MMAPG: A Training-Free Framework for Multimodal Multi-hop Question Answering via Adaptive Planning Graphs

Anonymous ACL submission

Abstract

Multimodal Multi-hop question answering requires integrating information from diverse sources, such as images and texts, to derive answers. Existing methods typically rely on sequential retrieval and reasoning, where each step builds on the previous output. However, this single-path paradigm makes them vulnerable to errors due to misleading intermediate steps. Moreover, developing multimodal models can be computationally expensive, often requiring extensive training. To address these limitations, we propose a training-free framework guided by an Adaptive Planning Graph, which consists of planning, retrieval and reasoning modules. The planning module analyzes the current state of the Adaptive Planning Graph, determines the next action and where to expand the graph, which enables dynamic and flexible exploration of reasoning paths. To handle retrieval of text to unspecified target modalities, we devise modality-specific strategies that dynamically adapt to distinct data types. Our approach preserves the characteristics of multimodal information without costly task-specific training, enabling seamless integration with up-to-date models. Finally, the experiments on MultimodalQA and WebQA show that our approach matches or outperforms existing models that rely on training.

1 Introduction

The field of question answering(QA) has gained significant attention and is increasingly applied across various domains, including customer support, healthcare, and education, particularly with the rapid advancements driven by large language models (LLMs) (Su et al., 2019; Lu et al., 2022; Wei et al., 2022; Shao et al., 2023; He et al., 2024). These models have demonstrated strong performance in single-hop QA. However, multimodal multi-hop QA (Talmor et al., 2021; Chang et al., 2022) presents a greater challenge, as it requires integrating diverse sources. As illustrated in Figure 1,

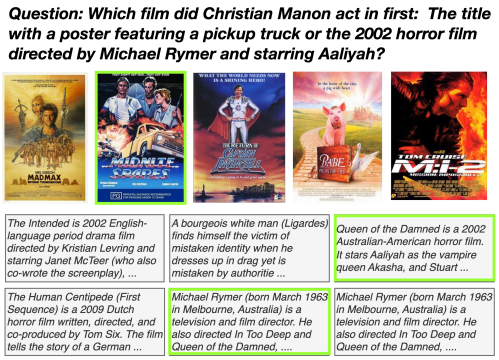

Figure 1: An example of multimodal multi-hop QA. It requires identifying relevant information (bounded by *green box*) from diverse sources to generate answers.

relevant information must be identified across multiple sources with different modalities to generate answers. In these settings, only a subset of sources is relevant, while others introduce noise. Solving this task requires approaches that effectively integrate both retrieval and reasoning capabilities.

Current research in this area centers around two main paradigms. The first approach adopts a two-stage framework (Yu et al., 2023; Liu et al., 2023; Lim et al., 2024). It retrieves all potentially relevant information in a single step, followed by answer generation based on the retrieved information. The two stages are trained independently with distinct objectives, which can lead to a misalignment. Specifically, the reasoning stage implicitly assumes that the retrieved sources are complete and accurate, introducing fragility as early retrieval failures cannot be rectified during generation. For example, in Figure 2, the retrieval results linked to *animated TV show* and *yellow gloves* involve multiple false positives, and overlook the correct poster. In the next stage, the system is forced to reason from the incorrect contexts, leading to inaccurate answer.

The second mainstream method employs an iterative approach (Trivedi et al., 2022; Yang et al., 2023a; Zhang et al., 2024), which offers greater

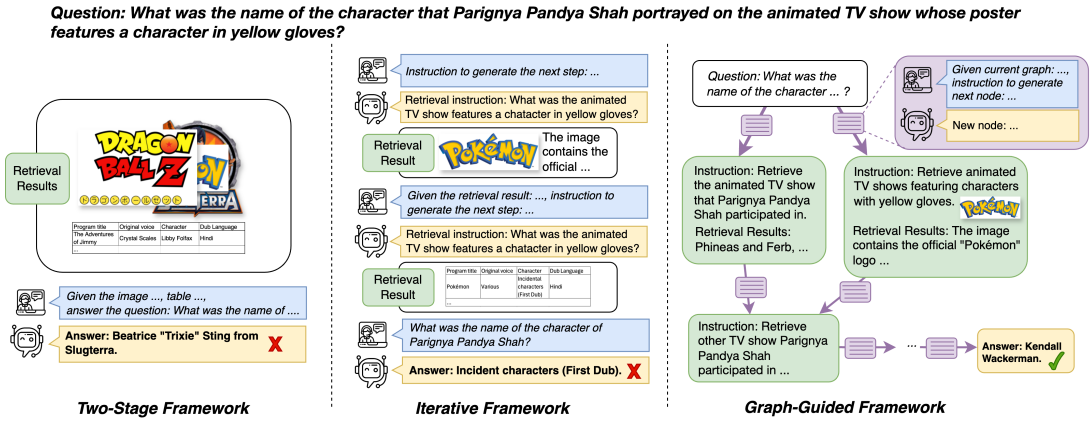

Figure 2: A representative case illustrating how each framework handles deviations in the reasoning process to reach the correct answer. We prompt the LLM to operate following the paradigm of each framework, comparing how structural differences affect reasoning under the same conditions.

flexibility as it has no restrictions on the number of steps, allowing for the integration of additional sources as needed. The structure of these methods follow a single-path paradigm, where actions proceed in a fixed pattern. This design introduces cascading error propagation, which relies on the model's inherent reasoning ability to correctify. For instance, in Figure 2, if an initial retrieval incorrectly identifies details about the TV show *Pokémon*, this mistake could mislead the next subquestion, ultimately leading to an incorrect result along a single-path with flaw. The challenge is further amplified in multimodal settings, where the modality involved in each step is often unknown. To handle different types of sources, SKURG (Yang et al., 2023a) fuses image and text embeddings to create an entity-centered representation, which relies on extensive training of existing models. ETG (Zhang et al., 2024) retrieves evidences by converting images into texts during preprocessing, which may result in information loss. and omit relevant contexts. These limitations highlight the need for developing methods that can dynamically adapt to multimodal sources.

