A Survey of Graph Meets Large Language Model: Progress and Future Directions

Yuhan Li\textsuperscript{1}, Zhixun Li\textsuperscript{2}, Peisong Wang\textsuperscript{3}, Jia Li\textsuperscript{1}, Xiangguo Sun\textsuperscript{2}, Hong Cheng\textsuperscript{2} and Jeffrey Xu Yu\textsuperscript{2}
\textsuperscript{1}The Hong Kong University of Science and Technology (Guangzhou)
\textsuperscript{2}SHIAE and SEEM, The Chinese University of Hong Kong
\textsuperscript{3}Tsinghua University

Abstract

Graph plays a significant role in representing and analyzing complex relationships in real-world applications such as citation networks, social networks, and biological data. Recently, Large Language Models (LLMs), which have achieved tremendous success in various domains, have also been leveraged in graph-related tasks to surpass traditional Graph Neural Networks (GNNs) based methods and yield state-of-the-art performance. In this survey, we first present a comprehensive review and analysis of existing methods that integrate LLMs with graphs. First of all, we propose a new taxonomy, which organizes existing methods into three categories based on the role (i.e., enhancer, predictor, and alignment component) played by LLMs in graph-related tasks. Then we systematically survey the representative methods along the three categories of the taxonomy. Finally, we discuss the remaining limitations of existing studies and highlight promising avenues for future research. The relevant papers are summarized and will be consistently updated at: https://github.com/yhLeeee/Awesome-LLMs-in-Graph-tasks.

1 Introduction

Graph, or graph theory, serves as a fundamental part of numerous areas in the modern world, particularly in technology, science, and logistics [Ji et al., 2021]. Graph data represents the structural characteristics between nodes, thus illuminating relationships within the graph’s components. Many real-world datasets, such as citation networks [Sen et al., 2008], social networks [Hamilton et al., 2017], and molecular [Wu et al., 2018], are intrinsically represented as graphs. To tackle graph-related tasks, Graph Neural Networks (GNNs) [Kipf and Welling, 2016; Velickovic et al., 2018] have emerged as one of the most popular choices for processing and analyzing graph data. The main objective of GNNs is to acquire expressive representations at the node, edge, or graph level for different kinds of downstream tasks through recursive message passing and aggregation mechanisms among nodes.

In recent years, significant advancements have been made in Large Language Models (LLMs) like Transformers [Vaswani et al., 2017], BERT [Kenton and others, 2019], GPT [Brown et al., 2020], and their variants. These LLMs can be easily applied to various downstream tasks with little adaptation, demonstrating remarkable performance across various natural language processing tasks, such as sentiment analysis, machine translation, and text classification [Zhao et al., 2023c]. While their primary focus has been on text sequences, there is a growing interest in enhancing the multi-modal capabilities of LLMs to enable them to handle diverse data types, including graphs, images, and videos.

LLMs help graph-related tasks. With the help of LLMs, there has been a notable shift in the way we interact with graphs, particularly those containing nodes associated with text attributes. As shown in Figure 1, the integration of graphs and LLMs demonstrates success in various downstream tasks across a myriad of graph domains. Integrating LLMs with traditional GNNs can be mutually beneficial and enhance graph learning. While GNNs are proficient at capturing structural information, they primarily rely on semantically constrained embeddings as node features, limiting their ability to express the full complexities of the nodes. Incorporating LLMs, GNNs can be enhanced with stronger node features that effectively capture both structural and contextual aspects. On the other hand, LLMs excel at encoding text but often struggle to capture structural information present in graph data. Combining GNNs with LLMs can leverage the robust textual understanding of LLMs while harnessing GNNs’ ability to capture structural relationships, leading to more powerful graph learning. To achieve a better systematic overview, as shown in Figure 2, we...
Graph Meets Large Language Models

Figure 2: A taxonomy of models for solving graph tasks with the help of large language models (LLMs) with representative examples.

