: Select $v_i \in G(v^*)$ with $H(v_i) \geq H(v^*)$ and $L(v^*) \geq T(v_i)$ constraints for $\mathcal{N}(v^*)$, where $G(v^*)$ is v^*'s one-hop neighbors and $H(\cdot)$ can be calculated according to Eq. 1;
3: Expand $\mathcal{W}(v^*)$ starting from v^* based on $\mathcal{G}$ within $L(v^*)$ and $H(v^*)$ constraints.



Target	*Key Neighbors*: $H(v_i) \geq H(v^*)$	*Key Walks*: $H(v_j) \geq H(v^*)$
$v^* = v_1$	$\{ v_3 , v_5 , v_6 , v_7 \}$	① $v_1 \rightarrow v_6 \rightarrow v_2 \rightarrow v_8 \rightarrow \cdots$ ② $v_1 \rightarrow v_3 \rightarrow v_{10}$
$v^* = v_2$	$\{ v_8 , v_9 \}$	① $v_2 \rightarrow v_8 \rightarrow v_{12}$ ② $v_2 \rightarrow v_8 \rightarrow v_{13}$ ③ $v_2 \rightarrow v_9 \rightarrow v_{11} \rightarrow \cdots$ ④ $v_2 \rightarrow v_9 \rightarrow \cdots$

Fig. 3. Two examples on IMDB illustrate how to extract the key information of nodes via neighbors and walks based on node energy $H(v^*)$ and the language token limitation $L(v^*)$.

achieve a holistic understanding of graph structures. Notably, this textualization method integrates GNN-inspired structural priors (i.e., the core principles of message passing and high-order propagation) into the graph description design, without requiring training specific GNNs. This enables more powerful and adaptive capabilities across diverse tasks and datasets.

Given LLM token limitations and the varied contributions of neighbors and walks to node understanding, we further develop an adaptive generation procedure to ensure the compactness of the description. Specifically, we first design a "node energy" metric $H(\cdot)$, assessing node information from two perspectives: token count in node attributes and node degree count. This metric enables us to effectively filter and select neighbor nodes and walk nodes, prioritizing those that are abundant in semantic information and possess a significant number of neighbor nodes, thus enhancing the expressiveness of the graph description. The calculation of node energy $H(v^*)$ is formulated as follows:

$$H(v^*) = T(v^*) \times \lceil \log(D(v^*) + 1) \rceil, \tag{1}$$

where v^* is the target node, $T(v^*)$ is the number of node tokens processed by a language tokenizer, $D(v^*)$ is the number of node degrees, and $\lceil \cdot \rceil$ is the ceiling operator.

Based on v^*, our method strategically incorporates neighbors and walks, as depicted in Algorithm 1. A neighbor v_i is chosen to describe the target v^* if its $H(v_i)$ surpasses $H(v^*)$, ensuring the included node can provide supplemental information. The process of adding walks concludes when encountering a node whose $H(v_i)$ does not meet the threshold set by the target's $H(v^*)$, thus refining the input to maximize related graph information within the constraints of the token limit. To further demonstrate the adaptive input generation for neighbors and walks tailored to each node, we present an example in Fig. 3. Node v_1's with a relatively low $H(v_1)$ value includes a wide range of neighbors to capture more context, excluding v_4 due to its even lower $H(v_4)$. Given the token limitation $L(v_1)$, only two walks are sampled for v_1 to maintain a compact description. Conversely, v_2 with its higher $H(v_2)$, inherently carries more information, prompting the selection of fewer neighbors. v_6 is excluded since $H(v_6) < H(v_2)$. This frees up tokens to detail more walks for v_2. The H metric thus effectively balances neighbor and walk inclusion for each node, marrying information-rich and token-efficient characteristics.

The process culminates in the textualization of each node's key information, producing a tailored and compact graph description as depicted in the upper right of Fig. 2. Note that, a graph-related task can involve multiple nodes (such as link prediction). In this case, we adaptively allocate the token limit requirement of each node involved by calculating the ratio of all node energies using the softmax function. Additionally, we provide the ablation in Section IV-C2 to further verify the effectiveness of the compact graph description.

Remark. Given an attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and an LLM token limitation L, the expected utility of including a node $v \in \mathcal{V}$ in the compact graph description can be approximated by the product of its semantic complexity $T(v)$ and structural centrality $D(v)$ as follows:

$$\begin{aligned} &\mathbb{E}[\phi(v)] \propto T(v) \times \lceil \log(D(v) + 1) \rceil = H(v), \\ &\text{s.t.} \sum_{v \in G} T(v) \leq L, \end{aligned} \tag{2}$$

where $\phi(v)$ denotes the information gain of including a node v. Under the LLM token constraint $\sum_{v \in G} T(v) \leq L$, prioritizing nodes by descending $H(v)$ value leads to a compact yet informative subgraph G, which captures both semantic richness and structural connectivity of the original graph $\mathcal{G}$. Notably, this formulation models node selection as a constrained expected utility maximization problem [30], where $H(v)$ acts as a practical proxy for the expected informativeness per node. Empirically, longer token spans tend to encode richer semantics [31], [32], while high-degree nodes often serve as key hubs that support global connectivity [33]–[36]. Therefore, the node energy $H(v)$ naturally balances token cost with information gain, making it a suitable metric for adaptive neighbor and walk selection within limited LLM tokens.

C. Diverse Instruction Generation

With the compact graph descriptions tailored to each node, another critical step in fine-tuning LLMs for generic graph

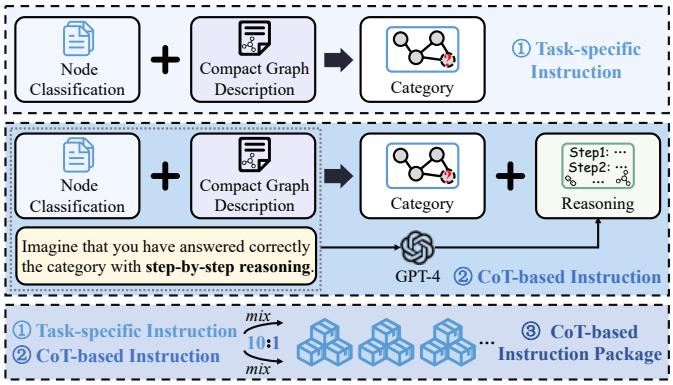


Fig. 4. A process showing how to conduct the Chain-of-Thought (CoT)-based instruction package for node classification. We leverage the reasoning ability distilled from advanced LLMs (e.g., GPT-4) and integrate them with task-specific instructions via a 1:10 mix ratio.

mining is crafting diverse and high-quality instructions [37], [38]. Benefiting from the key graph information captured by our designed compact graph descriptions, we can accurately and efficiently construct abundant graph-related task instructions (e.g., node classification, link prediction, and graph-to-text). While these instructions enable LLMs to effectively grasp compact graph descriptions, they may not always be easily comprehended when encountering unfamiliar graph structures or new tasks, potentially leading to inaccuracies and limited reasoning across various datasets. Moreover, manual construction of these instructions is often highly costly in terms of both time and resources.

To address these problems, we propose to distill the reasoning capabilities from advanced LLMs [39] (e.g., GPT-4 [40] with over 200 billion parameters) for graph-related tasks. Our approach, inspired by the Chain-of-Thought (CoT) processing methodology [41], prompts GPT-4 via a flexible template based on our compact graph descriptions for different tasks and then constructs task-specific CoT-based instruction packages. Different from the existing methods that leverage CoT in the prompting stage [41], [42], we directly construct CoT-based instruction packages, harnessing their diversity and reasoning abilities to facilitate instruction tuning with graph data.

Specifically, as shown in Fig. 4, we first design diverse task-specific instructions based on compact graph descriptions, such as node classification, link prediction, and graph-to-text. Then, we prompt GPT-4 with a small number of task-specific instructions to generate the corresponding step-by-step reasoning, which is used to conduct CoT-based instructions with the initial instructions across diverse tasks. This approach distills GPT-4's vast knowledge base to augment the reasoning and analytical abilities of our MuseGraph, enhancing the understanding of compact graphs.

To optimize the cost-effectiveness of querying GPT-4, we introduce the CoT-based instruction package. For every set of 1,000 standard task-specific instructions, we integrate 100 CoT-based instructions tailored to the same graph task. This approach not only proves to be economical but also broadens the diversity and flexibility of instructions, accommodating a range of graph-related tasks.

D. Graph-aware Instruction Tuning

With the proposed compact graph descriptions and diverse instruction generation mechanisms, identifying an effective instruction tuning method for LLMs remains crucial. Such tuning enables LLMs to understand and perform well a wide range of graph mining tasks across multiple datasets and contexts [38], [43]. However, a persistent challenge is the catastrophic forgetting issue commonly faced by LLMs [44], [45]. This phenomenon complicates the capabilities to maintain extensive task and dataset coverage without compromising previously acquired knowledge, necessitating strategies that balance new learning with memory retention.