To address these issues, we introduce a training-free Adaptive Planning Graph-guided approach. As illustrated in Figure 2, both the two-stage and iterative framework rely on a single-path paradigm, where each step strictly depends on the previous one. Consequently, errors in earlier steps can propagate and affect the final answers. In our framework, we adopt a Adaptive Planning Graph, where each node represents a thought or a result generated by module, and edges denote the dependencies between nodes. It presents a more flexible flow, allowing the new steps to be dynamically generated from any relevant node in the Adaptive Planning Graph. The proposed method consists of planning, retrieval and reasoning modules. At each step, The planning module generates a plan for one action on-the-fly. It continuously analyzing the current state to determine the next move. To facilitate search of contexts across multiple modalities, existing works either convert images into texts which may result in incomplete captions, or rely on resource-intensive pretraining. To address these limitations, our module constructs separate knowledge bases and uses modality-specific strategies to collect relevant information. It enables effective retrieval while preserving modality details and avoiding additional computational costs. The main contributions of our paper are presented as follows:

1. We introduce a novel framework, MMAPG (Multimodal Multi-hop Adaptive Planning Graph), that offers enhanced flexibility for tackling multimodal multi-hop QA. By constructing the Adaptive Planning Graph step-by-step, our approach facilitates dynamic exploration of different sources and supports a graph-based reasoning flow. To the best of our knowledge, this is the first work using graph-based planning for multimodal multi-hop QA.

2. We address multimodality with a training-free framework by employing distinct off-the-shelf within specialized modules. Our proposed modules preserve the details of each modality while leveraging the generated rationale to support the construction of the Adaptive Planning Graph.

3. We conduct experiments on MultimodalQA and WebQA datasets. The results demonstrate that our model performs comparably or better than trained models.

2 Related Works

2.1 Multimodal Multi-hop QA

The first mainstream approach to solve multimodal multi-hop QA is the two-stage framework. For instance, Solar (Yu et al., 2023) and UniRAG (Lim et al., 2024) both unify multimodal sources into texts, retrieve top-k results and employ language models to generate the final response. To handle multimodality, AutoRouting and ImplicitDecomp (Talmor et al., 2021) fine-tune models to answer questions depending on modality identified by a classifier. Meanwhile, PERQA (Yang et al., 2023b) employs an iterative evidence selection process and incorporates multimodal reasoning during the generation phase. In contrast to the methods previously discussed, which all involve training or fine-tuning, MMHQA-ICL (Liu et al., 2023) represents a training-free paradigm, which autonomously generates prompts for in-context learning.

The second approach focuses on an iterative framework, which is widely used in single-modality QA (Trivedi et al., 2022). However, its application in multimodal multi-hop QA is less common. A notable example is SKURG (Yang et al., 2023a), which introduces a unified retrieval and generation module that iteratively integrates multimodal information. A recent work, ETG (Zhang et al., 2024), proposes a mixture-of-experts approach to combine retrieval and generation, with reasoning represented through an entailment tree.

2.2 Chain of thought

The chain of thought (CoT) reasoning has significantly improved LLMs' reasoning abilities. It inspires approaches that prompt LLMs to generate full reasoning at once (Kojima et al., 2022) or incrementally (Xu et al., 2023; Shen et al., 2024). While these methods follow a single-path paradigm, multi-path approaches like CoT-SC (Wang et al., 2022), Tree-of-Thought (ToT) (Yao et al., 2024), and Graph-of-Thought (GoT) (Besta et al., 2024) explore multiple reasoning paths and decision-making processes. Notably, GoT requires users to manually define the execution plan, making it less flexible for QA tasks that require adaptive plans for different questions.

2.3 Multimodal Retrieval

Retrieval across various modalities has been extensively researched, usually with fixed source and target modalities, such as text-to-image, image-to-text, or image-text pair to image retrieval. However, retrieval without predefined target modalities has received less attention. Previous research (Mayilvahanan et al., 2023) has demonstrated that similarity scores between intra-modalities and inter-modalities exhibit different distributions, presenting inherent challenges in this area. MuRAG (Chen et al., 2022) pretrains a multimodal retrieval model, but requires collecting a large number of samples. REVEAL (Hu et al., 2023) involves pretraining and developing gating score to select dataset. These approaches illustrate that facilitating multi-modal retrieval often relies on costly resources. Other works (Yu et al., 2023; Liu et al., 2023) attempts to convert all the images into texts to address the challenges of multimodal retrieval. However, questions can focus on various details within an image, and these critical information may not be preserved when converting to texts.