2 Preliminary

2.1 Graph Neural Networks

Definitions. Most existing GNNs follow the message-passing paradigm which contains message aggregation and feature update, such as GCN [Kipf and Welling, 2016] and GAT [Velickovic et al., 2018]. They generate node representations by iteratively aggregating information of neighbors and updating them with non-linear functions. The forward process can be defined as:

\[ h_i^{(l)} = U \left( h_i^{(l-1)}, M\left( h_j^{(l-1)}, h_i^{(l-1)} | v_j \in N_i \right) \right) \]

where \( h_i^{(l)} \) is the feature vector of node \( i \) in the \( l \)-th layer, and \( N_i \) is a set of neighbor nodes of node \( i \). \( M \) denotes the message passing function of aggregating neighbor information. \( U \) denotes the update function with central node feature and neighbor node features as input. By stacking multiple layers, GNNs can aggregate messages from higher-order neighbors.

Definitions. While there is currently no clear definition for LLMs [Shayegani et al., 2023], here we provide a specific definition for LLMs mentioned in this survey. Two influential surveys on LLMs [Zhao et al., 2023c; Yang et al., 2023] distinguish between LLMs and pre-trained language models (PLMs) from the perspectives of model size and training approach. To be specific, LLMs are those huge language models (i.e., billion-level) that undergo pre-training on a significant amount of data, whereas PLMs refer to those early pre-trained models with moderate parameter sizes (i.e., million-level), which can be easily further fine-tuned on task-specific data to achieve better results to downstream tasks. Due to the relatively smaller parameter size of GNNs, incorporating GNNs and LLMs often does not require LLMs with large parameters. Hence, we follow [Liu et al., 2023a] to extend the definition of LLMs in this survey to encompass both LLMs and PLMs as defined in previous surveys.

Evolution. LLMs can be divided into two categories based on non-autoregressive and autoregressive language modeling. Non-autoregressive LLMs typically concentrate on natural language understanding and employ a “masked language modeling” pre-training task, while autoregressive LLMs focus more on natural language generation, frequently leveraging the “next token prediction” objective as their foundational task.

Classic encoder-only models such as BERT and RoBERTa fall under the category of non-autoregressive LLMs. Recently, autoregressive LLMs have witnessed continuous development.
Examples include Flan-T5 and ChatGLM, which are built upon the encoder-decoder structure, as well as GPT-3 and LLaMA, which are based on decoder-only architectures. Significantly, advancements in architectures and training methodologies of LLMs have given rise to emergent capabilities [Wei et al., 2022a], which is the ability to handle complex tasks in few-shot or zero-shot scenarios via some techniques such as in-context learning [Radford et al., 2021] and chain-of-thought [Wei et al., 2022b].

2.3 Proposed Taxonomy

We propose a taxonomy (as illustrated in Figure 2) that organizes representative techniques involving both graph and text modalities into three main categories: (1) LLM as Enhancer, where LLMs are used to enhance the classification performance of GNNs. (2) LLM as Predictor, where LLMs utilize the input graph structure information to make predictions. (3) GNN-LLM Alignment, where LLMs semantically enhance GNNs through alignment techniques. We note that in some models, due to the rarity of LLMs’ involvement, it becomes difficult to categorize them into these three main classes. Therefore, we separately organize them into the “Others” category with specific roles. For example, LLM-GNN [Chen et al., 2024] actively selects nodes for ChatGPT to annotate, thereby augmenting the GNN training by utilizing the LLM as an annotator. GPT4GNAS [Wang et al., 2023a] considers the LLM as an experienced controller in the task of graph neural architecture search, and utilizes GPT-4 [OpenAI, 2023] to explore the search space. Furthermore, ENG [Yu et al., 2023] empowers the LLM as a sample generator to generate additional training samples with labels to provide sufficient supervision signals for GNNs.

In the following sections, we present a comprehensive survey along the three main categories of our taxonomy for incorporating LLMs into graph-related tasks, respectively.

