UniGFM-RAG: Unify Graph Foundation Models and Large Language Models for End-to-End Retrieval and Reasoning

Zicheng Zhao*, Linhao Luo, Zongji Yu, Junnan Dong, Yanchao Tan,
Shirui Pan†, *Senior Member, IEEE*, and Chen Gong†, *Senior Member, IEEE*

***Abstract*—Large Language Models (LLMs) have shown remarkable capabilities across tasks, but they still struggle with complex reasoning that requires integrating scattered knowledge. Graph Retrieval-Augmented Generation (GraphRAG) addresses this challenge by organizing fragment knowledge as graphs, while recent Graph Foundation Models (GFMs) further improve GraphRAG with graph-native retrieval and reasoning capabilities. However, existing GraphRAG methods usually adopt decoupled retrieve-then-generate pipelines, which create objective misalignment between retrieval and reasoning, and training GFM retrievers typically depends on expensive human annotations. To address these limitations, we propose UniGFM-RAG, a unified end-to-end GraphRAG framework that tightly couples a GFM retriever with an LLM generator. Given a query, the GFM retriever first reasons over the graph to score and retrieve relevant document nodes. These node embeddings are then passed through a fully differentiable graph bridge, which projects them into soft-weighted graph prefixes according to retrieval scores and combines them with raw document texts to form graph-augmented prompts. In this way, structural information is injected directly into LLM generation, while the final answer loss can be back-propagated through the bridge to jointly optimize retrieval and reasoning. To stabilize training and reduce reliance on costly annotations, we further introduce a semantic-guided distillation module that provides auxiliary supervision using pseudo-labels derived from query-document semantic similarities. Extensive experiments on three multi-hop QA datasets and three out-of-domain GraphRAG benchmarks show that UniGFM-RAG consistently outperforms strong baseline methods and generalizes well, which demonstrates the effectiveness of end-to-end GraphRAG with tightly integrated GFM retrieval and LLM reasoning.**

Manuscript received March 28, 2026; revised October 25, 2026; accepted November 3, 2026. This research is supported by NSF of China (Grant Nos: 62336003, 12371510), and the Shanghai Municipal Special Program for Basic Research on General AI Foundation Models (Grant No. 2025SHZDZX025G12).

*This work was done while visiting Shanghai Jiao Tong University.

Z. Zhao, Z. Yu and C. Gong are with the School of Automation and Intelligent Sensing, Shanghai Jiao Tong University, Shanghai 200240, China. Email: {zicheng.zhao99@gmail.com, yuzongji@pjlab.org.cn, chen.gong@sjtu.edu.cn.}

L. Luo is with the Department of Data Science and AI, Monash University, Melbourne, Australia. E-mail: linhao.luo1@monash.edu.

J. Dong is with the Tencent Youtu Lab, Shanghai 200240, China. E-mail: hansonjdong@tencent.com.

Y. Tan is with School of Computer and Data Science, Fuzhou University, Fuzhou 350116, China. Email: yctan@fzu.edu.cn.

S. Pan is with the School of Information and Communication Technology, Griffith University, Queensland, Australia. Email: s.pan@griffith.edu.au.

†Corresponding Author.

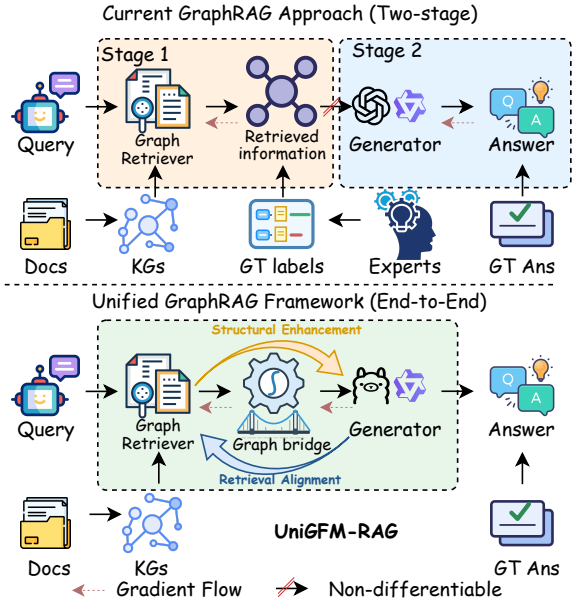


Fig. 1: Comparison between the current two-stage GraphRAG and the proposed UniGFM-RAG. Existing GraphRAG methods usually employ a decoupled retrieve-then-generate pipeline, which often results in objective misalignment. In contrast, UniGFM-RAG introduces a unified end-to-end GraphRAG framework that tightly couples a Graph Foundation Model (GFM) retriever with a Large Language Model (LLM) generator.

***Index Terms*—Large Language Models, Graph Retrieval-Augmented Generation, Graph Foundation Models**

I. Introduction

Large Language Models (LLMs) have demonstrated remarkable performance across a wide range of natural language processing tasks [1]–[5]. However, they still struggle with knowledge-intensive reasoning, where the evidence is often distributed across multiple documents [6]. Graphs [7]–[10] offer a natural and flexible way of modeling structural information, which makes them particularly suitable for capturing the dependencies within knowledge for complex reasoning. Motivated by this, Graph Retrieval-Augmented Generation (GraphRAG) [11]–[14] has emerged as a promising paradigm for enhancing LLM reasoning with graph-structured knowledge. As illustrated in the upper part of Figure 1, a typical GraphRAG system first constructs a graph from the external

corpus, such as a knowledge graph [15] or a document graph [16], where nodes correspond to documents or entities, and edges encode their relationships. It then employs a graph retriever to navigate the graph and identify relevant nodes, whose associated information is provided to an LLM generator to produce the final answer.

Early graph retrievers relied on the internal capabilities of LLMs to traverse graphs using heuristic search [17], global planning [18], or multi-step agentic reasoning [19], [20]. However, these approaches often suffer from inefficiency and limited structural understanding due to the unstructured nature of LLMs. Recently, Graph Foundation Models (GFMs) [21]–[24] have demonstrated strong graph reasoning and generalized abilities, which inspires a new wave of applying them to GraphRAG [25]. These methods utilize graph neural network (GNN)-based GFMs as native graph retrievers to capture intricate dependencies among nodes, which significantly enhances retrieval performance. Despite the promising results, *training GFMs requires large-scale, high-quality relevance labels, which are labor-intensive and difficult to obtain at scale* (**Limitation 1**).

Beyond the challenge of data acquisition, the architecture of existing GraphRAG frameworks poses another major bottleneck. Most GraphRAG methods adopt a decoupled, two-stage "retrieve-then-generate" pipeline [25]–[27]. In this paradigm, the retriever and generator are optimized separately for different objectives: the retriever is trained to maximize retrieval performance rather than improve final answer accuracy. This leads to significant objective misalignment, where *information considered highly relevant by the retriever may be suboptimal or redundant for the reasoning process of the generator* (**Limitation 2**). Furthermore, because document retrieval is a discrete operation, it prevents continuous gradient backpropagation from the generator to the retriever, which makes joint training even more difficult. Therefore, it is crucial to tightly integrate the GFM retriever and LLM generator into a unified framework that allows end-to-end optimization.

Beyond enabling joint optimization, a unified GFM-LLM framework can further improve the ability of LLM to leverage graph structure during reasoning [28]. Prior work has attempted to inject graph structural signals into LLMs through GNN-based graph encoders and graph-aware prompting [29]–[31]. Although effective for standalone graph tasks such as anomaly detection [32], node classification and link prediction [33], these methods assume a pre-defined graph as input and are not designed for retrieval-augmented settings. As a result, they *neither address how to retrieve query-relevant information from external graph nor capture the interaction between graph retrieval and answer generation* (**Limitation 3**), which is essential in GraphRAG.

In response to these challenges, we propose `UniGFM-RAG`, a unified end-to-end GraphRAG framework that tightly couples a GFM retriever with an LLM generator for joint retrieval and reasoning. As shown in the bottom of Figure 1, `UniGFM-RAG` fosters a reciprocal loop: the structural insights of GFM enhance LLM reasoning, while LLM reasoning refines graph retrieval. Given a query, the GFM retriever first reasons over the graph to score and retrieve top-k relevant document nodes. Instead of directly feeding the retrieved documents into the LLM, we introduce a novel *graph bridge* module to connect the GFM and LLM through a fully differentiable architecture. Specifically, the node embeddings of retrieved documents are extracted from the GFM and projected into "graph prefixes" that encode their structural roles and inter-document dependencies, and are then soft-weighted by the corresponding retrieval scores. These graph prefixes are concatenated with the raw document texts to form graph-augmented prompts for the LLM. In this way, the structural knowledge captured by the GFM is injected directly into the generation process, which enables the LLM to reason not only over the retrieved contents but also over their graph-structured dependencies (**Addressing Limitation 3**).

More importantly, the graph bridge remains fully differentiable, which allows gradients from the final answer generation loss to flow back through the graph prefixes and soft weights to refine both the LLM generator and GFM retriever. This directly aligns retrieval with downstream reasoning objectives and enables joint optimization of the two modules (**Addressing Limitation 2**). To further stabilize training, we introduce a *semantic-guided distillation* module that provides auxiliary supervision for the GFM retriever using pseudo-labels derived from query-document semantic similarities. This module helps the retriever learn robust topological navigation from semantic priors without requiring expensive fine-grained human annotations (**Addressing Limitation 1**).