3 Methods

We present our motivation for graph-guided framework in Section 3.1. Then we depict the main modules in our framework MMAPG, including knowledge base construction (Section 3.2), planning module (Section 3.3) and retrieval module (Section 3.4). The overall workflow is detailed in Appendix A.2. Since the reasoning module is simply implemented by calling an off-the-shelf model to generate the answer, we omit its discussion here, where its details can be found in Figure 3.

3.1 MMAPG Overview

In this section, we demonstrate how our graph-guided framework alleviates the limitations of two-stage and iterative framework based methods. To establish the framework, we prompt the LLM to analyze the current graph and generate instructions for next steps, as shown in Figure 2. Each step in the reasoning process is represented as a node, and we explicitly allow the new nodes to be created based on any existing node. This capability ensures that even if one path proves ineffective, the framework can still identify alternative paths based on the current nodes. In the example, when the retrieval of animated TV shows featuring two yellow gloves

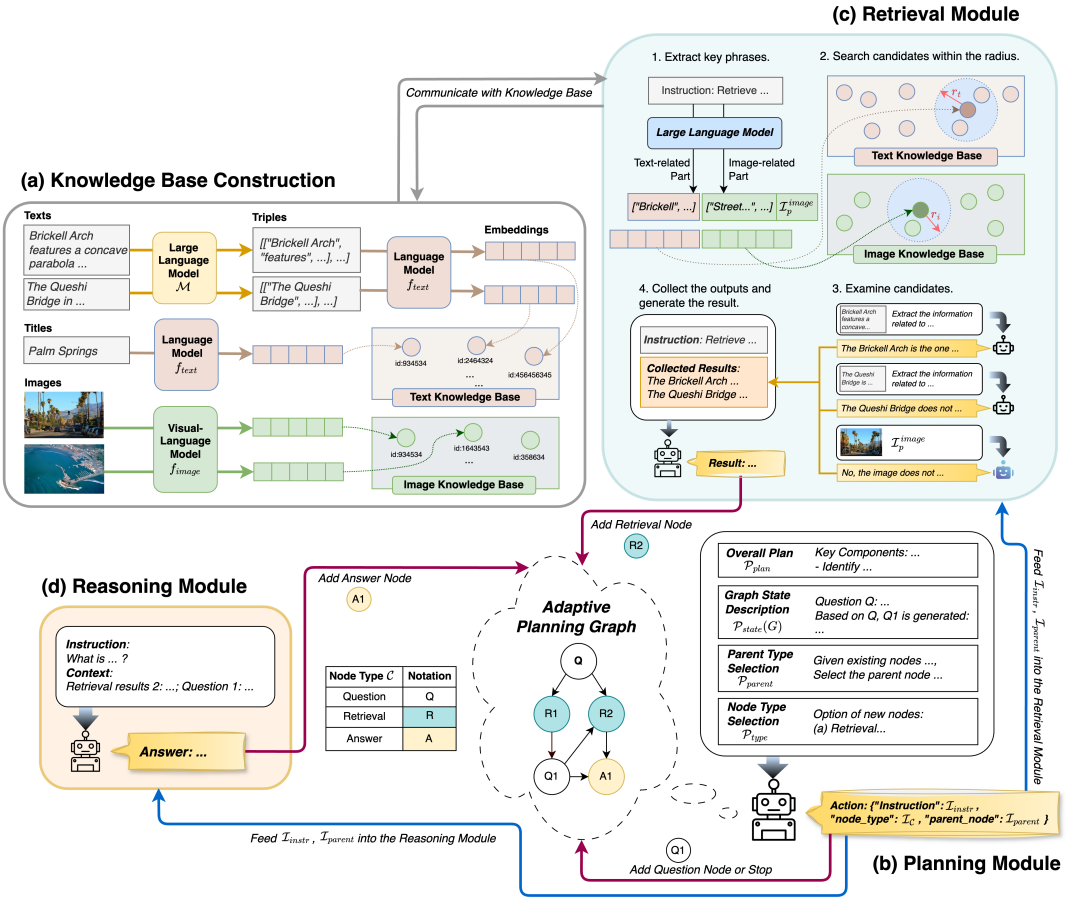

Figure 3: An overview of our framework MMAPG, which consists of four parts: **(a) Knowledge Base Construction**: precomputes embeddings for given sources; **(b) Planning Module**: determines the next action in the Adaptive Planning Graph and which module to call; **(c) Retrieval Module**: retrieve relevant sources and find key information; **(d) Reasoning Module**: derives the answer based on the provided instruction.

is unsuccessful, the system can combine existing nodes to explore other paths for retrieving relevant TV shows within a limited range. Compared to previous approaches, our paradigm enhances the flexibility and allows for a more adaptive process.

3.2 Knowledge Based Construction

Given a set of sources $S = \{S_1, \cdots, S_n\}$. Each source S_i may have associated text-based components T_i, (e.g., $T_i = \{T_i^{title}, T_i^{caption}\}$). We maintain separate knowledge base for text and images, denoted as KB_{text} and KB_{img}.