3 LLM as Enhancer

GNNs have become powerful tools to analyze graph-structure data. However, the most mainstream benchmark datasets (e.g., Cora and Ogbn-Arxiv) adopt naive methods to encode text information in text-attributed graphs (TAGs) using shallow embeddings, such as bag-of-words, skip-gram [Mikolov et al., 2013], or TF-IDF [Salton and Buckley, 1988]. This inevitably constrains the performance of GNNs on TAGs. LLM-as-enhancer approaches correspond to enhancing the quality of node embeddings with the help of powerful LLMs. The derived embeddings are attached to the graph structure to be utilized by any GNNs or directly inputted into downstream classifiers for various tasks. We naturally categorize these approaches into two branches: explanation-based and embedding-based, depending on whether they use LLMs to produce additional textual information.

3.1 Explanation-based Enhancement

To enrich the textual attributes, explanation-based enhancement approaches focus on utilizing the strong zero-shot capability of LLMs to capture higher-level information. As shown in Figure 3(a), generally they prompt LLMs to generate semantically enriched additional information, such as explanations, knowledge entities, and pseudo labels.

For instance, TAPE [He et al., 2024] is a pioneer work of explanation-based enhancement, which prompts LLMs to generate explanations and pseudo labels to augment textual attributes. After that, relatively small language models are fine-tuned on both original text data and explanations to encode text semantic information as initial node embeddings. [Chen et al., 2023] explore the potential competence of LLMs in graph learning. They first compare embedding-visible LLMs with shallow embedding methods and then propose KEA to enrich the text attributes. KEA prompts LLMs to generate a list of knowledge entities along with text descriptions and encodes them by fine-tuned PLMs and deep sentence embedding models.

LLM4Mol [Qian et al., 2023] attempts to employ LLMs to assist in molecular property prediction. Specifically, it uses LLMs to generate semantically enriched explanations for the original SMILES and then fine-tunes a small-scale language model to conduct downstream tasks. LLMRec [Wei et al., 2023] aims to utilize LLMs to figure out data sparsity and data quality issues in the graph recommendation system. It reinforces user-item interaction edges and generates user/item side information by LLMs. Lastly, it employs a lightweight GNN to encode the augmented recommendation network.

3.2 Embedding-based Enhancement

Refer to Figure 3(b), embedding-based enhancement approaches directly utilize LLMs to output text embeddings as initial node embeddings for GNN training. This kind of approach requires the use of embedding-visible or open-source LLMs because it needs to access text embeddings straightforward or fine-tune LLMs with structural information. Many of the current advanced LLMs (e.g., GPT4 [OpenAI, 2023] and PaLM [Chowdhery et al., 2022]) are closed-source and only provide online services. Strict restrictions prevent researchers from accessing their parameters and output embeddings. Embedding-based approaches mostly adopt a cascading form and utilizes structure information to assist the language model in pre-training or fine-tuning.

Typically, GaLM [Xie et al., 2023] pre-trains PLMs and GNN aggregator on a given large graph corpus to capture the information that can maximize utility towards massive applications and then fine-tunes the framework on a specific
downstream application to further improve the performance. Several works also aim to generate node embeddings by incorporating structural information into the fine-tuning phase of LLMs. GIANT [Chien et al., 2021], SimTeG [Duan et al., 2023] and TouchUp-G [Zhu et al., 2023] follow a similar way, they both fine-tune PLMs through link-prediction-like methods to help them perceive structural information. The subtle difference between them is that GIANT employs XR-Transformer to solve extreme multi-label classification over link prediction, TouchUp-G uses negative sampling during link prediction, while SimTeG employs parameter-efficient fine-tuning to accelerate the fine-tuning process. G-Prompt [Huang et al., 2023b] introduces a graph adapter at the end of PLMs to help extract graph-aware node features. Once trained, task-specific prompts are incorporated to produce interpretable node representations for various downstream tasks. WalkLM [Tan et al., 2023] is an unsupervised generic graph representation learning method. The first step of it is to generate attributed random walks on the graph and compose roughly meaningful textual sequences by automated textualization program. The second step is to fine-tune an LLM using textual sequences and extract representations from LLM. METERN [Jin et al., 2023a] introduces relation prior tokens to capture the relation-specific signals and uses one language encoder to model the shared knowledge across relations. LEADING [Xue et al., 2023] effectively fine tunes LLMs and transfers risk knowledge in LLM to downstream GNN model with less computation cost and memory overhead.

