Dynamic Graph Navigation via Triplet Chains for Structure-Aware Retrieval-Augmented Generation

Feng Zhao, Yufei Wu, Xianggan Liu*, Ruilin Zhao
Natural Language Processing and Knowledge Graph Lab,
School of Computer Science and Technology,
Huazhong University of Science and Technology, Wuhan, China
{zhaof, m202373966, d202287090, ruilinzhao}@hust.edu.cn

Abstract

Retrieval-Augmented Generation (RAG) was proposed to address the hallucination question of large language models (LLMs). However, the traditional RAG framework has certain limitations: for simple questions, the search results often introduce a large amount of irrelevant information; while for complex questions, the lengthy reference knowledge provided by the retrieval lacks structural information. Therefore, we proposed a structure-aware RAG, which achieves noise removal in retrieval through multi-chain graph navigation reasoning(Trig-Nav). This method constructs question triple reasoning chains and reference knowledge graphs with text attributes, allowing the system to retrieve three types of knowledge along different paths based on the requirements of LLM. It provides LLM with multi-angle and structured information input and significantly reduces noise. We conducted a comprehensive evaluation of Trig-Nav, comparing it with baseline methods across multiple datasets.Compared to traditional RAG, there is an average improvement of 6% in effectiveness. The results showed that Trig-Nav significantly enhances the model's performance, validating the effectiveness of this approach.

1 Introduction

Large language models (LLMs) (Brown et al., 2020; OpenAI et al., 2024) have demonstrated remarkable capabilities in language comprehension and reasoning. However, when confronting complex queries or topics beyond the temporal constraints of their training data, LLMs are prone to hallucinations and factual errors (Maynez et al., 2020; Zhou et al., 2021). Retrieval-Augmented Generation (RAG) (Wang et al., 2023) has emerged as a prevalent strategy to mitigate these issues by incorporating external knowledge from databases or the web. Despite its effectiveness, standard RAG is not universally beneficial. For straightforward questions requiring no external context, retrieval process may introduce irrelevant noise (Ni et al., 2024), inadvertently misleading the model. Conversely, in scenarios involving complex reasoning or extensive documents, the retrieved text often suffers from a disordered, unstructured format. Key information is typically scattered, hindering the LLM's ability to extract salient knowledge effectively. Consequently, adaptively retrieving low-noise, high-density information contingent on query complexity remains a critical challenge.

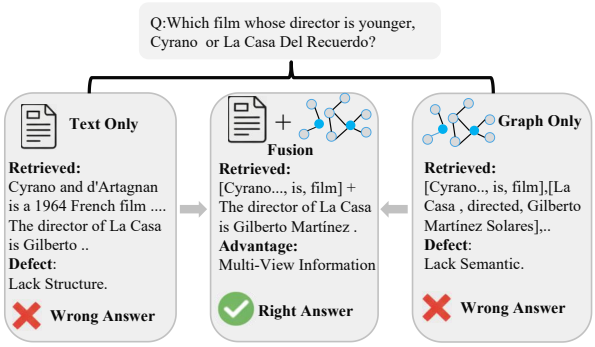

Figure 1: This figure illustrates that a single type of knowledge is insufficient to effectively guide an LLM, whereas combining graphs and text offers better guidance.

Recent advancements have sought to enhance RAG by optimizing text retrieval or integrating Knowledge Graphs (KGs). For instance, Yu et al. (2024) proposed training a reranker to filter the most relevant context, while Ru et al. (2024) developed an evaluator to assess the validity and utility of retrieved content. Alternatively, Xu et al. (2025) leveraged KG retrieval to reduce text-induced noise. However, as illustrated in Figure 1, exclusive reliance on either modality presents significant limitations. Text-based methods, while rich in semantic

*Co-first author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 27293–27303

detail, are hampered by inherent noise and by the difficulty LLMs face in capturing long-range dependencies. In contrast, KG-based methods offer noise reduction and interpretability through structured relationships, but often suffer from semantic sparsity and fail to provide the fine-grained context necessary for complex queries. Existing research has largely treated text and graph modalities in isolation, overlooking effective mechanisms to synergize their respective strengths.

To address the above issues, we focused on the integration of Knowledge Graphs and unstructured text. The overarching goal is to retain the semantic richness inherent in text while rigorously suppressing retrieval noise and bridging the structural gap often found in flat documents. Consequently, we propose a Structure-Aware Retrieval-Augmented Method (Trig-Nav). The cornerstone of Trig-Nav is its dynamic graph navigation reasoning. Instead of retrieving information indiscriminately, Trig-Nav transforms complex user queries into logical subquestion triple reasoning chains. It then constructs a targeted, dynamic knowledge graph derived from retrieved contexts, enabling synergistic inference that combines graph topology with textual semantics.

Furthermore, we taxonomize the knowledge required for complex reasoning into three distinct categories: factual (entity attributes), associative (inter-entity relationships), and causal (logical dependencies). Traditional paradigms often fall short here: text retrieval excels at factual details but struggles with long-range causal links due to noise, whereas static graph retrieval captures associations but may lack specific factual context. This dichotomy results in knowledge gap that hinders complex problem-solving. Trig-Nav bridges this gap. It not only synthesizes the semantic breadthof text retrieval with the reasoning depth of KGs but also features an adaptive mechanism that dynamically prioritizes different knowledge types according to the LLM's immediate reasoning state. This ensures a robust, noise-resilient, and comprehensive knowledge support system for Large Language Models.