The core contributions of this work are summarized as follows:

- We propose `UniGFM-RAG`, a unified end-to-end GraphRAG framework that unifies a GFM retriever with an LLM generator, which enables joint optimization of retrieval and reasoning instead of the conventional decoupled retrieve-then-generate pipeline.
- We design a differentiable graph bridge that transforms retrieved document nodes into soft-weighted graph prefixes and graph-augmented prompts, which injects graph structure into the LLM while allowing answer supervision to back-propagate to the GFM retriever.
- We introduce a semantic-guided distillation module that stabilizes GFM training with pseudo-labels derived from query-document semantic similarity, which reduces reliance on costly fine-grained human annotations.
- Extensive experiments on three multi-hop QA datasets and three out-of-domain GraphRAG benchmarks show that `UniGFM-RAG` consistently outperforms strong baselines and generalizes effectively across domains, which demonstrate the effectiveness of our unified framework.

II. Related Work

In this section, we review some representative prior works for Graph Retrieval Augmented Generation (GraphRAG) and Graph Foundation Models (GFMs).

A. Graph Retrieval Augmented Generation

Graph retrieval-augmented generation has emerged as a promising paradigm that retrieves query-relevant graph ele-

ments (e.g., node, triples, paths, or subgraphs) from a pre-constructed graph database (e.g., knowledge graphs [34], [35]). By explicitly modeling the inherent logical relationships among isolated pieces of knowledge, these methods significantly enhance the retrieval process beyond standard document retrieval. Recent approaches further integrate textual documents into knowledge graphs to facilitate more structured and semantically rich retrieval. For example, GraphRAG [15] employs community detection algorithms to summarize and organize documents, while LightRAG [36] introduces a dual-level retrieval mechanism that combines structural graph retrieval with semantic vector retrieval to efficiently locate entities and their relations. Similarly, HippoRAG [17] utilizes the Personalized PageRank (PPR) algorithm to identify salient graph entities and subsequently maps them to relevant documents. However, constructing these graphs is often computationally expensive and token-intensive, and the process can introduce noise, which limits the practical deployment of these methods in real-world scenarios. Concurrently, efforts to incorporate Graph Neural Networks (GNNs) into graph-enhanced RAG [37], [38] have yielded expressive results, largely due to the multi-hop graph reasoning capabilities of GNNs over incomplete graphs [39]–[41]. Nevertheless, these methods still struggle to generalize across diverse domains due to the absence of a unified graph foundation model.

TABLE I: Glossary of commonly used notations.

Notation	Description
Knowledge Graph & Query	
$\mathcal{G}$	The heterogeneous knowledge graph
$v \in \mathcal{V}$	A generic node v in the node set $\mathcal{V}$
$r \in \mathcal{R}$	A relation type r in the relation set $\mathcal{R}$
$(v, r, v') \in \mathcal{E}$	An edge (v, r, v') in the edge set $\mathcal{E}$
$t_v \in \mathcal{T}$	The node type of v, belonging to the type set $\mathcal{T}$
q, a	The user query and the final answer
$d \in \mathcal{D}$	A document d within the document subset $\mathcal{D} \subset \mathcal{V}$
Representations	
D_{gnn}	The hidden dimensions of the GNN embeddings
D_{llm}	The hidden dimensions of the LLM embeddings
L	The total number of GNN layers
$\mathbf{h}_q, \mathbf{h}_d$	Initial text embeddings of the query q and document d
$\mathbf{h}_v^{(l)}$	The embedding of node v at the l-th GNN layer
$\mathbf{m}_v^{(l)}$	The aggregated message received by node v at the l-th layer
$\mathbf{h}_d^{(L)}$	The final graph embedding of document node
$\tilde{\mathbf{h}}_d$	The structural prefix token projected from the graph embedding
$\mathbf{V}_d$	Dense token embedding sequence of document d
$\hat{\mathbf{V}}_d$	The soft-weighted, graph-augmented prompt representation for d
Retrieval & Distillation	
$\mathcal{D}_k$	The subset of the top-k retrieved document nodes
s_d	The predicted discrete relevance score for document d
w_d	The continuous soft weight for document d, $w_d \in (0, 1)$
p_t	The semantic pseudo-label distribution generated by the teacher
p_θ	The predicted retrieval distribution by the GFM student
Optimization	
λ	Hyperparameter balancing the generation and distillation losses
θ, ϕ	Trainable parameters of the GFM retriever and LLM generator
$\mathcal{L}_1$	The negative log-likelihood (NLL) generation loss of the LLM
$\mathcal{L}_2$	The semantic-guided KL divergence distillation loss
$\mathcal{L}$	The overall optimization objective

B. Graph Foundation Models

Graph Foundation Models (GFMs) leverage large-scale pre-training across massive datasets to create unified models with strong generalizability across various downstream tasks [21], [22], [42]. A primary challenge lies in defining graph tokens capable of capturing invariance across diverse and heterogeneous graph data [43]. For example, ULTRA [44] designs four fundamental relational interactions in knowledge graphs (KGs) to create a lightweight GFM with 0.2M parameters for link prediction. OpenGraph [45] develops a graph tokenizer to convert graphs into a unified node token representation, which enables transformer-like GFMs for tasks such as link prediction and node classification. GFT [46] introduces a transferable tree vocabulary to conduct a GFM that demonstrates effectiveness across diverse domains in graph learning. Despite these successful efforts, most methods primarily focus on conventional graph-related tasks and Transformer-like GFMs [29], [47] often struggle to scale to large graphs.

To combine the power of GFM and LLM, GraphLLM [30] integrates graph learning models with LLM to equip LLM with the ability to reason on graph data. GraphGPT [29] also incorporates a text-graph grounding component to establish connections between text and graph structures for LLMs. GraphTranslator [48] bridges GFM and LLM with a translator, aiming to transfer the node representations into tokens, based on which LLM can perform node classification and link prediction. Beyond conventional graph learning tasks, GFM-RAG [25] adopts a GFM-based retriever for complex question answering. However, this method relies heavily on the ground-truth query-document relevance labels for training the retriever and employs a disjoint two-stage pipeline. This inherently limits end-to-end optimization and often leads to sub-optimal reasoning performance. Therefore, designing a joint framework that seamlessly integrates a GFM-based retriever with a generator remains an open and critical question.

III. Approach

The overall framework of `UniGFM-RAG` is illustrated in Figure 2, which consists of three core components: (a) a *graph foundation model retriever* to effectively reason over the structured knowledge for document retrieval, (b) a *graph bridge module* to connect the retriever and generator into a unified framework, and (c) a *graph-conditioned LLM generator* that effectively incorporates graph information into the LLM generation process and enables end-to-end training. In the following sections, we will detail the formulation and design of each module.

GraphRAG Overview. To contextualize our proposed framework, we begin by formally defining the general GraphRAG process. Given a set of documents $\mathcal{D} = \{d_1, d_2, \cdots, d_n\}$, following common GraphRAG approaches [15], [25], we employ LLMs to extract structural knowledge and construct a knowledge graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R}, \mathcal{T})$. Here, $\mathcal{V}$ and $\mathcal{R}$ denote the sets of nodes and relations, respectively, and $\mathcal{E} = \{(v, r, v') \subseteq \mathcal{V} \times \mathcal{R} \times \mathcal{V}\}$ represents the set of edges that model the relationship among knowledge. Additionally, $\mathcal{T}$ specifies the node types (e.g., entities, documents).

Given the constructed graph $\mathcal{G}$ and a user query q, standard GraphRAG paradigms first retrieve query-relevant information and then leverage an LLM to reason over the retrieved context

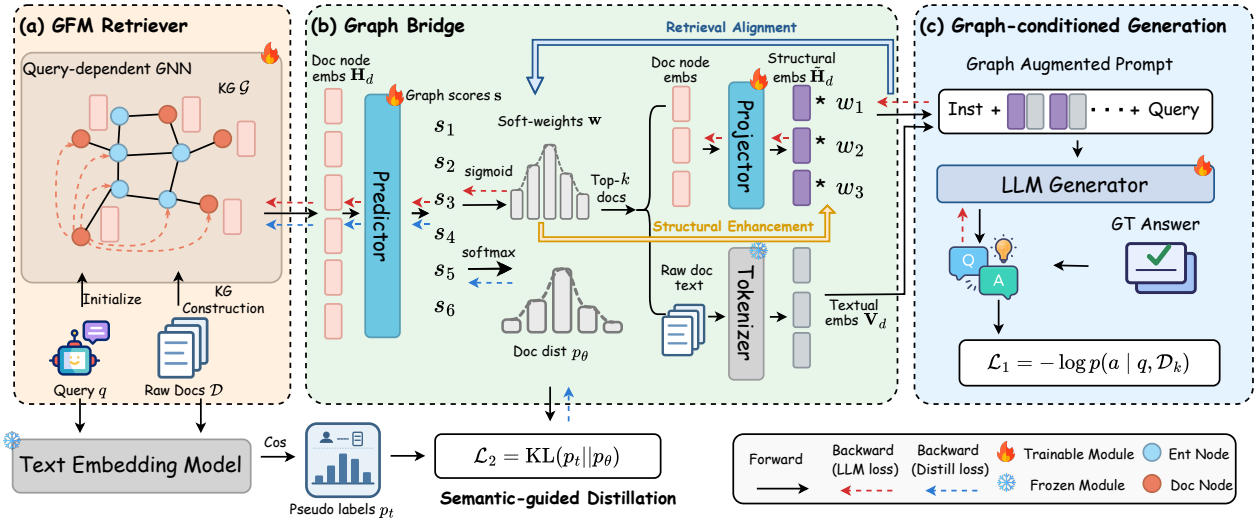


Fig. 2: The overall framework of UniGFM-RAG, which comprises of three core components: (a) *a graph foundation model retriever*, which reasons over the structural knowledge for accurate document retrieval; (b) *a graph bridge module*, which seamlessly connects the retriever and the generator into a unified framework; (c) *a graph-conditioned LLM generator*, which incorporates the structural information to guide the final answer generation. In addition, we introduce a semantic-guided distillation module to provide auxiliary supervision for training GFMs.

and generate the final answer a. Formally, a graph retriever R_θ identifies the top-k query-relevant document nodes $\mathcal{D}_k \subset \mathcal{D}$ according to the predicted probability $p_\theta(\mathcal{D}_k \mid q, \mathcal{G}, \mathcal{D})$, and a generator G_ϕ then produces the final answer a conditioned on both the query and the retrieved documents, which can be formulated as:

$$\mathcal{G} = \mathrm{GraphConstructor}\left(\mathcal{D}\right), \tag{1}$$

$$p_\Theta(a \mid q, \mathcal{G}, \mathcal{D}) = \sum_{\mathcal{D}_k \subseteq \mathcal{D}} p_\phi(a \mid q, \mathcal{D}_k) p_\theta(\mathcal{D}_k \mid q, \mathcal{G}, \mathcal{D}), \tag{2}$$

where $\Theta = \{\theta, \phi\}$ denote the combined parameters of the retriever and generator.