To this end, we propose a dynamic instruction package allocation strategy to adaptively adjust the volume of task-specific CoT-based instruction packages based on the complexities of tasks and datasets, which ensures that more complex tasks/datasets receive a proportionally larger set of instructions for detailed guidance. The calculation process includes two aspects as follows:

- For task complexity: We assess the complexity of tasks by calculating the average number of answer tokens for each task, which helps in tailoring the CoT-based instruction packages to specific needs within a dataset. A larger average number of answer tokens indicates a higher level of reasoning and response complexity, enabling us to proportionally allocate more instructions to such tasks.
- For dataset complexity: We calculate the total node energy $H(\cdot)$ (cf., Eq. 1) for each graph data, utilizing this metric to optimize the instruction distribution. A higher overall node energy reflects richer semantic and structural characteristics within a dataset, guiding the allocation of more instructions to these information-dense graphs.

By performing this dynamic allocation of instructions, we enhance LLMs' abilities to learn and retain extensive knowledge across a diverse range of graph mining challenges. We provide the ablation in Section IV-C4 to further assess the effectiveness of the dynamic instruction package allocation strategy.

To balance the effectiveness with computational efficiency, we adopt a graph-aware instruction tuning mechanism, which can sufficiently utilize diverse and high-quality instructions for fine-tuning LLMs [37]. Specifically, we adopt a general LLM LLaMA3-8B [46] with LoRA [47] as our initial fine-tuning point by default. Then, based on the diverse instruction package set $\mathcal{I} = \{I_1, I_2, \ldots, I_D\}$ as model input, we adopt the negative log-likelihood loss as the fine-tuning objective as follows:

$$p_\theta\left(Y_{j,k}|I_j, Y_{j,<k}\right) = LLM_\theta\left(I_j, Y_{j,<k}\right), \tag{3}$$

$$\mathcal{L}_\theta = -\sum_{k=1}^{|Y_j|} \log p_\theta\left(Y_{j,k}|I_j, Y_{j,<k}\right), \tag{4}$$

where θ is the learnable parameters of our proposed graph-aware LLM (i.e., MuseGraph), $I_j \in \mathcal{I}$ is the input of LLM, and Y_j is the output of LLM.

After obtaining the fine-tuned LLM for generic graph mining, we can apply it to various downstream tasks, such as node classification, link prediction, and graph-to-text. Notably, our MuseGraph can achieve superior performance across diverse

TABLE II
STATISTICS OF THE DATASETS FOR NODE CLASSIFICATION.

Dataset	IMDB	Freebase	Cora	Arxiv
# Nodes	21,420	180,098	2,708	169,343
# Edges	86,642	1,057,688	5,429	1,166,243
# Labeled nodes	4,573	7,954	2,708	169,343
# Classes	5	7	7	40

TABLE III
STATISTICS OF THE DATASETS FOR LINK PREDICTION.

Dataset	# Nodes	# Edges	# Link type	# Node type
Yelp	82,465	30,542,675	4	4
MIMIC-III	32,267	559,290	4	3

TABLE IV
STATISTICS OF THE DATASETS FOR GRAPH-TO-TEXT.

Dataset	# Graph	# Rel	Avg.# Nodes	Avg.# Triples	Avg. Length
AGENDA	40,720	7	12.37	4.48	140.36
WebNLG	8,783	246	5.91	2.95	13.02

tasks and datasets with few instruction packages based on the graph-aware instruction tuning mechanism, even in the few-shot and zero-shot settings (shown in Table VII and Fig. 6).

Model Extension. Our proposed MuseGraph establishes a generic approach for integrating LLMs with graph data. We can flexibly incorporate a variety of foundation LLMs, as discussed in Section IV-C1, which enhances its generation capabilities across different graph-related tasks. Building on this flexibility, we can further replace our adopted LoRA [47] with different parameter-efficient training approaches, such as QLoRA [48], AdaLoRA [49], and FourierFT [50]. These methods can be helpful to reduce the computational demands while maintaining high performance. Additionally, we can explore the incorporation of Reinforcement Learning from Human Feedback (RLHF) [37] to further enhance the learning process, adjusting model behaviors and improving decision-making in complex graph scenarios.

Datasets Extension. Leveraging the proposed compact graph description and diverse instruction generation mechanisms, we can effectively include a wider range of attributed graphs via our graph-aware instruction tuning, enriching the diversity of applications across various domains. Since our MuseGraph introduces a generic graph framework to simultaneously capture the semantic and structural information of attributed graphs across tasks and datasets, it can be applied for tasks across diverse datasets, including biological networks, social networks, and knowledge graphs. Furthermore, the zero-shot generalization capabilities of our MuseGraph, as evaluated via graph-informed DyVal (demonstrated in Fig. 6 in Section IV-B), highlight its potential to perform accurately on more datasets where it was not explicitly trained. This ability to generalize well in zero-shot scenarios underscores the practical utility of MuseGraph in navigating and analyzing unexplored graphs.

IV. EXPERIMENT

In this section, we conduct extensive experiments to validate the effectiveness and efficiency of our proposed MuseGraph under diverse conditions, aiming to answer the following three key research questions:

- **RQ1:** How does our MuseGraph framework perform in comparison to the representative graph-oriented methods for generic graph mining?
- **RQ2:** What are the effects of different model components?
- **RQ3:** Can the compact graph description help the model better understand the graph structure?

A. Experimental Setup

1) Datasets: To comprehensively evaluate the effectiveness and efficiency of our MuseGraph, we utilize two real-world datasets for Heterogeneous Node Classification (i.e., IMDB and Freebase), two for Homogeneous Node Classification (i.e., Cora and ogbn-arxiv (abbr. Arxiv)), two for Link Prediction (i.e., Yelp and MIMIC-III), and two for Graph-to-Text (i.e., AGENDA and WebNLG). Additionally, we apply graph-informed DyVal [51] to evaluate MuseGraph with two Dynamic Reasoning tasks (i.e., Reachability and Max Sum Path), including four levels with increasing complexity (i.e., D1, D2, D3, and D4). The detailed statistics are shown in Table II, Table III, and Table IV. Furthermore, the detailed descriptions of tasks and datasets are provided as follows:

i) **Heterogeneous Node Classification**: Heterogeneous Node Classification classifies the target node into pre-defined classes, leveraging diverse relationships and attributes within the graph, where the graph consists of multiple types of nodes and edges. Following [52], we split each dataset for Heterogeneous Node Classification into training/validation/testing sets with a ratio of 24%/6%/70%.

- IMDB[1] is a website about movies and related information, including a subset from the Action, Comedy, Drama, Romance, and Thriller genres. Each labeled movie has one or multiple labels.
- Freebase[2] is a knowledge graph of books, films, music, sports, people, locations, organizations, and businesses. Each labeled book has only one label.

ii) **Homogeneous Node Classification**: Different from Heterogeneous Node Classification, Homogeneous Node Classification forecasts the target node's label within the graph containing the same type of nodes and edges. We employ a 60%/20%/20% training/validation/testing split for Cora, which is consistent with [4]. For Arxiv, we adopt the public dataset split in [53], which is 54%/18%/28%.

- Cora[3] comprises 2,708 scientific publications classified into one of seven classes–case-based, genetic algorithms, neural networks, probabilistic methods, reinforcement learning, rule learning, and theory, with a citation network consisting of 5,429 links.
- Arxiv[4] represents the citation network among computer science arXiv papers. Each paper in the dataset is asso-

[1]https://www.kaggle.com/karrrimba/movie-metadatacsv
[2]http://www.freebase.com
[3]http://www.cora.justresearch.com/lander
[4]https://ogb.stanford.edu/docs/leader_nodeprop/#ogbn-arxiv

ciated with a research category, manually labeled by the authors and arXiv moderators. These research categories are chosen from 40 subject areas.

iii) **Link Prediction**: Link Prediction predicts the likelihood of a future or missing connection between two nodes in a graph. We train all methods using the randomly selected 80% of links and evaluate them on the remaining 20% held-out links as suggested in [14] and [54].

- Yelp[5] is a user-business network collected from Yelp, including four types of nodes, which are businesses, users, locations, and reviews.
- MIMIC-III[6] consists of a graph of diseases, patients, and visits, with nodes and relations derived from electronic health records.

iv) **Graph-to-Text**: Graph-to-Text generates a descriptive text based on the information and structure of the graph. Following [55], we design different few-shot settings with four training instance sizes ranging from 50, 100, 200, to 500.

- AGENDA[7] (Abstract Generation Dataset) is a dataset that links knowledge graphs with paper abstracts from scientific domains. The graphs in AGENDA are automatically extracted from the SciIE information extraction system. Each instance in AGENDA includes the paper's title, entities, graph, and abstract.
- WebNLG[8] is a crowd-sourced RDF triple-to-text dataset manually crafted by human annotators. The dataset includes graphs from DBpedia with up to seven triples paired with one or more reference texts. We adopt three large domains of data from WebNLG v1.5 for experiments (i.e., Airport, Building, and Food).

v) **Dynamic Reasoning**: Reachability and Max Sum Path are Dynamic Reasoning tasks generated by graph-informed DyVal [51], which generates test samples dynamically, mitigating the issues of data contamination and static complexity. We adopt a zero-shot setting consistent with DyVal [51], where we produce 500 samples for each task of varying complexity to balance test time and discrepancy.