We evaluated Trig-Nav on four different datasets. On all datasets, Trig-Nav demonstrated excellent performance compared to both single and multiple retrieval baselines, proving the effectiveness and generalizability of our approach. In summary, our contributions can be summarized as follows:

- We investigate retrieval strategy selection for different types of questions and complex question-solving scenarios.It is the first time that the method of combining question triple reasoning chains with graph retrieval has been proposed.
- We propose Trig-Nav,which enhances the efficiency of LLM's answering through the assistance of RAG, and solves the question of low-quality retrieval in complex scenarios.
- We conducted comprehensive experiments on various datasets and systematically verified the effectiveness of Trig-Nav.

2 Related Work

2.1 Text Retrieval Augmented Generation

Retrieval-augmented generation (RAG) (Khandelwal et al., 2019; Lewis et al., 2020; Guu et al., 2020; Jiang et al., 2023; Shi et al., 2024) enhances LLMs via external knowledge but remains susceptible to distraction from irrelevant context (Shi et al., 2023). Recent optimizations focus on retrieval quality and efficiency. For strategies to improve retrieval quality, query rewriting is usually combined.(Gao et al., 2022; Ma et al., 2023; Zhang et al., 2025), reranking (Zhang et al., 2022; Chen et al., 2024), and instruction fine-tuning (Lin et al., 2024; Liu et al., 2024; Zhang et al., 2025). Iterative approaches, such as Shao et al. (2023), alternate generation and retrieval for mutual optimization. Additionally, auxiliary models are employed for refinement: Yoran et al. (2024) utilize NLI models for context filtering, while Zhou et al. (2024) introduce expert models for answer evaluation. Regarding adaptive retrieval timing, methods rely on fixed intervals like RETRO (Borgeaud et al., 2022), token confidence like FLARE (Jiang et al., 2023), self-reflection tokens (Asai et al., 2023), or question complexity classification(Jeong et al., 2024).

2.2 Graph Retrieval Augment Generation

The current graph retrieval mainly enhances the generation effect by utilizing the graph as a knowledge base and optimizing the retrieval mechanism or subgraph processing procedure. For example, KG²RAG (Zhu et al., 2025) obtains structured answers by using the graph to retrieve corresponding text blocks; K-RagRec (Wang et al., 2025) improves the graph retrieval algorithm and introduces GNN and projectors to align the retrieved knowledge subgraphs with the semantic space of LLM,

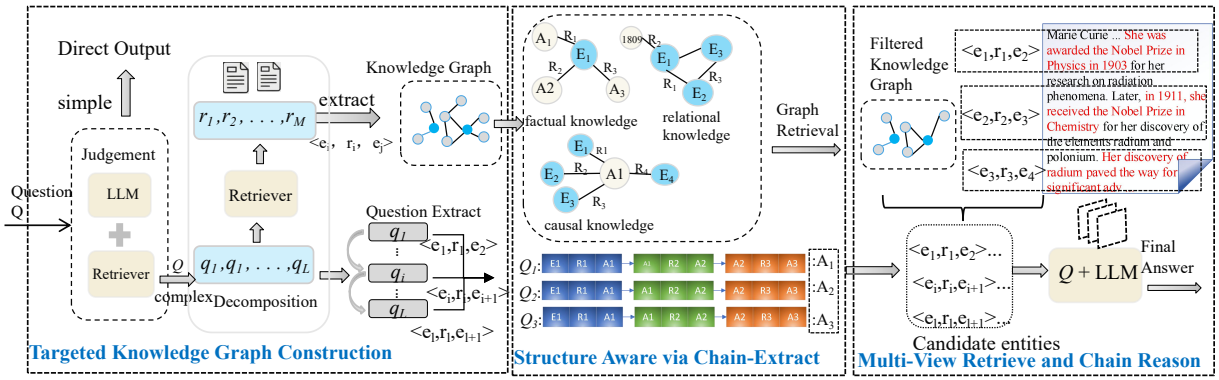

Figure 2: A display of the main process of Trig-Nav.The first module is responsible for decomposing and refining the questions to build a knowledge graph. The second module is responsible for organizing and extracting the questions to construct the triplesreasoning chains. The third module conducts multi-view retrieval and reasoning. After reasoning through the chain, it provides sub-question answers to the LLM to generate the final answer.

achieving knowledge-enhanced recommendation. Besides, there are also studies on optimizing subgraphs, such as Cai et al. (2025), which implements a two-stage alignment process from query to pattern and from pattern to subgraph, quantifying the alignment between the pattern and candidate subgraphs. Previous works mainly used graphs for retrieval, but their research perspectives mostly focused on how to use the graph for retrieval, while neglecting the fundamental issue of which types of knowledge the graph is best suited for retrieval. The essence of the graph advantage lies in its ability to depict semantic relationships between entities in a structured form. Given this, our research believes that knowledge graphs have unique structural advantages in retrieving relation and causal knowledge, which can better utilize the structural information of the graph.

3 Method

In this chapter, we first introduce some key symbols and definitions, and then elaborate on each module of our method, as shown in Figure 2.

3.1 Overview

Firstly, we will introduce the task process and the symbols used next.