A. Graph Foundation Model Retriever

Graph foundation models (GFMs) [21], [22] have demonstrated strong capabilities in capturing complex topological and semantic relationships within graph-structured data, which makes them ideal candidates for graph-based retrieval tasks. In UniGFM-RAG, we leverage a GFM as the backbone of our retriever to effectively reason over the knowledge graph and identify query-relevant document nodes.

Specifically, we adopt a query-dependent Graph Neural Network (GNN) architecture [25], [44] as the core of our GFM retriever. This design allows the model to explicitly incorporate both relational semantics and rich semantic features in node text into the message-passing process, which enables query-aware graph reasoning.

Text Encoding. Given a graph $\mathcal{G}$, we first utilize a pre-trained text embedding model (e.g., Qwen3-Embedding [49]) to respectively encode the textual content associated with each node $v \in \mathcal{V}$ and each relation $r \in \mathcal{R}$ into initial d-dimensional node embeddings $\mathbf{h}_v \in \mathbb{R}^d$ and relation embeddings $\mathbf{h}_r \in \mathbb{R}^d$, which can be formulated as:

$$\mathbf{h}_v = \mathrm{TextEmbedding}\left(\mathrm{Text}(v)\right), v \in \mathcal{V}, \tag{3}$$

$$\mathbf{h}_r = \mathrm{TextEmbedding}\left(\mathrm{Text}(r)\right), r \in \mathcal{R}, \tag{4}$$

where $\mathrm{Text}(\cdot)$ denotes the textual content associated with the nodes and relations. These semantic embeddings preserve the original textual knowledge, which provides a robust initialization for the subsequent GNN reasoning. Similarly, for a given user query q, we employ the same text embedding model to extract its dense representation $\mathbf{h}_q \in \mathbb{R}^d$, formulated as:

$$\mathbf{h}_q = \mathrm{TextEmbedding}\left(q\right). \tag{5}$$

By sharing the same encoding strategy, the query and graph elements are projected into the same latent semantic space. This approach not only enables generalization across different graphs and queries but also facilitates effective interaction between the query and the graph during GNN reasoning.

Query-dependent GNN Reasoning. After obtaining the initial embeddings, we apply an L-layer query-dependent GNN to jointly reason over text semantics and graph topology. The GNN updates the node representations iteratively by aggregating information from their neighbors, which allows the model to capture intricate dependencies among nodes and produce query-aware relevance scores for document retrieval. Formally, the GNN updates are defined as follows:

$$\mathbf{h}_v^{(0)} = \mathrm{Init}\left(\mathbf{h}_v, 1_{v \in \mathcal{V}_q} \mathbf{h}_q\right), \quad v \in \mathcal{V}, \tag{6}$$

$$\mathbf{m}_v^{(l)} = \mathrm{Agg}\left(\left\{\mathrm{Msg}\left(\mathbf{h}_v^{(l-1)}, \mathbf{h}_r^{(l)}, \mathbf{h}_{v'}^{(l-1)}\right) \mid (v, r, v') \in \mathcal{E}\right\}\right), \tag{7}$$

$$\mathbf{h}_v^{(l)} = \mathrm{Update}\left(\mathbf{h}_v^{(l-1)}, \mathbf{m}_v^{(l)}\right), \quad l \in [1, L], \tag{8}$$

where $\mathbf{h}_v^{(l)}$ denotes the embedding of node v at the l-th GNN layer. The `Init` operator initializes the node embeddings by a single-layer MLP, which fuses the original node embedding $\mathbf{h}_v$ and the query embedding $\mathbf{h}_q$ if the node v belongs to the set of query-related nodes $\mathcal{V}_q$ (i.e., $1_{v\in\mathcal{V}_q}$). Query-related node set $\mathcal{V}_q$ is constructed by semantic retrieval. Specifically, we encode the query and all graph nodes by an embedding model (e.g., Qwen3-Embedding-0.6B), compute their cosine similarities, and select the top-k most similar entity and document nodes as query-related nodes.

During the message-passing phase at each layer l, the `Msg` function employs DistMult [50] to compute the incoming messages from neighbors based on target node embedding $\mathbf{h}_v^{(l-1)}$, the source node embedding $\mathbf{h}_{v'}^{(l-1)}$ and the relation embedding $\mathbf{h}_r^{(l)}$. These messages are subsequently pooled by the `Agg` function (e.g., sum aggregation). Finally, the `Update` function computes the target node embedding $\mathbf{h}_v^{(l)}$ by combining the aggregated messages $\mathbf{m}_v^{(l)}$ with its previous representation $\mathbf{h}_v^{(l-1)}$ with another MLP. Concurrently, relation embeddings $\mathbf{h}_r^{(l)}$ are updated with a layer-specific MLP, defined as $\mathbf{h}_r^{(l)} = \mathrm{MLP}(\mathbf{h}_r^{(l-1)})$.

Document Retrieval. By encoding the textual content of graph nodes and relations into dense embeddings and applying L-layer of GNN, the GFM retriever can capture intricate dependencies among nodes and produce query-aware relevance scores for document retrieval. The final query-aware node representation $\mathbf{h}_v^{(L)}$ is then used to compute a relevance score s_v for document retrieval as follows:

$$s_v = \mathrm{Predictor}\left(\mathbf{h}_v^{(L)}, \mathbf{h}_v, \mathbf{h}_q\right), \tag{9}$$

where $\mathbf{h}_v$ and $\mathbf{h}_q$ denote the original text embeddings of node v and query q, respectively, and the predictor is implemented as a two-layer MLP to produce a relevance score s_v for each document node.

The top-k document nodes with the highest relevance scores are selected as the retrieved documents for subsequent generation. Formally, the retrieved document set $\mathcal{D}_k$ is defined as:

$$\mathcal{D}_k = \text{Top-}k(s_d), d \in \mathcal{D}. \tag{10}$$

However, this hard selection operation is inherently non-differentiable, which prevents end-to-end optimization between the retriever and the generator and prevents the abundant structural information encoded by the GFM from being effectively utilized by the LLM generator.

B. Graph Bridge Module

To construct a differentiable bridge between the GFM retriever and the LLM generator, we introduce a novel *graph bridge* module that transforms the retrieved document nodes into soft-weighted graph prefixes and seamlessly integrates them into the LLM generation process. The graph bridge serves as a continuous interface that allows the structural information from the GFM to be directly injected into the LLM while enabling gradients from the final answer loss to back-propagate to the GFM retriever for joint optimization.

We first relax the discrete retrieval scores s_d into continuous soft-weights w_d by passing them through a `Sigmoid` function, which preserves the independent relevance of each document without forcing competitive normalization. This can be formulated as:

$$w_d = \frac{1}{1+\exp(-s_d)}, d \in \mathcal{D}_k. \tag{11}$$

These differentiable soft-weights explicitly quantify the importance of each retrieved document and allow the retriever to learn which documents maximize the generator's generation performance.

To inject structural information, we reuse the node representations of retrieved documents $\{\mathbf{h}_d^{(L)} \mid d \in \mathcal{D}_k\}$ obtained from the final GNN layer in GFM retriever, which is defined in Eq. (8). These representations encapsulate the rich topological context of the document d within the graph, which will be transformed into the *graph prefixes* to guide the generation process of LLM.

A two-layer MLP (i.e., Projector) is first adopted as an alignment projector to map the structural representation into the semantic space of the LLM. Crucially, to allow the gradient to flow back to the retriever, we scale the projected representations by the differentiable soft-weighted w_d derived via Eq. (11), which is formulated as follows:

$$\tilde{\mathbf{h}}_d = w_d \cdot \mathrm{Projector}\left(\mathbf{h}_d^{(L)}\right), d \in \mathcal{D}_k. \tag{12}$$

This soft-weighted structural representation $\tilde{\mathbf{h}}_d \in \mathbb{R}^{D_{llm}}$ with D_{llm} dimensions serves as a special graph prefix, which explicitly encodes the structural information and relevance of the retrieved document to guide the LLM generation.

C. Graph-conditioned LLM Generator

By integrating the graph prefixes into the LLM generation process, we enable the LLM to effectively leverage the graph structure during reasoning, which significantly enhances its performance on complex tasks that require reasoning over multiple knowledge.

Graph-augmented Prompt Construction: For each retrieved document $d \in \mathcal{D}_k$, we first extract its textual content and tokenize it into a sequence of tokens. These tokens are then mapped to their dense embedding sequence $\mathbf{V}_d \in \mathbb{R}^{|d|\times D_{llm}}$ using the LLM's input embedding layer, where $|d|$ represents the sequence length of the document and D_{llm} denotes the hidden dimension of the LLM. This process captures the raw semantic information of the retrieved documents, which serves as the primary context for generation.