Specifically, the text knowledge space consists of embedding derived from textual sources and text-based image information, including image titles and image captions. Since different types of text-based information vary in length and contextual detail, which can impact the retrieval performance (Wang et al., 2024), to ensure consistent granularity, we decompose S_i into relational triplets $\mathcal{T}_i = \{\tau_{i1} \cdots, \tau_{im_i}\}$. Here, we employ a few-shot prompting method (Zhang and Soh, 2024) to extract triplets from S_i, and compute the embedding:

$$\mathcal{T}_i = \mathcal{M}(\mathcal{P}_{tri} \oplus \mathcal{F}_{tri} \oplus S_i), \quad (1)$$

$$\mathbf{e}_{ij}^{text} = f_{text}(\tau_{ij}), \quad (2)$$

where $\mathcal{M}$ represents the LLM used; $\mathcal{P}_{tri}$ is the prompt; $\mathcal{F}_{tri}$ is the few-shot examples; $\oplus$ means concatenation; f_{text} is the model to generate text embedding; $\mathbf{e}_{ij}^{text}$ is the computed embedding for triplet τ_{ij}. For short components, such as titles, we directly compute their embeddings. Finally, we construct the text knowledge base KB_{text} by storing all the text embeddings $\{\mathbf{e}_1^{text}, \cdots, \mathbf{e}_N^{text}\}$.

To construct image knowledge base KB_{img}, embeddings are directly computed as follows:

$$\mathbf{e}_i^{img} = f_{img}(S_i), \quad (3)$$

where f_{img} is the multimodal model to generate image embedding. Similarly, KB_{img} stores all the image embeddings $\{\mathbf{e}_1^{img}, \cdots, \mathbf{e}_M^{img}\}$.

3.3 Planning Module

In this section, we introduce the planning module, the core component for constructing the Adaptive Planning Graph, denoted as $G = (V, E)$. The node $v \in V$ represents the action at each stage, and directed edges $e \in E$ represent dependencies between nodes. Specifically, $e = (v_i, v_j)$ indicates that node v_j is built upon v_i, which reflects the logical flow of global reasoning. The Adaptive Planning Graph is built by adding new nodes along with their corresponding edges on the fly, where the generation of these nodes and edges are informed by the planning module at each step.

To harness the reasoning capability of LLMs for global planning, we design the prompt that serves two functions: analyzing the current state of the Adaptive Planning Graph as it evolves to answer the question and determining the next appropriate action. For the first function, we provide both an overall plan outlining the essential information needed and a summary of the current graph that reflects the progress made so far. Together, they offer the model a clear perspective of the information already gathered and what remains to be explored. For the second function, we present a set of options for expanding the graph. It allows the model to decide the most appropriate node to generate based on its prior analysis. To facilitate these processes, our prompts consists of four components: (1) Overall plan, denoted as $\mathcal{P}_{plan}$. Given that many LLMs exhibit limitations in handling long-term planning, we generate a high-level guide, which outlines the key components and possible global plans. It serves as a reference for expanding the Adaptive Planning Graph G. (2) Graph State Description, presented as $\mathcal{P}_{state}(G)$. It displays the current state of the Adaptive Planning Graph by describing the content of existing nodes and their dependencies. (3) Parent Node Selection Instruction, represented as $\mathcal{P}_{parent}$. It instructs the system to select parent nodes from all nodes in the graph G. (4) Node Type Selection Instruction, denoted as $\mathcal{P}_{\mathcal{C}}$. It provides a set of node types to select. We concatenate these components to build prompts that are fed into LLM $\mathcal{M}$:

$$\mathcal{I}_{\mathcal{C}}, \mathcal{I}_{parent}, \mathcal{I}_{instr} = \mathcal{M}(\mathcal{P}_{plan} \oplus \mathcal{P}_{state}(G) \oplus \mathcal{P}_{parent} \oplus \mathcal{P}_{\mathcal{C}}), \quad (4)$$

where $\mathcal{I}_{\mathcal{C}}, \mathcal{I}_{parent}, \mathcal{I}_{instr}$ are the type, parent nodes, and instruction for generating the content of the new node. The type of new node $\mathcal{C}_i \in \{Question, Answer, Retrieval, Stop\}$ is informed by $\mathcal{I}_{\mathcal{C}}$. It signifies the system of the next action to take. We present the instructions and actions according to each node type as follows:

- **Question**: The instruction is a direct question. Since no further processing is required, this question is taken as the new node content.
- **Answer**: The instruction specifies which question to be answered. The instruction and parent nodes are passed into the reasoning module to generate the corresponding answer. The answer is then added as the new node content.
- **Retrieval**: The instruction outlines the information to be retrieved. The instruction and parent nodes are passed to the retrieval module. If relevant information is found, it is added as the new node content. However, if no relevant information is retrieved, a node is created to indicate that no results were found.
- **Stop**: The action with this node is to terminate the process. It uses the content of the last answer node as the final answer. If an answer has not yet been generated, the LLM will generate one based on the overall graph.

The graph is then expanded by adding the new node along with the corresponding edges, which are determined by $\mathcal{I}_{parent}$. For instance, if $\mathcal{I}_{parent} = \{v_j\}$, a directed edge $e_{ij} = \{v_j, v_i\}$ will be added to the Adaptive Planning Graph, to indicate v_i is derived from the thought of v_j. The planning module will continuously plan the next step to update the graph until a stop node is generated or the maximum number of iteration is reached.

3.4 Retrieval Module

The retrieval module is responsible for extracting relevant information based on the given instruction and parent nodes. To eliminate training costs and enhance generalizability, we leverage off-the-shelf models to handle multimodality. However, retrieval across different source types can exhibit inherently different similarity score distributions, as we show in detail in Appendix A.1. To address this, we propose tailored strategies for different modalities. We firstly utilizes LLM to decompose the instruction into text-related and image-related components:

$$\mathcal{I}^{text}_{instr}, \mathcal{I}^{img}_{instr} = \mathcal{M}(\mathcal{P}_{decomp} \oplus \mathcal{I}_{parent} \oplus \mathcal{I}_{instr}), \quad (5)$$

where $\mathcal{P}_{decomp}$ is the prompt for decomposition. Next, we apply different methods to extract key elements from text-related and image-related parts.