For a user input query x_q, the goal of the RAG is to use the R retriever to search for the most relevant contexts $C_r = \{c_1, c_2, \cdots, c_n\}$ from a given corpus $D = \{c_1, c_2, \cdots, c_n\}$ of size N (e.g. a Wikipedia dump).Then, the context is provided to the generator G (either a black-box or white-box LLM) to produce the output answer $y_a = G(x_q, C_r)$. In our research, we found that LLMs are quite sensitive to whether the retrieved content C_r contains the necessary knowledge to solve the question. LLMs can determine what knowledge is still lacking for solving the question,and they have some awareness of their own state (Kadavath et al., 2022).Therefore, we provide C_r and x_q to the LLM to obtain the knowledge integrity judgment $J = G(x_q, C_r)$.Based on the content of J, we refine and decompose the original question $x_q$into L sub-questions $X_L = \{x_1, x_2, \cdots, x_l\}$. Each sub-question is then subjected to retrieval.We extract triples from the retrieved texts to construct a knowledge graph G_d, and conduct retrieval on both the knowledge graph and the knowledge base to obtain reference triples $T_t = \{\ldots, [S_i, P_i, O_i], \ldots\}$ and T_D. These are provided to LLM to generate the final answer $Y = G(Q, T_t, T_D)$.

Next, we will introduce the three types of knowledge proposed: factual knowledge, associative knowledge, and causal knowledge. There are sample images located in the middle position below Figure 2. Factual knowledge refers to atomic assertions that describe the inherent attributes of entities. If an entity has multiple attribute characteristics, they can be decomposed into multiple independent factual knowledge units. Associative knowledge represents semantic or structural relationships between multiple entities. Causal knowledge is used to describe the causal mechanisms or influence paths between multiple entities. We propose an adaptive knowledge selection mechanism that can dynamically identify and filter the most relevant knowledge categories based on the semantic struc-

ture and complexity of the question, providing accurate knowledge support for the language model.

Next, we will introduce the main modules of Trig-Nav: targeted knowledge graph construction, acquisition of sub-question triple chains, and integrated Graph-text retrieval and reasoning.

3.2 Targeted knowledge graph construction

We studied how to design a knowledge reconstruction method that can not only bridge the knowledge gap between LLM and questions, but also reduce the interference of irrelevant information. For the judgment of knowledge gaps, our method is the combined judgment of generation and retrieval. Specifically, for the input question X, we first obtain the cognitive representation $R = \{r_1, \ldots, r_i, \ldots, r_n\}$ of the original question X by the model.Then, by searching for relevant content in the document knowledge base based on (X, R), we systematically verify the accuracy of the model's cognition of the original question. When we detect that a cognitive unit r_i has deficiencies in correctness or completeness,we rephrased it as separate sub-questions. Through this mechanism, only by solving this series of filtered subquestions can we provide the necessary prerequisite knowledge system for the model to solve the original question X.

After obtaining the aforementioned set of semantically coherent sub-questions X_i, we perform retrieval operations for each sub-question X_i separately, and obtain the corresponding set of supporting documents Di. Considering that the original retrieval results D_i usually contain redundant information and noise, we further perform knowledge reconstruction operations, extracting structured triples from D_i and using them to construct the target knowledge graph G_i. This process can be formally represented as:

$$G_i = \text{KG-Construct}(\text{Triple-xtract}(D_i)). \quad (1)$$

The knowledge graph G_i not only has good structural readability but also effectively retains the semantic core, thereby providing more refined and clearly related knowledge support for LLM.

3.3 Structure-Aware via Triple-Chain Construction

Previously, we classified the knowledge required by the model into three categories: factual knowledge, associative knowledge, and causal knowledge. Although these types of knowledge are difficult to clearly distinguish at the text representation level, their structural differences can be clearly presented through the formal representation form of the knowledge graph. Based on this, each subquestion was mapped to the topological pattern of the knowledge graph, and an inference chain including intermediate or final answers was constructed. For example, the question "When did Valentin The Good's director die?" can be decomposed into two triplets: *[Valentin The Good, director, answer1]* and *[answer1, Date of death, answer2]*, thereby constructing the inference path "*Valentin The Good→answer1→answer2*", where the relations "director" and "Date of death" clarify the semantic direction of the path.

Specifically, factual knowledge corresponds to a single triplet structure, and the system performs direct single-step attribute queries. This characteristic can be formally expressed as:

$$f_{\text{fact}}(S) = I(|S| = 1) \quad (2)$$

where $|S|$ represents the number of triples in the sub-segment.Associative knowledge corresponds to a chain-like transmission structure (that is, the object of the preceding triplet is the subject of the subsequent triplet). This characteristic can be formally expressed as:

$$f_{\text{assoc}}(S) = \frac{1}{|S| - 1} \sum_{i=1}^{|S|-1} I(o_i = s_{i+1}) \quad (3)$$

where o_i and s_i respectively represent the object and the subject in the triplet.Causal knowledge corresponds to a star-shaped aggregation structure, and the system prioritizes aggregating multiple pieces of evidence pointing to the same entity.This characteristic can be formally expressed as:

$$f_{\text{causal}}(S) = \frac{\max_{o \in O} \sum_{i=1}^{|S|} I(o_i = o)}{|S|} \quad (4)$$

