Are Graphs Useful for LLMs? A Comprehensive Survey of Graph-Enhanced Large Language Models

XINYAN ZHU*, Beijing University of Posts and Telecommunications, China
CHENG YANG*, Beijing University of Posts and Telecommunications, China
QIUYU WANG, Beijing University of Posts and Telecommunications, China
ZEYUAN GUO, Beijing University of Posts and Telecommunications, China
ZEDI LIU, Beijing University of Posts and Telecommunications, China
YIDING WANG, Beijing University of Posts and Telecommunications, China
JIAWEI LIU, The Chinese University of Hong Kong, China
CHUNCHEN WANG, Beijing University of Posts and Telecommunications, China
MUHAN ZHANG, Peking University, China
CHUAN SHI, Beijing University of Posts and Telecommunications, China

Large language models (LLMs) have demonstrated strong capabilities across diverse tasks such as language understanding, reasoning, planning, and code generation. However, their sequence-based architectures limit their ability to capture complex relational structures, long-range dependencies, and multi-hop interactions. Graphs, which explicitly model entities and relationships, provide a natural complement to LLMs by enabling structured representation, multi-hop reasoning, and improved knowledge grounding. This synergy has led to a growing body of research on graph-enhanced LLMs, which we refer to as **Graph4LLM**. In this survey, we present a comprehensive, pipeline-oriented review of Graph4LLM methods, categorizing them into three stages of the LLM pipeline: (1) the *input phase*, where graphs structure prompts and incorporate external knowledge; (2) the *model phase*, where graphs guide word-level representations and agent-level coordination; and (3) the *output phase*, where graphs support structured reasoning, planning, and verification. For each phase, we provide a detailed review of the key methods and techniques. We further present a broad range of application scenarios, organizing them into *general* and *domain-specific* applications, and highlight how Graph4LLM methods demonstrate strong potential across diverse tasks and real-world settings. Finally, we outline the challenges and future research directions for developing more efficient and interpretable solutions. Resources for Graph4LLM are available at https://github.com/BUPT-GAMMA/Awesome-Graph4LLM.

CCS Concepts: • **Computing methodologies** → **Artificial intelligence**; **Natural language processing**; **Machine learning**; • **Information systems** → *Data mining*.

Additional Key Words and Phrases: Large Language Models, Graph Machine Learning

1 Introduction

Large language models (LLMs) [18, 31, 144] are foundation models with billions of parameters, typically built on the Transformer architecture [145] and pretrained on massive corpora. Under this paradigm, LLMs show impressive capabilities in natural language understanding, generation, and reasoning, enabling them to generalize across diverse tasks with minimal task-specific adaptation.

In practice, LLMs operate within a pipeline that organizes information flow from input to output. (1) In the input phase, task specifications and external knowledge are introduced. This is done using techniques like few-shot prompting [18], retrieval-augmented generation (RAG) [122], or by feeding the model curated training data. These methods

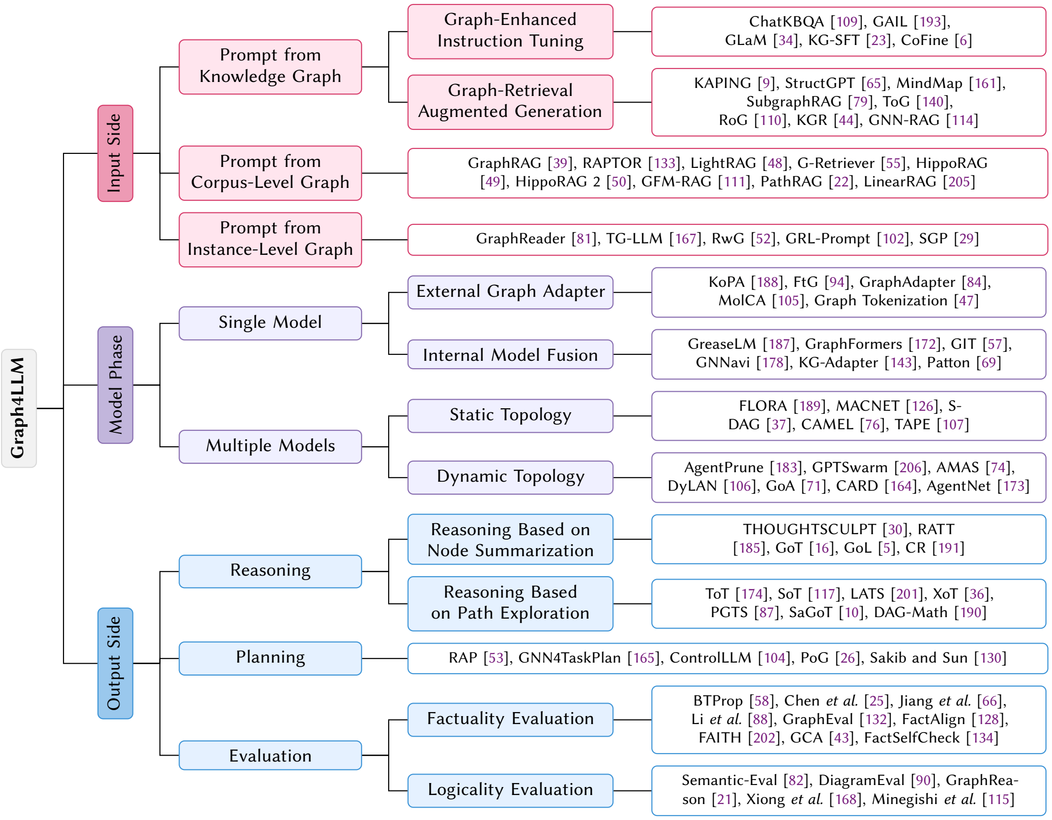


Fig. 1. A taxonomy of **Graph4LLM** with representative examples.

help shape how the raw information is presented to LLMs. (2) Next, the model focuses on processing these inputs. Transformer-based architectures use attention mechanisms and feedforward layers to process information sequentially. They can also be extended with multi-agent systems [54, 85] to coordinate multiple models through interaction. (3) Finally, in the output phase, LLMs generate task-specific responses. These responses can include question answering, executable code, or decision support, serving as the interface between model predictions and downstream applications.

Despite their impressive performance, LLMs have inherent limitations, primarily due to their reliance on linear token sequences. Such sequential models struggle to capture complex relational structures, long-range dependencies, and multi-hop interactions, which are crucial for many knowledge-intensive tasks [57]. Moreover, reasoning and planning processes are often encoded implicitly in latent representations. This makes intermediate states difficult to interpret, control, or verify systematically [174]. LLMs are also vulnerable to factual inconsistencies and hallucinations, especially when tasks require precise relational reasoning or reliable knowledge grounding. These challenges highlight the limitations of sequence-centric models in tasks that require explicit structure, transparency, and robustness [44].

---

[*] Both authors contributed equally to this research.

Authors' Contact Information: Xinyan Zhu, Beijing University of Posts and Telecommunications, China, zhuxinyan@bupt.edu.cn; Cheng Yang, Beijing University of Posts and Telecommunications, China, yangcheng@bupt.edu.cn; Qiuyu Wang, Beijing University of Posts and Telecommunications, China, autumn@bupt.edu.cn; Zeyuan Guo, Beijing University of Posts and Telecommunications, China, guozeyuan@bupt.edu.cn; Zedi Liu, Beijing

To address these limitations of LLMs, graphs provide complementary solutions. Graphs are non-Euclidean structures composed of nodes and edges that capture complex relationships. By explicitly encoding these relationships, they support multi-hop reasoning and represent non-linear structures effectively. Additionally, graphs offer transparent, structured intermediate states that improve interpretability and verification. By integrating external knowledge graphs (KGs) [56], they also enhance factual grounding, reducing hallucinations and improving reliability. According to the 2025 Gartner Data & Analytics Summit[1], graph technologies are projected to drive 80% of data and analytics innovations by 2026, acting as a foundational enabler to provide structured contextual knowledge for LLMs and systematically enable data to become AI-ready.

Building on the potential of graphs to address LLM limitations, graph-enhanced LLMs, which we refer to as **Graph4LLM**, leverage relational structures to handle complex, interconnected data more effectively. As a result, a growing body of work has emerged exploring diverse Graph4LLM paradigms and architectures. Despite this rapid development, research on Graph4LLM remains fragmented across communities and applications, making it necessary to provide a comprehensive categorization.

To better understand this gap, it is useful to distinguish two main research directions at the intersection of graphs and LLMs. In the LLM4Graph paradigm, LLMs are applied to enhance graph learning tasks, bringing strong semantic understanding and generative capabilities to traditional graph mining. Correspondingly, the bulk of existing surveys [68, 85, 91, 92, 129, 136, 153, 176] focus on this forward direction, systematically reviewing frameworks, methods, and applications of LLMs for graph tasks. In contrast, investigations into the reverse paradigm of graphs enhancing LLMs have received far less attention in existing surveys. Even the few surveys that cover Graph4LLM tend to focus on specific subareas rather than providing a unified view: some delve exclusively into KG-augmented LLMs [3, 20, 63, 120, 169], some concentrate on graph-based retrieval-augmented generation frameworks [112, 122, 125, 204], and others deploy graphs as structural scaffolds for multi-agent coordination and long-term memory. This fragmented coverage leaves a gap in understanding how graph structures can broadly address the inherent limitations of LLMs.

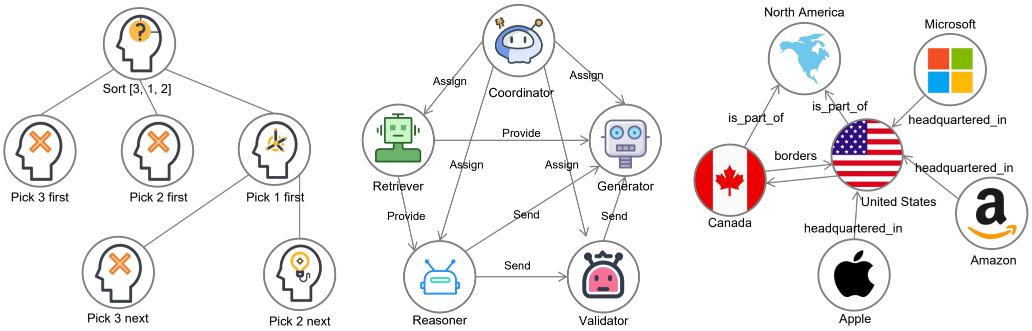
In contrast, this paper significantly expands our previous study [203] to provide the first comprehensive, pipeline-oriented survey of **Graph4LLM**, introducing a more refined taxonomy and coverage of the latest papers. Specifically, we categorize existing works according to the three phases of the LLM pipeline (Figure 1): (1) In the *input phase*, graphs are used to transform complex and scattered information into structured prompts, enabling LLMs to clearly capture key entities and relations. (2) In the *model phase*, graphs shape the internal processing of a single model or organize interactions across multiple models, enabling controlled information flow and task coordination. (3) In the *output phase*, graphs reorganize LLM responses into structured representations, making intermediate steps and dependencies easy to inspect and verify. Based on this taxonomy, we further organize methods within each phase and analyze their key design choices and trade-offs.

The remainder of this paper is organized as follows. Section 2 presents the preliminaries related to Graph4LLM, including the relevant technologies of graphs and LLMs, as well as an overview of Graph4LLM. Section 3 reviews multi-granularity prompt construction and knowledge incorporation techniques in the input phase. Section 4 examines single- and multi-model graph-enhanced systems in the model phase. Section 5 focuses on graph-structured reasoning,

---

[1] https://www.gartner.com/en/newsroom/press-releases/2025-03-05-gartner-data-and-analytics-summit-2025-orlando-day-3-highlights

University of Posts and Telecommunications, China, liuzedi@bupt.edu.cn; Yiding Wang, Beijing University of Posts and Telecommunications, China, wangyiding@bupt.edu.cn; Jiawei Liu, The Chinese University of Hong Kong, China, liujw@cse.cuhk.edu.hk; Chunchen Wang, Beijing University of Posts and Telecommunications, China, wangchunchen@bupt.edu.cn; Muhan Zhang, Peking University, China, muhan@pku.edu.cn; Chuan Shi, Beijing University of Posts and Telecommunications, China, shichuan@bupt.edu.cn.



(a) Tree (b) Directed Acyclic Graph (c) Knowledge Graph

Fig. 2. Examples of various types of graphs used in Graph4LLM. (a) depicts a tree structure that hierarchically arranges the sequential reasoning trajectories of LLMs for sorting tasks. (b) presents a directed acyclic graph to simulate the collaborative workflow of multi-agent systems. (c) shows a knowledge graph that captures structured factual relations involving the United States.

planning, and evaluation techniques in the output phase. Section 6 surveys representative Graph4LLM applications, and Section 7 discusses open challenges and future research directions. Overall, this organization follows the LLM pipeline and enables a comprehensive review of Graph4LLM methods.

2 Preliminary

2.1 Graphs

Graphs effectively model complex interactions in real-world systems, including social networks [155], transportation networks [15], and biomedical networks [13]. Formally, a graph $G = (V, E)$ consists of nodes V and edges $E \subseteq V \times V$ encoding pairwise relations, which may be directed or undirected with associated attributes. Unlike sequential or grid data, graphs naturally capture non-Euclidean relational structures. Figure 2 illustrates the common graph structures used in Graph4LLM, including tree structures where reasoning thoughts are organized as nodes, directed acyclic graphs (DAG) that model multi-agent collaboration, and KGs that serve as external knowledge bases.

Graph formalization. Practically, graphs are represented via adjacency matrices, adjacency lists, or edge lists. For $N = |V|$ nodes, the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ encodes connectivity ($\mathbf{A}_{ij}$ indicates the edge from i to j), while node features are stored in $\mathbf{X} \in \mathbb{R}^{N\times d}$. These representations underpin most graph learning algorithms.

Knowledge graphs. Knowledge graphs (KGs) [56] represent factual knowledge as structured triples (h, r, t), where h, t are entities and r denotes the relation between them. By capturing semantic relationships, KGs are pivotal for reasoning and question answering. Common learning techniques include embedding methods that map entities and relations into continuous vector spaces while preserving relational constraints.

Graph neural networks. Graph neural networks (GNNs) [135] are a family of neural models designed to learn representations from graph-structured data. The core idea of GNNs is message passing, where each node iteratively aggregates information from its neighbors to update its representation [171]. A generic message-passing layer can be written as:

$$\mathbf{h}_v^{(k)} = \phi^{(k)}\left(\mathbf{h}_v^{(k-1)}, \bigoplus_{u\in\mathcal{N}(v)} \psi^{(k)}\left(\mathbf{h}_u^{(k-1)}, \mathbf{e}_{uv}\right)\right). \tag{1}$$

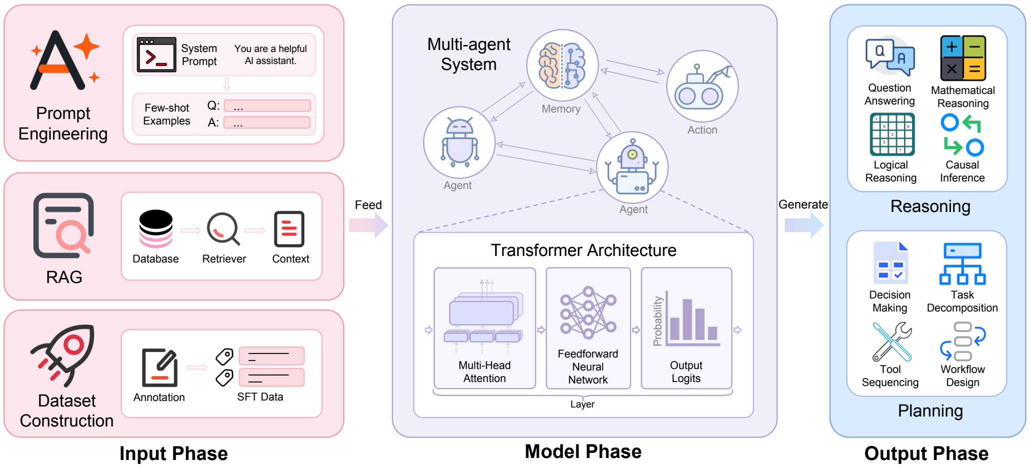


Fig. 3. The overall pipeline of LLM (input phase, model phase, and output phase).

Here, $\mathbf{h}_v^{(k)}$ is the representation of node v at layer k, and $\mathcal{N}(v)$ denotes its neighbors. $\mathbf{e}_{uv}$ represents edge attributes. $\psi^{(k)}$ computes messages from neighbors, $\phi^{(k)}$ updates the node by aggregating messages and its previous state, and $\bigoplus$ is a permutation-invariant aggregation (e.g., sum, mean, or max). Representative GNN architectures include GCNs [73], GATs [146], and GraphSAGE [51].