- Reachability determines if one node can reach another node in the graph. Respond with "True" or "False".
- Max Sum Path finds the maximum sum path between two nodes in a graph. The sum value is obtained by summing up the values of nodes in the path. If such a path does not exist, directly answer "N/A".

2) Evaluation Protocols: For Heterogeneous Node Classification and Homogeneous Node Classification, we use two commonly adopted evaluation metrics [14], [52], [54]: Macro-F1 (across all labels) and Micro-F1 (across all nodes). The F1 score is a metric of the model's accuracy in binary and multi-class classification tasks, which considers both precision and recall. For Link Prediction, we compute the AUC metric as suggested in [9], [14], [54]. AUC measures the area under the ROC curve, indicating the model's ability to distinguish between positive and negative classes across various thresholds. For Graph-to-Text, we report BLEU-4 [56] as our metric, where BLEU-4 measures the precision of four-word sequences (4-grams) in the generated text compared to the reference text. Moreover, we evaluate Reachability and Max Sum Path performance with Accuracy that is consistent with DyVal [51].

3) Methods for Comparison: The following characteristic baseline methods can be classified into three categories: i) GNN-based methods, ii) LLM-based methods, and iii) GNN+LLM-based methods.

i) **GNN-based methods**: We follow [9], [11], [14], [16], [52] in selecting GNN-based methods that are commonly used in homogeneous and heterogeneous network benchmarks.

- GraphSAGE [1] generates node embeddings by sampling and aggregating features from a node's local neighborhood, enabling scalable learning on large graphs.
- GCN [21] scales linearly in the number of graph edges and learns hidden layer representations that encode both local graph structure and features of nodes.
- GAT [57] utilizes masked self-attention mechanisms to enhance the processing of graph data by addressing limitations in traditional graph convolution methods.
- RevGNN [58] captures long-range interactions in graph data and reduces memory complexity with grouped reversible connections, enabling more effective training of deep and wide GNNs.
- HINormer [16] uses graph transformers to learn node representations on heterogeneous information networks by capturing both local structure and heterogeneity.
- R-GCN [59] improves traditional GCNs with relation-specific convolutions, enhancing learning from diverse edge types in knowledge graphs.
- HGT [60] extends the transformer architecture to handle heterogeneous graphs by incorporating type-specific parameters and an attention mechanism to capture diverse node and edge interactions.

ii) **LLM-based methods**: We choose representative and widely adopted LLM-based methods as baselines, covering both general-purpose and graph-specific LLMs. Note that we include the BART [61] and T5 [62] series as standard encoder-decoder baselines for Graph-to-Text, consistent with [55], [63].

- Baichuan2-7B-Base [64] is an open-source, bilingual language model developed by Baichuan Inc., trained on 2.6 trillion tokens with 7 billion parameters.
- Qwen2-7B-Instruct [65] is an instruction-tuned 7 billion parameter model, designed to excel in tasks like language understanding, generation, and more, with support for processing up to 131,072 tokens in context.
- LLaMA1-7B [66] is an open-source large language model developed by Meta. It is designed for natural language understanding and generation tasks, featuring 7 billion parameters for efficient and powerful text processing.
- LLaMA2-7B [67] is an improved version of LLaMA1, featuring 7 billion parameters with enhancements in data, training techniques, and model performance.
- LLaMA3-8B [46] succeeds LLaMA2, offering improved performance with 8 billion parameters through advancements in architecture, training data, and optimization.

[5]https://www.yelp.com/dataset
[6]https://physionet.org/content/mimiciii/1.4
[7]https://github.com/rikdz/GraphWriter/tree/master/data
[8]https://github.com/ThiagoCF05/webnlg

- GPT-3.5 [37] is a large-scale language model developed by OpenAI, trained on a vast corpus of text with 175 billion parameters, capable of generating human-like text and understanding complex contexts.
- GPT-4 [40] builds upon GPT-3.5, providing advanced language generation and understanding capabilities with greater scale and improved performance.
- BART-large [61] is a pre-trained language model based on the transformer architecture, featuring both encoder and decoder components, and encompasses 160 million parameters, making it a powerful tool for NLP tasks.
- T5-large [62] is a transformer-based pre-trained language model that uses a unified architecture for encoding and decoding, consisting of a 768-layer deep network with 11 billion parameters, effectively handling text-based tasks.

iii) **GNN+LLM-based methods**: We compare recent LLM-enhanced or instruction-tuned graph learning paradigms, as summarized in Table I. For methods such as GPT4Graph [8] and NLGraph [11], we exclude them as they are general-purpose benchmark frameworks rather than concrete graph models with training pipelines. LLM-GNN [3] and TAPE [4] rely on GPT-3.5 APIs to generate high-quality annotations and explanations for each dataset. Therefore, we only utilize the publicly released features for the Cora and Arxiv datasets[9][10], and mark "-" for the other datasets in Table V and Table VI. Additionally, since GHGRL [7] is designed with node-centric prompts for node classification, making it unsuitable for link prediction, we report "-" in Table VI.

- LLM-GNN [3] involves LLMs to generate confidence-aware annotations for a subset of nodes, which are then used to train GNNs for downstream prediction.
- TAPE [4] leverages LLMs' explanations to generate informative node features for text-attributed graphs, boosting the performance of various GNNs.
- OFA [5] describes diverse text-attributed graphs using human-understandable prompts and encodes them into a unified embedding space via LLMs, thereby guiding the training of a single GNN model.
- All-in-One [6] converts different-level tasks to the graph-level task with the unified graph and language prompts for improving the multi-task performance of GNNs.
- GHGRL [7] employs LLMs to automatically summarize and classify heterogeneous data, and apply a specialized GNN for task-specific learning.
- GraphGPT [9] integrates LLMs with graph knowledge using a graph structural instruction tuning paradigm, enhancing understanding through text-graph grounding and step-by-step reasoning.
- HiGPT [10] aligns LLMs with heterogeneous graph knowledge using instruction tuning, a specialized graph tokenizer, and a mixture-of-thought augmentation to improve understanding and tackle data sparsity.
- InstructGLM [12] tunes LLMs with highly scalable graph prompts that combine natural language instructions and node features from trained GNNs.

[9]https://github.com/CurryTang/LLMGNN
[10]https://github.com/XiaoxinHe/TAPE

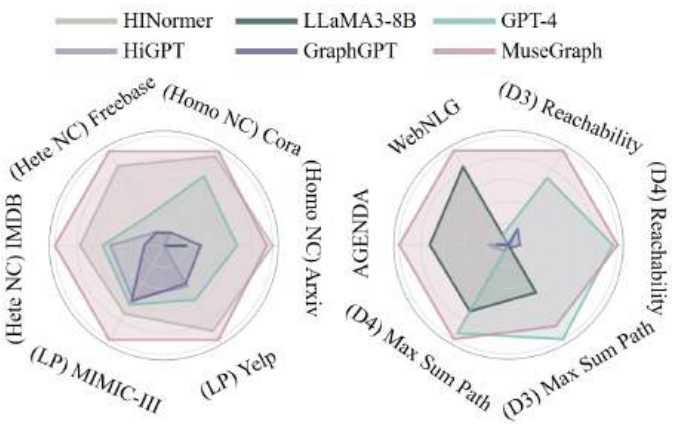


Fig. 5. Comprehensive performance of different models on various tasks and datasets. The radar charts compare the performance of six models (i.e., HiNormer, HiGPT, LLaMA3-8B, GraphGPT, GPT-4, and MuseGraph) across multiple tasks. Micro-F1 scores are shown for Node Classification on IMDB (Heterogeneous) and Cora (Homogeneous). Link Prediction on Yelp uses AUC scores. For limited-data Graph-to-Text tasks on AGENDA and WebNLG, we report BLEU-4 scores. Additionally, accuracy measures for Dynamic Reasoning tasks like Reachability and Max Sum Path (D3 to D4 complexities) from the DyVal benchmark are included. Results are normalized for clarity.

4) Implementation Details: For our MuseGraph, we utilize LLaMA-Factory [68] to train a unified framework using a mixture of instruction packages across various tasks and datasets. By default, we choose LLaMA3-8B[11] [46] as the foundation model for fine-tuning, and we perform parameter-efficient learning via LoRA [47] with $r = 32$ and $\alpha = 64$. The learning rate is set to $5e^{-5}$ and the maximum input length of LLM is set to 1200. The training process is carried out for two epochs. For GNN-based methods, we train and evaluate baselines based on CogDL [69], HGB [52], or HNE [14]. For LLM-based methods, we load the checkpoint of LLM from HuggingFace[12] or call official API from OpenAI[13] for evaluation. Additionally, for a fair comparison, we select GNN+LLM-based methods with public checkpoints or tuning details from their original papers. All LLM-based methods, GNN+LLM-based methods, and MuseGraph, are evaluated using the same test instructions. The hyperparameters of baselines are chosen carefully based on either grid search or their official source codes, and the learning rate is searched in $[1e^{-5}, 1e^{-2}]$. All experiments are conducted using only one NVIDIA GTX 3090 Ti GPU.