Let the unknown item to be solved in the triplet be denoted as A. Then, a question-type triplet can be formally represented as Triple⟨A, P, O⟩, where P is the predicate and O is the object. For simple questions consisting of only a single triplet, the knowledge requirements fall within the typical realm of factual knowledge, and the corresponding formula is:

$$A = \text{Triple}\langle A, P, O\rangle. \quad (5)$$

For complex questions involving multiple entities, if it is a multi-hop link structure, it can be represented as a recursive chain structure such as $\text{Triple}\langle S_1, P_1, A_1\rangle$, $\text{Triple}\langle A_1, P_2, A_2\rangle \ldots$, and at this time, each triplet needs to be solved in sequence according to the link order, and the obtained answers are passed to the subsequent triplets as input. The corresponding formula is as follows:

$$A = \sum_{i=1}^{n} \text{Triple}\langle A_i, P, O_{i+1}\rangle. \tag{6}$$

If it is a relationship network structure, it can be represented as $\text{Triple}\langle A_1, P_1, A_1\rangle$, $\text{Triple}\langle A_1, P_2, A_2\rangle$, $\text{Triple}\langle A_1, P_3, A_3\rangle \ldots$, which is a graph structure with one-to-many or many-to-one relationships. At this time, the first triplet to be answered in the network needs to be solved first, and after obtaining its answer, the corresponding variables in all related triplets are updated simultaneously, and then the entire question-solving process is completed by recursively following the network topology order.The corresponding formula is:

$$A = \text{Triple}\langle A_k, P, O\rangle + \sum_{i=1}^{n} \text{Triple}\langle A_i, P, A_k\rangle. \tag{7}$$

3.4 Multi-Source Retrieval and Chain Reasoning

After obtaining the sub-problem triplets sets $T = \{T_1, T_2, \ldots, T_n\}$, since their discrete structured representation is difficult for LLMs to directly understand, we propose a graph-text complementary retrieval augment method. This method combines the triplet structure with text semantics, providing multi-source knowledge input to the model.

For the multiple question triples $T_i = [S, P, A]$ obtained previously, where S is the subject, P is the predicate, and A is the answer to be found. These triples form inference links $L_j = [T_1 \rightarrow T_2 \rightarrow \cdots \rightarrow T_m]$ based on entity linking relationships, satisfying: $T_k.A = T_{k+1}.S, \forall k \in [1, m-1]$

All the links' endpoints point to the same target entity E_{target}, which is the solution objective of the original problem. To address the semantic sparsity issue of the knowledge graph, we adopt a text-enhanced graph reasoning approach. For each triple T_i, relevant text fragments D_i are retrieved through search, and the following solution process is constructed.

$$\begin{aligned} A_i = \text{LLM}\big(&\text{KG-Query}(S_i, P_i) \\ &\oplus \text{Retrieve}(D_i \mid S_i, P_i)\big) \end{aligned} \tag{8}$$

This process utilizes textual information to assist in understanding the graph structure and validates the rationality of the reasoning results through textual evidence. For multi-entailed entities, semantic disambiguation is carried out based on the text context. After obtaining the textual content of each triplet, the graph structure is further utilized to guide the text understanding. Define the anchor point set: $F = \{(S_i, P_i) \mid T_i \in T\}$. Based on these anchors, relevant semantic fragments are located in the text, and logical connections between these semantic text fragments are established based on the graph structure to form a coherent chain of reasoning evidence. Finally, after obtaining the enhanced knowledge representation $K = G \cup D$ (where G is the graph structure information and D is the textual semantic information), LLM can accurately answer each sub-question. Since the set of sub-questions completely covers the solution premise of the original problem, the final answer can be generated by integrating the answers to the sub-questions:

$$\text{Answer}(Q) = \text{LLM}\left(\bigcup_{i=1}^{n} \text{Answer}(T_i)\right) \tag{9}$$

4 Experiments

We evaluated Trig-Nav and a series of different benchmarks. Next, we will first introduce the datasets, benchmarks, evaluation metrics, and implementation details used in the experiment. Finally, we will present and analyze the experimental results.

4.1 Experiments Setup

Dataset. In terms of datasets, we use both single-hop and multi-hop datasets. For the single-hop datasets, we use: (1) Natural Questions (NQ) (Kwiatkowski et al., 2019): a dataset released by Google containing real user search queries; (2) Trivia-QA (Joshi et al., 2017): a dataset supporting complex reasoning and reading comprehension; (3) HotpotQA (Yang et al., 2018): A diverse and hybrid multi-hop question-answering dataset; (4) 2WikiMultiHopQA (Ho et al., 2020): a multi-hop question-answering dataset based on Wikipedia.

Baseline. We compare Trig-Nav with the following three types of baselines: traditional methods, text

processing RAG, and knowledge graph processing RAG.

For traditional method, we introduce (1)Vanilla LLM (Brown et al., 2020): directly using LLM to answer the questions. (2)Direct RAG(Lewis et al., 2020): this method retrieves the questions directly and does not process the retrieval results. It directly provides them to the LLM to generate answers based on the questions and the retrieval results.