Then, we concatenate the graph prefix $\tilde{\mathbf{h}}_d$ with the document embedding sequence $\mathbf{V}_d$ to form a graph-injected representation $\hat{\mathbf{V}}_d$ for each retrieved document, which is formulated as follows:

$$\hat{\mathbf{V}}_d = \left[\tilde{\mathbf{h}}_d \parallel \mathbf{V}_d\right], d \in \mathcal{D}_k, \hat{\mathbf{V}}_d \in \mathbb{R}^{(|d|+1)\times D_{llm}}, \tag{13}$$

where "$\parallel$" denotes sequence concatenation.

Finally, the formulated representations $\hat{\mathbf{V}}_d$ of all k retrieved documents are concatenated with the initial user query sequence to form the graph-augmented prompt for the LLM generator. The detailed prompt is illustrated in Figure 6 in

Appendix V-C, where the <graph_prefix> denotes the injected graph prefix $\tilde{\mathbf{h}}_d$.

Graph-conditioned Generation: Conditioned on this graph-augmented prompt, the LLM generator auto-regressively decodes the target answer $a = \{a_1, a_2, \ldots, a_{|a|}\}$. The probability of generating the entire answer sequence is factorized as:

$$p_\phi(a \mid q, \mathcal{D}_k) = \prod_{t=1}^{|a|} p_\phi(a_t \mid a_{<t}, q, \hat{\mathbf{V}}_{\mathcal{D}_k}), \tag{14}$$

where $\hat{\mathbf{V}}_{\mathcal{D}_k} = \{\hat{\mathbf{V}}_d\}_{d \in \mathcal{D}_k}$.

End-to-end Joint Optimization: During the training phase, the primary objective for the LLM generator is to minimize the standard negative log-likelihood (NLL) loss, denoted as $\mathcal{L}_1$:

$$\mathcal{L}_1 = -\sum_{t=1}^{|a|} \log p_\phi(a_t \mid a_{<t}, q, \hat{\mathbf{V}}_{\mathcal{D}_k}). \tag{15}$$

The applying graph prefix ensures that the original language modeling capabilities of LLM remain undisturbed, while still establishing a fully differentiable bridge, which allows gradients from the LLM loss to back-propagate to the GFM retriever's parameters θ through the prefix $\tilde{\mathbf{h}}_d$.

Building upon the chain rule, the gradient flow from each retrieved document explicitly splits into dual pathways as follows:

$$\frac{\partial \mathcal{L}_1}{\partial \theta} = \underbrace{\frac{\partial \mathcal{L}_1}{\partial \tilde{\mathbf{h}}_d} \frac{\partial \tilde{\mathbf{h}}_d}{\partial w_d} \frac{\partial w_d}{\partial \theta}}_{\text{Soft-weighting Pathway}} + \underbrace{\frac{\partial \mathcal{L}_1}{\partial \tilde{\mathbf{h}}_d} \frac{\partial \tilde{\mathbf{h}}_d}{\partial \mathbf{h}_d^{(L)}} \frac{\partial \mathbf{h}_d^{(L)}}{\partial \theta}}_{\text{Structural Pathway}}, d \in \mathcal{D}_k, \tag{16}$$

where the two terms on the right-hand side denote the back-propagated gradient through the continuous weight w_d and the structured embedding $\mathbf{h}_d^{(L)}$, respectively.

The soft-weighting mechanism offers two main advantages for joint optimization. During the forward pass, w_d acts as a continuous gate that scales the structural embedding and directs the attention of LLM to highly relevant documents while filtering out noise. In the backward pass, this pathway enables effective credit assignment to the retriever by explicitly measuring how changes in retrieval scores s_d affect the final generation loss $\mathcal{L}_1$. As a result, the retriever learns which documents best enhance generator performance, promoting synergistic optimization between retrieval and generation.

D. Semantic-guided Distillation

In real-world applications, obtaining fine-grained, ground-truth annotations for query-document relevance is prohibitively expensive and often infeasible. To overcome this strict absence of direct supervision, we draw inspiration from knowledge distillation [51]. Specifically, we propose to train GFM through weak supervision by using the abundant unlabeled nodes on the graph. The pre-trained text embedding models [52] exhibit robust semantic reasoning capabilities, which allow them to effectively capture the underlying relevance between the query and documents. Therefore, we utilize a frozen, pre-trained text encoder as a teacher to generate semantic pseudo-labels for all document nodes $\mathcal{D}$. The pseudo-label probability for a document d is formulated by applying a Softmax function over the inner products of their semantic embeddings, which can be formulated as

$$p_t(d \mid q, \mathcal{D}) = \frac{\exp(\mathbf{h}_d^\top \mathbf{h}_q / \tau)}{\sum_{d' \in \mathcal{D}} \exp(\mathbf{h}_{d'}^\top \mathbf{h}_q / \tau)}, \tag{17}$$

where $\mathbf{h}_q$ and $\mathbf{h}_d$ denote the original text embeddings of query q and the document d, respectively. The temperature parameter τ controls the smoothness of the probability distribution. A smaller temperature sharpens the distribution and assigns larger probabilities to more relevant documents, which helps provide high-confidence supervision signals during distillation.

Similarly, the student GFM retriever generates its predicted retrieval distribution $p_\theta(d \mid q, \mathcal{G})$ by applying the identical operation Softmax over its predicted relevance scores s_v, which is defined as Eq. (9). Finally, we optimize the GFM retriever by minimizing the Kullback-Leibler (KL) divergence between its predicted retrieval distribution and the semantically constructed pseudo-label distribution, which can be formulated as:

$$\mathcal{L}_2 = \mathbb{E}_{d \sim p_t(d|q,\mathcal{D})} \left[\log \frac{p_t(d \mid q, \mathcal{D})}{p_\theta(d \mid q, \mathcal{G})} \right]. \tag{18}$$

E. Training Strategy and Inference

To unify the semantic reasoning of the LLM generator and the structural retrieval of the GFM retriever, we train the entire UniGFM-RAG framework fully from end-to-end. The overall optimization objective is formulated as a linear combination of the generation loss $\mathcal{L}_1$ and the distillation loss $\mathcal{L}_2$:

$$\mathcal{L} = \mathcal{L}_1 + \lambda \mathcal{L}_2, \tag{19}$$

where λ is a tunable hyperparameter that balances the primary generation task and auxiliary retrieval mechanism.

Through this joint optimization paradigm, these two modules synergize effectively: (1) $\mathcal{L}_2$ provides robust prior guidance for the retriever via semantic pseudo-labels, resolving the scarcity of annotated labels. (2) Concurrently, $\mathcal{L}_1$ offers task-specific gradients through our differentiable soft-weighting bridge, which enables fine-tuning the retriever to fetch document nodes that directly maximize the final answering accuracy.

IV. Experiments

In this section, we conduct extensive experiments to answer the following research questions: **RQ1:** Can UniGFM-RAG achieve the state-of-the-art performance in complex reasoning? **RQ2:** Can UniGFM-RAG effectively generalize across diverse domains and data? **RQ3:** How do the key components of UniGFM-RAG contribute to overall performance? **RQ4:** How sensitive is UniGFM-RAG to different hyper-parameter settings? **RQ5:** How does the choice of backbone model impact the effectiveness of UniGFM-RAG? **RQ6:** How efficient is UniGFM-RAG in terms of inference?

A. Experimental Setup

Datasets. To comprehensively evaluate UniGFM-RAG, we conduct experiments on three widely adopted multi-hop QA

datasets, including HotpotQA [53], MuSiQue [54], and 2WikiMultiHopQA (2Wiki) [55]. For the joint training phase, we utilize the training data curated by GFM-RAG [25], where each dataset split consists of 1000 questions along with their corresponding documents. During the evaluation phase, we strictly follow the testing settings established by HippoRAG [17], GFM-RAG [25] for a fair comparison. We first assess the in-domain effectiveness of our proposed method `UniGFM-RAG` on test datasets of the aforementioned three multi-hop QA benchmarks. Furthermore, to assess the zero-shot generalization capabilities of our framework across diverse domains, we employ three out-of-domain GraphRAG benchmarks: G-bench (Novel) [56], G-bench (Medical) [56], and G-bench (CS) [57]. These benchmarks rigorously test the complex reasoning skills of the model over medical, novel, and computer science (CS) knowledge without any domain-specific fine-tuning. The detailed statistics of the evaluation datasets are summarized in Table II.

Baseline Methods. To rigorously evaluate the superiority of `UniGFM-RAG`, we compare it against a diverse set of strong baseline methods, which are broadly categorized into three groups. (1) *Pure LLM Methods*: This category directly employs the pure LLM for answer generation without relying on external retrieval, including zero-shot (ZS) GPT-4o-mini [58] and Qwen3-8B [2] as well as a fine-tuned (FT) Qwen3-8B on the training data. (2) *Traditional RAG Methods*: This category represents traditional sparse and dense retrieval models that rely solely on lexical or semantic matching without structural awareness, including BM25 [59], ColBERTv2 [60], and Qwen3-Emb (e.g., Qwen3-Embedding-0.6B) [49]. (3) *Graph-enhanced Methods*: This group consists of recent advancements that incorporate structural knowledge or graph reasoning into the retrieval and generation pipeline, including RAPTOR [61], GraphRAG (MS) [15], LightRAG [36], HippoRAG [17], SubgraphRAG [62], G-retriever [37], and GFM-RAG [25]. Detailed descriptions of these baseline methods are provided in Appendix V-B. Crucially, since our realistic problem setting assumes no access to ground-truth labels for query-document relevance, we meticulously re-run the experiments for SubgraphRAG, G-retriever, and GFM-RAG under identical condition to ensure a fair comparison. Meanwhile, all baseline methods employ gpt-4o-mini as reasoning models, with the exception of Qwen3-8B (ZS) and Qwen3-8B (FT). Further, due to computational constraints, the experiments for **RQ4**, **RQ5**, and **RQ6** were performed on a sampled subset of the original training datasets.