Notably, given the GPU constraints during the training phase, we leverage HiGPT [10] checkpoint from Vicuna-7B-v1.5, which was fine-tuned with 60-shot instruction data on the IMDB[14], GraphGPT [9] checkpoint tuned with Arxiv-PubMed-mix-NC-LP instruction data[15], and InstructGLM [12] public checkpoint[16] for conducting respective experiments. The full code for this work is available[17].

[11]https://ai.meta.com/blog/meta-llama-3
[12]https://huggingface.co
[13]https://platform.openai.com
[14]https://huggingface.co/Jiabin99/HiGPT
[15]https://huggingface.co/Jiabin99/GraphGPT-7B-mix-all
[16]https://github.com/agiresearch/InstructGLM
[17]https://github.com/Melinda315/MuseGraph

TABLE V
EXPERIMENTAL RESULTS ON FOUR BENCHMARK DATASETS FOR HETEROGENEOUS NODE CLASSIFICATION AND HOMOGENEOUS NODE CLASSIFICATION. THE BEST PERFORMANCES ARE HIGHLIGHTED IN **BOLDFACE** AND THE SECOND RUNNERS ARE UNDERLINED.

Method	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1	Micro-F1	Macro-F1
Dataset	(Hete NC) IMDB		(Hete NC) Freebase		(Homo NC) Cora		(Homo NC) Arxiv	
GraphSAGE	62.06	52.28	58.97	27.59	85.98	84.57	71.69	50.82
GCN	64.23	58.43	59.86	29.35	82.87	81.29	71.98	51.26
GAT	64.28	58.81	65.83	40.40	82.50	81.42	72.13	52.67
RevGNN	65.03	59.90	56.02	20.85	85.79	83.14	72.76	50.38
HINormer	67.63	64.29	66.51	48.87	82.84	81.28	71.01	51.81
R-GCN	62.98	58.90	58.57	47.03	80.63	79.14	71.04	50.79
HGT	66.95	62.71	60.46	29.33	81.52	80.76	71.80	51.43
Baichuan2-7B-Base	40.57	39.15	3.60	1.62	8.78	5.68	1.20	0.82
Qwen2-7B-Instruct	64.83	61.39	20.85	13.47	58.34	48.23	40.42	18.74
LLaMA1-7B	22.57	16.76	4.22	1.72	12.92	5.93	1.35	1.17
LLaMA2-7B	31.03	27.46	5.29	1.78	14.02	6.08	1.82	1.21
LLaMA3-8B	37.51	34.86	5.98	3.76	14.76	8.49	23.03	11.09
GPT-3.5	55.96	54.14	30.61	25.34	65.31	55.34	43.17	32.82
GPT-4	59.47	59.18	28.02	24.19	67.71	56.41	51.29	43.44
LLM-GNN	-	-	-	-	75.61	73.95	66.05	45.33
TAPE (GCN)	-	-	-	-	**88.19**	**87.16**	**74.36**	**55.74**
OFA	64.81	57.91	56.58	34.75	71.68	70.21	69.31	50.70
All-in-One	60.75	55.17	50.48	21.04	68.45	64.18	65.57	47.06
GHGRL	65.40	59.24	55.96	26.15	83.96	80.97	63.10	42.05
HiGPT	56.72	53.77	16.17	7.02	23.86	14.17	10.76	6.25
GraphGPT	44.52	43.70	15.80	6.14	24.57	15.26	31.08	17.12
InstructGLM	13.59	9.41	5.65	2.38	15.62	10.88	42.60	28.43
MuseGraph (Qwen2-7B-Instruct)	75.77	73.08	76.40	58.18	84.35	80.71	64.88	45.41
MuseGraph (LLaMA1-7B)	75.73	72.51	75.42	51.36	81.29	77.50	58.82	35.54
MuseGraph (LLaMA2-7B)	74.65	72.44	73.68	48.44	84.06	81.25	65.45	43.94
MuseGraph (LLaMA3-8B)	**76.57**	**73.78**	**78.01**	**59.62**	86.83	84.74	67.80	47.71

TABLE VI
AUC RESULTS ON TWO BENCHMARK DATASETS FOR LINK PREDICTION. THE BEST PERFORMANCES ARE HIGHLIGHTED IN **BOLDFACE** AND THE SECOND RUNNERS ARE UNDERLINED.

Dataset	Yelp	MIMIC-III
GraphSAGE	68.85	53.40
GCN	69.94	53.27
GAT	70.38	54.46
RevGNN	68.97	56.79
HINormer	75.03	57.82
R-GCN	72.17	57.31
HGT	79.02	64.01
Baichuan2-7B-Base	14.50	12.55
Qwen2-7B-Instruct	11.18	18.31
LLaMA1-7B	10.78	19.39
LLaMA2-7B	22.14	25.08
LLaMA3-8B	26.64	26.05
GPT-3.5	50.82	56.83
GPT-4	57.44	53.53
LLM-GNN	-	-
TAPE (GCN)	-	-
OFA	71.56	59.33
All-in-One	70.83	56.20
GHGRL	-	-
HiGPT	49.73	51.04
GraphGPT	48.24	52.01
InstructGLM	49.38	34.35
MuseGraph (Qwen2-7B-Instruct)	70.87	**80.21**
MuseGraph (LLaMA1-7B)	76.52	65.56
MuseGraph (LLaMA2-7B)	70.09	67.84
MuseGraph (LLaMA3-8B)	**80.07**	69.92

B. Main Results Across Different Tasks (RQ1)

In this subsection, we provide a comprehensive performance analysis of our proposed MuseGraph framework across various graph tasks, evaluating its generation and adaptability capabilities in general, few-shot, and zero-shot settings compared with state-of-the-art baselines.

Overall, our proposed MuseGraph consistently demonstrates superior performance, highlighting its robust understanding of graph data combined with strong language generation capabilities (shown in Fig. 5). In Node Classification and Link Prediction tasks across six datasets, MuseGraph achieves an average ranking of 2.17. In few-shot Graph-to-Text tasks and Dynamic Reasoning tasks in the zero-shot setting, MuseGraph secures the first and second rankings, respectively. Specifically, despite GPT-4's strength in handling attribute-rich datasets and outperforming our MuseGraph in most Dynamic Reasoning tasks, it often struggles with adaptability across unfamiliar tasks and settings. The ability of MuseGraph to match GPT-4 in complex Dynamic Reasoning tasks with significantly fewer trainable 13,631,488 parameters and training data highlights not just resource efficiency but also a practical solution for real-world graph applications requiring high adaptability and lower computational demands.

Similarly, domain-specific models such as HiGPT and GraphGPT perform commendably within their respective training contexts (e.g., IMDB) but falter when faced with scenarios outside these predefined datasets. HINormer, while effective at managing complex relationships in graph-specific tasks, lacks the versatility needed for broader applications (e.g., Graph-to-Text and Dynamic Reasoning tasks). In contrast, LLaMA3-8B excels in Graph-to-Text due to its extensive linguistic pre-training but, like others, it encounters limitations when stepping beyond its core competencies. MuseGraph, however, maintains robust performance across a variety of tasks, effectively integrating and generalizing graph data and language generation abilities where other models show constraints.

TABLE VII
BLEU-4 SCORES FOR THE GRAPH-TO-TEXT TASK ON AGENDA AND WEBNLG DATASETS. GPT-4, HIGPT, AND GRAPHGPT RESULTS REFLECT PERFORMANCES WITHOUT FINE-TUNING. IN CONTRAST, MODELS LIKE BART-LARGE, T5-LARGE, QWEN2-7B-INSTRUCT, LLAMA3-8B, AND MUSEGRAPH DEMONSTRATE VARIED PERFORMANCES ACROSS DIFFERENT TRAINING INSTANCE SIZES RANGING FROM 50 TO 500. THE BEST PERFORMANCES ARE HIGHLIGHTED IN **BOLDFACE** AND THE SECOND RUNNERS ARE UNDERLINED.

Dataset	AGENDA				WebNLG			
# Instances	50	100	200	500	50	100	200	500
GPT-4	7.87				5.43			
HiGPT	8.49				2.66			
GraphGPT	8.23				2.45			
InstructGLM	5.97				5.68			
BART-large	9.91	10.58	11.99	12.49	26.59	28.53	30.62	33.15
T5-large	1.82	5.59	7.75	9.55	21.27	23.57	26.48	31.64
Qwen2-7B-Instruct	7.95	8.28	9.90	12.41	27.24	32.29	34.90	36.02
LLaMA3-8B	10.35	10.44	13.07	13.36	26.53	33.43	33.55	36.54
MuseGraph (Qwen2-7B-Instruct)	10.77	11.34	12.78	12.86	31.15	32.35	**35.89**	**36.87**
MuseGraph (LLaMA3-8B)	**11.32**	**13.11**	**13.28**	**14.64**	**31.46**	**33.97**	34.25	36.83

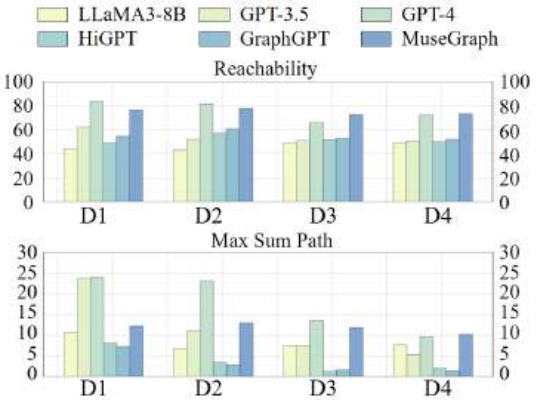


Fig. 6. Accuracy Results for the Reachability and Max Sum Path tasks on the graph with varying difficulty levels (i.e., D1 to D4), where "D" represents different degrees of complexity.