For text RAG method, we consider evaluating those that use text processing. Specifically, these include: (1) BlendFilter (Wang et al., 2024): This method trains a knowledge filtering model, through which the knowledge provided to the LLM is screened to enhance the generation quality of the LLM. (2) DRAGIN(Su et al., 2024): This method introduces an information demand detection module, which dynamically decides whether to introduce external knowledge by monitoring in real time the LLM's need for external knowledge when generating answers. (3) MetaRAG (Zhou et al., 2024): MetaRAG enables the model to monitor, evaluate and plan its response strategies through metacognition, thereby enhancing its introspective reasoning ability. (4)FKE (Han et al., 2025) : This method develop a decoding enhancement strategy to constrain the document-based decoding process using fine-grained knowledge, thereby facilitating more accurate generated answers. (5) Adaptively RAG(Jeong et al., 2024): This method trains a small model as a classifier to assess the complexity of a given query. Based on this classification, it decides whether retrieval is necessary and to what extent retrieval should be conducted. (6)MAIN-RAG(Chang et al., 2025): This method achieves retrieval optimization by establishing multiple agents.

For Graph Processing RAG, We evaluate: (1) SimGRAG (Cai et al., 2025): This method determines the retrieval strategy by quantifying the alignment between the query and the candidate subgraphs, and has developed an optimized retrieval algorithm.(2) Graph RAG : This method directly uses the knowledge graph for retrieval.

Metrics. We use exact Match (EM) and F1 as the evaluation metrics for our QA tasks. EM measures whether the answer generated by the LLM exactly matches the correct answer. F1 is the average of precision and recall, and it can comprehensively reflect the performance of the results. We use EM as our main metric.

Details. To reduce excessive computational overhead, we randomly sample 500 samples from each dataset for testing. For the retriever, we use BM25 (Askari et al., 2023), which retrieves documents based on term frequency, document length, and inverse document frequency. We use wikipedia-dpr-100w-2021 as our external knowledge base. LLaMa3-8B is employed as both the main model.The Qwen2.5-7B model was used for comparison verification.

4.2 Main Results

We conducted a comprehensive evaluation of Trig-Nav's performance across four benchmark datasets: 2WikiMultihopQA, HotpotQA, Natural Questions (NQ), and TriviaQA. The experimental results, detailed in Table 1, demonstrate that integrating retrieval augmentation significantly boosts LLM performance. Specifically, our proposed method achieved superior results across all datasets, with the most notable gains observed on the NQ dataset. Compared to the vanilla LLM baseline, Trig-Nav yielded a substantial performance increase of approximately 23%. This remarkable improvement is largely attributed to the single-hop, factoid nature of NQ questions. These queries typically feature clear, objective contexts but often require precise entity extraction. Standard retrieval methods often retrieve noisy passages that distract the model; however, by extracting the structure of the query and optimizing the retrieval granularity, Trig-Nav filters out irrelevant noise. This ensures the LLM is grounded in precise contextual evidence, thereby minimizing hallucinations and directly enhancing answer accuracy.

Conversely, the performance gains on the 2WikiMultihopQA dataset were relatively modest. This dataset demands robust multi-hop reasoning capabilities to navigate complex evidence chains. While standard RAG enhancement is beneficial, it is often insufficient to fully resolve intricate dependency paths without more structural guidance.

As indicated in Table 1, Adaptive-RAG performs strongly on multi-hop datasets like 2WikiMultihopQA and HotpotQA. This is likely due to its iterative retrieval mechanism, which aggregates information across multiple steps to support complex reasoning. However, this iterative process incurs a significant computational cost, reducing generation efficiency. In contrast, Trig-Nav strikes a superior balance between performance and efficiency. Notably, our method achieves state-of-the-art perfor-

Method Type	Method	NQ		Trivia-QA		HotpotQA		2WikiMultiHopQA	
		EM	F1	EM	F1	EM	F1	EM	F1
Traditional Method	Vanilla LLM (Brown et al., 2020)	25.80	11.62	46.40	44.65	23.00	25.38	25.20	26.20
	Direct RAG (Lewis et al., 2020)	50.42	44.31	62.02	62.61	34.80	33.86	29.60	30.24
Text Processing RAG	FKE(Han et al., 2025)	40.89	41.31	27.10	28.20	25.20	27.56	22.40	24.76
	Adaptive-RAG(Jeong et al., 2024)	37.80	38.23	52.20	52.71	42.00	41.09	**40.60**	**41.32**
	MAIN-RAG(Chang et al., 2025)	51.60	30.95	66.40	59.88	30.4	31.22	32.19	12.02
	BlendFilter (Wang et al., 2024)	49.79	49.02	53.22	55.75	31.40	30.98	24.00	22.74
	DRAGIN(Su et al., 2024)	48.32	49.20	62.2	59.03	31.40	32.03	30.40	31.00
	MetaRAG(Zhou et al., 2024)	44.07	43.32	60.32	61.05	28.03	29.32	26.02	25.32
Knowledge Graph Processing RAG	SimGRAG(Cai et al., 2025)	44.07	43.32	59.03	58.74	28.03	29.32	33.20	25.73
	Graph RAG	45.2	42.73	63.43	62.32	32.05	31.0	30.21	31.54
	Ours(Qwen)	52.21	38.17	65.6	61.11	41.4	45.63	34	34.08
	Ours	**56.85**	**48.26**	**67.00**	**67.55**	**43.8**	**48.04**	34.8	37.3

Table 1: Comparison of Trig-Nav's performance on four datasets and other baselines. The best results are in bold

	NQ	Trivia-QA	HotpotQA	2Wiki
judge	1.42	2.42	1.23	2.32
no_judge	1.53	3.85	1.38	2.92

Table 2: The average processing seconds for each question.