2.2 Large Language Models

Large language models (LLMs) [18, 31, 144] are a class of neural language models trained on massive text corpora to learn general-purpose linguistic and reasoning capabilities. Typically based on the Transformer [145] architecture, LLMs model token sequences via conditional probability and are trained with autoregressive objectives. Through large-scale pretraining, LLMs exhibit strong abilities in natural language understanding and generation. These capabilities are realized through various technologies at each phase of the LLM pipeline, as illustrated in Figure 3.

Input phase. The input phase acts as a bridge between raw task requirements, external knowledge, and the LLM's sequential processing paradigm. Key techniques include prompt engineering [18], which designs prompts to guide in-context learning and align its behavior with specific task goals. Retrieval-augmented generation (RAG) [75] is another key technique. It retrieves task-relevant knowledge from external databases and incorporates it into the input prompt, reducing reliance on parametric memory and mitigating factual hallucinations. Additionally, instruction tuning [157] and supervised fine-tuning (SFT) [118] with curated datasets help LLMs capture the semantic and structural features of input information across downstream tasks.

Model phase. The underlying technologies in the model phase are mainly based on the Transformer architecture and its extended optimizations, as well as the collaborative mechanisms of multiple models. For an individual LLM, the focus lies on enhancing its parametric memory and structural adaptability. Techniques such as parameter-efficient fine-tuning (PEFT) techniques [59, 60] can be viewed as lightweight modules that adapt internal representations without changing the weights of core models. For multiple LLM collaboration, LLM-based multi-agent systems (MAS) [54, 85] have emerged as a key extension. They coordinate interactions between multiple LLMs through structured communication and enable task decomposition by defining agent roles and interaction rules.

Output phase. The output phase of LLMs focuses on generating coherent and task-relevant text from latent representations. Beyond simple next-token prediction, various techniques guide the generation process to improve reasoning quality. Reasoning methods [62], such as Chain-of-Thought (CoT) [158], encourage the model to generate explicit intermediate steps, improving performance on tasks requiring multi-step inference or logical deduction. These capabilities enable LLMs to handle complex tasks including multi-hop question answering, mathematical/logical reasoning, and causal inference. Building on this, planning techniques further extend generation from reasoning traces to structured action sequences. LLM planning methods [156] decompose goals into sequences of interdependent sub-tasks, allowing models to generate coherent plans rather than isolated responses. By decomposing complex tasks into manageable steps, planning supports tool use, workflow design, and strategic decision-making, leading to more accurate and executable results.

2.3 Graph4LLM

Graph4LLM refers to a class of methods that explicitly integrate graph structures into LLM–centric systems in order to enhance reasoning, knowledge utilization, and reliability. In this survey, we use *Graph4LLM* as an umbrella term to denote methods where graphs are not merely auxiliary data formats, but play a functional role in the overall system design and inference process.

At a high level, a Graph4LLM system consists of three core components: graph construction, graph–LLM interaction, and task execution. First, graph construction defines how structured representations are built. These graphs may encode entities and relations, intermediate reasoning states, agent interactions, or task dependencies. Second, graph–LLM interaction specifies how the graph is incorporated into the LLM workflow, including how graph information is queried, updated, or used to guide generation. Third, task execution describes how the combined system generates final outputs, potentially involving iterative reasoning, verification, or multi-step decision making.

3 Input Phase

The input phase of LLMs involves processing raw text, which is typically fed in a sequential manner. Graph4LLM input-phase methods extract knowledge from graphs or use them to index and organize text content. In this section, we categorize these methods based on the source of the graph structure (as shown in Figure 4): **Prompt from Knowledge Graph**, which utilizes pre-existing KGs; **Prompt from Corpus-Level Graph**, which constructs a global graph index from the text corpus; and **Prompt from Instance-Level Graph**, which induces ad-hoc structures based on specific input instances to guide generation.

3.1 Prompt from Knowledge Graph

Prompt from Knowledge Graph methods integrate existing KGs into LLM prompting pipelines to provide structured and reliable knowledge support. These methods rely on task-agnostic KGs and typically fall into two categories: Graph-Enhanced Instruction Tuning, which collects training data from KG facts and relations, and Graph-Retrieval Augmented Generation (Graph-RAG), which treats KGs as external knowledge databases for structured retrieval.

3.1.1 Graph-Enhanced Instruction Tuning. **Graph-Enhanced Instruction Tuning** modifies LLM parameters to better align internal representations with KG structures. This is achieved by constructing structure-aware instruction tuning data. Such data introduces explicit relational and logical information into training, enabling the model to generalize to more complex scenarios. Existing methods differ mainly in how training data are derived and formatted from KGs.

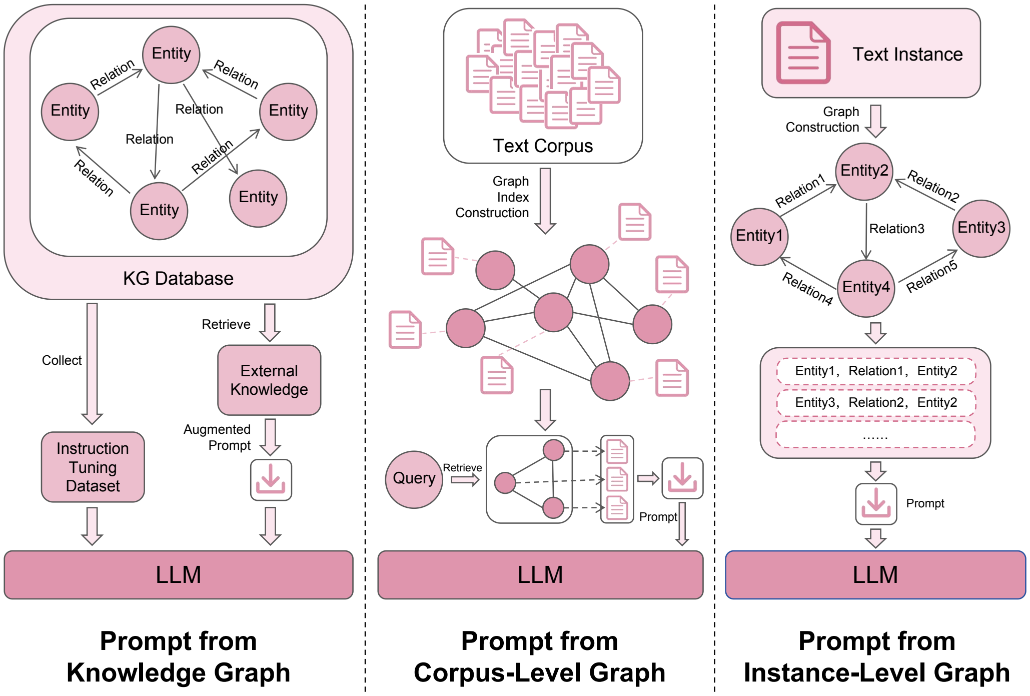


Fig. 4. Different frameworks of Graph4LLM in the input phase.

A line of work derives supervision signals from formal query languages (e.g., SPARQL), where KG reasoning processes are explicitly expressed as executable queries. To address low-resource knowledge graph question answering (KGQA), GAIL [193] employs Generative Adversarial Imitation Learning to fine-tune LLMs on synthetic data. This data is generated by translating sampled SPARQL queries into natural language to approximate real query distributions. ChatKBQA [109] reduces the syntactic complexity of graph query languages by fine-tuning LLMs to translate natural language questions into intermediate logical forms, rather than directly generating SPARQL queries. This semantic parsing paradigm effectively reduces syntax errors and improves query execution.

Another line of work avoids explicit query languages and instead constructs training data by directly traversing KG structures, such as subgraphs, neighborhoods, or paths, to capture relational patterns. GLaM [34] initiates this by transforming domain-specific KGs into text-based Q&A pairs. Graph neighborhoods are linearized into natural language to construct instruction tuning data. KG-SFT [23] uses subgraph extraction to identify relevant reasoning structures and generate explanations for each Q&A pair. It incorporates a knowledge conflict detector to filter unreliable paths, thus improving the model's ability to handle complex relational knowledge during reasoning. Expanding from local subgraphs to global topology, CoFine [6] leverages KG community structures for fine-tuning. It partitions the graph into semantically homogeneous clusters using algorithms like Louvain [17]. Then, it utilizes random walks to capture relational chains within each community, converting these paths into chat-based interactions as training data that expose deep thematic patterns to the LLMs.

3.1.2 Graph-Retrieval Augmented Generation. Building on the same goal of leveraging KG structure, **Graph-Retrieval Augmented Generation** differs by keeping LLM parameters unchanged. It retrieves task-relevant subgraphs and leverages graph topology to support evidence-based reasoning in real time. Existing methods mainly differ in their retrieval and reasoning strategies to construct effective prompts.

One line of work treats the KG purely as a structured evidence source, focusing on retrieving relevant facts and feeding them to the LLMs while leaving all reasoning to the model itself. KAPING [9], for instance, retrieves relevant facts from KGs for the input question, prepends these facts to the question as a prompt, and feeds them to off-the-shelf LLMs for zero-shot KGQA. For unified reasoning over diverse structured data, StructGPT [65] designs specialized interfaces for KGs, tables and databases to extract evidence. It iterates an invoke-linearization-generation procedure, letting LLMs focus on reasoning to reach the answer. MindMap [161] retrieves multi-hop and neighborhood evidence from KGs, converts these subgraphs into natural language "mind maps", and feeds them as prompts to guide inference. To address the trade-off between retrieval efficiency and effectiveness, SubgraphRAG [79] employs a lightweight multi-layer perceptron (MLP) with parallel triple-scoring to flexibly retrieve structure-aware subgraphs for zero-shot LLM reasoning.

In contrast, another line of work treats KGs not only as evidence but also as active guides that steer or constrain the reasoning process of LLMs. ToG [140] tightly couples LLMs with KGs by allowing the model to dynamically explore reasoning paths. It uses beam search over KG triples, enabling the LLMs to evaluate and prune paths in real time for optimal reasoning. RoG [110] frames reasoning as KG-grounded relation path planning: it first prompts the LLMs to generate relation-only templates, then retrieves matching entity paths from the KGs to ensure interpretable inference. To further enhance reliability, KGR [44] introduces iterative KG-based verification, treating KGs as an external fact checker that corrects LLM outputs across multiple rounds to reduce hallucinations. GNN-RAG [114] utilizes GNNs to process KG subgraphs, assigning relevance weights to nodes and extracting shortest paths to potential answer candidates. These paths are then verbalized as context for LLM RAG reasoning, with the system offering optional retrieval augmentation and routing.

3.1.3 Discussion. Prompt from Knowledge Graph methods provide LLMs with explicit factual and relational knowledge from curated sources. This helps the model ground its reasoning on reliable facts, improves performance on knowledge-intensive tasks, and makes the reasoning process more interpretable by exposing entity–relation structures.

However, these methods face challenges that are closely tied to the nature of KGs. KGs are often incomplete, inconsistent, or outdated, which can directly mislead the model. Aligning KG structures with natural language inputs is also non-trivial, as symbolic relations do not always match how information is expressed in text. In addition, retrieving useful subgraphs requires understanding both the query intent and the graph structure, which is difficult when relations are complex or sparsely connected.

Future work should focus on improving the alignment between structured knowledge and natural language understanding. Developing methods that can automatically detect and resolve inconsistencies in KGs will be important for reliability. More intelligent retrieval mechanisms that consider both semantic meaning and graph structure could help select truly relevant knowledge. Another direction is to enable continuous updating of KGs so that models can stay consistent with evolving real-world information.

3.2 Prompt from Corpus-Level Graph

In contrast to Prompt from Knowledge Graph, Prompt from Corpus-Level Graph constructs corpus-specific graph indices over unstructured text collections rather than relying on existing KGs. In this setting, nodes correspond to documents, entities, or concepts extracted from the corpus, and edges encode semantic or structural relations. Such graph indices enable LLMs to efficiently locate the associated textual content.

Traditional RAG methods often struggle with queries that require global understanding of the corpus. To address this limitation, recent work explores hierarchical structures for global information aggregation. GraphRAG [39] targets query-specific summarization over massive private corpora by constructing an entity KG from source documents. Crucially, the graph is partitioned into groups of closely related entities (communities) via community detection. Hierarchical summaries are then generated through a two-stage process: the system first prompts the LLMs to generate partial responses for each community, and subsequently synthesizes these into a cohesive global answer. RAPTOR [133], on the other hand, constructs a bottom-up hierarchical tree by recursively embedding, clustering, and summarizing text chunks. During inference, it retrieves directly from multiple abstraction levels within this tree, enabling the seamless integration of global themes and local details for complex reasoning. Despite these advances, hierarchical aggregation may still introduce retrieval redundancy, increase token usage (longer reasoning chains), and reduce control over intermediate summarization errors.

Recent work has shifted the focus from merely organizing information to optimizing the specific pathways and efficiency of the retrieval process. These methods seek to progressively exploit and reuse graph structures to improve generation efficiency while substantially reducing retrieval noise. LightRAG [48] employs graph-enhanced indexing and a dual-level retrieval paradigm to capture complex entity dependencies, enabling simultaneous low-level local and high-level global queries. Its incremental update algorithm ensures seamless data integration without costly full-graph reconstruction. Similarly, G-Retriever [55] formulates retrieval as a Prize-Collecting Steiner Tree optimization. It extracts compact subgraphs to reduce context length and mitigate hallucinations. Drawing inspiration from mammalian memory, HippoRAG [49] and its successor HippoRAG 2 [50] employ a neuro-symbolic approach that combines LLM extraction with Personalized PageRank, propagating relevance to diffuse relevance signals across KGs. To overcome domain-specific constraints, GFM-RAG [111] trains a general graph foundation model (GFM) [98] through a two-stage process on massive datasets comprising 60 KGs and 700k documents to learn transferable topological representations. PathRAG [22] directly targets retrieval redundancy by using flow-based pruning to extract critical relational paths, converting them into structured prompts that enhance logical coherence.

3.2.1 *Discussion.*

Prompt from Corpus-Level Graph methods organize large text collections into structured graphs, allowing LLMs to navigate information beyond linear context windows. By capturing both local connections and global structure, these methods help the model retrieve more coherent and relevant information for complex queries.

A key challenge in this setting lies in the construction and abstraction of graphs from raw text. Hierarchical structures often introduce redundant information across different levels, which increases token usage and slows down inference. Errors from intermediate steps such as clustering and summarization can also propagate and affect final outputs. At the same time, methods that aggressively optimize retrieval may simplify the graph too much, for example, through excessive pruning or compression, causing important context to be missed. Achieving a good balance between global coverage and local precision is difficult in practice.

Future work can focus on improving how corpus information is structured rather than simply making construction more efficient. Better methods are needed to preserve important details during abstraction while still enabling scalable retrieval. Designing adaptive graph structures that can reorganize based on query intent may also improve performance. Another promising direction is to jointly optimize graph construction and retrieval, so that the structure directly supports downstream usage instead of being built independently.

3.3 Prompt from Instance-Level Graph

Unlike the previous two paradigms, Prompt from Instance-Level Graph does not rely on pre-built KGs or corpus-level indices. Instead, it constructs graphs on the fly for each input instance, transforming a logically complex query into a structured representation. This graph is then linearized into semi-structured text, preserving structure while staying compatible with standard LLM inputs.

Basically, graph structures are used to improve how models select and organize in-context examples. For example, GRL-Prompt [102] constructs a task-specific KG to encode correlations between the query and candidate demonstrations. A policy network, trained via reinforcement learning with reward shaping, selects and orders in-context examples to construct optimal prompts. By leveraging this instance-driven, structured approach, the framework can identify informative examples and arrange them effectively.

Beyond example selection, graphs are further used to expose and guide the reasoning process. Instead of treating reasoning as implicit token generation, these methods explicitly organize intermediate logic. To address chronological confusion in narrative understanding, TG-LLM [167] converts textual narratives into temporal graphs to enhance the learning of temporal dependencies. This structured representation allows the model to reason over diverse temporal concepts through strategies such as CoT bootstrapping and graph data augmentation. Similarly, RwG [52] constructs explicit graphs directly from the context to bridge information gaps in logical reasoning tasks. By structuring implicit knowledge into pairwise entity relationships, the framework enhances the ability of LLMs to understand multi-hop dependencies and implicit connections.

At a larger scale, graph structures become a way to organize the entire reasoning workflow. Rather than optimizing individual prompts or reasoning steps, these approaches treat graphs as a global control mechanism. SGP [29] provides a task-agnostic three-stage prompting framework that directs models to convert unstructured text into graph representations, plan navigation strategies, and execute reasoning by traversing the graph step by step. This decomposition enables more controllable multi-step reasoning across diverse natural language scenarios. For extreme long-context challenges, GraphReader [81] structures extensive documents into navigable graphs to facilitate processing. Instead of feeding the entire document into the model, the framework deploys an autonomous agent that iteratively explores the graph in a query-guided manner. This mechanism allows the model to handle massive contexts up to 256k tokens while operating within a limited 4k context window, demonstrating superior performance on various long-text benchmarks.