Metrics. We assess the performance of `UniGFM-RAG` across two primary dimensions: final QA reasoning performance and intermediate retrieval performance. For QA reasoning performance, we report Exact Match (EM) and F1 scores, aligning with established practices in prior works [17], [25]. Conversely, for the G-bench datasets, we report standard Accuracy (ACC) to maintain strict consistency with their original benchmarks [56], [57]. Regarding retrieval performance, we adopt document recall@2 (R@2) and recall@5 (R@5) for multi-hop QA and evidence recall (Recall) for G-bench datasets as evaluation metrics.

Implementation Details. We train `UniGFM-RAG` using the large-scale corpus curated from GFM-RAG, which consists of 60,000 query samples and 737,310 documents. For the architecture, the GFM retriever is implemented as a 6-layer query-dependent GNN with a hidden dimension of 1024. During message passing, we employ a DistMult function combined with sum aggregation, while the relation update mechanism is parameterized by a 2-layer MLP. The initial text representations are encoded using Qwen3-Embedding-0.6B. The total parameter of the retriever is 34M. The LLM generator is instantiated with the Qwen3-8B model.

Our proposed `UniGFM-RAG` is trained with 8 NVIDIA H200 GPUs (141G) for 10 epochs with a per-device batch size 1. We optimize the framework using AdamW with differential learning rates to accommodate the varying convergence dynamics: 1e-4 for the randomly initialized GFM retriever and 2e-5 for the LLM generator. Finally, the loss-balancing hyperparameter λ is set to 1. We set $k = 5$ for multi-hop QA datasets and $k = 10$ for G-bench datasets. We set the teacher temperature to $\tau_t = 0.05$ to sharpen the teacher distribution and provide high-confidence relevance supervision, while the student temperature is set to $\tau_s = 1.0$ during training.

TABLE II: Statistics of the evaluation datasets.

Dataset	# Query	# Document
HotpotQA [53]	1,000	9,221
MuSiQue [54]	1,000	6,119
2Wiki [55]	1,000	11,656
G-bench (Novel) [56]	2,010	461
G-bench (Medical) [56]	2,062	2,406
G-bench (CS) [57]	1,018	24,534

*B. Main Results **(RQ1)***

QA Results. As shown in Table III, `UniGFM-RAG` outperforms all baseline methods across three multi-hop question answering datasets, demonstrating its effectiveness in reasoning over graph-structured knowledge for complex reasoning. Pure LLM methods (e.g., GPT-4o-mini), even after fine-tuning (Qwen3-8B (FT)), perform poorly on these tasks due to deficiency in inherent knowledge and complex reasoning. Traditional RAG methods (e.g, BM25, Qwen3-Emb) mitigate knowledge gap by incorporating external knowledge, which generally outperform pure LLMs. However, they still struggle with multi-hop QA due to their inability to capture complex structural relationships. Graph-enhanced methods (e.g., HippoRAG, GFM-RAG) show improved performance by effectively leveraging graph structures. However, these methods either incur significant API cost (e.g, HippoRAG) or rely on high-quality query-document labels to train the retriever (e.g., GFM-RAG). Moreover, these two-stage methods lead to misalignment between the retriever and generator, resulting in suboptimal QA performance as the retrieved information may not be perfectly optimized for the downstream reasoning.

`UniGFM-RAG` overcomes these limitations with a unified end-to-end framework that tightly integrates the GFM retriever and LLM generator, aligning their objectives to directly optimize for the final QA performance. By modeling complex

TABLE III: QA reasoning performance comparison on multi-hop datasets.

Method	HotpotQA		MuSiQue		2Wiki	
	EM	F1	EM	F1	EM	F1
Pure LLM Methods						
GPT-4o-mini (ZS) [58]	28.6	41.0	11.2	36.3	30.2	36.3
Qwen3-8B (ZS) [2]	15.1	23.7	1.3	8.1	23.2	27.9
Qwen3-8B (FT) [2]	19.1	27.0	2.7	10.7	25.8	29.0
Traditional RAG Methods						
BM25 [59]	52.0	63.4	20.3	28.8	47.9	51.2
ColBERTv2 [60]	43.4	57.7	15.5	26.4	33.4	43.3
Qwen3-Emb [49]	46.9	60.4	24.0	35.8	49.5	57.1
Graph-enhanced Methods						
RAPTOR [61]	50.6	64.7	27.7	39.2	39.7	48.4
GraphRAG (MS) [15]	51.4	67.6	27.0	42.0	34.7	61.0
LightRAG [36]	9.9	20.2	2.0	9.3	2.5	12.1
HippoRAG [17]	46.3	60.0	24.0	35.9	59.4	67.3
SubgraphRAG [62]	44.5	57.0	25.1	35.7	62.7	69.0
G-retriever [37]	41.4	53.4	23.6	34.3	33.5	39.6
GFM-RAG [25]	46.3	60.2	23.5	34.8	60.0	66.8
UniGFM-RAG	**65.6**	**77.2**	**31.1**	**44.0**	**69.7**	**73.9**

TABLE IV: Retrieval performance comparison on multi-hop QA datasets. Recall@k (R@k) is used as the evaluation metric.

Method	HotpotQA		MuSiQue		2Wiki	
	R@2	R@5	R@2	R@5	R@2	R@5
Traditional RAG Methods						
BM25 [59]	55.4	72.2	32.3	41.2	51.8	61.9
ColBERTv2 [60]	64.7	79.3	37.9	49.2	59.2	68.2
Qwen3-Emb [49]	67.5	80.0	41.9	55.7	63.6	70.0
Graph-enhanced Methods						
RAPTOR [61]	58.1	71.2	35.7	45.3	46.3	53.8
GraphRAG (MS) [15]	58.3	76.6	35.4	49.3	61.6	77.3
LightRAG [36]	38.8	54.7	24.8	34.7	45.1	59.1
HippoRAG [17]	60.1	78.5	41.2	53.2	68.4	**87.0**
SubgraphRAG [62]	57.6	71.7	37.7	47.6	**70.2**	85.3
G-retriever [37]	51.8	63.8	35.4	43.5	61.0	66.6
GFM-RAG [25]	56.1	76.4	37.0	51.2	63.5	82.4
UniGFM-RAG	**68.7**	**92.4**	**45.5**	**67.3**	65.7	81.7

relationships in the retrieved knowledge graph and ensuring that the retrieved information is highly relevant for reasoning, UniGFM-RAG significantly outperforms all baselines and achieves the state-of-the-art across all evaluated datasets.

Retrieval Results. As shown in Table IV, UniGFM-RAG achieves the best retrieval results on the MuSiQue and HotpotQA datasets, while maintaining highly competitive performance on the 2Wiki dataset. Although advanced embedding-based methods (e.g., Qwen3-Emb) yield strong results by effectively capturing semantic similarity, they consistently fall short of graph-enhanced approaches when handling complex, multi-hop queries. Notably, despite being distilled from the Qwen3-Emb model, UniGFM-RAG significantly outperforms its semantic base model across all evaluated datasets. This substantial performance gap highlights a dual advantage in our approach. First, by utilizing the GFM to model topological relationships, UniGFM-RAG successfully overcomes the inherent limitations of the semantic-only retrieval, proving that structural awareness is essential for complex reasoning. Second, our framework addresses the limitations of existing graph-based retrievers (e.g., GFM-RAG). Although GFM-RAG also adopts a GFM, UniGFM-RAG still surpasses it by aligning the answer objectives directly into the retrieval phase. This demonstrates the superior efficacy of our end-to-end training paradigm over disjointed two-stage pipelines, ensuring that the retrieved information are perfectly optimized for the downstream reasoning task without relying on any human labeled ground-truth.

TABLE V: Performance comparison on the G-bench dataset [56], [57]. QA reasoning is measured by Accuracy (ACC), and retrieval is measured by Recall.

Method	G-bench (ACC)			G-bench (Recall)	
	Novel	Medical	CS	Novel	Medical
Pure LLM Methods					
GPT-4o-mini (ZS) [58]	51.4	67.1	70.7	-	-
Qwen3-8B (ZS) [2]	31.2	38.5	63.5	-	-
Qwen3-8B (FT) [2]	42.2	39.4	24.9	-	-
Traditional RAG Methods					
BM25 [59]	56.5	68.7	71.7	82.1	87.9
ColBERTv2 [60]	56.2	71.8	71.9	82.4	89.5
Qwen3-Emb [49]	46.0	40.5	73.4	86.4	**95.2**
Graph-enhanced Methods					
RAPTOR [61]	43.2	57.1	73.6	66.1	84.2
GraphRAG (MS) [15]	50.9	45.2	72.5	67.4	56.4
LightRAG [36]	45.1	63.9	71.2	79.6	82.6
HippoRAG [17]	44.8	59.1	72.6	81.2	84.0
GFM-RAG [25]	56.7	71.9	72.6	83.5	89.6
UniGFM-RAG	**60.0**	**72.9**	**73.9**	**87.8**	92.6

C. Generalization across Diverse Domains (*RQ2*)

To evaluate the generalizability of UniGFM-RAG, we conduct experiments on G-bench datasets in diverse domains, including novel, medical and computer science (CS). As shown in Table V, UniGFM-RAG consistently achieves top-tier performance in most metrics, which demonstrate strong robustness and cross-domain adaptability. Such superior generalization stems from the effective fusion of Graph Foundation Models and Large Language Models, which jointly leverage transferable graph representation ability and powerful reasoning and generalization capabilities. This synergy allows UniGFM-RAG to perform reliably across distinct domains, which highlights its strong domain-agnostic nature.

D. Ablation Study (*RQ3*)

In this section, we conduct ablation studies to validate the effectiveness of each core component in UniGFM-RAG: (1) the LLM generator objective (L_1), (2) the semantic guided distillation loss (L_2), and (3) graph foundation model (GFM) retriever to the overall performance. The detailed results are summarized in Figure 3.