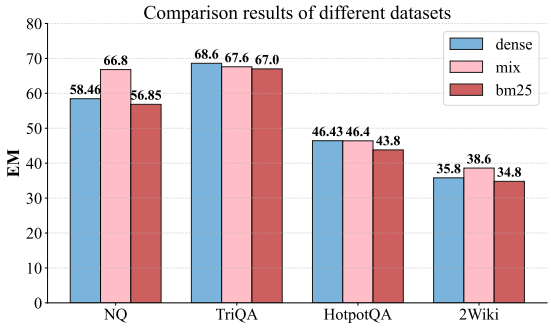

Figure 3: The impact of different retrieval methods on the performance of Trig-Nav.

mance on the TriviaQA dataset. This success stems from our optimization strategy, which refines and simplifies complex problem structures. By decomposing questions and reducing the cognitive load on the LLM, Trig-Nav enables more effective analysis and response generation for compositionally complex queries.

4.3 Ablation Study

The experimental results show that with the increase of the amount of retrieved knowledge, LLM acquires more comprehensive knowledge, thereby significantly improving its accuracy in answering multi-hop questions.This also indicates the necessity of our research. By conducting a necessary analysis of questions of different complexities and then providing the necessary knowledge for the question, it prevents LLM from being misled by redundant knowledge on the premise of obtaining sufficient reference information.

We conducted ablation studies to evaluate the effectiveness of different modules in our method. In the experiment, we removed retrieval modules (w/o. retrieval), text module (w/o. text), graph module (w/o. graph),multi-link retrieval module, and decompose module (w/o. decompose). We used Exact Match (EM) as the metric for evaluation. The results, as shown in Table 3, indicate that when the LLM completely forgoes retrieval, its performance in answering questions is significantly worse, with an approximate decline of 25%. To verify the roles of text knowledge and graph knowledge in our method, we attempted the effects of not using text knowledge and not using graph knowledge respectively. It can be seen that the performance has declined to varying degrees compared with combining the two types of knowledge, indicating that integrating both can better assist LLMs in solving questions.

To examine the impact of different types of knowledge on LLMs' question-solving, we tested the effect of removing multi-link retrieval and found that retrieving only one type of knowledge would cause LLMs' performance to drop by approximately 3%. To test the role of sub-question answers in our method, we retrieved the sub-questions and let LLMs generate answers, then provided the answers as reference information to LLMs and asked them to generate answers based only on the sub-question answers and their own knowledge without the reference of retrieved knowledge. The results showed that the performance improved by about 5% on all datasets except HotpotQA, and by 9% on the HotpotQA dataset. We speculate that

	NQ	2WikiMultiHopQA	Trivia-QA	HotpotQA
w/o. retrieval	31.52	26.4	46	20.44
w/o. text	55.64	30.2	63.2	37.4
w/o. graph	54.83	31	65.5	35.6
w/o. multi-link	52.45	32.8	63.4	39.6
w/o. decompose	50.42	29.60	62.02	34.80
Ours	**56.85**	**34.8**	**67.32**	**43.8**

Table 3: Ablation study for removing each single module on the Trig-Nav in terms of EM.

this is because complex questions often involve multiple aspects of knowledge, and direct retrieval is prone to returning redundant or irrelevant content. Decomposed sub-questions focus on a single aspect, making the retrieval results more precise.

To verify the influence of the judge module on the system's inference latency, we conducted a targeted efficiency experiment. As evidenced by the data in Table 2, integrating the judge module significantly accelerates the processing speed for each question, leading to reduced response times and an enhanced user experience. This efficiency gain stems from the module's ability to preemptively filter out unnecessary retrieval operations, thereby avoiding the substantial computational overhead associated with excessive context searching. Furthermore, our ablation study on the question decomposition module reveals a critical insight: removing this component results in a performance degradation of up to approximately 8%. This sharp decline confirms that decomposing complex queries into granular sub-tasks is essential for reducing reasoning difficulty, ultimately ensuring higher accuracy in the final answers.

To evaluate the impact of retrieval strategies on Trig-Nav, We compared the performance of Trig-Nav using dense, sparse (based on BM25) and hybrid retrievers on four datasets, as shown in Figure 3. We find that Trig-Nav possesses excellent adaptability independent of the underlying retriever. On TriQA and HotpotQA, the EM scores are highly stable (67.0%–68.6% and 43.8%–46.4%), proving our model's resilience to retrieval quality fluctuations. On NQ and 2Wiki, although hybrid retrieval further improves the upper bound, we show that using only dense or BM25 still yields competitive performance. Ultimately, we conclude that Trig-Nav's effectiveness stems from its core architectural design rather than specific high-precision retrievers, allowing for robust adaptation to various real-world systems.