For the text-related part, we employ few-shot examples mined from the dataset to identify key phrase. The queries utilized for text retrieval are generated as follows:

$$\mathcal{Q}_{text} = \mathcal{M}(\mathcal{P}_{extract} \oplus \mathcal{F}_{text} \oplus \mathcal{I}^{text}_{instr}), \quad (6)$$

where $\mathcal{P}_{extract}$ is the prompt for key phrase extraction, and $\mathcal{F}_{text}$ represents the set of few-shot examples to identify key phrases or words.

For the image-related part, the type of image retrieval is identified from $\mathcal{I}^{img}_{instr}$ firstly, which determines the action to be taken next. There are two types of image retrieval defined. The first one is targeted image retrieval. It occurs when the instruction $\mathcal{I}^{text}_{instr}$ mentions specific identifiers of a particular image, such as title. We expect these identifiers to be extracted and used to search the text knowledge base KB_{text}. The query $\mathcal{Q}'_{text}$ is generated by LLM as follows:

$$\mathcal{Q}'_{text}, \mathcal{I}^{img}_{tgt} = \mathcal{M}(\mathcal{P}_{tgt} \oplus \mathcal{F}_{tgt} \oplus \mathcal{I}^{img}_{instr}), \quad (7)$$

where $\mathcal{P}_{tgt}$ and $\mathcal{F}_{targe}$ are the prompt and few-shot examples for targeted image retrieval. The generated $\mathcal{I}^{img}_{tgt}$ will be used in later steps for candidate examination. The final query to search in text knowledge base KB_{text} is then the combination of $\mathcal{Q}_{text}$ and $\mathcal{Q}'_{text}$:

$$\mathcal{Q}_{text} = \mathcal{Q}_{text} \oplus \mathcal{Q}'_{text}. \quad (8)$$

The other type is descriptive image retrieval, which is utilized when a description about the image content is provided. In this case, the image-related part instruction $\mathcal{I}^{img}_{instr}$ usually contains descriptive text to guide the system in locating images that best match the description in KB_{img}. The queries are generated as follows:

$$\mathcal{Q}_{img}, \mathcal{I}^{img}_{descr} = \mathcal{M}(\mathcal{P}_{descr} \oplus \mathcal{F}_{descr} \oplus \mathcal{I}^{img}_{instr}), \quad (9)$$

where $\mathcal{P}_{descr}$, $\mathcal{F}_{descr}$ are the prompt and few-shot examples for descriptive image retrieval.

Next, we generate the corresponding embeddings for extracted queries. These embeddings are then searched in the corresponding knowledge base to identify matches within a defined radius r_t and r_i for KB_{text} and KB_{img} respectively. For example, when searching within the text knowledge base, we first compute the embedding for the extracted phrase q^{text}_i as follows:

$$\mathbf{e}^{text}_i = f_{text}(q^{text}_i). \quad (10)$$

Next, we identify the candidates from KB_{text} that are within the defined radius r_t from $\mathbf{e}^{text}_i$:

$$C^{text} = \{\mathbf{e}_j \in KB_{text} \mid \|\mathbf{e}_j - \mathbf{e}^{text}_i\| \leq r_t\}. \quad (11)$$

Then, we could derive k candidates as $C^{text} = \{c^{text}_1, \cdots, c^{text}_k\}$.

While retrieval based solely on similarity score may introduce many outliers, a more refined examination is conducted using off-the-shelf models. These models assist in verifying whether the content of the candidates is relevant to the instruction. For text-related parts, we directly instruct LLM to extract useful information O^{text}_i from c^{text}_i:

$$O^{text}_i = \mathcal{M}(\mathcal{P}^{text}_{exam} \oplus \mathcal{I}^{text}_{instr} \oplus c^{text}_i), \quad (12)$$

where $\mathcal{P}^{text}_{exam}$ is the prompt used for examining textual candidates. For image-related part, we instruct vision-language model $\mathcal{M}_{vl}$ to examine the image:

$$O^{img}_j = \mathcal{M}_{vl}(\mathcal{P}^{img}_{exam} \oplus \mathcal{I}^{img}_{p} \oplus c^{img}_j). \quad (13)$$

where $p \in \{descr, target\}$. Finally, all the examination results $O = \{O^{text}_1, \cdots, O^{img}_1, \cdots\}$ will be collected together. These compiled information, along with $\mathcal{I}_{instr}$, is fed into the LLM for information extraction:

$$R = \mathcal{M}(P_{retr} \oplus \mathcal{I}_{instr} \oplus O). \quad (14)$$

The final results R are then served as the content of the new retrieval node.

4 Experiments

4.1 Experimental Setup

We utilize MultimodalQA (Talmor et al., 2021) and WebQA (Chang et al., 2022) datasets for multimodal multi-hop QA evaluation. MultimodalQA is a dataset designed for question answering across text, tables, and images, where each question is accompanied by a set of distractors. The answers are evaluated using the F1 and exact match (EM) scores. WebQA consists of QA pairs along with images or text snippets, including distractors. The evaluation metrics for WebQA involve QA-Acc, keyword-based accuracy and QA-FL, which assesses fluency using BARTScore. The implementation details of our framework can be found in Appendix A.5.