3.3.1 Discussion. Prompt from Instance-Level Graph methods dynamically construct structured representations for each input, organizing complex relationships such as temporal, logical, or dependency connections. This helps the model better understand how different pieces of information are related, especially in long or interdependent inputs, leading to more informed decisions.

The main challenge here lies not only in efficiency but also in reasoning stability. Since the graph is built on the fly, small mistakes in early steps, such as missing nodes or incorrect relations, can quickly affect later decisions. This makes the overall process sensitive to noise and difficult to debug. In addition, different tasks may require very different graph structures, and there is no clear standard for how these graphs should be constructed, which limits generalization across tasks.

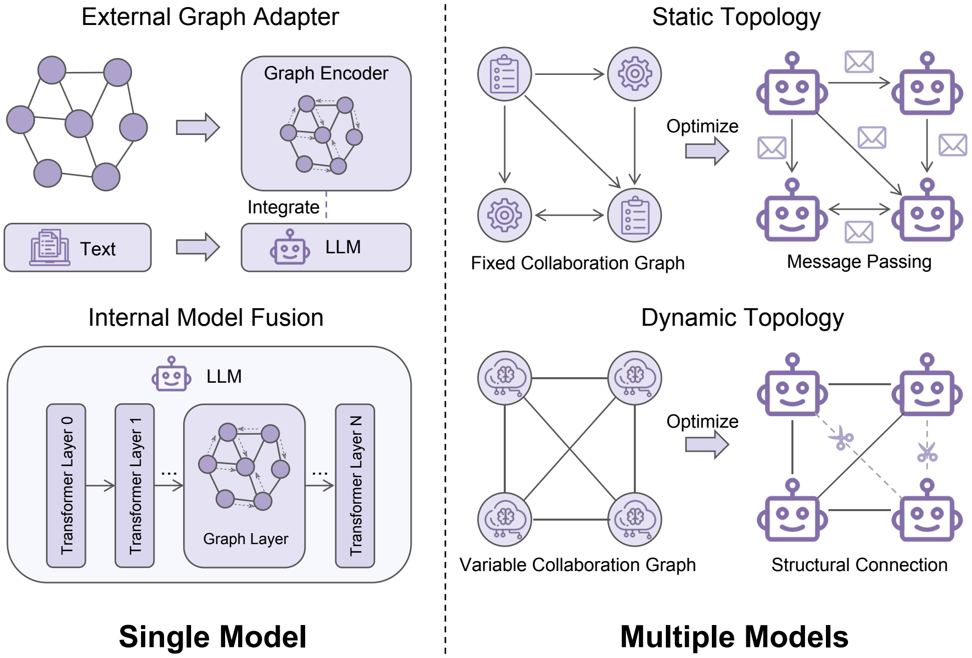


Fig. 5. Different frameworks of Graph4LLM in the model phase.

Future work should focus on making instance-level graph reasoning more stable and adaptive. Methods that can verify or refine intermediate graph structures during reasoning could help reduce error accumulation. It is also important to develop more unified frameworks that can flexibly adjust graph structures without heavy task-specific design. Finally, integrating lightweight feedback mechanisms into the reasoning loop may help the model actively correct itself and improve robustness.

4 Model Phase

The model phase of LLMs concerns both the internal architecture of the model and the way multiple agents collaborate. Graph4LLM model-phase methods introduce graph structures as explicit relational priors that complement the sequence-centric inductive bias of LLMs. Existing methods can be broadly categorized into two paradigms (as shown in Figure 5): **Single Model**, where graph modules are integrated into one LLM backbone with varying depth of fusion, and **Multiple Models**, where graphs specify or learn the communication topology and task dependencies among multiple models/agents.

4.1 Single Model

In the field of graph-enhanced single model, the core goal is to integrate structured graph signals within the LLM framework. This aims to improve the model's ability to process relational and structural information within a unified architecture. Broadly, existing strategies can be grouped into two paradigms: External Graph Adapter, which adds graph encoders before LLMs to fuse graph features, and Internal Model Fusion, which embeds structural interactions deeply into the model's layers for bidirectional influence.

4.1.1 External Graph Adapter. **External Graph Adapter** aims to inject structural signals through auxiliary graph encoding modules, while keeping the LLM backbone parameters (e.g., multi-head attention and feed-forward layers) largely frozen or only lightly fine-tuned. These methods typically employ dedicated graph encoders (e.g., GCNs, GATs) to encode graph topologies, and subsequently map the structured features into the natural language representation space of the LLM through lightweight projection or adaptation mechanisms. Existing works exhibit high diversity in specific feature alignment and projection mechanisms.

A dominant line of work encodes graphs into dense embeddings and injects them into LLMs through lightweight projection or adaptation modules, without explicitly preserving graph topology. KoPA [188] introduces a Knowledge Prefix Adapter that projects KG-derived embeddings into virtual tokens prepended to the prompt, enabling structure-aware reasoning through prefix conditioning. GraphAdapter [84] extends this strategy to text-vision settings by constructing a dual KG over textual and visual structures, and injecting cross-modal structural knowledge via a textual adapter to improve downstream classification. In domain-specific scenarios, MolCA [105] aligns molecular graphs with text via a cross-modal projector (e.g., Q-Former) and employs lightweight low-rank adaptation (LoRA) for model optimization.

In contrast, recent approaches aim to preserve graph topology more explicitly when interfacing with LLMs. To mitigate LLM hallucinations, FtG [94] adopts a filter-then-generate framework, formulating KG construction as a multiple-choice task. It further uses ego-graph serialization and a structure-text adapter to better align topology with text. Graph Tokenization [47] advances the interface to the vocabulary level by introducing reversible graph serialization and graph byte pair encoding (BPE), enabling the learning of structure-aware tokens without breaking subgraph connectivity. This allows standard Transformers to process graph structures similarly to text, without modifying the backbone model.

4.1.2 Internal Model Fusion. In contrast to External Graph Adapter that remains at shallow alignment, **Internal Model Fusion** advocates breaking the architectural boundary between LLMs and GNNs, embedding graph structure interaction mechanisms into the internal computational layers of the LLM. A common design interleaves Transformer layers with GNN-style message passing or introduces cross-stream modules for bidirectional exchange between token and graph layers.

A highly representative fusion strategy involves constructing explicit layer-wise interleaving and bidirectional interaction modules. GreaseLM [187] interleaves pretrained language model (PLM) and GNN layers to enable iterative exchange between token and graph representations, achieving bidirectional grounding between language context and structured knowledge. Similarly, GraphFormers [172] binds Transformer and GNN modules layer by layer, injecting neighborhood aggregation after each layer to progressively enhance structure awareness.

Beyond such layer-wise coupling, another line of work internalizes graph signals by modifying the self-attention mechanism. GIT [57] introduces a graph-induced attention mechanism that encodes graph connectivity and shortest-path distances as attention patterns. This injects structural inductive biases into the Transformer, encouraging the model to prioritize topologically closer entities during attention computation.

Furthermore, some works explore internal graph routing and fusion mechanisms within the model. GNNavi [178] adopts a lightweight yet effective strategy by inserting a small number of GNN navigation layers into a frozen LLM decoder, where prompt-induced temporary graphs act as routing signals to guide message passing and regulate information flow in few-shot scenarios. KG-Adapter [143] extends internal fusion to decoder-only LLMs by introducing a dedicated adapter that encodes both node-centric and relation-centric knowledge, performing structural integration before each attention computation. At the pre-training level, Patton [69] learns structure-aware representations by jointly modeling text and large-scale network links, implicitly encoding topology into the foundation model.

4.1.3 Discussion. Single-model methods directly integrate graph structure into the internal computation process of a model. This allows the model to learn structured representations together with language understanding in one unified system. As a result, the model can better capture relational information and improve reasoning on tasks that depend on structured inputs, without relying heavily on external retrieval or extra prompting.

At the same time, these methods face several practical challenges. External adapters often rely on projecting graph features into the text space, which can weaken structural signals when graph structures become complex. Internal fusion methods introduce additional components such as GNN layers or modified attention mechanisms, increasing model complexity and training cost. This makes them harder to scale and deploy. Compatibility is another concern, since different graph types and tasks may require different designs, limiting generalization. When graph and language representations are tightly coupled, it also becomes more difficult to identify and analyze errors.

Future work can focus on designing more efficient ways to integrate graph structure without adding significant computational overhead. Improving how structural information is preserved during alignment will help maintain important graph signals. More flexible model designs are needed so that a single framework can handle different graph types and tasks. It is also promising to explore hybrid strategies that combine lightweight adaptation with selective deeper integration. Finally, better pre-training approaches with diverse graph and text data may help models learn more general and robust structural patterns.

4.2 Multiple Models

Unlike single-model methods, graph-enhanced multiple models leverage graphs to coordinate interactions, communication flows, and task dependencies across multiple models or agents. These methods are categorized into Static and Dynamic Topology. Static Topology relies on predefined graph structures to ensure controllable orchestration and reduce redundant communication. Dynamic Topology, by contrast, adapts the graph at runtime through learning, pruning, or generation mechanisms, providing greater flexibility and responsiveness in multi-agent collaboration.

4.2.1 Static Topology. **Static Topology** uses a fixed collaboration graph to define communication links and artifact flows in multi-agent systems. The graph is predefined based on human priors or task logic (e.g., directed acyclic pipelines, hierarchical structures, or fixed role graphs), focusing on optimizing message passing and role execution. Within this paradigm, existing methods mainly differ in the source of the predefined collaboration graph.

Task-dependency graphs (TDGs) are a core application of static topology. In multi-step reasoning or code generation, tasks often follow a strict sequential order. FLORA [189] abstracts agent workflows into a DAG, using graph message passing to efficiently evaluate and predict multi-agent collaboration. To improve scalability, MACNET [126] organizes hundreds or thousands of LLM agents into a topologically sorted DAG, allowing unidirectional message passing to support asynchronous parallel collaboration and avoid the communication explosion of fully connected architectures. S-DAG [37] extends this to cross-disciplinary problem solving by pre-constructing a subject-based DAG across multiple knowledge bases and statically assigning expert models along with their derivation sequences.

Beyond task-flow-based partitions, static topologies also frequently manifest as predefined social role networks. CAMEL [76] adopts a hard-coded binary role interaction pattern, achieving highly enclosed autonomous exploration mechanisms through a pre-set “AI User” and “AI Assistant” role graph. TAPE [107] utilizes hierarchically structured preset collaboration trees, establishing strict top-down communication links between high-level planners and low-level executors.

4.2.2 Dynamic Topology. Addressing the rigidity of Static Topology in open-domain tasks, **Dynamic Topology** uses a variable collaboration graph that can be learned or generated at runtime. These methods allow agents to dynamically adjust connections based on current inputs, states, or feedback, offering greater flexibility in multi-agent coordination. Existing methods mainly differ in how and when the collaboration structure is adjusted or constructed.

One representative line of work focuses on graph topology sparsification and communication cost optimization. AgentPrune [183] learns to remove redundant edges from dense interaction graphs, showing that sparse topologies can reduce API costs while even improving decision accuracy by filtering noise. GPTSwarm [206] further treats agents and their connections as a differentiable system, jointly optimizing node prompts and edge-level data flow to discover task-specific graph structures during execution.

Another important line of work emphasizes instant graph generation and selection based on input sample features. AMAS [74] is based on the observation that there is no one-size-fits-all optimal communication topology for different input samples within the same dataset. Therefore, it constructs a dynamic graph selector that adaptively picks the most suitable collaboration graph structure from a pre-maintained pool of candidate topologies upon receiving a new input. DyLAN [106] focuses on evolution during the execution process, dynamically evaluating the current contribution scores of each agent across multiple rounds of reasoning, selectively culling marginal nodes and rewiring core nodes in real-time. In the domain of inference-time structure generation, Graph-of-Agents (GoA) [71] breaks the limitations of traditional single-thread dialogues by dynamically instantiating agents and establishing graph connections, achieving massive nonlinear context extension. Recent frameworks like CARD [164] and AgentNet [173] have also systematically explored how to leverage the LLMs' own meta-reasoning capabilities to allocate role nodes and automatically construct interaction networks on the fly during runtime.

4.2.3 Discussion. Multi-model methods use graph structures to coordinate multiple agents, where the graph defines how agents communicate, share information, and split tasks. This turns the system into a collaborative network rather than a single-model reasoning system. Such designs are effective for complex problems because they allow parallel processing, flexible role assignment, and better handling of long or multi-step workflows. They are especially useful when different subtasks require different expertise.

However, this setting also introduces new challenges. Designing an effective collaboration structure is not straightforward, as static graphs may be too rigid for open-ended tasks, while dynamic graphs can be unstable and harder to control. Communication overhead becomes a key issue when many agents are involved, especially if messages are not well filtered. Coordination errors may arise when agents have inconsistent views or produce conflicting outputs. In addition, dynamically adjusting the graph requires extra computation and reliable signals for decision-making, which is still difficult in practice. Evaluating these systems is also challenging, since performance depends not only on individual agents but also on their interactions.

Future work can focus on improving the balance between structure and flexibility, so that systems remain stable while adapting to different tasks. More efficient communication strategies are needed to reduce unnecessary message passing. Better mechanisms for conflict resolution and coordination will help maintain consistency across agents. It is also promising to explore learning-based approaches that can automatically design and refine collaboration graphs over time. Finally, developing clearer evaluation frameworks may provide better insights into how multi-agent interactions contribute to overall performance.

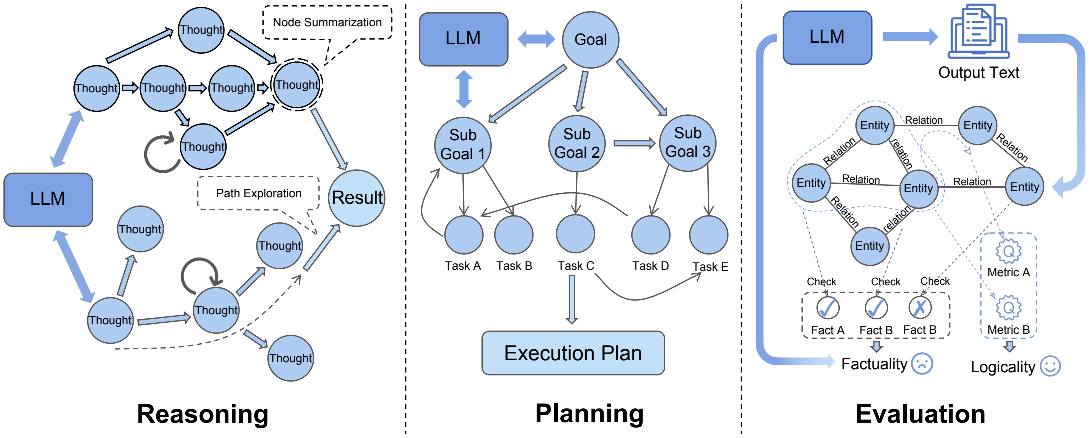


Fig. 6. Different frameworks of Graph4LLM in the output phase.

5 Output Phase

The output phase of LLMs generates text sequences based on the internal representation and the received context. Graph4LLM output-phase methods represent LLM responses as graphs to clarify the thinking process, enabling more structured and verifiable outputs. We categorize existing methods into three directions based on the function of the graphs (as shown in Figure 6): **Reasoning**, which represents and traverses intermediate inference states; **Planning**, which organizes multi-step decision making and action sequences; and **Evaluation**, where graphs serve as structured references for verifying and scoring generated content.

5.1 Reasoning

Graph-based reasoning uses graph structures to organize reasoning thoughts as relationships and dependencies between entities, improving the clarity and interpretability of model outputs. Methods in this category fall into two primary categories: Reasoning Based on Node Summarization, which consolidates information around key nodes to guide reasoning, and Reasoning Based on Path Exploration, which performs reasoning by selecting and traversing paths in a structured space.

5.1.1 Reasoning Based on Node Summarization. **Reasoning Based on Node Summarization** organizes the reasoning process around central nodes that aggregate information from multiple steps. Each node acts as an information hub, condensing partial conclusions. The reasoning process converges to a final node where relevant information is summarized and synthesized into the output. Existing methods differ mainly in how reasoning nodes are structured and updated.

At the structured reasoning level, several methods adopt hierarchical or tree-based organizations to iteratively refine node summaries. THOUGHTSCULPT [30] adopts a search-driven formulation within a Monte Carlo tree search (MCTS) framework. It integrates a thought generator, an evaluator, and a decision simulator, while iteratively revising node-level summaries. RATT [185] similarly leverages tree-structured reasoning, but emphasizes path evaluation and branch selection, using node representations to identify promising reasoning trajectories.

Beyond explicit tree structures, node summarization can be extended to more flexible graph-based or compositional reasoning paradigms. CR [191] organizes intermediate propositions into a dynamically constructed DAG, where nodes represent verified reasoning units. Through a proposer, a verifier, and reporter roles, it incrementally validates and aggregates node-level conclusions, enabling reliable accumulation of evidence toward the final output. GoL [5] further enhances this paradigm by incorporating symbolic logic, with nodes encoding logical statements and edges capturing formal inference relations. By enforcing precise logical rules during updates, it improves faithfulness and supports systematic hallucination detection while remaining scalable. GoT [16] generalizes this paradigm by modeling dependencies among thoughts, organizing reasoning as a graph where nodes represent units of information and edges encode inter-thought relations. This enables flexible thought transformations, including aggregation of multiple reasoning paths, refinement via feedback loops, and distillation of key insights from interconnected thought networks.