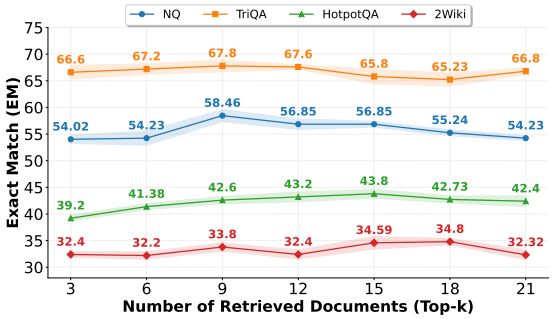

Figure 4: The impact of the number of retrieved texts on the performance of LLM under different datasets.The shadow indicates the error range.

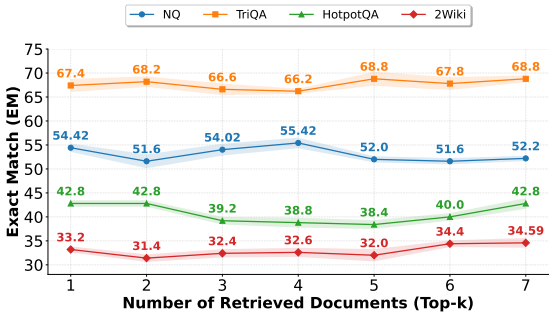

Figure 5: The impact of the reduction of noise in knowledge graphs on the performance of LLMs, with the number of texts changing from 9 to 21 for top-k.

4.4 Retrieval Analysis

We evaluated the impact of varying the top-k retrieved results on LLM performance. As shown in Figure 4, increasing k had negligible or even negative effects on single-hop datasets (NQ, TriviaQA) due to potential noise. Conversely, multi-hop datasets (2WikiMultihopQA, HotpotQA) showed a positive correlation between k and EM, indicating a benefit from broader context. These trends were further validated using Qwen-7B to ensure model agnosticism (Table 1).

Further analysis of larger retrieval sets, as illustrated in Figure 5, revealed a distinct rise-then-fall performance trajectory. This pattern suggests the

existence of an information capacity limit within the generation process: while a moderate amount of context bridges knowledge gaps, continuing to increase retrieval quantity eventually reaches a tipping point where the introduction of irrelevant noise outweighs the benefits of additional information. The varying thresholds for performance decline observed across different datasets reflect their distinct knowledge requirements and sensitivity to noise. These findings validate our proposed strategy of classifying knowledge types for targeted retrieval, demonstrating that blindly increasing the retrieval window is suboptimal compared to a precision-oriented approach.

The experimental results show that with the increase of the amount of retrieved knowledge, LLM acquires more comprehensive knowledge, thereby significantly improving its accuracy in answering multi-hop questions.This also indicates the necessity of our research. By conducting a necessary analysis of questions of different complexities and then providing the necessary knowledge for the question, it prevents LLM from being misled by redundant knowledge on the premise of obtaining sufficient reference information.

5 Conclusion

We introduce Trig-Nav, an adaptive retrieval-augmented method based on the integration of multi-perspective graph-text knowledge. It first dynamically decomposes the original question into a series of interrelated sub-questions based on the difficulty of the question and the internal knowledge of the LLM. Knowledge graphs are constructed based on the retrieved knowledge for each sub-question, and each sub-question is further divided into multiple triples to be processed. For each sub-question, the system simultaneously initiates three structured retrieval chains. Subsequently, the framework aligns and associates the retrieved structured triples with relevant unstructured text descriptions to generate refined text evidence blocks rich in semantic associations. In the answer generation stage, LLM answers the original question based on the answers to the sub-questions. We evaluate our method on four datasets using multiple metrics, and the results show that our method effectively improves the shortcomings of simple RAG and achieves significant advantages.

6 Limitations

Our approach relies on the necessity judgment of RAG and the construction of knowledge graphs through the utilization of LLM. When the internal knowledge base of the LLM is relatively limited or the LLM has difficulty following the given prompt for output, it will significantly affect the effect of our method. Furthermore, when dealing with complex multi-hop reasoning questions, our method fails to effectively enhance the inherent reasoning ability of LLM.

Acknowledgments

This work was supported in part by the National Key R&D Program of China under Grant 2023YFF0905503, National Natural Science Foundation of China under Grants No.62472188.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *Preprint*, arXiv:2310.11511.

Arian Askari, Amin Abolghasemi, Gabriella Pasi, Wessel Kraaij, and Suzan Verberne. 2023. Injecting the bm25 score as text improves bert-based re-rankers. In *European Conference on Information Retrieval*, pages 66–83. Springer.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, and 9 others. 2022. Improving language models by retrieving from trillions of tokens. *Preprint*, arXiv:2112.04426.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Yuzheng Cai, Zhenyue Guo, Yiwen Pei, Wanrui Bian, and Weiguo Zheng. 2025. Simgrag: Leveraging similar subgraphs for knowledge graphs driven retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 3139–3158.

Chia-Yuan Chang, Zhimeng Jiang, Vineeth Rakesh, Menghai Pan, Chin-Chia Michael Yeh, Guanchu Wang, Mingzhi Hu, Zhichao Xu, Yan Zheng, Mahashweta Das, and 1 others. 2025. Main-rag: Multi-agent filtering retrieval-augmented generation. In

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2607–2622.

Zhanpeng Chen, Chengjin Xu, Yiyan Qi, and Jian Guo. 2024. Mllm is a strong reranker: Advancing multimodal retrieval-augmented generation via knowledge-enhanced reranking and noise-injected training. *Preprint*, arXiv:2407.21439.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2022. Precise Zero-Shot Dense Retrieval without Relevance Labels. *arXiv e-prints*, arXiv:2212.10496.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-Augmented Language Model Pre-Training. *arXiv e-prints*, arXiv:2002.08909.