From the table, the experimental results clearly demonstrate the necessity of each component in UniGFM-RAG. First, removing the generator objective (*w/o* $\mathcal{L}_1$) yields a noticeable performance drop across all datasets, which confirms its essential role in ensuring high-quality answer generation. Secondly, excluding the semantic-guided distillation loss (*w/o* $\mathcal{L}_2$) leads to severe degradation. This highlights that, in the absence of ground-truth document retrieval labels, our pseudo-label supervision is vital for effectively training GFM retriever. Crucially, we observe a catastrophic performance drop when removing the GFM retriever entirely (*w/o* GFM). This underscores the absolute necessity of reasoning capability over the graph-structured knowledge for resolving complex reasoning.

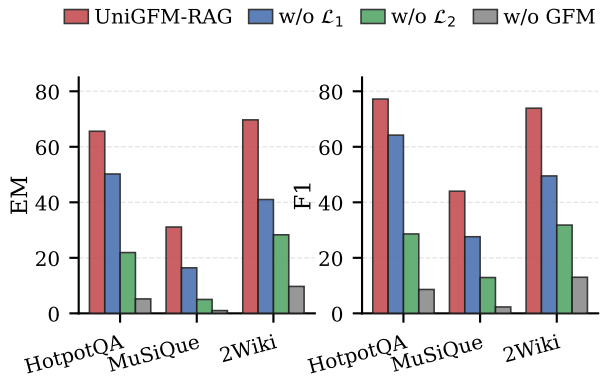


Fig. 3: Ablation studies of UniGFM−RAG.

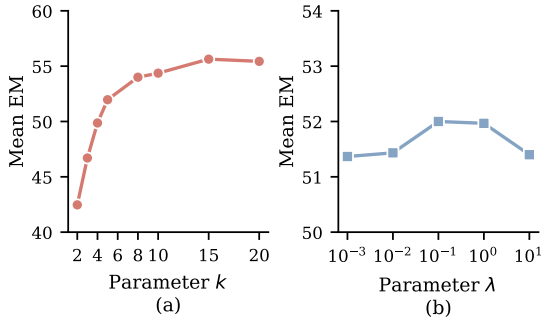


Fig. 4: Parameter analysis on hyperparameter on k and λ.

E. Parameter Analysis (RQ4)

To investigate the robustness of UniGFM−RAG, we analyze the impact of critical hyper-parameters, the number of the retrieved document $k \in \{2, 5, 8, 10, 15, 20\}$ and the distillation loss weight $\lambda \in \{0.001, 0.01, 0.1, 1.0, 10\}$. As illustrated in Figure 4(a), while overall performance improves as k increases, UniGFM−RAG already achieves highly competitive results at $k = 5$. Beyond this point, the marginal performance gains diminish rapidly, whereas the computational overhead grows significantly. Consequently, to strike an optimal trade-off between model effectiveness and computational efficiency, we adopt $k = 5$ as the default setting. Additionally, Figure 4(b) demonstrates that the framework remains largely stable regarding λ. UniGFM−RAG exhibits minor performance fluctuations under varying λ values. Based on these observations, we set $\lambda = 1.0$ as the optimal default setting for our experiments.

F. Different Backbone Models (RQ5)

To verify the model-agnostic nature of UniGFM−RAG, we evaluate it with multiple LLM backbones of different families and parameter scales. As shown in Table VI, replacing the default generator with Qwen3-4B, Llama-3.1-8B, or Llama-3.2-3B still yields consistently strong performance across all three benchmarks. In particular, all backbone variants substantially outperform GFM-RAG on HotpotQA, and also achieve clear gains on the more challenging MuSiQue benchmark, indicating that the effectiveness of UniGFM−RAG is not tied to a specific generator. On 2Wiki, the performance remains highly competitive, with the lightweight Llama-3.2-3B variant even achieving the best EM score among all compared backbones. These results show that the gains of UniGFM−RAG mainly come from the framework itself, namely the tight integration of the GFM retriever and the LLM generator, rather than from relying on one particular backbone model.

TABLE VI: Different backbone models in UniGFM−RAG.

Variant	HotpotQA		MuSiQue		2Wiki	
	EM	F1	EM	F1	EM	F1
GFM-RAG	46.3	59.5	23.5	34.8	60.0	66.8
UniGFM−RAG						
Qwen3-8B	**63.4**	**75.7**	**31.1**	**44.0**	61.7	**67.1**
Qwen3-4B	61.2	73.2	26.7	37.5	59.3	64.67
Llama-3.1-8B	59.5	72.3	29.4	39.2	59.0	63.5
Llama-3.2-3B	58.0	70.8	28.3	39.9	**62.4**	67.0

G. Efficiency Analysis (RQ6)

To evaluate the inference efficiency of UniGFM−RAG, we conduct experiments on G-bench (CS) to compare both inference cost (seconds per sample) and reasoning accuracy against various baseline methods. As shown in Figure 5, UniGFM−RAG achieves the highest reasoning accuracy while also incurring the lowest inference time per sample. This advantage mainly comes from its end-to-end graph-aware reasoning design. Existing graph-based RAG methods usually involve additional online reasoning steps. For example, ToG requires multi-round graph exploration and iterative LLM prompting, while HippoRAG performs multi-hop graph retrieval with Personalized PageRank, which also introduces iterative graph propagation during inference. GraphRAG and LightRAG also rely on external LLMs to summarize, reason over, or generate responses from retrieved graph contexts, which increases the online inference cost. In contrast, UniGFM−RAG performs retrieval and reasoning through a unified GFM, which avoids repeated external LLM invocations. Although graph encoding and message passing introduce additional computation, these operations are highly parallelizable on GPUs and are more efficient than multi-step LLM prompting during inference. Therefore, UniGFM−RAG achieves a favorable balance between effectiveness and efficiency for reasoning over graph-structured knowledge.

H. Retrieval Case Study

To further illustrate the effectiveness of UniGFM−RAG, we present a qualitative retrieval case study. As shown in Table VII, both UniGFM−RAG and GFM-RAG successfully retrieve the initial relevant document for “Rip Curl Pro Portugal”, which mentions John John Florence. However, GFM-RAG fails to retrieve the subsequent relevant information (e.g., date of birth) for “John John Florence”, which directly derives the final answer. In contrast, UniGFM−RAG successfully identifies the critical supporting document for “John John Florence”. This demonstrates that, with the guidance of the LLM generator, our GFM retriever effectively navigates

TABLE VII: Case study comparing the retrieval and reasoning performance of UniGFM-RAG with GFM-RAG on the HotpotQA dataset.

Question	What American professional Hawaiian surfer born 18 October 1992 won the Rip Curl Pro Portugal?	
Gold Answer	John John Florence	
Supporting Docs.	["MEO Rip Curl Pro Portugal", "John John Florence"]	
GFM-RAG (Baseline Method)		
Retrieved Docs.	["MEO Rip Curl Pro Portugal", "Reef Hawaiian Pro"]	**(Recall@2=0.5)**
Predicted Answer	document does not contain information about John John Florence	**✗ Incorrect**
UniGFM-RAG (Ours)		
Retrieved Docs.	["MEO Rip Curl Pro Portugal", **"John John Florence"**]	**(Recall@2=1.0)**
Predicted Answer	John John Florence	**✓ Correct**

TABLE VIII: Case study comparing the retrieval and reasoning performance of UniGFM-RAG with GFM-RAG on the HotpotQA dataset.

Question	Which season did Brian Billick coach the Baltimore Ravens though in 2001?	
Gold Answer	6th season	
Supporting Docs.	["2001 Baltimore Ravens season", "Brian Billick"]	
GFM-RAG (Baseline Method)		
Retrieved Docs.	["2015 Baltimore Ravens season", "2011 Baltimore Ravens season", "Brian Billick", "2000 Baltimore Ravens season", "2001 Baltimore Ravens season"]	**(Recall@5=1.0)**
Predicted Answer	2001 Baltimore Ravens season	**✗ Incorrect**
UniGFM-RAG (Ours)		
Retrieved Docs.	[**"2001 Baltimore Ravens season"** (0.42), **"Brian Billick"** (0.38), "2000 Baltimore Ravens season" (0.34), "2015 Baltimore Ravens season" (0.19), "2001 St. Louis Rams season" (0.03)]	**(Recall@5=1.0)**
Predicted Answer	6th season	**✓ Correct**

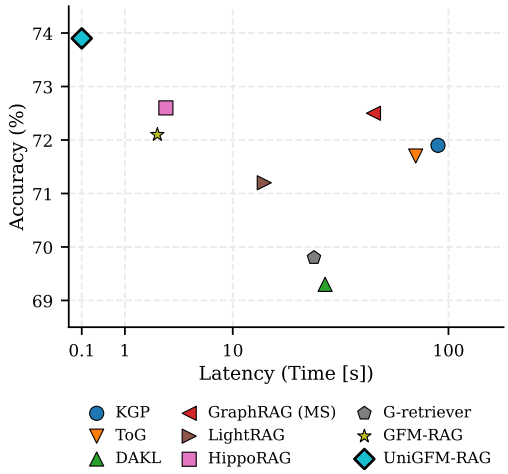


Fig. 5: Efficiency and performance comparison on G-bench (CS).

complex graph structures to locate critical multi-hop evidence for complex reasoning.

I. QA Case Study

Furthermore, we present a case study on answer generation to validate the critical impact of document ordering driven by our soft-weighting mechanism. As shown in Table VIII, both UniGFM-RAG and GFM-RAG successfully retrieve all supporting documents within their top-5 results (Recall@5=1.0). However, UniGFM-RAG places the relevant documents at the top of the ranked list, whereas GFM-RAG ranks them 3rd and 5th. Consequently, UniGFM-RAG derives the correct answer, while GFM-RAG fails, likely as it suffers from the well-known "lost in the middle" phenomenon [63]. Ultimately, this demonstrates that introducing the soft-weighting mechanism explicitly reveals the importance of each retrieved document, guiding the generator to focus on the most critical evidence.