Methods	Trained Models	Single-Modal		Multi-Modal		Overall	
		F1	EM	F1	EM	F1	EM
Fine-tuned							
AutoRouting (Talmor et al., 2021)	*RoBERTa, ViLBERT*	58.5	51.7	40.2	34.2	51.1	44.7
ImplicitDecomp (Talmor et al., 2021)	*RoBERTa, ViLBERT*	58.8	51.1	51.7	46.5	55.9	49.3
Solar (Yu et al., 2023)	*BERT, T5*	74.8	69.7	65.4	55.5	66.1	59.8
PERQA (Yang et al., 2023b)	*BERT, ViT+Llama+Lora*	74.1	69.7	60.3	54.7	67.8	62.8
SKURG (Yang et al., 2023a)	*OFA,BART*	69.7	66.1	57.2	52.5	64.0	59.8
ETG (Zhang et al., 2024)	*T5(MoE)*	74.9	**69.8**	**65.7**	**64.7**	66.5	**68.2**
W/o fine-tuning							
MMHQA-ICL (Liu et al., 2023)	-	72.9	60.5	55.5	46.2	65.8	54.8
MMAPG (ours)	-	**75.4**	65.2	65.0	51.9	**70.6**	59.1

Table 1: The comparison of different methods on MultimodalQA dataset. We report the F1 and EM scores for single-modal, multi-modal and overall questions. The trained models used for each model are also presented.

Methods	QA-Acc	QA-FL
Fine-tuned		
Solar	58.9	60.9
MuRAG	54.6	55.7
SKURG	63.4	47.8
PERQA	63.9	**61.7**
W/o fine-tuning		
MMAPG (ours)	**65.9**	56.4

Table 2: The performance comparison on WebQA.

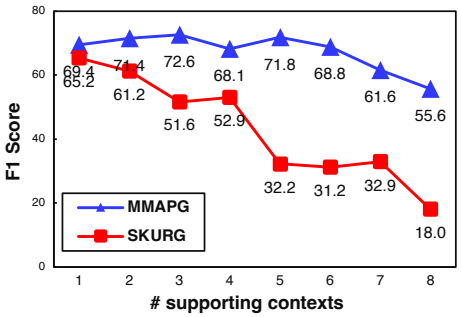

Figure 4: The F1 scores of our method and SKURG across different numbers of supporting documents.

4.2 Main Results

We present our results on the MultimodalQA in Table 1. Compared to baseline models, our method achieves the highest F1 scores in the overall evaluation. However, our exact match scores fall short of the current state-of-the-art. This disparity arises since our approach operates without fine-tuning, and thus, does not align precisely with the ground-truth labels provided by the dataset (See Appendix A.9 for more details). Our single-modality F1 score surpasses existing methods, although the multimodal F1 result remains slightly below that of fine-tuned models.

It is likely due to the the inherent challenges of image reasoning. Without fine-tuning, vision-language models may exhibit greater variability in performance, affecting their ability to precisely match reference answers. Despite this, our approach maintains comparable performance of existing fine-tuned model.

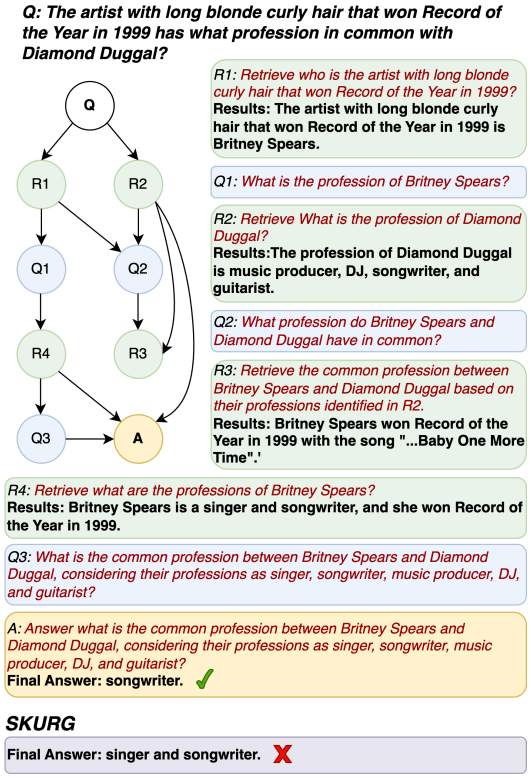

Figure 5: Case study of a multimodal multi-hop QA where MMAPG answers correctly while SKURG fails. The question, retrieval and answer nodes are presented in blue, green and yellow boxes respectively. The instruction is displayed in red. The SKURG result is shown in a purple box.

Table 2 presents the results of WebQA dataset. We observe that our method achieves accuracy comparable to that of other fine-tuned approaches. However, in terms of fluency, our method yields lower scores. Similar to the exact match scores,

Methods	Single-hop		Multi-hop		Overall	
	F1	EM	F1	EM	F1	EM
MMAPG	**75.3**	**65.7**	**70.4**	**57.6**	**73.7**	**63.9**
w/o graph-guided planning	72.4	61.2	58.6	45.1	65.3	53.0
w/o triplet conversion	68.1	56.1	57.6	46.3	63.9	52.2
w/o retrieval module	54.0	44.9	45.8	33.3	49.8	39.1

Table 3: The ablation studies on graph-guided planning and retrieval module. We present the F1 and EM score for single-hop, multi-hop and overall questions.

without fine-tuning, the paraphrased answers struggle to closely match the ground truth.