Jingxuan Han, Zhendong Mao, Yi Liu, Yexuan Che, Zheren Fu, and Quan Wang. 2025. Fine-grained knowledge enhancement for retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 10031–10044.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C. Park. 2024. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. *Preprint*, arXiv:2403.14403.

Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active Retrieval Augmented Generation. *arXiv e-prints*, arXiv:2305.06983.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, and 1 others. 2022. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2019. Generalization through Memorization: Nearest Neighbor Language Models. *arXiv e-prints*, arXiv:1911.00172.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Rich James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, Luke Zettlemoyer, and Scott Yih. 2024. Ra-dit: Retrieval-augmented dual instruction tuning. *Preprint*, arXiv:2310.01352.

Zihan Liu, Wei Ping, Rajarshi Roy, Peng Xu, Chankyu Lee, Mohammad Shoeybi, and Bryan Catanzaro. 2024. Chatqa: Surpassing gpt-4 on conversational qa and rag. *Preprint*, arXiv:2401.10225.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting for retrieval-augmented large language models. *Preprint*, arXiv:2305.14283.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Shiyu Ni, Keping Bi, Jiafeng Guo, and Xueqi Cheng. 2024. When do llms need retrieval augmentation? mitigating llms' overconfidence helps retrieval augmentation. *Preprint*, arXiv:2402.11457.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Dongyu Ru, Lin Qiu, Xiangkun Hu, Tianhang Zhang, Peng Shi, Shuaichen Chang, Cheng Jiayang, Cunxiang Wang, Shichao Sun, Huanyu Li, Zizhao Zhang, Binjie Wang, Jiarong Jiang, Tong He, Zhiguo Wang, Pengfei Liu, Yue Zhang, and Zheng Zhang. 2024. Ragchecker: A fine-grained framework for diagnosing retrieval-augmented generation. *Preprint*, arXiv:2408.08067.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. *Preprint*, arXiv:2305.15294.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. 2023. Large Language Models Can Be Easily Distracted by Irrelevant Context. *arXiv e-prints*, arXiv:2302.00093.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2024. REPLUG: Retrieval-augmented black-box language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8371–8384, Mexico City, Mexico. Association for Computational Linguistics.

Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. 2024. Dragin: dynamic retrieval augmented generation based on the information needs of large language models. *arXiv preprint arXiv:2403.10081*.

Haoyu Wang, Ruirui Li, Haoming Jiang, Jinjin Tian, Zhengyang Wang, Chen Luo, Xianfeng Tang, Monica Xiao Cheng, Tuo Zhao, and Jing Gao. 2024. Blendfilter: Advancing retrieval-augmented large language models via query generation blending and knowledge filtering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1009–1025.

Shijie Wang, Wenqi Fan, Yue Feng, Lin Shanru, Xinyu Ma, Shuaiqiang Wang, and Dawei Yin. 2025. Knowledge graph retrieval-augmented generation for llm-based recommendation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 27152–27168.

Yile Wang, Peng Li, Maosong Sun, and Yang Liu. 2023. Self-knowledge guided retrieval augmentation for large language models. *Preprint*, arXiv:2310.05002.

Derong Xu, Xinhang Li, Ziheng Zhang, Zhenxi Lin, Zhihong Zhu, Zhi Zheng, Xian Wu, Xiangyu Zhao, Tong Xu, and Enhong Chen. 2025. Harnessing large language models for knowledge graph question answering via adaptive multi-aspect retrieval-augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25570–25578.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024. Making retrieval-augmented language models robust to irrelevant context. *Preprint*, arXiv:2310.01558.

Yue Yu, Wei Ping, Zihan Liu, Boxin Wang, Jiaxuan You, Chao Zhang, Mohammad Shoeybi, and Bryan Catanzaro. 2024. Rankrag: Unifying context ranking with retrieval-augmented generation in llms. *Preprint*, arXiv:2407.02485.

Yanzhao Zhang, Dingkun Long, Guangwei Xu, and Pengjun Xie. 2022. Hlatr: Enhance multi-stage text retrieval with hybrid list aware transformer reranking. *Preprint*, arXiv:2205.10569.

Zhuocheng Zhang, Yang Feng, and Min Zhang. 2025. Levelrag: Enhancing retrieval-augmented generation with multi-hop logic planning over rewriting augmented searchers. *Preprint*, arXiv:2502.18139.

Chunting Zhou, Graham Neubig, Jiatao Gu, Mona Diab, Paco Guzman, Luke Zettlemoyer, and Marjan Ghazvininejad. 2021. Detecting hallucinated content in conditional neural sequence generation. *Preprint*, arXiv:2011.02593.

Yujia Zhou, Zheng Liu, Jiajie Jin, Jian-Yun Nie, and Zhicheng Dou. 2024. Metacognitive retrieval-augmented large language models. *Preprint*, arXiv:2402.11626.

Xiangrong Zhu, Yuexiang Xie, Yi Liu, Yaliang Li, and Wei Hu. 2025. Knowledge graph-guided retrieval augmented generation. *Preprint*, arXiv:2502.06864.