V. Conclusion

In this paper, we present UniGFM-RAG, a unified, end-to-end GraphRAG framework that seamlessly couples a graph foundation model (GFM) retriever with a large language model (LLM) generator. By leveraging GFM, we effectively reason over structural knowledge to conduct accurate document retrieval. Through the proposed graph bridge module, we connect the retriever and generator via a fully differentiable architecture. This integration fosters a reciprocal learning loop: the structural insights from GFM enhance the reasoning capabilities of LLM, while the gradient feedback from the LLM refines the graph retrieval process. Furthermore, by employing soft-weighting mechanism, we enable the LLM to dynamically focus on the most critical documents within the graph-augmented prompt. Extensive experiments on six complex reasoning benchmarks demonstrate that UniGFM-RAG consistently outperforms existing baseline methods, which establishes new state-of-the-art performance. Additional analyses confirm its exceptional inference efficiency and strong generalizability. Ultimately, UniGFM-RAG offers a promising paradigm for unifying GFMs and LLMs for knowledge-intensive applications. In future work, we plan to extend UniGFM-RAG in two directions. First, we will explore multi-modal knowledge injection, which enables the topology-aware

reasoning process to incorporate heterogeneous information beyond pure text. Second, we aim to adapt the unified end-to-end framework to specialized and high-stakes domains that require both structural evidence and reliable generation.

REFERENCES

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

[2] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv *et al.*, "Qwen3 technical report," *arXiv preprint arXiv:2505.09388*, 2025.

[3] A. Liu, B. Feng, B. Xue, B. Wang, B. Wu, C. Lu, C. Zhao, C. Deng, C. Zhang, C. Ruan *et al.*, "Deepseek-v3 technical report," *arXiv preprint arXiv:2412.19437*, 2024.

[4] H. Ding, Z. Fan, I. Guehring, G. Gupta, W. Ha, J. Huan, L. Liu, B. Omidvar-Tehrani, S. Wang, and H. Zhou, "Reasoning and planning with large language models in code development," in *KDD*, 2024, p. 6480–6490.

[5] B. Du, P. Li, S. Zhang, H. Liu, and Y. Yu, "Trust-aware human–robot collaborative decision-making with llm-generated data," *IEEE TCDS*, vol. 18, no. 3, pp. 763–778, 2026.

[6] Z. Li, X. Chen, H. Yu, H. Lin, Y. Lu, Q. Tang, F. Huang, X. Han, L. Sun, and Y. Li, "Structrag: Boosting knowledge intensive reasoning of llms via inference-time hybrid information structurization," in *ICLR*, 2025.

[7] S. Pan, L. Luo, Y. Wang, C. Chen, J. Wang, and X. Wu, "Unifying large language models and knowledge graphs: A roadmap," *IEEE TKDE*, vol. 36, no. 7, pp. 3580–3599, 2024.

[8] X. Sun, H. Cheng, J. Li, B. Liu, and J. Guan, "All in one: Multi-task prompting for graph neural networks," in *KDD*, 2023, pp. 2120–2131.

[9] L. Yang, J. Zheng, H. Wang, Z. Liu, Z. Huang, S. Hong, W. Zhang, and B. Cui, "Individual and structural graph information bottlenecks for out-of-distribution generalization," *IEEE TKDE*, vol. 36, p. 682–693, 2024.

[10] L. Jiang, J. Ren, Z. Zhou, Y. Qu, Z. Zheng, H. Lu, and M. Wu, "Graph reinforcement learning-based reachability map for generalized mobile manipulation," *IEEE TCDS*, vol. 18, no. 2, pp. 438–451, 2026.

[11] B. Peng, Y. Zhu, Y. Liu, X. Bo, H. Shi, C. Hong, Y. Zhang, and S. Tang, "Graph retrieval-augmented generation: A survey," *TOIS*, vol. 44, no. 2, pp. 1–52, 2025.

[12] Q. Zhang, S. Chen, Y. Bei, Z. Yuan, H. Zhou, Z. Hong, H. Chen, Y. Xiao, C. Zhou, J. Dong *et al.*, "A survey of graph retrieval-augmented generation for customized large language models," *arXiv preprint arXiv:2501.13958*, 2025.

[13] Y. Huang, S. Zhang, and X. Xiao, "Ket-rag: A cost-efficient multi-granular indexing framework for graph-rag," in *KDD*, 2025, p. 1003–1012.

[14] Z. Zhao, L. Luo, J. Dong, H. Luo, X. Li, S. Pan, and C. Gong, "Hyper-reco: Retrieving and connecting evidence with hypergraph neural networks for llm multi-hop reasoning," *arXiv preprint arXiv:2609.32327*, 2026.

[15] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, D. Metropolitansky, R. O. Ness, and J. Larson, "From local to global: A graph rag approach to query-focused summarization," *arXiv preprint arXiv:2404.16130*, 2024.

[16] Y. Wang, N. Lipka, R. A. Rossi, A. Siu, R. Zhang, and T. Derr, "Knowledge graph prompting for multi-document question answering," in *AAAI*, vol. 38, no. 17, 2024, pp. 19 206–19 214.

[17] B. Jimenez Gutierrez, Y. Shu, Y. Gu, M. Yasunaga, and Y. Su, "Hipporag: Neurobiologically inspired long-term memory for large language models," *NeurIPS*, vol. 37, pp. 59 532–59 569, 2024.

[18] Y. Li, D. Song, Y. Tian, H. Wang, C. Zhou, and S. Zhang, "A framework of knowledge graph-enhanced large language model based on global planning," *IEEE TKDE*, vol. 38, no. 2, pp. 736–748, 2025.

[19] S. Ma, C. Xu, X. Jiang, M. Li, H. Qu, C. Yang, J. Mao, and J. Guo, "Think-on-graph 2.0: Deep and faithful large language model reasoning with knowledge-guided retrieval augmented generation," in *ICLR*, 2025.

[20] S. Wei, Z. Zhao, X. Feng, and H. Yu, "A cooperative multiagent reinforcement learning method based on reward decomposition for scalable agent environments," *IEEE TCDS*, vol. 18, no. 2, pp. 398–410, 2026.

[21] H. Mao, Z. Chen, W. Tang, J. Zhao, Y. Ma, T. Zhao, N. Shah, M. Galkin, and J. Tang, "Position: Graph foundation models are already here," in *ICML*, 2024.

[22] J. Liu, C. Yang, Z. Lu, J. Chen, Y. Li, M. Zhang, T. Bai, Y. Fang, L. Sun, P. S. Yu *et al.*, "Graph foundation models: Concepts, opportunities and challenges," *IEEE TPAMI*, vol. 47, no. 6, pp. 5023–5044, 2025.

[23] J. Yang, L. Yang, Z. Guo, J. Gao, J. Wu, T. Chai, H. Huang, C. Yang, and C. Shi, "Benchmarking graph foundation models," in *KDD*, 2025, p. 5866–5875.

[24] Y. Fang, H. Miao, Y. Liang, L. Deng, Y. Cui, X. Zeng, Y. Xia, Y. Zhao, T. B. Pedersen, C. S. Jensen *et al.*, "Unraveling spatio-temporal foundation models via the pipeline lens: A comprehensive review," *IEEE TKDE*, vol. 38, pp. 2040–2063, 2026.

[25] L. Luo, Z. Zhao, G. Haffari, D. Phung, C. Gong, and S. Pan, "Gfm-rag: Graph foundation model for retrieval augmented generation," in *NeurIPS*, 2025.

[26] Y. Yu, W. Ping, Z. Liu, B. Wang, J. You, C. Zhang, M. Shoeybi, and B. Catanzaro, "Rankrag: Unifying context ranking with retrieval-augmented generation in llms," *NeurIPS*, vol. 37, pp. 121 156–121 184, 2024.

[27] J. Dong, B. Fatemi, B. Perozzi, L. F. Yang, and A. Tsitsulin, "Don't forget to connect! improving rag with graph-based reranking," *arXiv preprint arXiv:2405.18414*, 2024.

[28] B. Jin, G. Liu, C. Han, M. Jiang, H. Ji, and J. Han, "Large language models on graphs: A comprehensive survey," *IEEE TKDE*, vol. 36, no. 12, pp. 8622–8642, 2024.

[29] J. Tang, Y. Yang, W. Wei, L. Shi, L. Su, S. Cheng, D. Yin, and C. Huang, "Graphgpt: Graph instruction tuning for large language models," in *SIGIR*, 2024, pp. 491–500.

[30] Z. Chai, T. Zhang, L. Wu, K. Han, X. Hu, X. Huang, and Y. Yang, "Graphllm: Boosting graph reasoning ability of large language model," *IEEE TBD*, 2025.

[31] N. Chen, Y. Li, J. Tang, and J. Li, "Graphwiz: An instruction-following language model for graph computational problems," in *KDD*, 2024, pp. 353–364.

[32] H. Qiao, C. Niu, L. Chen, and G. Pang, "Anomalygfm: Graph foundation model for zero/few-shot anomaly detection," in *KDD*, 2025, pp. 2326–2337.

[33] R. Chen, T. Zhao, A. Jaiswal, N. Shah, and Z. Wang, "Llaga: large language and graph assistant," in *ICML*, 2024.

[34] L. LUO, Y.-F. Li, G. Haffari, and S. Pan, "Reasoning on graphs: Faithful and interpretable large language model reasoning," in *ICLR*, 2024.

[35] L. Luo, Z. Zhao, G. Haffari, Y.-F. Li, C. Gong, and S. Pan, "Graph-constrained reasoning: Faithful reasoning on knowledge graphs with large language models," in *ICML*, 2025.