To assess our method in long-range reasoning, we compare its performance with an iterative framework, SKURG in Figure 4, which shows the F1 score across varying numbers of required supporting contexts in MultimodalQA. As the number of supporting contexts increases, requiring more steps, SKURG's performance drops significantly, especially beyond five contexts. In contrast, our model maintains stable performance, which shows its robustness in complex reasoning scenarios.

4.3 Ablation Study

We present the ablation studies for Adaptive Planning Graph-guided planning , triplet conversion and retrieval module on MultimodalQA. For single-hop tasks, removing graph-guided mechanism results in minor performance decline. However, for multi-hop questions, it leads to a substantial drop of over 10 points in both F1 and exact match scores. It demonstrates that our planning module has a critical role in handling questions that require long-range inference steps. Eliminating triplet conversion during knowledge base construction results in consistent drops across all metrics, demonstrating its role in aligning data granularity. For the retrieval module, we observe a more significant performance drop without our modality-specific strategies, as incorrect retrieval intensifies hallucination for both reasoning and global planning. It validates that our retrieval module improve the performance even without additional training.

4.4 Case Study

We show a case study in Figure 5, which presents how the Adaptive Planning Graph is constructed. At the initial few steps, appropriate nodes are generated. For the third retrieval node, R_3, the system attempts to retrieve information regarding *common profession*, but mistakenly returns results related to *Britney Spears* without identifying her profession, leading to a deviation from the intended path. As *Britney Spears' profession* is identified as an essential information in the overall plan, the planning module analyzes the graph state at this point and determines that the last retrieval action does not capture this information. Consequently, it decides to focus on retrieving *Britney Spears' profession* directly in the next step. Ultimately, this leads to the correct answer. The case study highlights that, even when the graph temporarily deviates, the system is capable of recovering and producing accurate results.

5 Conclusion

In this paper, we introduce a Adaptive Planning Graph-guided framework for multimodal multi-hop QA. It comprises planning, retrieval, and reasoning modules, which leverage off-the-shelf models without fine-tuning. Compared to existing approaches, the proposed method enables flexible reasoning path exploration and plug-and-play model integration. Experimental results show that our method achieves competitive performance against fine-tuned models even without additional training.

6 Limitations

Despite the effectiveness of MMAPG, the absence of fine-tuning leads to lower exact match and fluency scores. Additionally, frequent model calls increases inference costs. The flexibility of our Adaptive Planning Graph leads to longer exploration times and additional steps. Future work will focus on improving module efficiency and reducing computational costs during inference.

References

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. 2024. Graph of thoughts: Solving elaborate problems with large language models. In

Proceedings of the AAAI Conference on Artificial Intelligence, volume 38, pages 17682–17690.

Yingshan Chang, Mridu Narang, Hisami Suzuki, Guihong Cao, Jianfeng Gao, and Yonatan Bisk. 2022. Webqa: Multihop and multimodal qa. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16495–16504.

Wenhu Chen, Hexiang Hu, Xi Chen, Pat Verga, and William W Cohen. 2022. Murag: Multimodal retrieval-augmented generator for open question answering over images and text. *arXiv preprint arXiv:2210.02928*.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *arXiv preprint arXiv:2402.07630*.

Ziniu Hu, Ahmet Iscen, Chen Sun, Zirui Wang, Kai-Wei Chang, Yizhou Sun, Cordelia Schmid, David A Ross, and Alireza Fathi. 2023. Reveal: Retrieval-augmented visual-language pre-training with multi-source multimodal knowledge memory. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 23369–23379.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Qi Zhi Lim, Chin Poo Lee, Kian Ming Lim, and Ahmad Kamsani Samingan. 2024. Unirag: Unification, retrieval, and generation for multimodal question answering with pre-trained language models. *IEEE Access*, 12:71505–71519.

Weihao Liu, Fangyu Lei, Tongxu Luo, Jiahe Lei, Shizhu He, Jun Zhao, and Kang Liu. 2023. Mmhqa-icl: Multimodal in-context learning for hybrid question answering over text, tables and images. *arXiv preprint arXiv:2309.04790*.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems*, 35:2507–2521.

Prasanna Mayilvahanan, Thaddäus Wiedemer, Evgenia Rusak, Matthias Bethge, and Wieland Brendel. 2023. Does clip's generalization performance mainly stem from high train-test similarity? *arXiv preprint arXiv:2310.09562*.

Zhenwei Shao, Zhou Yu, Meng Wang, and Jun Yu. 2023. Prompting large language models with answer heuristics for knowledge-based visual question answering. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 14974–14983.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2024. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *Advances in Neural Information Processing Systems*, 36.

Dan Su, Yan Xu, Genta Indra Winata, Peng Xu, Hyeondey Kim, Zihan Liu, and Pascale Fung. 2019. Generalizing question answering system with pre-trained language model fine-tuning. In *Proceedings of the 2nd workshop on machine reading for question answering*, pages 203–211.

Alon Talmor, Ori Yoran, Amnon Catav, Dan Lahav, Yizhong Wang, Akari Asai, Gabriel Ilharco, Hannaneh Hajishirzi, and Jonathan Berant. 2021. Multimodalqa: complex question answering over text, tables and images. In *International Conference on Learning Representations*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *arXiv preprint arXiv:2212.10509*.