5.1.2 Reasoning Based on Path Exploration. By comparison, **Reasoning Based on Path Exploration** frames reasoning as a traversal over multiple candidate paths. Information is not centralized at a single node but distributed along a path. Each node on the path corresponds to an independent action or intermediate decision. The final output is constructed by accumulating information along the selected path. Existing methods mainly differ in how paths are generated and selected during exploration.

A line of existing methods instantiates path exploration over tree-structured spaces, where reasoning proceeds by branching and selecting among alternative trajectories. ToT [174] models reasoning as a tree, exploring branches and selecting promising trajectories. SoT [117] improves efficiency with a skeleton-first strategy, generating concise outlines that are later expanded in parallel. LATS [201] incorporates MCTS to support long-horizon decision making, enabling systematic exploration of combinatorial reasoning–action spaces and unifying reasoning, acting, and planning. XoT [36] augments MCTS with pretraining signals and external knowledge, allowing more informed traversal of reasoning paths. PGTS [87] integrates reinforcement learning with tree search, learning to balance exploration and exploitation via reward optimization instead of relying on hand-crafted heuristics or costly self-evaluation.

In contrast, recent methods move beyond the tree-structure constraint and generalize path exploration to graph-structured spaces, enabling richer dependencies beyond single trajectories. SaGoT [10] constructs a thought graph during decoding via a graph-structured self-attention mechanism, allowing reasoning steps to form flexible interconnections rather than being restricted to a single path, thus implicitly guiding exploration through inter-step dependencies. DAG-Math [190] instead imposes a directed acyclic graph structure over reasoning trajectories, where nodes represent intermediate states and edges encode rule-based transitions, constraining valid paths to follow structured derivations and enabling more faithful and controllable reasoning.

5.1.3 Discussion. Graph-based reasoning introduces structure into the reasoning process, making intermediate steps more organized and easier to track. This structured organization helps models handle multi-step problems more reliably, as information can be accumulated, compared, or refined in a controlled way. It also makes the reasoning process more transparent, which is useful for understanding model behavior and diagnosing errors.

At the same time, several challenges limit its effectiveness. Node-based methods rely heavily on accurate summarization, and early mistakes can affect all later steps. Path-based methods face a large search space, which can lead to high computational cost or inefficient exploration. Controlling the trade-off between exploration and correctness is still difficult, especially when many possible reasoning paths exist. In more flexible graph settings, dependencies between steps can become complex, making it harder to maintain consistency. Another issue is that structured reasoning often requires additional design choices, such as how to define nodes, edges, or update rules, which may not generalize well across tasks.

Future work can explore more efficient ways to manage the reasoning process, such as reducing unnecessary paths or improving the quality of intermediate summaries. Better strategies for guiding exploration may help models focus on more promising reasoning directions. It is also important to design more general frameworks that can adapt to different tasks without heavy manual design. Combining node-based and path-based reasoning in a unified approach may further improve flexibility and performance. Finally, improving how reasoning structures are aligned with model outputs could make results both more accurate and easier to interpret.

5.2 Planning

Building on reasoning capabilities, graph-based planning focuses on task organization rather than performing step-by-step inference. It leverages graphs to organize and manage interdependent goals, tasks and actions, enabling LLMs to handle complex multi-step tasks through plan executions. Existing methods mainly differ in how they use graphs for task decomposition and execution.

A common perspective models planning as a hierarchical structure, starting with methods that emphasize structural decomposition. In these approaches, high-level goals are decomposed into sub-tasks represented as graph nodes, where edges explicitly capture logical and temporal dependencies. Typically, RAP [53] treats graphs as task decomposition tools, with nodes as sub-tasks and edges encoding dependencies. GNNs support sub-task retrieval and execution, while the same LLM serves dual roles: generating potential actions and predicting resulting states to evaluate progress through reward functions. GNN4TaskPlan [165] formulates task planning as a graph-based decision-making problem, using learned representations to guide sub-task selection and execution. Following this line of work, ControlLLM [104] introduces the Thoughts-on-Graph paradigm, which searches for optimal paths on tool graphs modeling parameter and dependency relations, while supporting the decomposition of complex tasks for multimodal execution.

While the aforementioned methods focus on establishing a blueprint, a more adaptive paradigm has emerged that treats graphs as active feedback structures. Rather than relying on static representations, these approaches emphasize the model's ability to revise, backtrack, or expand plans based on real-time environmental feedback. PoG [26] uses graphs as feedback structures during planning. Unlike conventional approaches that rely on fixed exploration breadth or unidirectional reasoning, PoG adopts a self-correcting planning mechanism. It incrementally constructs and updates multi-level sub-goal graphs to detect deviations and enable correction through three components: Guidance, Memory, and Reflection. In robotic task planning, recent work by Sakib and Sun [130] utilizes graphs as execution-oriented interfaces. Multiple GPT-4–generated task trees are consolidated into a unified plan, which is then converted into executable low-level actions through cost-aware selection and symbolic planning.

5.2.1 *Discussion.*

Graph-based planning organizes goals, sub-tasks, and their dependencies into structured plans, helping models manage complex workflows in a more systematic way. This structure supports clear task decomposition, explicit dependency management, and coordinated execution across multiple steps. It is particularly useful for long-horizon tasks, where maintaining a consistent execution order and adapting plans based on feedback are critical.

Several challenges remain in practice. Designing a good task structure is difficult, as an incorrect decomposition can lead to inefficient or even failed execution. Plans that rely on fixed structures may lack flexibility when the environment changes, while adaptive methods introduce additional complexity and may become unstable. Keeping track of dependencies across many sub-tasks can also be challenging, especially when tasks interact in non-trivial ways.

In addition, planning often depends on accurate predictions of future outcomes, and errors in these predictions can lead to poor decisions during execution.

Future work can focus on making planning more robust and adaptive without adding too much computational overhead. Better methods for automatic task decomposition could reduce the need for manual design. Improving how models update plans based on feedback will help them recover from mistakes more effectively. It is also promising to explore tighter connections between planning and execution, so that plans remain aligned with real outcomes. Finally, combining planning with stronger reasoning and memory mechanisms may further improve performance on complex, long-term tasks.

5.3 Evaluation

Unlike reasoning and planning, graph-based evaluation focuses on assessing LLM outputs. It uses graph structures to capture richer associations among entities, facts, and sentences. Specifically, evaluation methods can be categorized into two main types: Factuality Evaluation, which assesses the accuracy and reliability of generated content through explicit entity and relation modeling; Logicality Evaluation, which measures logical coherence and reasoning consistency using graph-based relational representations.

5.3.1 Factuality Evaluation. **Factuality Evaluation** assesses whether LLM outputs are correct by modeling content as structured graphs to verify dependencies, propagate uncertainty, and detect inconsistencies. Existing methods mainly differ in the form of evidence they construct and how it is aligned with model outputs.

Some works focus on leveraging graph structures to model and reason about uncertainty for factuality evaluation. BTProp [58] formulates hallucination detection as probabilistic inference over a belief tree, jointly estimating latent claim truth via a hidden Markov tree conditioned on logical relations and confidence signals. Chen *et al.* [25] model uncertainty propagation using entity- and sentence-level graphs, calibrating uncertainty through semantic dependencies and contradiction probabilities. From a global perspective, Jiang *et al.* [66] estimate claim uncertainty via graph centrality over bipartite generation–claim graphs, showing that higher-order metrics better capture global support than simple self-consistency. Similarly, Li *et al.* [88] construct consistency graphs over sampled outputs and apply GNNs to map uncertainty to correctness probabilities for robust factuality estimation.

Another line of work converts generated text into structured knowledge for fine-grained verification: GraphEval [132] represents responses as KG triples and evaluates them via graph traversal and natural language inference (NLI)-based reasoning, enabling interpretable localization of hallucinated facts. FactAlign [128] formulates hallucination detection as a KG alignment problem, supporting fact-level detection and intrinsic–extrinsic error classification without additional training. Domain-specific extensions such as FAITH [202] show that KG-based evaluation is particularly effective in high-stakes domains like healthcare, where interpretability and expert alignment are essential.

Graph structures are also leveraged for sample-based evaluation. GCA [43] constructs contextual KGs over multiple sampled outputs, modeling fact dependencies via relational graph convolution to improve alignment and consistency in long-form generation. FactSelfCheck [134] similarly represents responses as fact-level graphs and detects hallucinations through cross-sample inconsistency analysis, enabling more fine-grained and effective correction than sentence-level self-consistency.

5.3.2 Logicality Evaluation. At a higher level, **Logicality Evaluation** assesses the quality of reasoning in LLM outputs. It goes beyond factual correctness to examine semantic coherence, inferential structure, and the organization of logical transitions using explicit relational representations. Existing methods mainly differ in the stage and granularity of the structures they target.

Semantic-Eval [82] is a training-free framework for assessing reasoning quality via semantic comprehension rather than surface matching. It constructs semantic graphs from pairwise similarity and uses a graph-based weighting scheme (SemanticRank) to capture sentence-level contributions. A pretrained NLI model refines relations, producing scores better aligned with semantic coherence and human judgments. In a different modality, DiagramEval [90] abstracts LLM-generated diagrams into directed graphs, with elements as nodes and connections as edges. It evaluates logical soundness via node and path alignment, measuring whether relational structures preserve intended reasoning flow and dependencies.

Another line of work targets the explicit verification of reasoning processes. GraphReason [21] aggregates multiple reasoning paths produced by an LLM into a unified reasoning graph, merging shared intermediate steps across solutions. By applying a graph neural verifier over this merged structure, the method evaluates logical consistency across reasoning trajectories and selects solutions that exhibit stronger structural support, leading to improved reasoning accuracy without additional model training.

Beyond direct verification, graph-based representations are also employed to analyze and characterize reasoning behavior. Xiong *et al.* [168] cluster chain-of-thought outputs into semantic steps and construct directed graphs to analyze structural properties, showing that features like branching and convergence correlate with reasoning performance and prompting strategies. At a finer granularity, Minegishi *et al.* [115] propose Circuit-based Reasoning Verification. This method extracts attribution-based computational graphs from hidden states, revealing distinct structural patterns for correct versus incorrect reasoning and linking graph topology to reasoning outcomes.

5.3.3 Discussion. Graph-based evaluation provides a structured way to assess LLM outputs by explicitly modeling facts, relationships, and reasoning processes. This makes evaluation more fine-grained, as it can check not only whether an answer is correct, but also where errors occur and how they propagate. Such structured analysis improves interpretability, supports more reliable hallucination detection, and allows evaluation to move beyond surface-level matching toward deeper semantic and logical assessment.

However, this line of work still faces several challenges. Constructing accurate graphs from generated text is not trivial, and errors in extraction can affect the reliability of evaluation results. Different methods rely on different forms of structure, such as triples, graphs over samples, or reasoning paths, which makes it hard to compare results across approaches. Modeling uncertainty and consistency across multiple outputs can also be computationally expensive. For logicality evaluation, capturing complex reasoning patterns in a clean and consistent structure is difficult, especially when reasoning steps are implicit or loosely expressed. In addition, many methods depend on external components such as NLI models or GNNs, which introduces extra complexity and potential sources of error.

Future work can focus on improving the robustness of graph construction from text, so that evaluation is less sensitive to extraction errors. More unified evaluation frameworks may help compare different methods and provide clearer benchmarks. Reducing computational cost while maintaining fine-grained analysis is another important direction. It is also promising to design evaluation methods that better align with human judgment, especially for complex reasoning tasks. Finally, combining factuality and logicality evaluation into a single framework may provide a more complete view of model performance.

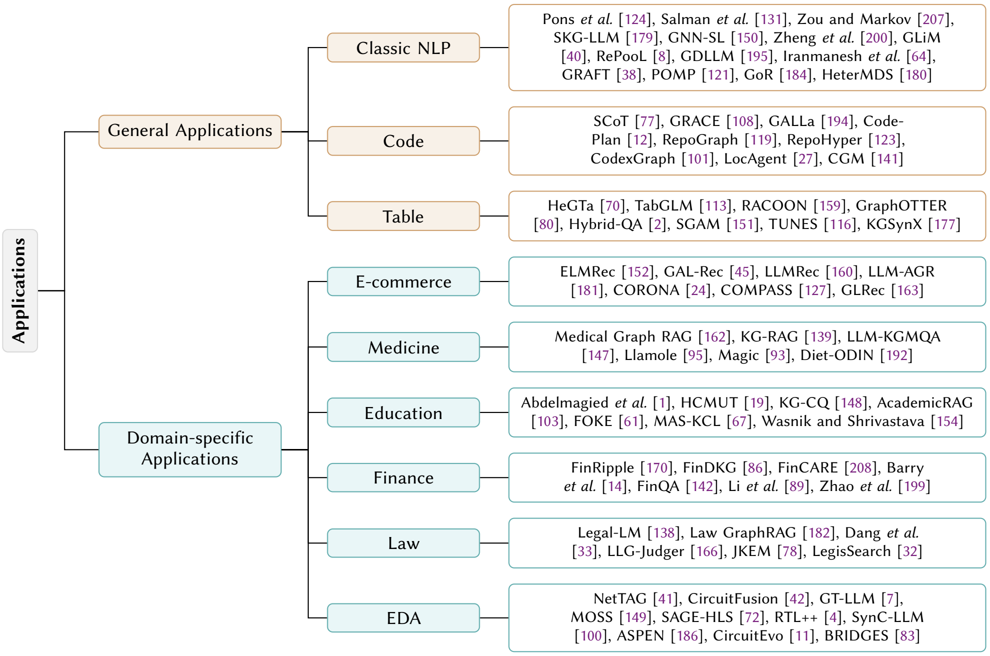


Fig. 7. A taxonomy of **Graph4LLM** Application.

6 Applications

Graph4LLM methods has been widely applied across diverse domains, substantially improving performance on both general and domain-specific tasks.

6.1 General Applications

6.1.1 Classic Natural Language Processing. Classic natural language processing (NLP) tasks provide a natural testbed for studying how graph structures can enhance LLMs. These tasks cover multiple linguistic levels, ranging from word-level semantics to sentence-level representations and sequence-level predictions [198]. In these scenarios, Graph4LLM methods leverage graph structures to encode domain knowledge and provide explicit guidance for model reasoning.

Word-level tasks. Word-level tasks focus on modeling token semantics and relationships, such as word sense disambiguation and lexical analysis. In these tasks, graphs encode lexical relations, co-occurrence structures, or KG connections, grounding token representations in explicit relational structures to enhance contextual understanding. Pons *et al.* [124] improve zero-shot entity disambiguation by leveraging KG hierarchies and entity descriptions to prune candidates and enrich prompts, reducing ambiguity while maintaining cross-domain adaptability. Similarly, Salman *et al.* [131] integrate LLMs with Wikidata, retrieving candidates via SPARQL and selecting the most appropriate entity through generative matching, combining symbolic retrieval with contextual reasoning for strong disambiguation without task-specific training.

Sentence-level tasks. Sentence-level tasks aim to capture holistic semantics, including classification, similarity, and entailment. By constructing graphs over syntactic or inter-sentence relations, these methods provide a structured context that enriches sentence-level embeddings. For example, Zou and Markov [207] combine BERT [35] embeddings with GNNs and augment node features using GPT-generated emotion descriptions, leveraging both contextual encoding and structural modeling for improved emotion recognition. SKG-LLM [179] injects sentiment KG priors by constructing graphs over product features and emotion categories, integrating graph-derived signals into prompt optimization with adaptive weighting, enabling more fine-grained sentiment classification.

Sequence tagging. Sequence tagging tasks require assigning labels to each token in a sequence, such as named entity recognition and part-of-speech tagging. Graphs model dependencies beyond linear order, such as syntactic or long-range relations, to enable more globally consistent predictions. GNN-SL [150] addresses long-tail labeling by constructing a heterogeneous graph that links input sequences with similar annotated examples, using GNNs to propagate label information for token-level prediction. For cross-domain settings, Zheng *et al.* [200] model label relationships as graphs in both source and target domains and integrate them into BERT representations, improving named entity recognition (NER) under label mismatch and data scarcity.

Information extraction. Information extraction aims to identify structured elements such as entities, relations, and events. Graphs represent elements and their interactions, enabling structured reasoning over complex text. GLiM [40] constructs dynamic entity-pair graphs to constrain the search space of potential entities and uses LLMs to recover missing relations under incomplete labeling. RePooL [8] models tokens, spans, and relations in a dual-view graph, supporting relation-aware composition and hierarchical validation. Extending to event-level extraction, GDLLM [195] introduces distance-aware event graphs to capture long-range temporal dependencies. Beyond text, Iranmanesh *et al.* [64] apply graph-based retrieval to structured industry foundation classes (IFC) data, enabling graph-guided extraction of object properties and relations.