1) Compared with GNN-based Methods: As shown in Table V and Table VI, our MuseGraph outperforms GNN-based methods across most tasks and datasets, demonstrating its accurate comprehension of graph data. Notably, MuseGraph achieves significant performance gains in Heterogeneous Node Classification on IMDB and Freebase with an average of 13.99% and 19.64%. Moreover, MuseGraph consistently outperforms all GNN-based baselines in Link Prediction with an average performance improvement of 17.54%.

Specifically, while approaches like HINormer excel in Heterogeneous Node Classification, GraphSAGE and GAT in Homogeneous settings, and HGT in Link Prediction, they generally struggle with Graph-to-Text and Dynamic Reasoning tasks due to their limited adaptability and lack of linguistic features. In contrast, MuseGraph that can be easily tuned on different foundation models with a small number of corpora demonstrates a comprehensive understanding of graph data coupled with language generation capabilities. The overall performance across a variety of tasks and datasets makes MuseGraph a more practical solution than traditional GNN-based methods like HINormer and GAT.

2) Compared with LLM-based Methods: Generally, MuseGraph surpasses all LLM-based methods with significant improvements on both general graph tasks and few-shot ones, highlighting MuseGraph's superior capabilities in generic graph mining (shown in Table V, Table VI, and Table VII). Note that, by effective graph-aware tuning, our MuseGraph outperforms its foundation model LLaMA3-8B, ranging from 0.79% in BLEU-4 under 500 training instances on WebNLG to 1485.64% in Macro-F1 on Freebase.

In detail, GPT-4, known for its extensive parameter set and strong generalization ability, excels in Dynamic Reasoning tasks and attribute-rich datasets (as depicted in Fig. 6 and Table V). However, its large scale and closed-source nature often preclude fine-tuning, which limits its adaptability to complex graph structures and unfamiliar tasks. For example, when accessed via its official API without fine-tuning, GPT-4 produces verbose and inaccurate text in few-shot Graph-to-Text tasks, resulting in suboptimal BLEU-4 scores due to hallucinatory content. By comparison, open-source LLMs (e.g., Qwen2-7B-Instruct and LLaMA3-8B) that can be slightly fine-tuned demonstrate improved performance and reduced training costs. For instance, BART-large and LLaMA3-8B, tailored with a specific corpus of Graph-to-Text, gain average improvements over GPT-4 with 245.11% and 274.38%, respectively. However, without our proposed diverse instruction generation mechanism, these LLMs still encounter challenges in fully comprehending complex graph structures, reflecting inherent limitations in their adaptability to diverse graph data.

Note that, MuseGraph employs a graph-aware instruction tuning mechanism that utilizes fewer but more diverse CoT-based instruction packages and fewer trainable parameters. This approach not only enhances the precision of graph-oriented downstream tasks with a single NVIDIA GTX 3090 Ti GPU but also boosts generation capabilities, enabling MuseGraph to effectively match GPT-4 in complex Dynamic Reasoning tasks with improved efficiency and adaptability.

3) Compared with GNN+LLM-based Methods: Overall, our proposed MuseGraph exceeds the majority of GNN+LLM-based methods across various tasks and datasets, demonstrating its robust understanding of diverse graph data with powerful language generation capabilities (as depicted in Table V, Table VI, Table VII, and Fig. 6). For Heterogeneous Node Classification, the performance improvements range from 17.08% in Micro-F1 on IMDB to 71.57% in Macro-F1 on

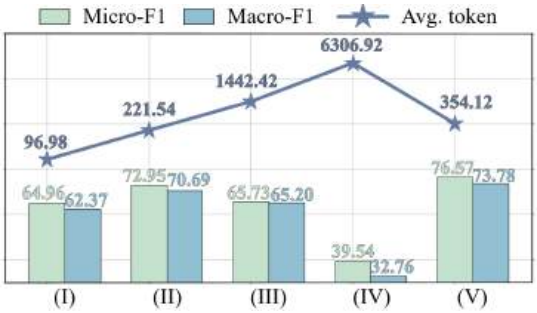


Fig. 7. Comparison of results for Heterogeneous Node Classification on IMDB across different graph descriptions as input. (I) through (IV) denote varying levels of neighbor inclusion: (I) utilizes central node attributes; (II) integrates one-hop neighbors; (III) includes up to two-hop neighbors; and (IV) encompasses three-hop neighbors. (V) employs a proposed method combining one-hop neighbors and random walks. Micro-F1 and Macro-F1 scores are shown alongside the average number of tokens used on IMDB.

TABLE VIII
ABLATION STUDY RESULTS FOR HETEROGENEOUS NODE CLASSIFICATION (NC) ON IMDB UNDER DIFFERENT SETTINGS OF FINE-TUNING. "NC PKG W/O. COT" EXCLUDES CHAIN OF THOUGHT (COT)-BASED INSTRUCTIONS, WHILE "NC PKG", "NC PKG + LP PKG (1:1)", AND "NC PKG + LP PKG (3:1)" INCLUDE THEM. NOTABLY, "NC PKG + LP PKG (1:1)" AND "NC PKG + LP PKG (3:1)" INTEGRATE LINK PREDICTION (LP) TASKS SPECIFICALLY TAILORED FOR THE IMDB DATASET. THE ALLOCATION RATIO 3:1 DENOTES THAT FOR EVERY FOUR COT-BASED INSTRUCTION PACKAGES USED, THREE ARE FROM THE NC TASK AND ONE IS FROM THE LP TASK. IN CONTRAST, OUR MUSEGRAPH (LLAMA3-8B) EMPLOYS COT-BASED INSTRUCTIONS DRIVEN BY MULTIPLE DATASETS AND TASKS, NOT LIMITED TO IMDB. PERFORMANCE IS MEASURED IN MICRO-F1 AND MACRO-F1 SCORES, WITH THE AVERAGE TOKEN COUNT ON IMDB PROVIDED.

Setting	Micro-F1	Macro-F1	Avg. token
NC pkg w/o. CoT	71.23	70.62	317.87
NC pkg	73.03	71.45	339.73
NC pkg + LP pkg (1:1)	72.81	71.07	346.09
NC pkg + LP pkg (3:1)	74.86	72.13	354.12
MuseGraph (LLaMA3-8B)	76.57	73.78	354.12

Freebase. Additionally, MuseGraph significantly outperforms all GNN+LLM-based methods in both Graph-to-Text and Dynamic Reasoning tasks, benefiting from its superior language generation and reasoning capabilities.

Some methods (e.g., LLM-GNN, TAPE, OFA, and GHGRL) train GNNs as predictors with LLM-enhanced features, combining the interpretative strengths of LLMs with GNNs to improve text-attributed graph representation learning. Particularly, OFA unifies different graph data by describing nodes and edges using natural language and converts them into a shared embedding space via LLMs. TAPE achieves top performance on node classification datasets like Cora and Arxiv by encoding LLMs' explanations. However, it relies on manually customized prompts and GPT-3.5 APIs to generate high-quality explanations for each dataset, which incurs high training costs and restricts its adaptability across diverse tasks and datasets. Moreover, these methods do not apply to Graph-to-Text and Dynamic Reasoning tasks due to their lack of language understanding and generation abilities. This further highlights the adaptability and flexibility of MuseGraph beyond traditional graph prediction tasks.

Another category of methods, like HiGPT, GraphGPT, and InstructGLM, utilizes LLMs to directly perform graph tasks. These methods typically depend on task/dataset-specific GNNs for graph encoding and integrate node representations into the input instructions, which leads to significant computational overhead and limited cross-dataset adaptability. By contrast, as illustrated in Table VII, MuseGraph achieves substantial performance gains by efficiently fine-tuning with even a limited number of examples. This ability highlights its effectiveness in adapting to different Graph-to-Text tasks and securing significant improvements in model performance. Note that HiGPT demonstrates flexibility in zero-shot learning across heterogeneous graph datasets, like when trained on IMDB and tested on DBLP [10]. However, it shows significant performance drops on homogeneous datasets such as Cora (shown in Table V), which highlights its limited adaptability to different graph structures. Additionally, HiGPT struggles with Link Prediction tasks due to a lack of task-specific training corpus, often yielding incorrect "Yes" responses.

These issues underline the critical need for GNN+LLM-based methods to incorporate task-specific training, boost computational efficiency, and achieve generalization across diverse graph structures. In contrast, MuseGraph introduces a compact graph description without specific GNNs to generate fewer diverse and high-quality instruction data under limited language tokens for fine-tuning. Our proposed approach effectively alleviates the content truncation problem while facilitating efficiency and adaptability across various graph tasks and datasets.