A recent work, OFA [Liu et al., 2024], attempts to propose a general graph learning framework, which can utilize a single graph model to conduct adaptive downstream prediction. It describes all nodes and edges using human-readable texts and encodes them from different domains into the same space by LLMs. Subsequently, the framework is adaptive to perform different tasks by inserting task-specific prompting substructures into the input graph.

3.3 Discussions

LLM-as-enhancer approaches have demonstrated superior performance on TAG, being able to effectively capture both textual and structural information. Moreover, they also exhibit strong flexibility, as GNNs and LLMs are plug-and-play, allowing them to leverage the latest techniques to address the encountered issues. However, despite some papers claiming strong scalability, in fact, LLM-as-enhancer approaches entail significant overhead when dealing with large-scale datasets. Taking explanation-based approaches as an example, they need to query LLMs’ APIs for $N$ times for a graph with $N$ nodes, which is indeed a substantial cost.

4 LLM as Predictor

The core idea behind this category is to utilize LLMs to make predictions for a wide range of graph-related tasks, such as classifications and reasonings, within a unified generative paradigm. However, applying LLMs to graph modalities presents unique challenges, primarily because graph data often lacks straightforward transformation into sequential text, as different graphs define structures and features in different ways. We classify the models broadly into flatten-based and GNN-based predictions, depending on whether they employ GNNs to extract structural features for LLMs.

4.1 Flatten-based Prediction

The majority of the existing attempts that utilize LLMs as predictors employ the strategy of flattening the graph into textual descriptions, which facilitates direct processing of graph data by LLMs through text sequences. As shown in Figure 4(a), flatten-based prediction typically involves two steps: (1) utilizing a flatten function to transform a graph structure into a sequence of nodes or tokens, and (2) a parsing function is then applied to retrieve the predicted label from the output generated by LLMs. As the core of flatten-based prediction, a variety of flatten functions has been leveraged.

GPT4Graph [Guo et al., 2023] utilizes graph description languages such as GML and GraphML to represent graphs. These languages provide standardized syntax and semantics for representing the nodes and edges within a graph. Inspired by linguistic syntax trees, GraphText [Zhao et al., 2023b] leverages graph-syntax trees to convert a graph structure to a sequence of nodes, which is then fed to LLMs for training-free graph reasoning. Furthermore, ReLM [Shi et al., 2023] uses SMILES strings to provide one-dimensional linearizations of molecular graph structures. GIMLET [Zhao et al., 2023a] adopts distance-based position embedding to extend the capability of LLMs to perceive graph structures. Graph data can be also represented through methods like adjacency matrices and adjacency lists. Several methods [Wang et al., 2023b; Fatemi et al., 2023; Liu and Wu, 2023; Zhang et al., 2023a] directly employ numerically organized node and edge lists to depict the graph data in plain text. GraphTMI [Das et al., 2023] further explores different modalities such as motif and image to integrate graph data with LLMs.

The use of natural narration to express graph structures is also making steady progress. [Chen et al., 2023] and [Hu et al., 2023] both integrate the structural information of citation networks into the prompts, which is achieved by explicitly representing the edge relationship through the word “cite” and representing the nodes using paper indexes or titles. [Huang et al., 2023a], on the other hand, does not use the word “cite” to represent edges but instead describes the relationships via enumerating randomly selected $k$-hop neighbors of the current node. Similarly, InstructGLM [Ye et al., 2023] designs a series of scalable prompts based on the maximum hop level. These prompts allow a central paper node to establish direct associa-
tions with its neighbors up to any desired hop level by utilizing the described connectivity relationships expressed in natural language. In addition, GPT4Graph [Guo et al., 2023] and [Chen et al., 2023] imitate the aggregation behavior of GNNs and summarize the current neighbor’s attributes as additional inputs, aiming to provide more structural information.