[36] Z. Guo, L. Xia, Y. Yu, T. Ao, and C. Huang, "Lightrag: Simple and fast retrieval-augmented generation," *EMNLP*, 2025.

[37] X. He, Y. Tian, Y. Sun, N. Chawla, T. Laurent, Y. LeCun, X. Bresson, and B. Hooi, "G-retriever: Retrieval-augmented generation for textual graph understanding and question answering," *NeurIPS*, vol. 37, pp. 132 876–132 907, 2024.

[38] C. Mavromatis and G. Karypis, "Gnn-rag: Graph neural retrieval for efficient large language model reasoning on knowledge graphs," in *ACL*, 2025, pp. 16 682–16 699.

[39] M. Yasunaga, H. Ren, A. Bosselut, P. Liang, and J. Leskovec, "Qa-gnn: Reasoning with language models and knowledge graphs for question answering," in *NAACL*, 2021, pp. 535–546.

[40] X. Ren, J. Tang, D. Yin, N. Chawla, and C. Huang, "A survey of large language models for graphs," in *KDD*, 2024, pp. 6616–6626.

[41] T. He, Z. Chen, L. Liao, Y. Cao, Y. Liu, W. Tang, X. Mao, K. Lv, M. Liu, and B. Qin, "Subgraph-centric multi-agent reinforcement learning for multi-hop knowledge graph reasoning," *IEEE TKDE*, vol. 38, no. 2, pp. 1319–1333, 2025.

[42] Y. Li, Y. Wang, J. Tang, H. Chang, Y. Ren, and J. Li, "Advancing graph foundation models: A data-centric perspective," in *KDD*, 2025, pp. 1635–1646.

[43] X. Jiang, T. Jia, Y. Fang, C. Shi, Z. Lin, and H. Wang, "Pre-training on large-scale heterogeneous graph," in *KDD*, 2021, pp. 756–766.

[44] M. Galkin, X. Yuan, H. Mostafa, J. Tang, and Z. Zhu, "Towards foundation models for knowledge graph reasoning," in *ICLR*, 2024.

[45] L. Xia, B. Kao, and C. Huang, "Opengraph: Towards open graph foundation models," *arXiv preprint arXiv:2403.01121*, 2024.

[46] Z. Wang, Z. Zhang, N. V. Chawla, C. Zhang, and Y. Ye, "Gft: graph foundation model with transferable tree vocabulary," in *NeurIPS*, 2024.

[47] J. Tang, Y. Yang, W. Wei, L. Shi, L. Xia, D. Yin, and C. Huang, "Higpt: Heterogeneous graph language model," in *KDD*, 2024, pp. 2842–2853.

[48] M. Zhang, M. Sun, P. Wang, S. Fan, Y. Mo, X. Xu, H. Liu, C. Yang, and C. Shi, "Graphtranslator: Aligning graph model to large language model for open-ended tasks," in *WWW*, 2024, pp. 1003–1014.

[49] Y. Zhang, M. Li, D. Long, X. Zhang, H. Lin, B. Yang, P. Xie, A. Yang, D. Liu, J. Lin *et al.*, "Qwen3 embedding: Advancing text embedding and reranking through foundation models," *arXiv preprint arXiv:2506.05176*, 2025.
[50] B. Yang, S. W.-t. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," in *ICLR*, 2015.
[51] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.
[52] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2019, pp. 4171–4186.
[53] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. Cohen, R. Salakhutdinov, and C. D. Manning, "Hotpotqa: A dataset for diverse, explainable multi-hop question answering," in *EMNLP*, 2018, pp. 2369–2380.
[54] H. Trivedi, N. Balasubramanian, T. Khot, and A. Sabharwal, "Musique: Multihop questions via single-hop question composition," *TACL*, vol. 10, pp. 539–554, 2022.
[55] X. Ho, A.-K. D. Nguyen, S. Sugawara, and A. Aizawa, "Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps," in *COLING*, 2020, pp. 6609–6625.
[56] Z. Xiang, C. Wu, Q. Zhang, S. Chen, Z. Hong, X. Huang, and J. Su, "When to use graphs in rag: A comprehensive analysis for graph retrieval-augmented generation," *arXiv preprint arXiv:2506.05690*, 2025.
[57] Y. Xiao, J. Dong, C. Zhou, S. Dong, Q.-w. Zhang, D. Yin, X. Sun, and X. Huang, "Graphrag-bench: Challenging domain-specific reasoning for evaluating graph retrieval-augmented generation," *arXiv preprint arXiv:2506.02404*, 2025.
[58] OpenAI, "Hello gpt-4o," 2024. [Online]. Available: https://openai.com/index/hello-gpt-4o/
[59] S. E. Robertson and S. Walker, "Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval," in *SIGIR*, 1994, pp. 232–241.
[60] K. Santhanam, O. Khattab, J. Saad-Falcon, C. Potts, and M. Zaharia, "Colbertv2: Effective and efficient retrieval via lightweight late interaction," in *NAACL*, 2022, pp. 3715–3734.
[61] P. Sarthi, S. Abdullah, A. Tuli, S. Khanna, A. Goldie, and C. D. Manning, "Raptor: Recursive abstractive processing for tree-organized retrieval," in *ICLR*, 2024.
[62] M. Li, S. Miao, and P. Li, "Simple is effective: The roles of graphs and large language models in knowledge-graph-based retrieval-augmented generation," in *ICLR*, 2025.
[63] G. A. Baker, A. Raut, S. Shaier, L. E. Hunter, and K. von der Wense, "Lost in the middle, and in-between: Enhancing language models' ability to reason over long contexts in multi-hop qa," *arXiv preprint arXiv:2412.10079*, 2024.

VI. Biography Section



Zicheng Zhao is currently pursuing the Ph.D. degree in computer science and technology. He is currently a visiting student with the School of Automation and Intelligent Sensing, Shanghai Jiao Tong University. His research interests include knowledge graphs, large language models, and graph neural networks.



Linhao Luo is a Postdoctoral Research Fellow with the Department of Data Science & AI at Monash University, Australia. He obtained his PhD degree in computer science at Monash University and received his B.S. degree from Harbin Institute of Technology, China. He has published over 30 research papers in top-tier journals and conferences, including the IEEE Transactions on Knowledge and Data Engineering (TKDE), Knowledge and Information Systems, Knowledge-Based Systems, ICLR, ICML, NeurIPS, ACL, EMNLP, AAAI, IJCAI, WSDM, CIKM, SIGIR, and The Web Conference.



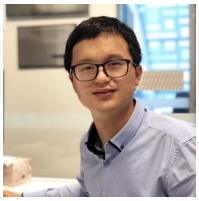
Zongji Yu will receive the bachelor's degree from the School of Control Science and Engineering, Shandong University in Jinan, China. Since 2026, he has been pursuing a doctoral degree at Shanghai Jiao Tong University in Shanghai, China. His research interests include continual learning, reinforcement learning, and AI safety.



Junnan Dong is now a senior researcher at Tencent Youtu Lab. He obtained his Ph.D. at the Hong Kong Polytechnic University. He focuses on Large Language Models (LLMs), Diffusion Language Models (DLMs), Retrieval Augmented Generation (RAG) and Agents for advanced domain-specific reasoning. He has published more than 20 papers on top-tier conferences, including ICLR, ICML, NeurIPS, ACL, WWW, KDD, SIGIR, AAAI and WSDM. He was also nominated as the 'Best Research Postgraduate Student' by both the Faculty and Department of Computing at PolyU. He has been continuously serving as a program committee member for ICLR, NeurIPS, ICML, ICLR, ACL, AAAI and WWW, etc.



Yanchao Tan (Member, IEEE) is currently an Associate Professor in the College of Computer and Data Science, Fujian Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University. She obtained her PhD degree in the College of Computer Science, Zhejiang University, China. Her research interests include data mining, healthcare, graph mining, and recommender system.



Shirui Pan (Senior Member, IEEE) received the PhD degree in computer science from the University of Technology Sydney (UTS), Ultimo, NSW, Australia. He is a professor with the School of Information and Communication Technology, Griffith University, Australia. His research interests include data mining and machine learning. His research received the 2024 CIS IEEE TNNLS Oustanding Paper Award and the 2020 IEEE ICDM Best Student Paper Award. He is recognized as one of the AI 2000 AAAI/IJCAI Most Influential Scholars in Australia. He was the awardee for the 2024 IEEE ICDM Tao Li Award. He is an ARC Future fellow and a fellow of Queensland Academy of Arts and Sciences (FQA).



Chen Gong (Senior Member, IEEE) received his dual doctoral degree from Shanghai Jiao Tong University (SJTU) and University of Technology Sydney (UTS), respectively. Currently, he is a full professor of Shanghai Jiao Tong University. His research interests mainly include machine learning, data mining, and learning-based vision problems. He has published more than 130 technical papers at prominent journals and conferences such as IEEE T-PAMI, JMLR, IJCV, IEEE T-NNLS, IEEE T-IP, ICML, NeurIPS, ICLR, CVPR, ICCV, ECCV, AAAI, IJCAI, ICDM, etc. He serves as the associate editor for IEEE T-CSVT, Neural Networks, and NePL, and also the Area Chair or Senior PC member of several top-tier conferences such as AAAI, IJCAI, ICML, ICLR, ECML-PKDD, AISTATS, ICDM, ACM MM, etc. He won the ICDM Best Student Paper Runner-Up Award, the second prize of Natural Science Award of the Chinese Institute of Electronics, "Excellent Doctorial Dissertation Award" of Chinese Association for Artificial Intelligence, "Wu Wen-Jun AI Excellent Youth Scholar Award", and the Scientific Fund for Distinguished Young Scholars of Jiangsu Province. He was also selected as the "Global Top Chinese Young Scholars in AI" released by Baidu, and "World's Top 2% Scientists" released by Stanford University.