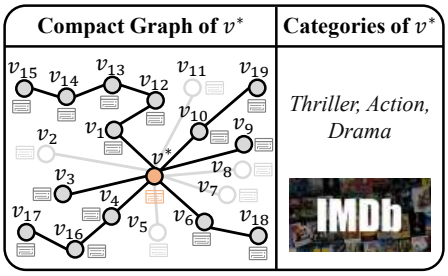
C. Ablation Studies (RQ2)

To demonstrate the impact of various model components, we conduct a series of ablation studies, including (1) Variations in Foundation Models, (2) Variations in Compact Graph Descriptions, (3) Variations in CoT-based Instruction Packages, and (4) Variations in Dynamic Instruction Package Allocation Strategies. We have the following observations.

1) Variations in Foundation Models: We explore several foundation models, including Qwen2-7B-Instruct, LLaMA1-7B, LLaMA2-7B, and LLaMA3-8B, across various graph-related tasks and datasets.

According to Table V, Table VI, and Table VII, MuseGraph (LLaMA3-8B) outperforms other variations of MuseGraph across most tasks and datasets, with performance improvements ranging from 0.96% in Macro-F1 on IMDB to 15.61% in BLEU-4 under 100 training instances on AGENDA. This indicates that a foundation model with larger parameters and extensive training can achieve better and more stable results with our graph-aware instruction tuning. Moreover, MuseGraph (Qwen2-7B-Instruct) also achieves remarkable results on MIMIC-III and WebNLG, underscoring the adaptability of our framework to harness the distinct capabilities of various LLMs, enhancing their robust understanding of graph data. Through the above results, MuseGraph proves to be cost-effective, by utilizing fine-tuning of diverse LLMs rather than relying on extensive new data and training from scratch.

2) Variations in Compact Graph Descriptions: We assess various compact graph descriptions on Heterogeneous Node



Name	Type	H
v^*: *Echo Dr.*	*MOVIE*	15
v_1: *Artificial Intelligence*	*KEYWORD*	17
v_2: *Hugh Mun*	*ACTOR*	4
v_3: *Security System*	*KEYWORD*	15
v_4: *Patrick Ryan Sims*	*DIRECTOR*	20
v_5: *Family Relationships*	*KEYWORD*	12
v_6: *Tech Noir*	*KEYWORD*	18
v_7: *Dane Bowman*	*ACTOR*	9
v_8: *John Pirkis*	*ACTOR*	5
...	...	...

Compact Graph Description: *The compact graph description of this MOVIE is listed as follows: Title: {v^*} Ego Graph Nodes: {v^*, v_1, v_3, v_4, v_6, ... , v_{18}, v_{19}} One-hop Neighbors: {v_4, v_1, v_9, v_3, v_{10}, v_6} Random Walks: {A. $v^* \rightarrow v_1 \rightarrow v_{12} \rightarrow v_{13} \rightarrow v_{14} \rightarrow v_{15}$; B. $v^* \rightarrow v_4 \rightarrow v_{16} \rightarrow v_{17}$; C. ...}*

HiGPT: ***Answer:*** ***[Romance, Thriller, Comedy, Action, Drama]***; ***Reasoning:*** *Based on the provided compact graph description, this MOVIE "Echo Dr." has an actor named Claire Gordon-Harper, who is known for her roles in romantic and dramatic movies. Additionally, it has the keywords "Home Invasion" and "Android", which is a common element in thriller and action movies. Finally, this MOVIE has an actor named Johnathan Hurley, who has been in comedic roles in the past.*

GPT-4: ***Answer:*** ***[Thriller, Drama, Science Fiction]***; ***Reasoning:*** *Analyzing the compact graph description for "Echo Dr.", the presence of keywords such as "Artificial Intelligence", "Tech Noir", and "Android" strongly suggest a setting focused on technology and its implications, typical of sci-fiction. The keyword "Home Invasion" aligns with the Thriller genre, as it involves ... in thriller movies. The keyword "Human Android Relationship" also suggests elements of Drama, exploring ....*

Qwen2-7B-Instruct: ***Answer:*** ***[Thriller, Action, Drama]***; ***Reasoning:*** *The MOVIE "Echo Dr." is associated with keywords such as "Artificial Intelligence", "Home Invasion", "Tech noir", ... These keywords suggest a futuristic setting with elements of technology and human-robot relationships, which are common in genres like Drama and Action. The "Home Invasion" keyword indicates a thriller aspect, as it often .... The "Android" keyword also points towards a Sci-Fi or Action genre.*

MuseGraph: ***Answer:*** ***[Thriller, Action, Drama]***; ***Reasoning:*** ***(1) Ego Graph Nodes:*** *The MOVIE "Echo Dr." is connected to other movies like "Blade Runner" and "Impostor", which are known for their elements of action, thriller, and drama, such as ....* ***(2) One-hop Neighbors:*** *The one-hop neighbors include elements like "Tech Noir", "Home Invasion", ..., These are characteristic of the Thriller and Action genres. The "Human Android Relationship" also add a dramatic element to the movie, reinforcing the Drama genre.* ***(3) Random Walks:*** *Walk A: This suggests a storyline that is likely dark, ..., typical of a Thriller. Walk B: connects "Echo Dr." with .... Walk C: ....*

Fig. 8. Heterogeneous Node Classification Results on IMDB from HiGPT, GPT-4, Qwen2-7B-Instruct, and MuseGraph, all utilizing our compact graph description as input. These models, selected for their generation and reasoning capabilities of LLMs, are compared based on their ability to analyze graphs. Best viewed in color.

Classification on the IMDB dataset, exploring models with different levels of neighborhood and walk integration. The simplest model (I) uses only central node attributes. Models (II) through (IV) gradually integrate more extensive neighborhood attributes, enhancing the model's context-awareness. Our advanced model (V) uniquely merges one-hop neighbors with random walks, striking an optimal balance between richness of information and token efficiency, as shown in Fig. 7.

Specifically, compared with description (I), which only has attribute information of the central node, description (II) integrates all one-hop neighbor attributes to achieve up to 13.34% improvements in Macro-F1. However, when extending the neighborhood to two-hop or multi-hop without sampling decreases effectiveness (i.e., (III) and (IV)), the performance significantly deteriorates due to an overload of input tokens. Conversely, model (V) effectively combines one-hop neighbors with random walks within a controlled token budget by calculating the node energy (cf., Eq. 1). This ensures a rich information intake and maintains token efficiency, significantly improving performance on complex graph structures.

3) Variations in CoT-based Instruction Packages: In the ablation study detailed in Table VIII, we investigate the impact of different variations in Chain of Thought (CoT)-based instruction packages on the IMDB dataset for the Heterogeneous Node Classification task.

When CoT-based instructions are incorporated with task-specific instructions (i.e., NC pkg w/o. CoT vs. NC pkg), there is a notable improvement in both Micro-F1 and Macro-F1 scores with only a 21.86 increase in the average number of tokens, indicating enhanced model understanding and reasoning capabilities with minimal impact on computational efficiency. Note that appropriately integrating CoT-instruction packages for Link Prediction tasks on IMDB (e.g., NC pkg + LP pkg (3:1)) demonstrates a compounded beneficial effect in Heterogeneous Node Classification, as further analyzed in Section IV-C4. This not only further improves the model's adaptability across various graph tasks, but also underscores the synergistic effect of diverse yet balanced CoT-based instruction packages on comprehension capabilities for generic graphs.

4) Variations in Dynamic Instruction Package Allocation Strategies: As shown in Table VIII, we assess the impact of varying CoT-based instruction package compositions, incorporating instruction packages from multiple tasks and datasets, and applying them to perform graph-aware instruction tuning.

By adjusting the allocation ratios of Link Prediction CoT-based instructions (e.g., a 3:1 ratio compared to a 1:1 ratio) within the NC pkg + LP pkg, we observed performance gains for Heterogeneous Node Classification, with Micro-F1 rising from 72.81 to 74.86. This demonstrates the critical need for carefully balancing instruction packages based on the complexity and requirements of each task, enabling LLMs to accurately understand and adapt to diverse graph-based tasks. Furthermore, when we diversify the instruction set with various graph tasks and datasets, our MuseGraph (LLaMA3-8B) can surpass NC pkg + LP pkg (3:1) with a superior performance improvement of 2.29% on average. This strongly indicates that the strategic dynamic instruction package allocation with different task complexities and dataset complexities can enhance model adaptability across different graph tasks and datasets while mitigating the risk of catastrophic forgetting.

D. Case Studies (RQ3)

To evaluate the effectiveness of compact graph descriptions in enhancing LLMs' comprehension of graph structures, we provide a Heterogeneous Node Classification scenario on the IMDB dataset. Fig. 8 contrasts the responses from HiGPT, GPT-4, Qwen2-7B-Instruct, and MuseGraph, which all interpret the same compact graph description but exhibit varying analytical abilities due to distinct LLM architectures.

Overall, the utilization of compact graph descriptions facilitates a deeper and more accurate analysis by encapsulating key graph data into a more manageable and interpretable format. This not only preserves critical information but also enhances the models' understanding and reasoning capabilities to generate relevant and context-aware responses, allowing models to focus on the pertinent details without the noise commonly associated with extensive graph data. As shown in the blue responses of Fig. 8, both Qwen2-7B-Instruct and MuseGraph benefit from this compact graph description, achieving correct predictions and analyses that reflect the insightful understanding of graph data.