4.2 GNN-based Prediction

GNNs have demonstrated impressive capabilities in understanding graph structures through recursive information exchange and aggregation among nodes. As illustrated in Figure 4(b), in contrast to flatten-based prediction, which converts graph data into textual descriptions as inputs to LLMs, GNN-based prediction leverages the advantages of GNNs to incorporate inherent structural characteristics and dependencies present in graph data with LLMs, allowing LLMs to be structure-aware. GNN-based prediction also relies on a parser to extract the output from LLMs. Integrating GNN representations into LLMs often requires tuning, making it easier to standardize the prediction format of LLMs by providing desirable outputs during training.

Various strategies have been proposed to fuse the structural patterns learned by GNNs and the contextual information captured by LLMs. For instance, GIT-Mol [Liu et al., 2023b] and MolCA [Liu et al., 2023c] both implement BLIP-2’s Q-Former [Li et al., 2023a] as the cross-modal projector to map the graph encoder’s output to the LLM’s input text space. Multiple objectives with different attention masking strategies are employed for effective graph-text interactions. GraphLLM [Chai et al., 2023] derives the graph-enhanced prefix by applying a linear projection to the graph representation during prefix tuning, allowing the LLM to synergize with the graph transformer to incorporate structural information crucial to graph reasoning. Additionally, both GraphGPT [Tang et al., 2023] and InstructMol [Cao et al., 2023] employ a simple linear layer as the lightweight alignment projector to map the encoded graph representation to some graph tokens, while the LLM excels at aligning these tokens with diverse text information. DGTL [Qin et al., 2023] injects the disentangled graph embeddings directly into each layer of the LLM, highlighting different aspects of the graph’s topology and semantics.

4.3 Discussions

Utilizing LLMs directly as predictors shows superiority in processing textual attributes of graphs, especially achieving remarkable zero-shot performance compared with traditional GNNs. The ultimate goal is to develop and refine methods for encoding graph-structured information into a format that LLMs can comprehend and manipulate effectively and efficiently. Flatten-based prediction may have an advantage in terms of effectiveness, while GNN-based prediction tends to be more efficient. In flatten-based prediction, the input length limitation of LLMs restricts each node’s access to only its neighbors within a few hops, making it challenging to capture long-range dependencies. Additionally, without the involvement of GNNs, inherent issues of GNNs such as heterophily cannot be addressed. On the other hand, for GNN-based prediction, training an additional GNN module and inserting it into LLMs for joint training is challenging due to the problem of vanishing gradients in the early layers of deep transformers.

5 GNN-LLM Alignment

The alignment of GNNs and LLMs offers an efficient method for integrating graph and text data. This alignment retains the distinct capabilities of each encoder by synchronizing their embedding spaces at a certain point. We categorize alignment techniques into symmetric and asymmetric: symmetric alignment treats GNNs and LLMs equally, whereas asymmetric alignment prioritizes one over the other.

5.1 Symmetric

Symmetric alignment refers to the equal treatment of the graph and text modalities during alignments. These approaches ensure that the encoders of both modalities achieve comparable performance in their respective applications.

A typical symmetric alignment architecture shown in Figure 5(a) adopts a two-tower style, employing separate encoders to individually encode the graph and text. Interaction between these modalities occurs only once during alignment. Approaches like SAFER [Chandra et al., 2020] typically use basic concatenation for combining these embeddings. Recent developments in two-tower models increasingly leverage contrastive learning, similar to the CLIP [Radford et al., 2021], for more effective alignment of different modalities. The methods generally involve a two-step process: initially extracting graph and text representations, followed by applying contrastive learning with a modified InfoNCE loss [Oord et al., 2018]. Text2Mol [Edwards et al., 2021] introduces a cross-modal attention mechanism for early fusion of graph and text embeddings. It employs a transformer decoder, using LLM output as the source sequence and GNN output as the target sequence.