Text generation. Text generation tasks include summarization, data-to-text generation, and question answering. Graphs act as planning or control structures to organize content, guide decoding, and enforce structural constraints. GRAFT [38] models discourse units as a DAG to maintain cross-sentence consistency in document-level translation. POMP [121] builds a dynamic meta-graph over auxiliary languages to guide prompting in low-resource translation. For long-context summarization, GoR [184] links retrieved chunks with historical responses to jointly refine retrieval and generation. HeterMDS [180] further constructs a heterogeneous graph over sentences, paragraphs, and documents to capture hierarchical structure for coherent multi-document summarization.

6.1.2 Code. Code-related applications focus on understanding, generating, and reasoning over program structures, where syntax, control flow, and semantic dependencies play a central role. Unlike natural language, source code exhibits structures such as abstract syntax trees, control-flow graphs, and call graphs. Graph4LLM methods leverage these structures to complement token-level representations, enabling more precise and context-aware code understanding and generation.

Function-level tasks. These tasks operate at the level of individual functions or snippets, including code summarization, completion, and defect detection. Graphs typically model intra-function structures such as syntax trees, data-flow, and control-flow dependencies, exposing execution logic and variable interactions. SCoT [77] introduces structure-aware reasoning by aligning chains of thought with program constructs (e.g., branches and loops), improving generation accuracy. GRACE [108] enhances vulnerability detection by injecting code graph structures into in-context learning, while GALLa [194] aligns LLM representations with graph-based semantics during training, incorporating structural information without inference-time overhead.

Repository-level tasks. These tasks consider entire codebases and model cross-file and cross-module relationships through graphs such as call graphs, dependency graphs, and module interaction graphs. CodePlan [12] treats large-scale code editing as a planning problem, constructing dependency-aware edit sequences across files. RepoGraph [119] builds a unified code graph to support repository-level reasoning, while RepoHyper [123] performs graph-based search and expansion to retrieve relevant code beyond local scopes. CodexGraph [101] enables explicit structure-aware querying via a graph database interface, and LocAgent [27] applies multi-hop reasoning over heterogeneous graphs to align queries with code entities. Finally, CGM [141] injects graph structures into the attention mechanism, enabling holistic reasoning without relying on agent-based interaction.

6.1.3 Table. Table-related applications focus on understanding and reasoning over structured data, including table interpretation, table question answering, and synthetic data generation. Unlike unstructured text, tabular inputs encode meaning through rows, columns, cells, and schema relations, and prior table-specific models such as TaBERT [175] have shown the importance of modeling such structures. Graph4LLM methods extend this line of research by converting tables and their associated schemas or evidence into graphs, enabling LLMs to better capture structural dependencies that are easily lost in naive row-wise serialization.

Table understanding and representation. These approaches aim to enable LLMs to capture the semantic and structural properties of tables through explicit graph-based encoding. In this setting, graphs represent relationships among cells, headers, and rows. HeGTa [70] uses a heterogeneous graph with multi-grained self-supervised objectives to align tabular structure with LLM parametric knowledge for few-shot table understanding. TabGLM [113] combines graph-encoded structural embeddings with text-based semantic features to learn transferable table representations across datasets. For metadata-oriented tasks, RACOON [159] injects triples from external KGs into the LLM context to improve column type annotation.

Table-based question answering and reasoning. These tasks require navigating complex schemas or multimodal contexts, where answers are often distributed across tables and external evidence. Graphs typically serve as reasoning scaffolds or unified indices for cross-table dependencies. GraphOTTER [80] structures LLM reasoning over tabular graphs via a “Thought-Action-Update” loop. For joint table-text reasoning, Hybrid-QA [2] constructs a unified graph linking cells and passages for subgraph-based answer discovery. In large-scale databases, SGAM [151] encodes schema-level relations such as foreign keys and derivation paths, while TUNES [116] integrates entity-oriented search to enable graph-based reasoning over complex tables.

Synthetic tabular data generation. These tasks explore how graph-guided constraints can improve the fidelity and downstream utility of generated records. KGSynX [177] models attribute dependencies with a KG and feeds graph-derived structure into the LLM during synthesis. It further uses explainable feedback to iteratively refine generated samples, helping preserve relational consistency and task-relevant utility in low-data settings.

6.2 Domain-specific Applications

6.2.1 E-commerce. E-commerce applications primarily rely on recommendation systems to match users with relevant products, making recommendation a central task in this domain. These systems model user preferences from historical interactions between users and items, where collaborative signals, high-order connectivity, and semantic contexts play a key role. Unlike free-form text, e-commerce data is naturally organized as bipartite or heterogeneous graphs that encode user behaviors, product attributes, and relational signals such as co-purchase or co-view patterns [97]. Graph4LLM methods leverage these structures to integrate graph-based interaction modeling with the semantic understanding of LLMs, yielding more accurate, explainable, and context-aware product recommendations.

Collaboration and behavior. These applications operate on user-item interaction graphs to capture community patterns and high-order collaborative filtering signals in e-commerce platforms. In this setting, methods align structural graph representations with LLM representations to address challenges such as data sparsity and weak ID semantics. ELMRec [152] uses graph-aware whole-word embeddings and propagated structural positions to improve high-order interaction modeling. GAL-Rec [45] strengthens collaborative semantics through graph-aware learning, while LLMRec [160] and LLM-AGR [181] use LLMs to supervise graph augmentation or restructuring before recommendation. Beyond representation learning, graph topology can also reshape the computational space. CORONA [24] constructs a structured retrieval space from user–item graphs with coarse-to-fine filtering, narrowing the search space and aligning generation with high-order collaborative signals.

Knowledge and reasoning. These applications incorporate structured knowledge and reasoning to support more interpretable and domain-aware recommendations in e-commerce. Graphs in this category encode explicit relations such as product attributes, category hierarchies, and user preferences, allowing LLMs to reason beyond interaction history. In conversational recommendation, COMPASS [127] uses knowledge-graph-augmented LLMs to perform explainable reasoning over user preferences by extracting and aligning preference subgraphs. For domain-specific scenarios, GLRec [163] combines heterogeneous graphs with LLM prompting to improve zero-shot understanding of structured signals.

6.2.2 Medicine. Medical applications require knowledge-intensive tasks like clinical question answering, drug discovery, and diagnosis support, where factual accuracy and multi-step reasoning are critical. Biomedical data is structured, including molecular networks, clinical records, and medical ontologies [95, 139]. Graph4LLM methods leverage biomedical KGs for question answering and grounding [147, 162], and multiscale molecular graphs for drug development. By incorporating structured knowledge into retrieval, reasoning, and generation, they mitigate factual drift, enhance interpretability, and improve the reliability of medical decision support.

Medical question answering and reasoning. Methods in this category focus on retrieving and reasoning over biomedical literature and knowledge bases to support reliable answer generation. Medical Graph RAG [162] uses graph-based retrieval to anchor LLM outputs in verified medical evidence and reduce hallucinations. KG-RAG [139] further incorporates graph structure into prompt construction to better align model responses with clinical knowledge. LLM-KGMQA [147] tackles this by augmenting LLMs with a medical KG for multi-hop question answering.

Drug discovery and molecular design. These applications focus on generating or reasoning over candidate molecules under biochemical and synthesis constraints. Llamole [95] incorporates retrosynthetic planning into multimodal LLMs for inverse molecular design, where chemical synthesis pathways serve as graph-structured constraints during generation.

Clinical diagnosis and healthcare analytics. These applications use patient data and clinical graphs to support diagnosis and healthcare analytics. Magic [93] activates graph reasoning in a multi-agent diagnostic workflow, allowing LLM agents to navigate structured disease evidence for liver disease diagnosis. Diet-ODIN [192] models dietary patterns and patient behaviors as interpretable graphs, showing how graph-enhanced LLM pipelines can support public health monitoring tasks such as opioid misuse detection.

6.2.3 Education. Education-related applications focus on tasks such as resource discovery, intelligent tutoring, and personalized learning path planning. Unlike general information retrieval, educational settings are structured by prerequisite relations, curriculum constraints, and heterogeneous learning resources [19, 67]. Recent Graph4LLM methods use course KGs, learner knowledge component graphs, and graph-induced learning paths to support tutoring and recommendation [1, 148]. When operating without explicit structure, standalone LLMs may produce responses that are pedagogically inconsistent or weakly aligned with curriculum requirements. By grounding tutoring and recommendation in educational graphs, Graph4LLM methods improve coherence, interpretability, and personalization.

Educational knowledge discovery and question answering. These tasks aim to provide students with curriculum-aligned answers by retrieving relevant structural context. In MOOCs, Abdelmagied *et al.* [1] navigate course-specific KGs in GraphRAG-based tutoring to keep LLM explanations consistent with the syllabus. In institutional settings, cross-data KG construction [19] unifies heterogeneous educational resources for question answering, while KG-CQ [148] uses educational KGs to support self-directed tutoring. AcademicRAG [103] further treats academic resources as a graph-enhanced retrieval problem, enabling multi-hop access to learning materials.

Personalized learning and cognitive modeling. These applications tailor educational content by modeling the structure of knowledge components and learner states. FOKE [61] integrates foundation models with KGs to provide explainable feedback aligned with a student's cognitive profile. MAS-KCL [67] uses an agentic workflow to learn latent knowledge component graphs, and Wasnik and Shrivastava [154] integrate LLMs and KGs to build adaptive educational frameworks that adjust difficulty and content distribution based on the learner's graph-encoded proficiency profile.

6.2.4 Finance. Financial applications focus on tasks such as market trend analysis, financial question answering, fraud or risk detection, and personalized financial services. Unlike standalone document analysis, financial settings rely on transaction graphs, inter-corporate relation graphs, event propagation structures, and dynamic market knowledge [86, 170]. Recent Graph4LLM methods use these structures for market trend detection, financial retrieval, and graph-based risk analysis [14, 142]. When processing financial data without explicit relational structure, standalone LLMs often miss systemic dependencies and weakly capture temporal propagation patterns. By combining graph-based market structure with LLM semantic reasoning, Graph4LLM methods provide more structured support for financial analysis and services.

Market trend analysis and event ripple effects. These tasks focus on modeling how market signals propagate across related entities and events. FinRipple [170] aligns LLMs with financial market structure to improve awareness of event ripple effects across equities. FinDKG [86] combines dynamic KGs with LLMs for global trend detection, while FinCARE [208] uses graph-enhanced reasoning to analyze causal relations behind market movements.

Financial question answering and document intelligence. These tasks require LLMs to navigate financial documents and dynamic market knowledge under factual constraints. Barry *et al.* [14] use graph-based indices in GraphRAG for finance to improve retrieval efficiency and factual grounding over financial data. FinQA [142] treats financial question answering as navigation over dynamic KGs and uses LLM-based revision to refine responses without task-specific fine-tuning.

Risk management and financial services. These applications use transaction graphs, behavioral relations, and user profiles to support risk detection and personalized services. For financial risk governance, Li *et al.* [89] present an end-to-end risk assessment model. By integrating text mining and financial risk KGs into an LLM with RAG and multi-feature fusion, the model enables precise credit risk prediction and rational default probability estimation for effective risk quantification. On the consumer-facing side, Zhao *et al.* [199] combine LLMs with GNNs for personalized financial product recommendation by integrating user-item interaction structure with semantic profiles.

6.2.5 Law. Applications in the legal domain focus on tasks such as legal consulting, statutory question answering, judgment prediction, and legal article recommendation. Unlike general text, legal data is governed by statutory hierarchies, precedent relations, temporal validity, and normative constraints [138]. Recent Graph4LLM systems construct legal KGs, citation graphs, and graph-based legal indices to support retrieval and analysis [182]. When processing such data, standalone LLMs are prone to hallucinations, weak temporal grounding, and unreliable alignment with formal legal constraints. By integrating legal graphs with LLM reasoning, Graph4LLM methods provide more structured support for legal retrieval, analysis, and decision assistance.

Legal consulting and question answering. These tasks aim to provide legally grounded answers under current rules and precedents. Legal-LM [138] integrates legal KGs with language models for legal consulting. Law GraphRAG [182] and Dang *et al.* [33] construct structured graph indices for multi-hop legal retrieval.

Judgment prediction and judicial recommendation. These applications use case graphs and statutory relations to predict outcomes or recommend legal provisions. LLG-Judger [166] combines LLMs with graph contrastive learning over similar-case structures for legal judgment prediction. For law article recommendation, JKEM [78] builds a case-enhanced legal KG that links case facts with criminal law statutes for graph-guided retrieval and recommendation. LegisSearch [32] further combines legislative KGs with LLMs and text embeddings for graph-aware legislative retrieval.

6.2.6 EDA. Electronic design automation (EDA) applications leverage circuit topology to enhance hardware design and analysis with LLMs. Unlike natural language, hardware description languages (HDL) and circuit netlists exhibit strict interconnectivity [99], fan-in/fan-out constraints [96], and hierarchical dependencies [196], forming rich multi-source structural data [46, 197]. Early works [28, 137] primarily used GNNs to model circuit topology, but overlooked textual semantics in design documents and comments. Standalone LLMs struggle with long-range dependencies and may generate hallucinations that violate syntax or logic. By introducing structure-aware graphs, Graph4LLM integrates circuit topology with LLM reasoning, improving code generation, power-performance-area (PPA) prediction, and automated optimization.

Multimodal circuit modelling and performance analysis. By aligning structural features from graph encoders with the semantic understanding of LLMs, these applications enable early prediction of circuit performance metrics such as timing and power. NetTAG [41] represents netlists as text-attributed graphs, using LLMs to encode gate-level logic semantics and graph Transformers to capture global structure, resulting in a universal netlist foundation model. CircuitFusion [42] further fuses hardware code, structure graphs, and functional summaries via multimodal alignment, improving the generalization of PPA prediction. For timing analysis, GT-LLM [7] uses an LLM as a parser to convert register-transfer level (RTL) into graph structures, paired with a graph learning engine for early timing prediction. MOSS [149] targets sequential circuits, integrating GNNs and LLMs to resolve long-range information dependencies and strengthen modeling of complex logical behaviors.

Structure-enhanced hardware code generation. These applications use the syntactic and logical graph structures of circuits to guide LLMs in generating high-quality hardware description code. Given the stringent requirements for logical correctness in hardware design, SAGE-HLS [72] introduces a syntax-aware, abstract syntax tree (AST)-guided framework that improves the quality of high-level synthesis (HLS) code generation via structured prompting. For RTL modeling, RTL++ [4] converts code into control flow graphs (CFGs) and data flow graphs (DFGs) with textual encoding, capturing internal hierarchies and dependencies to substantially boost the RTL generation quality of open-source models. SynC-LLM [100] further breaks through the limitations of LLMs in circuit generation scale, and proposes a hierarchical synthetic circuit code generation framework: first, the global structure is generated through the graph diffusion model, and then the LLMs are guided to fill the code layer by layer with hierarchical controllable prompts.

Graph-guided design optimization. These applications harness LLM reasoning to guide sophisticated graph transformations or provide generic graph enhancement across tasks. For datapath optimization, ASPEN [186] uses LLMs to direct E-Graph rewriting in equality saturation, automatically searching for optimal area-performance trade-offs along the Pareto frontier. In logic synthesis, CircuitEvo [11] innovatively employs LLMs to iteratively evolve graph-structured programs, automating circuit compactness optimization while preserving functional correctness. To address the difficulty of integrating graph modalities in EDA tasks, BRIDGES [83] constructs a large-scale multimodal dataset and generic framework. It connects DFGs and netlist graphs to LLMs via lightweight cross-modal projectors, providing a standardized graph-enhanced foundation for diverse EDA workflows.

7 Challenges and Future Directions

Graph construction quality and robustness. The effectiveness of Graph4LLM methods critically depends on the quality and robustness of the constructed graphs. In practice, graphs are often derived from noisy corpora or imperfect knowledge bases, making them prone to incompleteness, spurious relations, and structural bias. These imperfections can propagate through downstream components, leading to misleading retrieval results and distorted reasoning trajectories. Future work should therefore prioritize reliable and adaptive graph construction, capable of handling varying data quality and uncertainty. This can be achieved, for instance, by incorporating uncertainty quantification methods applied to graph nodes and edges, enabling LLMs to reason over probabilistic or graded relations. Additionally, iterative refinement mechanisms that leverage feedback from LLM outputs to revise or prune unreliable structures represent a promising direction.

Complexity challenges from explicit structural expansion. Larger graphs can capture more entities and relations, but increasing size, density, and diversity also brings higher computational cost and reasoning difficulty. In practice, large or highly connected graphs can lead to a combinatorial explosion of paths, making search inefficient and sometimes unstable. Future work should therefore focus on complexity-aware graph design, such as pruning and hierarchical structures, to better match graph granularity with task requirements. This allows the model to focus on the most relevant parts of the graph at each stage. A key challenge is to determine how much structure to expose to the model and when to use it, which is critical for scalable and reliable reasoning. Ultimately, the goal is to make graph structure not just richer, but more useful.