Note that, HiGPT and GPT-4 perform wrong predictions and provide the additional categories for v^* (shown in the red responses of Fig. 8). HiGPT struggles with the compact graph description due to the absence of specific learned graph tokens for input. This limitation causes HiGPT to revert to its existing knowledge base for category analysis, leading to irrelevant and misleading suggestions, such as attributing the category "Comedy" to Johnathan Hurley based on his past roles, which does not align with v^*'s actual categories. GPT-4, with its extensive parameters and knowledge, excels in studying the semantic information of keywords presented in v^*'s compact graph description to predict the categories. However, this makes it ignore graph structures and leads to hallucinations, resulting in erroneous predictions like "Science Fiction". Although Qwen2-7B-Instruct shows remarkable performances on IMDB, it is hard to adapt to different tasks and datasets without further fine-tuning (depicted in Table V, Table VI, and Table VII).

In contrast, MuseGraph can accurately understand graph structures via compact graph description. For example, the one-hop neighbor "Home Invasion" and the walk "A" suggest a strong thriller element of node "v^*". These elements collectively enable MuseGraph to provide a detailed and contextually enriched genre prediction, showcasing its ability to synthesize complex graph structures into coherent and accurate analyses.

V. Conclusion and Future Work

In this paper, we introduce MuseGraph, an effective and generic approach for graph mining, which can enable the accurate understanding abilities of graph data across various tasks and datasets. Through the innovative design of compact graph descriptions with adaptive generation procedure, the generation of diverse task-specific Chain-of-Thought (CoT)-based instruction packages, and the implementation of graph-aware instruction tuning, our MuseGraph integrates the strengths of Graph Neural Networks (GNNs) and Large Language Models (LLMs) into one single foundation model. Our comprehensive experimental results demonstrate MuseGraph's superior performance against state-of-the-art baselines in five graph tasks and ten datasets, illustrating its ability not only to improve the precision of graph-related downstream tasks but also to enhance the generation capabilities of LLMs, which is further consolidated with our real case study results.

The primary limitation of our MuseGraph lies in the reliance on input graphs with rich semantic information and the careful selection of training datasets. Looking ahead, it is interesting to broaden the scope of MuseGraph by incorporating it with a wider range of graph types and exploring its applications in more diverse tasks. For instance, applying MuseGraph to biological graphs could be intriguing, where integrating domain-specific knowledge and expert feedback is crucial.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants (No.62302098) and Fujian Provincial Natural Science Foundation of China under Grants (2025J01540). Carl Yang was not supported by any fund from China.

References

[1] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in Neural Information Processing Systems*, vol. 30, pp. 1025–1035, 2017.

[2] G. Cui, J. Zhou, C. Yang, and Z. Liu, "Adaptive graph encoder for attributed graph embedding," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 976–985.

[3] Z. Chen, H. Mao, H. Wen, H. Han, W. Jin, H. Zhang, H. Liu, and J. Tang, "Label-free node classification on graphs with large language models (llms)," in *Proceedings of the 12th International Conference on Learning Representations*, 2024, pp. 31 632–31 655.

[4] X. He, X. Bresson, T. Laurent, A. Perold, Y. LeCun, and B. Hooi, "Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning," in *Proceedings of the 12th International Conference on Learning Representations*, 2024, pp. 5711–5732.

[5] H. Liu, J. Feng, L. Kong, N. Liang, D. Tao, Y. Chen, and M. Zhang, "One for all: Towards training one graph model for all classification tasks," in *Proceedings of the 12th International Conference on Learning Representations*, 2024, pp. 20 188–20 210.

[6] X. Sun, H. Cheng, J. Li, B. Liu, and J. Guan, "All in one: Multi-task prompting for graph neural networks," *In Proceedings of the 29rd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2120–2131, 2023.

[7] H. Gao, C. Zhang, F. Wu, C. Zheng, J. Zhao, and H. Liu, "Bootstrapping heterogeneous graph representation learning via large language models: A generalized approach," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025, pp. 16 717–16 726.

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

[9] J. Tang, Y. Yang, W. Wei, L. Shi, L. Su, S. Cheng, D. Yin, and C. Huang, "Graphgpt: Graph instruction tuning for large language models," in *Proceedings of the 47th ACM SIGIR International Conference on Research and Development in Information Retrieval*, 2024, pp. 491–500.

[10] J. Tang, Y. Yang, W. Wei, L. Shi, L. Xia, D. Yin, and C. Huang, "Higpt: Heterogeneous graph language model," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2024, pp. 2842–2853.

[11] H. Wang, S. Feng, T. He, Z. Tan, X. Han, and Y. Tsvetkov, "Can language models solve graph problems in natural language?" *Advances in Neural Information Processing Systems*, vol. 36, pp. 30 840–30 861, 2023.

[12] R. Ye, C. Zhang, R. Wang, S. Xu, and Y. Zhang, "Language is all a graph needs," in *Findings of the Association for Computational Linguistics: EACL 2024*, 2024, pp. 1955–1973.

[13] Q. Huang, H. Ren, P. Chen, G. Kržmanc, D. Zeng, P. S. Liang, and J. Leskovec, "Prodigy: Enabling in-context learning over graphs," *Advances in Neural Information Processing Systems*, vol. 36, pp. 16 302–16 317, 2023.

[14] C. Yang, Y. Xiao, Y. Zhang, Y. Sun, and J. Han, "Heterogeneous network representation learning: A unified framework with survey and benchmark," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 10, pp. 4854–4873, 2020.

[15] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1616–1637, 2018.

[16] Q. Mao, Z. Liu, C. Liu, and J. Sun, "Hinormer: Representation learning on heterogeneous information networks with graph transformer," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 599–610.

[17] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2016, pp. 855–864.

[18] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2017, pp. 135–144.

[19] S. Ivanov and E. Burnaev, "Anonymous walk embeddings," in *Proceedings of the 35th International Conference on Machine Learning*, 2018, pp. 2186–2195.

[20] J. Li, H. Peng, Y. Cao, Y. Dou, H. Zhang, S. Y. Philip, and L. He, "Higher-order attribute-enhancing heterogeneous graph neural networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 1, pp. 560–574, 2021.

[21] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proceedings of the 5th International Conference on Learning Representations*, 2017.

[22] C. Yang, J. Zhang, H. Wang, S. Li, M. Kim, M. Walker, Y. Xiao, and J. Han, "Relation learning on social networks with multi-modal graph edge variational autoencoders," in *Proceedings of the 13th International Conference on Web Search and Data Mining*, 2020, pp. 699–707.

[23] W.-Z. Li, C.-D. Wang, J.-H. Lai, and S. Y. Philip, "Towards effective and robust graph contrastive learning with graph autoencoding," *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 2, pp. 868–881, 2023.

[24] T. Qian, Y. Liang, Q. Li, and H. Xiong, "Attribute graph neural networks for strict cold start recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 8, pp. 3597–3610, 2020.

[25] H. Zhang, Y. Ren, L. Fu, X. Wang, G. Chen, and C. Zhou, "Multi-scale self-supervised graph contrastive learning with injective node augmentation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 1, pp. 261–274, 2023.

[26] X. Jiang, T. Jia, Y. Fang, C. Shi, Z. Lin, and H. Wang, "Pre-training on large-scale heterogeneous graph," in *Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2021, pp. 756–766.

[27] N. Liu, X. Wang, H. Han, and C. Shi, "Hierarchical contrastive learning enhanced heterogeneous graph neural network," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 10, pp. 10 884–10 896, 2023.

[28] M. Sun, K. Zhou, X. He, Y. Wang, and X. Wang, "Gppt: Graph pre-training and prompt tuning to generalize graph neural networks," in *Proceedings of the 28th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2022, pp. 1717–1727.

[29] Y. Li, Z. Li, P. Wang, J. Li, X. Sun, H. Cheng, and J. X. Yu, "A survey of graph meets large language model: Progress and future directions," in *Proceedings of the 33th International Joint Conference on Artificial Intelligence*, 2024, pp. 8123–8131.

[30] M. Fazel and M. Chiang, "Network utility maximization with nonconcave utilities using sum-of-squares method," in *Proceedings of the 44th IEEE Conference on Decision and Control*. IEEE, 2005, pp. 1867–1874.

[31] H. Lian, Y. Xiong, Z. Lin, J. Niu, S. Mo, H. Chen, P. Liu, and G. Ding, "Lbpe: Long-token-first tokenization to improve large language models," in *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2025, pp. 1–5.

[32] S. Acevedo, A. Mascaretti, R. Rende, M. Mahaut, M. Baroni, and A. Laio, "An approach to identify the most semantically informative deep representations of text and images," *arXiv preprint arXiv:2505.17101*, 2025.

[33] J. Melton and S. Krishnan, "muxgnn: Multiplex graph neural network for heterogeneous graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 9, pp. 11 067–11 078, 2023.