Figure 5: The illustration of GNN-LLM-Alignment approaches.
The decoder’s output is used for contrastive learning, paired with GNN-processed outputs. MoMu [Su et al., 2022], MoleculeST M [Liu et al., 2022], ConGraT [Brannon et al., 2023], and RLMRec [Ren et al., 2023] share a similar framework, which adapts paired graph embeddings and text embeddings to implement contrastive learning. MoMu and MoleculeST both source molecules from PubChem. MoMu pairs these with texts from scientific papers, whereas MoleculeST uses molecules’ descriptions. ConGraT extends this graph-text pairing approach to social, knowledge, and citation networks. RLMRec aligns LLMs’ semantic space with user-item interaction signals in recommendation systems. G2P2 [Wen and Fang, 2023] and Grenade [Li et al., 2023b] have further advanced the use of contrastive learning. G2P2 enhances the granularity by employing contrastive learning at three levels during the pre-training stage: node-text, text-text summary, and node-node summary. Grenade is optimized by integrating graph-centric contrastive learning with dual-level graph-centric knowledge alignment, which includes both node-level and neighborhood-level alignment.

The iterative alignment, depicted in Figure 5(b), distinguishes itself by allowing iterative interaction between the modalities. For example, GLEM [Zhao et al., 2022] employs the Expectation-Maximization (EM) framework, where one encoder iteratively generates pseudo-labels for the other, allowing the alignment between two representation spaces.

5.2 Asymmetric

Symmetric alignment balances both modalities equally, whereas asymmetric alignment prioritizes one, often using GNNs’ structural processing to bolster LLMs. Current approaches mainly include graph-nested transformers and graph-aware distillation.

The graph-nested transformer, as exemplified by Graphformer [Yang et al., 2021] in Figure 5(c), demonstrates asymmetric alignment through the integration of GNNs into each transformer layer. Within each layer of the LLM, the node embedding is obtained from the first token-level embedding, which corresponds to the [CLS] token. The process involves gathering embeddings from all relevant nodes and applying them to a graph transformer. The output is then concatenated with the input embeddings and passed on to the next layer of the LLM. Patton [Jin et al., 2023b] extends GraphFormer by proposing network-contextualized masked language modeling and masked node prediction and shows strong performance in various downstream tasks, including classification, retrieval, reranking, and link prediction.

GRAD [Mavromatis et al., 2023] employs graph-aware distillation for aligning two modalities, depicted in Figure 5(d). It utilizes a GNN as a teacher model to generate soft labels
for an LLM. Since the LLMs share parameters, the GNN can also benefit from improved textual encodings after the updates to the LLMs’ parameters. Through iterative updates, a graph-aware LLM is developed, resulting in enhanced scalability in inference due to the absence of the GNN. Similar to GnAD, THLM [Zou et al., 2023] employs a heterogeneous GNN to enhance LLMs with multi-order topology learning capabilities. It involves pretraining a GNN alongside an auxiliary GNN through Context Graph Prediction and Masked Language Modeling tasks. After the pretraining process, the auxiliary GNN is discarded and the LLM is fine-tuned for downstream tasks.

5.3 Discussions
To align GNNs and LLMs, symmetric alignments treat each modality equally to both enhance GNNs and LLMs. This enables encoders to efficiently manage multimodal tasks, utilizing their unique strengths to enhance modality-specific representations. Asymmetric methods enhance LLMs by inserting graph encoders into transformers or using GNNs as teachers. However, alignment techniques face severe data scarcity, since only a few graph datasets contain native graph-text pairs, limiting the applicability of these methods.

6 Future Directions
Table 1 summarizes the models that leverage LLMs to assist graph-related tasks according to the proposed taxonomy. Based on the above review and analysis, we believe that there is still much space for further enhancement in this field.

Dealing with non-TAG. Text-attributed graphs (TAGs) have shown improved learning with LLM assistance, yet many real-world graphs lack textual information. For instance, traffic networks (e.g., PeMS03 [Song et al., 2020]) use nodes for sensors, and superpixel graphs (e.g., PascalVOC-SP [Dwivedi et al., 2022]) for superpixels, without text attributes for nodes, making semantic descriptions challenging. While OFA [Liu et al., 2024] proposes using texts to describe nodes and edges, embedding them with LLMs isn’t always effective across all domains. Research into utilizing LLMs for graph foundation models without rich text data remains promising.