Scaling laws for structural information. While scaling laws for model size and training data are well understood, it remains unclear how performance scales with different graph structures in Graph4LLM systems. In this context, scaling involves graph size, the number of agents, and the expansion of intermediate reasoning steps during inference. Increasing these factors can improve reasoning depth, but may also introduce noise and degrade overall efficiency. A key challenge is to understand how structural properties such as graph density and reasoning depth affect performance. In particular, it remains unclear whether increasing structure leads to consistent gains or quickly reaches saturation due to redundancy and error propagation. Future work should aim to characterize these scaling trends and identify efficient regimes, enabling better control over graph expansion and more reliable reasoning.

Towards self-improving Graph4LLM: distillation, feedback, and co-evolution. Despite their effectiveness, most existing Graph4LLM methods use a unidirectional process, where graphs assist LLMs at specific stages but remain external to the model. The information in these graphs is consumed only during generation and not retained afterward. As a result, valuable structural signals must be repeatedly reconstructed, which limits efficiency and long-term knowledge transfer. To address these limitations, future work should focus on self-improving Graph4LLM systems. One promising method is graph-to-model distillation, where useful relational patterns from the Graph4LLM process are integrated into the model's representations through fine-tuning or knowledge distillation. This would allow the model to internalize structural priors and reduce reliance on explicit graphs during inference. Model-to-graph feedback can also refine graph construction, enabling representations to evolve based on utility and support more robust, reusable structures.

8 Conclusion

This paper presents a comprehensive survey of Graph4LLM, categorizing existing graph-enhanced LLM methods in the input, model, and output phases. It covers representative methods across different paradigms and their diverse application scenarios. Specifically, it also introduces both general and domain-specific applications of Graph4LLM, illustrating its wide applicability in universal scenarios. Additionally, the paper summarizes key challenges facing current Graph4LLM research and outlines future directions. By providing a coherent structural perspective on the integration of graphs and LLMs, this survey offers a concise overview of the field, facilitating a clear understanding of how graphs enhance the capabilities of LLMs.

References

[1] Mohamed Abdelmagied, Mohamed Amine Chatti, Shoeb Joarder, Qurat Ul Ain, and Rawaa Alatrash. 2025. Leveraging graph retrieval-augmented generation to support learners' understanding of knowledge concepts in MOOCs. In *European MOOCs Stakeholders Summit*. Springer, 108–118.

[2] Ankush Agarwal, Chaitanya Devaguptapu, et al. 2025. Hybrid graphs for table-and-text based question answering using llms. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. 858–875.

[3] Garima Agrawal, Tharindu Kumarage, Zeyad Alghamdi, and Huan Liu. 2024. Can knowledge graphs reduce hallucinations in llms?: A survey. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. 3947–3960.

[4] Mohammad Akyash, Kimia Azar, and Hadi Kamali. 2025. Rtl++: Graph-enhanced llm for rtl code generation. In *2025 IEEE International Conference on LLM-Aided Design (ICLAD)*. IEEE, 44–50.

[5] Fatimah Alotaibi, Adithya Kulkarni, and Dawei Zhou. 2024. Graph of logic: Enhancing llm reasoning with graphs and symbolic logic. In *2024 IEEE International Conference on Big Data (BigData)*. IEEE, 5926–5935.

[6] Alessia Amelio, Christopher Buratti, Michele Marchetti, Davide Traini, Domenico Ursino, and Luca Virgili. 2025. Exploiting knowledge graph communities to fine-tune large language models. *Expert Systems with Applications* (2025), 129816.

[7] Fahad Rahman Amik, Yousef Safari, Zhanguang Zhang, and Boris Vaisband. 2025. Graph-based timing prediction at early-stage RTL using large language model. In *Proceedings of the 30th Asia and South Pacific Design Automation Conference (ASP-DAC)*. 393–400.

[8] Hye-Yoon Baek, Jinho Choi, Jimyeung Seo, Xiongnan Jin, Dongcheon Lee, and Byungkook Oh. 2025. Relation-Faceted Graph Pooling with LLM Guidance for Dynamic Span-Aware Information Extraction. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*. 109–118.

[9] Jinheon Baek, Alham Fikri Aji, and Amir Saffari. 2023. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. In *Proceedings of the 1st Workshop on Natural Language Reasoning and Structured Explanations (NLRSE)*. 78–106.

[10] Ruiqiao Bai, Xue Han, Shuo Lei, Junlan Feng, Yanyan Luo, and Chao Deng. 2025. Self-attention-based Graph-of-Thought for Math Problem Solving. In *Findings of the Association for Computational Linguistics: ACL 2025*. 6112–6125.

[11] Yinqi Bai, Jie Wang, Lei Chen, Zhihai Wang, Yumeng Li, Mingxuan Yuan, Jianye HAO, Defu Lian, and Enhong Chen. 2026. Evolving Graph Structured Programs for Circuit Generation with Large Language Models. In *The Fourteenth International Conference on Learning Representations (ICLR)*.

[12] Ramakrishna Bairi, Atharv Sonwane, Aditya Kanade, Vageesh D C, Arun Iyer, Suresh Parthasarathy, Sriram Rajamani, Balasubramanyan Ashok, and Shashank Shet. 2024. Codeplan: Repository-level coding using llms and planning. *Proceedings of the ACM on Software Engineering* 1, FSE (2024), 675–698.

[13] Albert-Laszlo Barabasi and Zoltan N Oltvai. 2004. Network biology: understanding the cell's functional organization. *Nature reviews genetics* 5, 2 (2004), 101–113.

[14] Mariam Barry, Gaëtan Caillaut, Pierre Halftermeyer, Raheel Qader, Mehdi Mouayad, Fabrice Le Deit, Dimitri Cariolaro, and Joseph Gesnouin. 2025. GraphRAG: leveraging graph-based efficiency to minimize hallucinations in LLM-driven RAG for finance data. In *Proceedings of the Workshop on Generative AI and Knowledge Graphs (GenAIK)*. 54–65.

[15] Michael GH Bell and Yasunori Iida. 1997. *Transportation network analysis*.

[16] Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 38. 17682–17690.

[17] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[18] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[19] Tuan Bui, Oanh Tran, Phuong Nguyen, Bao Ho, Long Nguyen, Thang Bui, and Tho Quan. 2024. Cross-data knowledge graph construction for LLM-enabled educational question-answering system: A case study at HCMUT. In *Proceedings of the 1st ACM Workshop on AI-Powered Q&A Systems for Multimedia*. 36–43.

[20] Linyue Cai, Chaojia Yu, Yongqi Kang, Yu Fu, Heng Zhang, and Yong Zhao. 2025. Practices, opportunities and challenges in the fusion of knowledge graphs and large language models. *Frontiers in computer science* 7 (2025), 1590632.

[21] Lang Cao. 2024. Graphreason: Enhancing reasoning capabilities of large language models through a graph-based verification approach. In *Proceedings of the 2nd Workshop on Natural Language Reasoning and Structured Explanations (@ ACL 2024)*. 1–12.

[22] Boyu Chen, Zirui Guo, Zidan Yang, Yuluo Chen, Junze Chen, Zhenghao Liu, Chuan Shi, and Cheng Yang. 2026. Pathrag: Pruning graph-based retrieval augmented generation with relational paths. In *Proceedings of the AAAI conference on artificial intelligence*.

[23] Hanzhu Chen, Xu Shen, Jie Wang, Zehao Wang, Qitan Lv, Junjie He, Rong Wu, Feng Wu, and Jieping Ye. 2025. Knowledge graph finetuning enhances knowledge manipulation in large language models. In *The Thirteenth International Conference on Learning Representations*.

[24] Junze Chen, Xinjie Yang, Cheng Yang, Junfei Bao, Zeyuan Guo, Yawen Li, and Chuan Shi. 2025. Corona: A coarse-to-fine framework for graph-based recommendation with large language models. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2048–2058.

[25] Kedi Chen, Qin Chen, Jie Zhou, Xinqi Tao, Bowen Ding, Jingwen Xie, Mingchen Xie, Peilong Li, and Zheng Feng. 2025. Enhancing uncertainty modeling with semantic graph for hallucination detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 23586–23594.

[26] Liyi Chen, Panrong Tong, Zhongming Jin, Ying Sun, Jieping Ye, and Hui Xiong. 2024. Plan-on-graph: Self-correcting adaptive planning of large language model on knowledge graphs. *Advances in Neural Information Processing Systems* 37 (2024), 37665–37691.

[27] Zhaoling Chen, Robert Tang, Gangda Deng, Fang Wu, Jialong Wu, Zhiwei Jiang, Viktor Prasanna, Arman Cohan, and Xingyao Wang. 2025. Locagent: Graph-guided llm agents for code localization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 8697–8727.

[28] HongTao Cheng, Jiawei Liu, Jianwang Zhai, Mingyu Zhao, Cheng Yang, and Chuan Shi. 2024. SATGL: An Open-Source Graph Learning Toolkit for Boolean Satisfiability. In *2024 2nd International Symposium of Electronics Design Automation (ISEDA)*. IEEE, 746–751.

[29] Kewei Cheng, Nesreen K Ahmed, Theodore L Willke, and Yizhou Sun. 2024. Structure guided prompt: Instructing large language model in multi-step reasoning by exploring graph structure of the text. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. 9407–9430.

[30] Yizhou Chi, Kevin Yang, and Dan Klein. 2025. Thoughtsculpt: Reasoning with intermediate revision and search. In *Findings of the Association for Computational Linguistics: NAACL 2025*. 7685–7711.

[31] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research* 24, 240 (2023), 1–113.

[32] Andrea Colombo, Anna Bernasconi, Luigi Bellomarini, Luigi Guiso, Claudio Michelacci, and Stefano Ceri. 2025. LegisSearch: navigating legislation with graphs and large language models. *Artificial Intelligence and Law* (2025), 1–27.

[33] Dung V Dang, Hau Nguyen, Thao TN Le, Hung Do, Hung Nguyen, Hung Q Ngo, and Hien D Nguyen. 2025. Knowledge graph-based legal query system with llm and retrieval augmented generation. In *Asian Conference on Intelligent Information and Database Systems*. Springer, 161–172.

[34] Stefan Dernbach, Khushbu Agarwal, Alejandro Zuniga, Michael Henry, and Sutanay Choudhury. 2024. Glam: Fine-tuning large language models for domain knowledge graph alignment via neighborhood partitioning and generative subgraph encoding. In *Proceedings of the AAAI Symposium Series*, Vol. 3. 82–89.

[35] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 4171–4186.

[36] Ruomeng Ding, Chaoyun Zhang, Lu Wang, Yong Xu, Minghua Ma, Wei Zhang, Si Qin, Saravan Rajmohan, Qingwei Lin, and Dongmei Zhang. 2024. Everything of thoughts: Defying the law of penrose triangle for thought generation. In *Findings of the Association for Computational Linguistics: ACL 2024*. 1638–1662.

[37] Jiangwen Dong, Zehui Lin, Wanyu Lin, and Mingjin Zhang. 2026. S-DAG: A Subject-Based Directed Acyclic Graph for Multi-Agent Heterogeneous Reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[38] Himanshu Dutta, Sunny Manchanda, Prakhar Bapat, Meva Ram Gurjar, and Pushpak Bhattacharyya. 2025. GRAFT: A Graph-based Flow-aware Agentic Framework for Document-level Machine Translation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*. 2405–2428.

[39] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130* (2024).

[40] Hao Fang, Yuejie Zhang, Rui Feng, Yingwen Wang, Qing Wang, Wen He, Xiaobo Zhang, Tao Zhang, and Shang Gao. 2025. GLiM: integrating graph transformer and LLM for document-level biomedical relation extraction with incomplete labeling. In *Findings of the Association for Computational Linguistics: ACL 2025*. 14131–14146.

[41] Wenji Fang, Wenkai Li, Shang Liu, Yao Lu, Hongce Zhang, and Zhiyao Xie. 2025. Nettag: A multimodal rtl-and-layout-aligned netlist foundation model via text-attributed graph. In *2025 62nd ACM/IEEE Design Automation Conference (DAC)*. IEEE, 1–7.

[42] Wenji Fang, Shang Liu, Jing Wang, and Zhiyao Xie. 2025. CircuitFusion: Multimodal Circuit Representation Learning for Agile Chip Design. In *The Thirteenth International Conference on Learning Representations (ICLR)*.

[43] Xinyue Fang, Zhen Huang, Zhiliang Tian, Minghui Fang, Ziyi Pan, Quntian Fang, Zhihua Wen, Hengyue Pan, and Dongsheng Li. 2025. Zero-resource hallucination detection for text generation via graph-based contextual knowledge triples modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 23868–23877.

[44] Xinyan Guan, Yanjiang Liu, Hongyu Lin, Yaojie Lu, Ben He, Xianpei Han, and Le Sun. 2024. Mitigating large language model hallucinations via autonomous knowledge graph-based retrofitting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 18126–18134.

[45] Zhong Guan, Likang Wu, Hongke Zhao, Ming He, and Jianpin Fan. 2025. Enhancing collaborative semantics of language model-driven recommendations via graph-aware learning. *IEEE Transactions on Knowledge and Data Engineering* (2025).

[46] Feng Guo, Yueyue Xi, Jianwang Zhai, Jingyu Jia, Jiawei Liu, Kang Zhao, and Chuan Shi. 2025. IRGNN: A Graph-based Framework Integrating Numerical Solution and Point Cloud for Static IR Drop Prediction. In *2025 62nd ACM/IEEE Design Automation Conference (DAC)*. IEEE, 1–7.

[47] Zeyuan Guo, Enmao Diao, Cheng Yang, and Chuan Shi. 2026. Graph Tokenization for Bridging Graphs and Transformers. In *The Fourteenth International Conference on Learning Representations*.

[48] Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2025. LightRAG: Simple and Fast Retrieval-Augmented Generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*. 10746–10761.

[49] Bernal J Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. *Advances in neural information processing systems* 37 (2024), 59532–59569.

[50] Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to Memory: Non-Parametric Continual Learning for Large Language Models. In *Forty-second International Conference on Machine Learning*.

[51] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).

[52] Haoyu Han, Yaochen Xie, Hui Liu, Xianfeng Tang, Sreyashi Nag, William Headden, Yang Li, Chen Luo, Shuiwang Ji, Qi He, et al. 2025. Reasoning with graphs: Structuring implicit knowledge to enhance llms reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*. 25698–25714.

[53] Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 8154–8173.

[54] Junda He, Christoph Treude, and David Lo. 2025. Llm-based multi-agent systems for software engineering: Literature review, vision, and the road ahead. *ACM Transactions on Software Engineering and Methodology* 34, 5 (2025), 1–30.

[55] Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *Advances in Neural Information Processing Systems* 37 (2024), 132876–132907.

[56] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. 2021. Knowledge graphs. *ACM Computing Surveys (Csur)* 54, 4 (2021), 1–37.

[57] Giwon Hong, Jeonghwan Kim, Junmo Kang, and Sung-Hyon Myaeng. 2022. Graph-induced transformers for efficient multi-hop question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. 10288–10294.

[58] Bairu Hou, Yang Zhang, Jacob Andreas, and Shiyu Chang. 2025. A probabilistic framework for llm hallucination detection via belief tree propagation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. 3076–3099.

[59] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *International conference on machine learning*. PMLR, 2790–2799.

[60] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

[61] Silan Hu and Xiaoning Wang. 2024. Foke: A personalized and explainable education framework integrating foundation models, knowledge graphs, and prompt engineering. In *China national conference on big data and social computing*. Springer, 399–411.

[62] Jie Huang and Kevin Chen-Chuan Chang. 2023. Towards reasoning in large language models: A survey. In *Findings of the association for computational linguistics: ACL 2023*. 1049–1065.

[63] Nourhan Ibrahim, Samar Aboulela, Ahmed Ibrahim, and Rasha Kashef. 2024. A survey on augmenting knowledge graphs (KGs) with large language models (LLMs): models, evaluation metrics, benchmarks, and challenges. *Discover Artificial Intelligence* 4, 1 (2024), 76.

[64] Sima Iranmanesh, Hadeel Saadany, and Edlira Vakaj. 2025. LLM-assisted Graph-RAG Information Extraction from IFC Data. In *Proceedings of the 2025 European Conference on Computing in Construction*, Vol. 6.

[65] Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Structgpt: A general framework for large language model to reason over structured data. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 9237–9251.

[66] Mingjian Jiang, Yangjun Ruan, Prasanna Sattigeri, Salim Roukos, and Tatsunori Hashimoto. 2024. Graph-based uncertainty metrics for long-form language model generations. *Advances in Neural Information Processing Systems* 37 (2024), 32980–33006.

[67] Yuan-Hao Jiang, Kezong Tang, Zi-Wei Chen, Yuang Wei, Tian-Yi Liu, and Jiayi Wu. 2025. MAS-KCL: Knowledge component graph structure learning with large language model-based agentic workflow. *The Visual Computer* 41, 9 (2025), 6453–6464.