[34] Y. Wang, T. Zhao, Y. Zhao, Y. Liu, X. Cheng, N. Shah, and T. Derr, "A topological perspective on demystifying gnn-based link prediction performance," in *Proceedings of the 12th International Conference on Learning Representations*, 2024, pp. 20 884–20 922.

[35] J. Gao, X. Zou, Y. Ai, D. Li, Y. Niu, B. Qi, and J. Liu, "Graph counselor: Adaptive graph exploration via multi-agent synergy to enhance llm reasoning," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 24 650–24 668.

[36] Y. Gao, X. Wang, X. He, Z. Liu, H. Feng, and Y. Zhang, "Addressing heterophily in graph anomaly detection: A perspective of graph spectrum," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 1528–1538.

[37] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback," *Advances in Neural Information Processing Systems*, vol. 35, pp. 27 730–27 744, 2022.

[38] Y. Wang, H. Ivison, P. Dasigi, J. Hessel, T. Khot, K. Chandu, D. Wadden, K. MacMillan, N. A. Smith, I. Beltagy *et al.*, "How far can camels go? exploring the state of instruction tuning on open resources," *Advances in Neural Information Processing Systems*, pp. 74 764–74 786, 2023.

[39] K. Shridhar, A. Stolfo, and M. Sachan, "Distilling reasoning capabilities into smaller language models," in *Findings of the Association for Computational Linguistics: ACL 2023*, 2023, pp. 7059–7073.

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

[41] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 824–24 837, 2022.

[42] Z. Xiang, F. Jiang, Z. Xiong, B. Ramasubramanian, R. Poovendran, and B. Li, "Badchain: Backdoor chain-of-thought prompting for large language models," in *Proceedings of the 12th International Conference on Learning Representations*, 2024, pp. 28 289–28 316.

[43] Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, and H. Hajishirzi, "Self-instruct: Aligning language models with self-generated instructions," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 2023, pp. 13 484–13 508.

[44] M. Mosbach, M. Andriushchenko, and D. Klakow, "On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines," in *Proceedings of the 8th International Conference on Learning Representations*, 2020.

[45] S. Vijay and A. Priyanshu, "Nerda-con: Extending ner models for continual learning–integrating distinct tasks and updating distribution shifts," in *Proceedings of the Updatable Machine Learning (UpML) Workshop at the 39th International Conference on Machine Learning (ICML)*. PMLR, 2022.

[46] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan *et al.*, "The llama 3 herd of models," *arXiv e-prints*, pp. arXiv–2407, 2024.

[47] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," in *Proceedings of the 10th International Conference on Learning Representations*, 2022.

[48] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "Qlora: Efficient finetuning of quantized llms," *Advances in Neural Information Processing Systems*, vol. 36, pp. 10 088–10 115, 2023.

[49] Q. Zhang, M. Chen, A. Bukharin, P. He, Y. Cheng, W. Chen, and T. Zhao, "Adaptive budget allocation for parameter-efficient fine-tuning," in *Proceedings of the 11th International Conference on Learning Representations*, 2023.

[50] Z. Gao, Q. Wang, A. Chen, Z. Liu, B. Wu, L. Chen, and J. Li, "Parameter-efficient fine-tuning with discrete fourier transform," in *Proceedings of the 41st International Conference on Machine Learning*, 2024, pp. 14 884–14 901.

[51] K. Zhu, J. Chen, J. Wang, N. Z. Gong, D. Yang, and X. Xie, "Dyval: Dynamic evaluation of large language models for reasoning tasks," in *Proceedings of the 12th International Conference on Learning Representations*, 2024, pp. 18 091–18 128.

[52] Q. Lv, M. Ding, Q. Liu, Y. Chen, W. Feng, S. He, C. Zhou, J. Jiang, Y. Dong, and J. Tang, "Are we really making much progress? revisiting, benchmarking and refining heterogeneous graph neural networks," in *Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2021, pp. 1150–1160.

[53] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec, "Open graph benchmark: Datasets for machine learning on graphs," *Advances in Neural Information Processing Systems*, vol. 33, pp. 22 118–22 133, 2020.
[54] Y. Tan, Z. Zhou, H. Lv, W. Liu, and C. Yang, "Walklm: A uniform language model fine-tuning framework for attributed graph embedding," *Advances in Neural Information Processing Systems*, vol. 36, pp. 13 308–13 325, 2023.
[55] J. Li, T. Tang, W. X. Zhao, Z. Wei, N. J. Yuan, and J.-R. Wen, "Few-shot knowledge graph-to-text generation with pretrained language models," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021, pp. 1558–1568.
[56] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 2002, pp. 311–318.
[57] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proceedings of the 6th International Conference on Learning Representations*, 2018.
[58] G. Li, M. Müller, B. Ghanem, and V. Koltun, "Training graph neural networks with 1000 layers," in *Proceedings of the 38th International Conference on Machine Learning*. PMLR, 2021, pp. 6437–6449.
[59] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *15th International Conference on Extended Semantic Web Conference*, 2018, pp. 593–607.
[60] Z. Hu, Y. Dong, K. Wang, and Y. Sun, "Heterogeneous graph transformer," in *Proceedings of the ACM Web Conference 2020*, 2020, pp. 2704–2710.
[61] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 7871–7880.
[62] L. F. Ribeiro, M. Schmitt, H. Schütze, and I. Gurevych, "Investigating pretrained language models for graph-to-text generation," in *Proceedings of the 3rd Workshop on Natural Language Processing for Conversational AI*, 2021, pp. 211–227.
[63] A. Colas, M. Alvandipour, and D. Z. Wang, "Gap: A graph-aware language model framework for knowledge graph-to-text generation," in *Proceedings of the 29th International Conference on Computational Linguistics*, 2022, pp. 5755–5769.
[64] A. Yang, B. Xiao, B. Wang, B. Zhang, C. Bian, C. Yin, C. Lv, D. Pan, D. Wang, D. Yan *et al.*, "Baichuan 2: Open large-scale language models," *arXiv preprint arXiv:2309.10305*, 2023.
[65] A. Yang, B. Yang, B. Hui, B. Zheng, B. Yu, C. Zhou, C. Li, C. Li, D. Liu, F. Huang *et al.*, "Qwen2 technical report," *arXiv preprint arXiv:2407.10671*, 2024.
[66] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.
[67] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.
[68] Y. Zheng, R. Zhang, J. Zhang, Y. Ye, and Z. Luo, "Llamafactory: Unified efficient fine-tuning of 100+ language models," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, 2024, pp. 400–410.
[69] Y. Cen, Z. Hou, Y. Wang, Q. Chen, Y. Luo, Z. Yu, H. Zhang, X. Yao, A. Zeng, S. Guo *et al.*, "Cogdl: A comprehensive library for graph deep learning," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 747–758.

Yanchao Tan is currently working as an Associate Professor with the College of Computer and Data Science, and Fujian Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, Fuzhou, China. She obtained her Ph.D. degree from the College of Computer Science, Zhejiang University, Hangzhou, China, in 2022. She was a Visiting Scholar at The Chinese University of Hong Kong from 2024 to 2025. Her research interests include data mining, healthcare, recommender systems, and large language models.

Hang Lv received a BS degree in computer science from Fuzhou University, China, in 2023. He is currently working toward a Ph.D. degree in computer science and technology at Fuzhou University, China. His research interests include data mining, healthcare, recommender systems, large language models, and graph representation learning.

Pengxiang Zhan received a BS degree in computer science from Fuzhou University, China, in 2023. He is currently pursuing a Master's degree in computer science and technology at Fuzhou University, China. His research interests include language models and multimodal large models.

Shiping Wang is currently working as a Professor with the College of Computer and Data Science, Fuzhou University, Fuzhou, China, and the Director of the Key Laboratory of Intelligent Metro, Fujian Province University, Fuzhou, China. He received his Ph.D. degree from the University of Electronic Science and Technology of China, Chengdu, China in 2014. He was a Visiting Scholar in University of Alberta, Edmonton, Canada from August 2013 to August 2014. He worked as a Research Assistant in National University of Singapore from January 2014 to August 2014, and a Research Fellow in Nanyang Technological University of Singapore from August 2015 to August 2016. He was also a Visiting Researcher in Peking University, Beijing, China from August 2019 to August 2020. His research interests include machine learning, deep learning, feature representation and multi-modal fusion.

Carl Yang is an Assistant Professor of Computer Science at Emory University, jointly appointed in the Rollins School of Public Health and Nell Hodgson Woodruff School of Nursing. He received his Ph.D. in Computer Science at University of Illinois, Urbana-Champaign in 2020, and B.Eng. in Computer Science and Engineering at Zhejiang University in 2014. His research interests span data mining, knowledge graphs, multimodality foundation models, and trustworthy AI, with applications in network sciences, neuroscience, biomedicine, and healthcare. Carl's research results have led to 200+ peer-reviewed publications in top venues across AI/ML and medicine/healthcare. He is also a recipient of the SIGKDD Rising Star Award in 2025, NSF CAREER Award in 2025, NIH K25 (Career) Award in 2023, and multiple Best Paper Awards such as of MedInfo 2025, KDD Health Day 2022, ML4H 2022, and ICDM 2020.