Dealing with data leakage. Data leakage in LLMs has become a focal point of discussion. Given that LLMs undergo pre-training on extensive text corpora, it’s likely that LLMs may have seen and memorized at least part of the test data of the common benchmark datasets, especially for citation networks. [Chen et al., 2023] proves that specific prompts could potentially enhance the “activation” of LLMs’ corresponding memory, thereby influencing the evaluation. Hence, it’s crucial to reconsider the methods employed to accurately evaluate the performance of LLMs on graph-related tasks. A fair, systematic, and comprehensive benchmark is also needed.

Improving transferability. Transferability in graphs is tough because each graph has its unique features and structure. Differences in size, nodes, edges, and topology make it hard to apply what’s learned from one graph to another. While LLMs have demonstrated promising zero/few-shot abilities in language tasks due to their extensive pre-training on vast amounts of corpora, their knowledge for graph tasks isn’t well-explored. OFA [Liu et al., 2024] tries to make this easier by describing graphs in a unified way for better cross-domain transferability. However, improving the transferability in graphs is still an open area for more research.

Improving explainability. Explainability, also known as interpretability, denotes the ability to explain or present the behavior of models in human-understandable terms. LLMs exhibit improved explainability compared to GNNs when handling graph-related tasks, primarily due to the reasoning and explaining ability of LLMs to produce user-friendly explanations for graph reasoning. Several studies have examined explaining techniques within the prompting paradigm, such as in-context learning [Radford et al., 2021] and chain-of-thought [Wei et al., 2022b]. Further explorations should be conducted to enhance explainability.

Improving efficiency. While LLMs have demonstrated their effectiveness in learning on graphs, they may face inefficiencies in terms of time and space, particularly compared to dedicated graph learning models such as GNNs that inherently process graph structures. This is especially obvious when LLMs rely on sequential graph descriptions for predictions discussed in Section 4. Existing studies have tried to enable LLMs’ efficient adaption via adopting parameter-efficient fine-tuning strategies, such as LoRA [Hu et al., 2021] and prefix tuning [Li and Liang, 2021]. We believe that more efficient methods may unlock more power of applying LLMs on graph-related tasks with limited computational resources.

Analysis and improvement of expressive ability. Despite the recent achievements of LLMs in graph-related tasks, their theoretical expressive power remains largely unexplored. It is widely acknowledged that standard message-passing neural networks are as expressive as the 1-Weisfeiler-Lehman (WL) test, meaning that they fail to distinguish non-isomorphic graphs under 1-hop aggregation [Xu et al., 2018]. Therefore, two fundamental questions arise: How effectively do LLMs understand graph structures? Can their expressive ability surpass those of GNNs or the WL-test?

7 Conclusion
The application of LLMs to graph-related tasks has emerged as a prominent area of research in recent years. In this survey, we aim to provide an in-depth overview of existing strategies for adapting LLMs to graphs. Firstly, we introduce a novel taxonomy that categorizes techniques involving both graph and text modalities into three categories based on the different roles played by LLMs, i.e., enhancer, predictor, and alignment component. Secondly, we systematically review the representative studies according to the taxonomy. Finally, we discuss some limitations and highlight several future research directions. Through this comprehensive review, we aspire to shed light on the advancements and challenges in the field of graph learning with LLMs, thereby encouraging further enhancements in this domain.

Acknowledgements
This work was supported by NSFC Grant No. 62206067, HKUST-HKUST(GZ) 20 for 20 Cross campus Collaborative...
Research Scheme C019, Guangzhou-HKUST(GZ) Joint Funding Scheme 2023A0330673, project MMT-p2-23 of the Shun Hing Institute of Advanced Engineering, the Research Grant Council of the Hong Kong Special Administrative Region, China (No. CUHK 14217622).

Contribution Statement
Yuhan Li, Zhixun Li, and Peisong Wang contribute equally to this work. Jia Li is the corresponding author (jiallee@ust.hk).

References