[68] Bowen Jin, Gang Liu, Chi Han, Meng Jiang, Heng Ji, and Jiawei Han. 2024. Large language models on graphs: A comprehensive survey. *IEEE Transactions on Knowledge and Data Engineering* (2024).

[69] Bowen Jin, Wentao Zhang, Yu Zhang, Yu Meng, Xinyang Zhang, Qi Zhu, and Jiawei Han. 2023. Patton: Language model pretraining on text-rich networks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 7005–7020.

[70] Rihui Jin, Yu Li, Guilin Qi, Nan Hu, Yuan-Fang Li, Jiaoyan Chen, Jianan Wang, Yongrui Chen, Dehai Min, and Sheng Bi. 2025. Hegta: Leveraging heterogeneous graph-enhanced large language models for few-shot complex table understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 24294–24302.

[71] Taejong Joo, Shu Ishida, Ivan Sosnovik, Bryan Lim, Sahand Rezaei-Shoshtari, Adam Gaier, and Robert Giaquinto. 2026. Graph of Agents: Principled Long Context Modeling by Emergent Multi-Agent Collaboration. In *The Thirteenth International Conference on Learning Representations (ICLR)*.

[72] M Zafir Sadik Khan, Nowfel Mashnoor, Mohammad Akyash, Kimia Azar, and Hadi Kamali. 2025. Sage-hls: Syntax-aware ast-guided llm for high-level synthesis code generation. In *2025 IEEE 43rd International Conference on Computer Design (ICCD)*. IEEE, 574–581.

[73] Thomas N Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*.

[74] Hui Yi Leong, Yuheng Li, Yuqing Wu, Wenwen Ouyang, Wei Zhu, Jiechao Gao, and Wei Han. 2025. Amas: Adaptively determining communication topology for llm-based multi-agent system. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*. 2061–2070.

[75] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems* 33 (2020), 9459–9474.

[76] Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems* 36 (2023), 51991–52008.

[77] Jia Li, Ge Li, Yongmin Li, and Zhi Jin. 2025. Structured chain-of-thought prompting for code generation. *ACM Transactions on Software Engineering and Methodology* 34, 2 (2025), 1–23.

[78] Jun Li, Lu Qian, Peifeng Liu, and Taoxiong Liu. 2024. Construction of legal knowledge graph based on knowledge-enhanced large language models. *Information* 15, 11 (2024), 666.

[79] Mufei Li, Siqi Miao, and Pan Li. 2025. Simple is Effective: The Roles of Graphs and Large Language Models in Knowledge-Graph-Based Retrieval-Augmented Generation. In *The Thirteenth International Conference on Learning Representations*.

[80] Qianlong Li, Chen Huang, Shuai Li, Yuanxin Xiang, Deng Xiong, and Wenqiang Lei. 2025. Graphotter: Evolving llm-based graph reasoning for complex table question answering. In *Proceedings of the 31st International Conference on Computational Linguistics*. 5486–5506.

[81] Shilong Li, Yancheng He, Hangyu Guo, Xingyuan Bu, Ge Bai, Jie Liu, Jiaheng Liu, Xingwei Qu, Yangguang Li, Wanli Ouyang, et al. 2024. Graphreader: Building graph-based agent to enhance long-context abilities of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*. 12758–12786.

[82] Shusheng Li, Jiale Li, Yifei Qu, Xinwei Shi, Yanliang Guo, Ziyi He, Yubo Wang, and Wenjun Tan. 2025. Semantic-Eval: A Semantic Comprehension Evaluation Framework for Large Language Models Generation without Training. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 9675–9690.

[83] Wei Li, Yang Zou, Christopher Ellis, Ruben Purdy, Shawn Blanton, and José MF Moura. 2025. BRIDGES: bridging graph modality and large language models within EDA tasks. In *2025 IEEE International Conference on LLM-Aided Design (ICLAD)*. IEEE, 77–84.

[84] Xin Li, Dongze Lian, Zhihe Lu, Jiawang Bai, Zhibo Chen, and Xinchao Wang. 2023. Graphadapter: Tuning vision-language models with dual knowledge graph. *Advances in Neural Information Processing Systems* 36 (2023), 13448–13466.

[85] Xinyi Li, Sai Wang, Siqi Zeng, Yu Wu, and Yi Yang. 2024. A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges. *Vicinagearth* 1, 1 (2024), 9.

[86] Xiaohui Victor Li and Francesco Sanna Passino. 2024. Findkg: Dynamic knowledge graphs with large language models for detecting global trends in financial markets. In *Proceedings of the 5th ACM International Conference on AI in Finance*. 573–581.

[87] Yang Li. 2025. Policy Guided Tree Search for Enhanced LLM Reasoning. In *Forty-second International Conference on Machine Learning*.

[88] Yukun Li, Sijia Wang, Lifu Huang, and Liping Liu. 2025. Graph-based Confidence Calibration for Large Language Models. *Transactions on Machine Learning Research* (2025).

[89] Zhenglin Li, Mingxiu Sui, Chen Yang, Sa Liu, Bo Guan, and Xinjin Li. 2025. Construction of Financial Risk Assessment Model Based on Text Mining and LLM Architecture. In *Proceedings of the 2025 2nd International Conference on Digital Economy and Computer Science*. 152–158.

[90] Chumeng Liang and Jiaxuan You. 2025. Evaluating LLM-Generated Diagrams as Graphs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (Eds.). 12678–12690.

[91] Xinxin Liang, Zuoxu Wang, Mingrui Li, and Zhijie Yan. 2024. A survey of LLM-augmented knowledge graph construction and application in complex product design. *Procedia CIRP* 128 (2024), 870–875.

[92] Bingchen Liu, Yuanyuan Fang, Naixing Xu, Shihao Hou, Xin Li, and Qian Li. 2025. Large language models for knowledge graph embedding: A survey. *Mathematics* 13, 14 (2025), 2244.

[93] Bowen Liu, Yaqing Nie, Hong Song, Yucong Lin, Jingtao Li, Xutao Weng, Zhaoli Su, Yuhong Suo, Tingting Lv, Xinyan Zhao, et al. 2025. Magic: AN LLM-based multi-agent activated graph-reasoning intelligent collaboration model for liver disease diagnosis. *Information Fusion* (2025), 103557.

[94] Ben Liu, Jihai Zhang, Fangquan Lin, Cheng Yang, and Min Peng. 2025. Filter-then-generate: Large language models with structure-text adapter for knowledge graph completion. In *Proceedings of the 31st International Conference on Computational Linguistics*. 11181–11195.

[95] Gang Liu, Michael Sun, Wojciech Matusik, Meng Jiang, and Jie Chen. 2025. Multimodal Large Language Models for Inverse Molecular Design with Retrosynthetic Planning. In *The Thirteenth International Conference on Learning Representations*.

[96] Jiawei Liu, Zhiyan Liu, Xun He, Jianwang Zhai, Zhengyuan Shi, Qiang Xu, Bei Yu, and Chuan Shi. 2025. WideGate: Beyond Directed Acyclic Graph Learning in Subcircuit Boundary Prediction. In *2025 Design, Automation & Test in Europe Conference (DATE)*. IEEE, 1–7.

[97] Jiawei Liu, Chuan Shi, Cheng Yang, Zhiyuan Lu, and Philip S Yu. 2022. A survey on heterogeneous information network based recommender systems: Concepts, methods, applications and resources. *AI Open* 3 (2022), 40–57.

[98] Jiawei Liu, Cheng Yang, Zhiyuan Lu, Junze Chen, Yibo Li, Mengmei Zhang, Ting Bai, Yuan Fang, Lichao Sun, Philip S Yu, et al. 2025. Graph foundation models: Concepts, opportunities and challenges. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2025).

[99] Jiawei Liu, Jianwang Zhai, Mingyu Zhao, Zhe Lin, Bei Yu, and Chuan Shi. 2024. Polargate: Breaking the functionality representation bottleneck of and-inverter graph neural network. In *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design*. 1–9.

[100] Shang Liu, Yao Lu, Wenji Fang, Jing Wang, and Zhiyao Xie. 2025. Sync-llm: Generation of large-scale synthetic circuit code with hierarchical language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 17361–17376.

[101] Xiangyan Liu, Bo Lan, Zhiyuan Hu, Yang Liu, Zhicheng Zhang, Fei Wang, Michael Qizhe Shieh, and Wenmeng Zhou. 2025. Codexgraph: Bridging large language models and code repositories via code graph databases. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. 142–160.

[102] Yuze Liu, Tingjie Liu, Tiehua Zhang, Youhua Xia, Jinze Wang, Zhishu Shen, Jiong Jin, Zhijun Ding, and Fei Richard Yu. 2025. GRL-Prompt: Towards Prompts Optimization via Graph-Empowered Reinforcement Learning Using LLMs' Feedback. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 426–438.

[103] Zhuchenyang Liu. 2025. AcademicRAG: Knowledge Graph Enhanced Retrieval-Augmented Generation for Academic Resource Discovery. (2025).

[104] Zhaoyang Liu, Zeqiang Lai, Zhangwei Gao, Erfei Cui, Ziheng Li, Xizhou Zhu, Lewei Lu, Qifeng Chen, Yu Qiao, Jifeng Dai, et al. 2024. Controlllm: Augment language models with tools by searching on graphs. In *European Conference on Computer Vision*. Springer, 89–105.

[105] Zhiyuan Liu, Sihang Li, Yanchen Luo, Hao Fei, Yixin Cao, Kenji Kawaguchi, Xiang Wang, and Tat-Seng Chua. 2023. Molca: Molecular graph-language modeling with cross-modal projector and uni-modal adapter. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 15623–15638.

[106] Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. 2024. A dynamic llm-powered agent network for task-oriented agent collaboration. In *First Conference on Language Modeling*.

[107] Xingzhou Lou, Junge Zhang, Timothy J Norman, Kaiqi Huang, and Yali Du. 2024. TAPE: leveraging agent topology for cooperative multi-agent policy gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 17496–17504.

[108] Guilong Lu, Xiaolin Ju, Xiang Chen, Wenlong Pei, and Zhilong Cai. 2024. GRACE: Empowering LLM-based software vulnerability detection with graph structure and in-context learning. *Journal of Systems and Software* 212 (2024), 112031.

[109] Haoran Luo, E Haihong, Zichen Tang, Shiyao Peng, Yikai Guo, Wentai Zhang, Chenghao Ma, Guanting Dong, Meina Song, Wei Lin, et al. 2024. Chatkbqa: A generate-then-retrieve framework for knowledge base question answering with fine-tuned large language models. In *Findings of the association for computational linguistics: ACL 2024*. 2039–2056.

[110] Linhao Luo, Yuan Fang Li, Gholamreza Haffari, and Shirui Pan. 2024. Reasoning on Graphs: Faithful and Interpretable Large Language Model Reasoning. In *International Conference on Learning Representations 2024*. International Conference on Learning Representations (ICLR).

[111] Linhao Luo, Zicheng Zhao, Gholamreza Haffari, Dinh Phung, Chen Gong, and Shirui Pan. 2025. GFM-RAG: Graph Foundation Model for Retrieval Augmented Generation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[112] Yusong Ma, Hongxuan Nie, Chao Chen, Jiujie Zhang, Jiali Jiang, Bisheng Wang, and Yuqin Xia. 2025. A Survey of Retrieval-Augmented Generation (RAG) for Large Language Models. In *2025 International Conference on Trustworthy Big Data and Artificial Intelligence (ICTBAI)*. IEEE, 7–13.

[113] Anay Majee, Maria Xenochristou, and Wei-Peng Chen. 2025. Tabglm: Tabular graph language model for learning transferable representations through multi-modal consistency minimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 19387–19395.

[114] Costas Mavromatis and George Karypis. 2025. GNN-RAG: Graph neural retrieval for efficient large language model reasoning on knowledge graphs. In *Findings of the Association for Computational Linguistics: ACL 2025*. 16682–16699.

[115] Gouki Minegishi, Hiroki Furuta, Takeshi Kojima, Yusuke Iwasawa, and Yutaka Matsuo. 2025. Topology of Reasoning: Understanding Large Reasoning Models through Reasoning Graph Properties. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[116] Thi-Nhung Nguyen, Hoang Ngo, Dinh Phung, Thuy-Trang Vu, and Dat Quoc Nguyen. 2025. Improving Table Understanding with LLMs and Entity-Oriented Search. In *Second Conference on Language Modeling*.

[117] Xuefei Ning, Zinan Lin, Zixuan Zhou, Zifu Wang, Huazhong Yang, and Yu Wang. 2024. Skeleton-of-thought: Prompting LLMs for efficient parallel generation. In *Proceedings 12th international conference on learning representations-ICLR 2024*. openreview. net.

[118] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35 (2022), 27730–27744.

[119] Siru Ouyang, Wenhao Yu, Kaixin Ma, Zilin Xiao, Zhihan Zhang, Mengzhao Jia, Jiawei Han, Hongming Zhang, and Dong Yu. 2025. RepoGraph: Enhancing AI Software Engineering with Repository-level Code Graph. In *The Thirteenth International Conference on Learning Representations*.

[120] Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering* 36, 7 (2024), 3580–3599.

[121] Shilong Pan, Zhiliang Tian, Liang Ding, Haoqi Zheng, Zhen Huang, Zhihua Wen, and Dongsheng Li. 2024. POMP: Probability-driven meta-graph prompter for LLMs in low-resource unsupervised neural machine translation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 9976–9992.

[122] Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. 2024. Graph retrieval-augmented generation: A survey. *ACM Transactions on Information Systems* (2024).

[123] Huy N Phan, Hoang N Phan, Tien N Nguyen, and Nghi DQ Bui. 2025. Repohyper: Search-expand-refine on semantic graphs for repository-level code completion. In *2025 IEEE/ACM Second International Conference on AI Foundation Models and Software Engineering (Forge)*. IEEE, 14–25.

[124] Gerard Pons, Besim Bilalli, and Anna Queralt. 2024. Knowledge graphs for enhancing large language models in entity disambiguation. In *International Semantic Web Conference*. Springer, 162–179.

[125] Tyler Thomas Procko and Omar Ochoa. 2024. Graph retrieval-augmented generation for large language models: A survey. In *2024 Conference on AI, science, engineering, and technology (AIxSET)*. IEEE, 166–169.

[126] Chen Qian, Zihao Xie, YiFei Wang, Wei Liu, Kunlun Zhu, Hanchen Xia, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, et al. 2024. Scaling Large Language Model-based Multi-Agent Collaboration. In *The Thirteenth International Conference on Learning Representations*.

[127] Zhangchi Qiu, Linhao Luo, Shirui Pan, and Alan Wee-Chung Liew. 2025. Reasoning Over User Preferences: Knowledge Graph-Augmented LLMs for Explainable Conversational Recommendations. In *2025 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1495–1504.

[128] Mohamed Rashad, Ahmed Zahran, Abanoub Amin, Amr Abdelaal, and Mohamed AlTantawy. 2024. FactAlign: Fact-level hallucination detection and classification through knowledge graph alignment. In *Proceedings of the 4th Workshop on Trustworthy Natural Language Processing (TrustNLP 2024)*. 79–84.

[129] Xubin Ren, Jiabin Tang, Dawei Yin, Nitesh Chawla, and Chao Huang. 2024. A survey of large language models for graphs. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 6616–6626.

[130] Md Sadman Sakib and Yu Sun. 2024. Consolidating trees of robotic plans generated using large language models to improve reliability. *International Journal of Artificial Intelligence and Robotics Research* 1, 01 (2024), 2450002.

[131] Muhammad Salman, Haoting Chen, Sergio José Rodríguez Méndez, and Armin Haller. 2024. An LLM-SPARQL Hybrid Framework for Named Entity Linking and Disambiguation to Wikidata. In *China Conference on Knowledge Graph and Semantic Computing*. Springer, 210–226.

[132] Hannah Joelle Sansford, Nicholas Richardson, Hermina Petric Maretic, and Juba Nait Saada. 2024. GraphEval: A Knowledge-Graph Based LLM Hallucination Evaluation Framework. In *Fourth Workshop on Knowledge-infused Learning*.

[133] Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D Manning. 2024. Raptor: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations*.

[134] Albert Sawczyn, Jakub Binkowski, Denis Janiak, Bogdan Gabrys, and Tomasz Kajdanowicz. 2026. FactSelfCheck: Fact-Level Black-Box Hallucination Detection for LLMs. In *Findings of the Association for Computational Linguistics: EACL 2026*.

[135] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2008. The graph neural network model. *IEEE transactions on neural networks* 20, 1 (2008), 61–80.

[136] Wenbo Shang and Xin Huang. 2025. A survey of large language models on generative graph analytics: Query, learning, and applications. *IEEE Transactions on Knowledge and Data Engineering* (2025).

[137] Shan Shen, Shenglu Hua, Jiajun Zou, Jiawei Liu, Jianwang Zhai, Chuan Shi, and Wenjian Yu. 2025. Transferable Parasitic Estimation via Graph Contrastive Learning and Label Rebalancing in AMS Circuits. In *2025 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. IEEE, 1–9.

[138] Juanming Shi, Qinglang Guo, Yong Liao, Yuxing Wang, Shijia Chen, and Shenglin Liang. 2024. Legal-lm: Knowledge graph enhanced large language models for law consulting. In *International conference on intelligent computing*. Springer, 175–186.

[139] Karthik Soman, Peter W Rose, John H Morris, Rabia E Akbas, Brett Smith, Braian Peetoom, Catalina Villouta-Reyes, Gabriel Cerono, Yongmei Shi, Angela Rizk-Jackson, et al. 2024. Biomedical knowledge graph-optimized prompt generation for large language models. *Bioinformatics* 40, 9 (2024), btae560.

[140] Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. 2024. Think-on-Graph: Deep and Responsible Reasoning of Large Language Model on Knowledge Graph. In *The Twelfth International Conference on Learning Representations*.

[141] Hongyuan Tao, Ying Zhang, Zhenhao Tang, Hongen Peng, Xukun Zhu, Bingchang Liu, Yingguang Yang, Ziyin Zhang, Zhaogui Xu, Haipeng Zhang, et al. 2025. Code Graph Model (CGM): A Graph-Integrated Large Language Model for Repository-Level Software Engineering Tasks.

Advances in Neural Information Processing Systems (2025).

[142] Wenbiao Tao, Hanlun Zhu, Keren Tan, Jiani Wang, Yuanyuan Liang, Huihui Jiang, Pengcheng Yuan, and Yunshi Lan. 2024. Finqa: A training-free dynamic knowledge graph question answering system in finance with llm-based revision. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 418–423.

[143] Shiyu Tian, Yangyang Luo, Tianze Xu, Caixia Yuan, Huixing Jiang, Chen Wei, and Xiaojie Wang. 2024. KG-adapter: Enabling knowledge graph integration in large language models through parameter-efficient fine-tuning. In *Findings of the Association for Computational Linguistics: ACL 2024*. 3813–3828.

[144] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[145] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[146] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

[147] FeiLong Wang, Donghui Shi, Jose Aguilar, Xinyi Cui, Jinsong Jiang, Longjian Shen, and Mengya Li. 2025. LLM-KGMQA: large language model-augmented multi-hop question-answering system based on knowledge graph in medical field. *Knowledge and Information Systems* 67, 8 (2025), 6461–6503.

[148] Guixia Wang, Zehui Zhan, and Shouyuan Qin. 2025. Synergizing Knowledge Graphs and LLMs: An Intelligent Tutoring Model for Self-Directed Learning. *Education Sciences* 15, 9 (2025), 1102.

[149] Mingjun Wang, Bin Sun, Jianan Mu, Feng Gu, Boyu Han, Tianmeng Yang, Xinyu Zhang, Silin Liu, Yihan Wen, Hui Wang, et al. 2025. MOSS: Multi-Modal Representation Learning on Sequential Circuits. In *2025 62nd ACM/IEEE Design Automation Conference (DAC)*. IEEE, 1–7.

[150] Shuhe Wang, Yuxian Meng, Rongbin Ouyang, Jiwei Li, Tianwei Zhang, Lingjuan Lyu, and Guoyin Wang. 2023. Gnn-sl: Sequence labeling based on nearest examples via gnn. In *Findings of the Association for Computational Linguistics: ACL 2023*. 12679–12692.

[151] Xixi Wang, Miguel Costa, Jordanka Kovaceva, Shuai Wang, and Francisco C Pereira. 2025. Plugging schema graph into multi-table qa: A human-guided framework for reducing llm reliance. In *Findings of the Association for Computational Linguistics: EMNLP 2025*. 5829–5842.

[152] Xinfeng Wang, Jin Cui, Fumiyo Fukumoto, and Yoshimi Suzuki. 2024. Enhancing high-order interaction awareness in llm-based recommender model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. 11696–11711.

[153] Zehong Wang, Chuxu Zhang, Jundong Li, Nitesh Chawla, and Yanfang Ye. 2025. Graph Foundation Models: Challenges, Methods, and Open Questions. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 6184–6194.

[154] Harsh Homdeo Wasnik and Tushar Shrivastava. 2025. Adaptive Machine Learning Frameworks Integrating LLMs and Knowledge Graphs for Tailored Education. In *2025 8th International Conference on Circuit, Power & Computing Technologies (ICCPCT)*. IEEE, 1509–1514.

[155] Stanley Wasserman and Katherine Faust. 1994. Social network analysis: Methods and applications. (1994).

[156] Hui Wei, Zihao Zhang, Shenghua He, Tian Xia, Shijia Pan, and Fei Liu. 2025. Plangenllms: A modern survey of llm planning capabilities. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 19497–19521.

[157] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2022. Finetuned Language Models are Zero-Shot Learners. In *International Conference on Learning Representations*.

[158] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* 35 (2022), 24824–24837.

[159] Lindsey Linxi Wei, Guorui Xiao, and Magdalena Balazinska. 2024. RACOON: An LLM-based Framework for Retrieval-Augmented Column Type Annotation with a Knowledge Graph. In *NeurIPS 2024 Third Table Representation Learning Workshop*.

[160] Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. Llmrec: Large language models with graph augmentation for recommendation. In *Proceedings of the 17th ACM international conference on web search and data mining*. 806–815.

[161] Yilin Wen, Zifeng Wang, and Jimeng Sun. 2024. Mindmap: Knowledge graph prompting sparks graph of thoughts in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 10370–10388.

[162] Junde Wu, Jiayuan Zhu, Yunli Qi, Jingkun Chen, Min Xu, Filippo Menolascina, Yueming Jin, and Vicente Grau. 2025. Medical graph rag: Evidence-based medical large language model via graph retrieval-augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 28443–28467.

[163] Likang Wu, Zhaopeng Qiu, Zhi Zheng, Hengshu Zhu, and Enhong Chen. 2024. Exploring large language model for graph data understanding in online job recommendations. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 38. 9178–9186.

[164] Tongtong Wu, Yanming Li, Ziye Tang, Chen Jiang, Linhao Luo, Guilin Qi, Shirui Pan, and Gholamreza Haffari. 2026. CARD: Towards Conditional Design of Multi-agent Topological Structures. In *The Fourteenth International Conference on Learning Representations*.

[165] Xixi Wu, Yifei Shen, Caihua Shan, Kaitao Song, Siwei Wang, Bohang Zhang, Jiarui Feng, Hong Cheng, Wei Chen, Yun Xiong, et al. 2024. Can graph learning improve planning in LLM-based agents? *Advances in Neural Information Processing Systems* 37 (2024), 5338–5383.

[166] Yangbin Xia and Xudong Luo. 2024. Legal Judgment Prediction with LLM and Graph Contrastive Learning Networks. In *Proceedings of the 2024 8th International Conference on Computer Science and Artificial Intelligence*. 424–432.

[167] Siheng Xiong, Ali Payani, Ramana Kompella, and Faramarz Fekri. 2024. Large language models can learn temporal reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 10452–10470.

[168] Zhen Xiong, Yujun Cai, Zhecheng Li, and Yiwei Wang. 2025. Mapping the Minds of LLMs: A Graph-Based Analysis of Reasoning LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. 17762–17774.

[169] Ran Xu, Patrick Jiang, Linhao Luo, Cao Xiao, Adam Cross, Shirui Pan, Jimeng Sun, and Carl Yang. 2025. A survey on unifying large language models and knowledge graphs for biomedicine and healthcare. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 6195–6205.

[170] Yuanjian Xu, Jianing Hao, Kunsheng Tang, Jingnan Chen, Anxian Liu, Peng Liu, and Guang Zhang. 2025. Finripple: Aligning large language models with financial market for event ripple effect awareness. In *Findings of the Association for Computational Linguistics: ACL 2025*. 9377–9398.

[171] Cheng Yang, Jiawei Liu, and Chuan Shi. 2021. Extract the knowledge of graph neural networks and go beyond it: An effective knowledge distillation framework. In *Proceedings of the web conference 2021*. 1227–1237.

[172] Junhan Yang, Zheng Liu, Shitao Xiao, Chaozhuo Li, Defu Lian, Sanjay Agrawal, Amit Singh, Guangzhong Sun, and Xing Xie. 2021. Graphformers: Gnn-nested transformers for representation learning on textual graph. *Advances in Neural Information Processing Systems* 34 (2021), 28798–28810.

[173] Yingxuan Yang, Huacan Chai, Shuai Shao, Yuanyi Song, Siyuan Qi, Renting Rui, and Weinan Zhang. 2025. AgentNet: Decentralized Evolutionary Coordination for LLM-based Multi-Agent Systems. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[174] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems* 36 (2023), 11809–11822.

[175] Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th annual meeting of the association for computational linguistics*. 8413–8426.

[176] Yuxin You, Zhen Liu, Xiangchao Wen, Yongtao Zhang, and Wei Ai. 2025. Large language models meet graph neural networks: a perspective of graph mining. *Mathematics* 13, 7 (2025), 1147.

[177] K Yu, Shigeru Ishikura, Yukari Usukura, Yuki Shigoku, and Teruaki Hayashi. 2025. KGSynX: Knowledge Graph and Explainable Feedback Guided LLMs for Synthetic Tabular Data Generation. ISWC.

[178] Shuzhou Yuan, Ercong Nie, Michael Färber, Helmut Schmid, and Hinrich Schütze. 2024. Gnnavi: Navigating the information flow in large language models by graph neural network. In *Findings of the Association for Computational Linguistics: ACL 2024*. 3987–4001.

[179] Yixuan Yuan and Bixuan Li. 2025. SKG-LLM: Enhancing Large Language Models with Sentiment Knowledge Graphs for Fine-Grained Sentiment Analysis. In *International Conference on Knowledge Science, Engineering and Management*. Springer, 282–290.

[180] Gui-Huang Zeng, Yan-Qin Liu, Chun-Yang Zhang, Hai-Chun Cai, and CL Philip Chen. 2024. Adaptive Multi-Document Summarization Via Graph Representation Learning. *IEEE Transactions on Cognitive and Developmental Systems* (2024).

[181] Xinji Zha, Yumin Dong, Haomiao Jiang, Zihui Xu, and Chao Wang. 2025. LLM-AGR: Large Language Model Augmented Graph Representation Learning for Recommendation. *Knowledge-Based Systems* (2025), 114791.

[182] Haoxing Zhai. 2025. Law GraphRAG: An Advanced Legal Question-Answering System. In *2025 5th International Conference on Artificial Intelligence and Industrial Technology Applications (AIITA)*. IEEE, 1407–1410.

[183] Guibin Zhang, Yanwei Yue, Zhixun Li, Sukwon Yun, Guancheng Wan, Kun Wang, Dawei Cheng, Jeffrey Xu Yu, and Tianlong Chen. 2024. Cut the Crap: An Economical Communication Pipeline for LLM-based Multi-Agent Systems. In *The Thirteenth International Conference on Learning Representations*.

[184] Haozhen Zhang, Tao Feng, and Jiaxuan You. 2025. Graph of records: Boosting retrieval augmented generation for long-context summarization with graphs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 23780–23799.

[185] Jinghan Zhang, Xiting Wang, Weijieying Ren, Lu Jiang, Dongjie Wang, and Kunpeng Liu. 2025. Ratt: A thought structure for coherent and correct llm reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 26733–26741.

[186] Niansong Zhang, Chenhui Deng, Johannes Maximilian Kuehn, Chia-Tung Ho, Cunxi Yu, Zhiru Zhang, and Haoxing Ren. 2025. ASPEN: LLM-Guided E-Graph Rewriting for RTL Datapath Optimization. In *2025 ACM/IEEE 7th Symposium on Machine Learning for CAD (MLCAD)*. IEEE, 1–9.

[187] Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. 2022. GreaseLM: Graph REASoning Enhanced Language Models. In *International Conference on Learning Representations*.

[188] Yichi Zhang, Zhuo Chen, Lingbing Guo, Yajing Xu, Wen Zhang, and Huajun Chen. 2024. Making large language models perform better in knowledge graph completion. In *Proceedings of the 32nd ACM international conference on multimedia*. 233–242.

[189] Yuanshuo Zhang, Yuchen Hou, Bohan Tang, Shuo Chen, Muhan Zhang, Xiaowen Dong, and Siheng Chen. 2025. FLORA: GNNs as Predictors of Agentic Workflow Performances. In *The Fourth Learning on Graphs Conference*.

[190] Yuanhe Zhang, Ilja Kuzborskij, Jason D. Lee, Chenlei Leng, and Fanghui Liu. 2026. DAG-Math: Graph-Guided Mathematical Reasoning in LLMs. In *The Fourteenth International Conference on Learning Representations*.

[191] Yifan Zhang, Jingqin Yang, Yang Yuan, and Andrew C Yao. 2025. Cumulative Reasoning with Large Language Models. *Transactions on Machine Learning Research* (2025).

[192] Zheyuan Zhang, Zehong Wang, Shifu Hou, Evan Hall, Landon Bachman, Jasmine White, Vincent Galassi, Nitesh V Chawla, Chuxu Zhang, and Yanfang Ye. 2024. Diet-odin: A novel framework for opioid misuse detection with interpretable dietary patterns. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 6312–6323.

[193] Zhiqiang Zhang, Liqiang Wen, and Wen Zhao. 2024. A gail fine-tuned llm enhanced framework for low-resource knowledge graph question answering. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management.* 3300–3309.

[194] Ziyin Zhang, Hang Yu, Sage Lee, Peng Di, Jianguo Li, and Rui Wang. 2025. Galla: Graph aligned large language models for improved source code understanding. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 13784–13802.

[195] Jie Zhao, Wanting Ning, Yuxiao Fei, Yubo Feng, and Lishuang Li. 2025. Gdllm: A global distance-aware modeling approach based on large language models for event temporal relation extraction. In *Findings of the Association for Computational Linguistics: EMNLP 2025*.

[196] Mingyu Zhao, Xun He, Jiawei Liu, Jianwang Zhai, and Chuan Shi. 2026. Topology Matters in RTL Circuit Representation Learning. In *The Fourteenth International Conference on Learning Representations (ICLR)*.

[197] Mingyu Zhao, Jiawei Liu, Jianwang Zhai, and Chuan Shi. 2025. MILS: Modality Interaction Driven Learning for Logic Synthesis. In *Proceedings of the Great Lakes Symposium on VLSI 2025 (GLSVLSI)*. 64–70.

[198] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* 1, 2 (2023), 1–124.

[199] Yushang Zhao, Yike Peng, Dannier Li, Yuxin Yang, Chengrui Zhou, and Jing Dong. 2025. Research on Personalized Financial Product Recommendation by Integrating Large Language Models and Graph Neural Networks. In *Proceedings of the 2025 International Conference on Software Engineering and Computer Applications*. 53–59.

[200] Junhao Zheng, Haibin Chen, and Qianli Ma. 2022. Cross-domain named entity recognition via graph matching. In *Findings of the Association for Computational Linguistics: ACL 2022*. 2670–2680.

[201] Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2024. Language agent tree search unifies reasoning, acting, and planning in language models. In *Proceedings of the 41st International Conference on Machine Learning*. 62138–62160.

[202] Shasha Zhou, Mingyu Huang, Jack Cole, Charles Britton, Ming Yin, Jan Wolber, and Ke Li. 2026. Assessing Automated Fact-Checking for Medical LLM Responses with Knowledge Graphs. In *Proceedings of the AAAI conference on artificial intelligence*.

[203] Xinyan Zhu, Cheng Yang, Qiuyu Wang, Zeyuan Guo, Yiding Wang, Zedi Liu, Chunchen Wang, and Chuan Shi. 2026. Graph4LLM: A Systematic Survey of Graph-Enhanced Large Language Models. *Authorea Preprints* (2026).

[204] Zulun Zhu, Tiancheng Huang, Kai Wang, Junda Ye, Xinghe Chen, and Siqiang Luo. 2025. Graph-based approaches and functionalities in retrieval-augmented generation: A comprehensive survey. *Comput. Surveys* (2025).

[205] Luyao Zhuang, Shengyuan Chen, Yilin Xiao, Huachi Zhou, Yujing Zhang, Hao Chen, Qinggang Zhang, and Xiao Huang. 2026. LinearRAG: Linear Graph Retrieval Augmented Generation on Large-scale Corpora. In *The Fourteenth International Conference on Learning Representations*.

[206] Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. 2024. Gptswarm: Language agents as optimizable graphs. In *Forty-first International Conference on Machine Learning*.

[207] Xinhao Zou and Konstantin Markov. 2024. Combining Graph NN and LLM for Improved Text-Based Emotion Recognition. In *International Conference on Artificial Intelligence: Methodology, Systems, and Applications*. Springer, 143–154.

[208] Alejandro Michel Zuniga, Abhinav Arun, Bhaskarjit Sarmah, and Stefano Pasquali. 2026. FinCARE: Financial Causal Analysis with Reasoning & Evidence. In *NeurIPS 2025 Workshop: Generative AI in Finance*.