Xiaohua Wang, Zhenghua Wang, Xuan Gao, Feiran Zhang, Yixin Wu, Zhibo Xu, Tianyuan Shi, Zhengyuan Wang, Shizheng Li, Qi Qian, et al. 2024. Searching for best practices in retrieval-augmented generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17716–17736.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

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

Binfeng Xu, Zhiyuan Peng, Bowen Lei, Subhabrata Mukherjee, Yuchen Liu, and Dongkuan Xu. 2023. Rewoo: Decoupling reasoning from observations for efficient augmented language models. *arXiv preprint arXiv:2305.18323*.

Qian Yang, Qian Chen, Wen Wang, Baotian Hu, and Min Zhang. 2023a. Enhancing multi-modal multi-hop question answering via structured knowledge and unified retrieval-generation. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 5223–5234.

Shuwen Yang, Anran Wu, Xingjiao Wu, Luwei Xiao, Tianlong Ma, Cheng Jin, and Liang He. 2023b. Progressive evidence refinement for open-domain multimodal retrieval question answering. *arXiv preprint arXiv:2310.09696*.

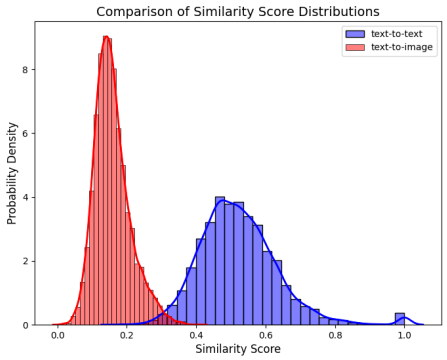

Figure 6: Comparison of similarity score distributions.

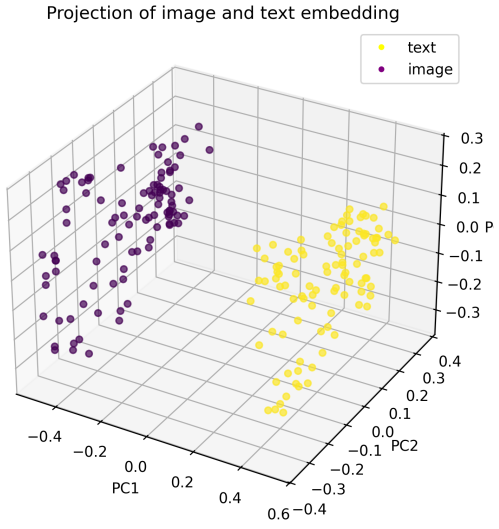

Figure 7: Projection of image and text embeddings.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Bowen Yu, Cheng Fu, Haiyang Yu, Fei Huang, and Yongbin Li. 2023. Unified language representation for question answering over text, tables, and images. *arXiv preprint arXiv:2306.16762*.

Bowen Zhang and Harold Soh. 2024. Extract, define, canonicalize: An llm-based framework for knowledge graph construction. *arXiv preprint arXiv:2404.03868*.

Qing Zhang, Haocheng Lv, Jie Liu, Zhiyun Chen, JianYong Duan, Hao Wang, Li He, and Mingying Xu. 2024. An entailment tree generation approach for multimodal multi-hop question answering with mixture-of-experts and iterative feedback mechanism. In *ACM Multimedia 2024*.

A Appendices

A.1 Challenges of Modality-Agnostic Retrieval

In this section, we examine the limitations of modality-agnostic retrieval when using textual queries, using 100 random selected image-caption pairs from the MSCOCO dataset. Figure 7 shows a 3D projection of CLIP embeddings for both images and their corresponding captions. The embeddings exhibit a clear separation by modality, which highlights the modality gap in the shared representation space. Furthermore, Figure 6 presents the similarity distributions for text-text and text-image pairs derived from same set of samples, which reveals a distribution shift between text-text and text-image similarity scores.

A.2 Overall Workflow

We introduce the overall workflow in Algorithm 1. For simplicity, we denote the prompts used for planning, retrieval and reasoning modules as $P_{planning}$, $P_{retrieval}$ and P_{reason}. Initially, separate knowledge bases are constructed for text and image modalities (line 1), and a graph is initialized with the given question (line 2). Following this, the graph construction procedure begins (line 3-22). At each step, the planning module provides several instructions for the generation of new node (line 4). Depending on the new node type $\mathcal{C}_i$, the system decides which module to invoke and executes the action (Line 7-17). During this step, the content of the new node is determined and the edges are added to update the Adaptive Planning Graph. This process continues until either a stopping condition is met or the maximum number of turns k is reached. Once the graph construction phase is completed, if the final node is not of the answer type, an answer will be generated based on the current state of the graph (line 24). Otherwise, the last answer node will be returned as the final output (line 26).

A.3 Detailed Prompts

We include the prompts from the planning module mentioned in Section 3.3 in Table 5. Since the overall plan $\mathcal{P}_{plan}$ is generated based on the given question, we provide the prompt for generating the plan as $\mathcal{P}_{plan\_gen}$ instead. The prompts from the retrieval module in Section 3.4 are listed in Table 6.

The few-shot examples of $\mathcal{F}_{tgt}$ and $\mathcal{F}_{descr}$ in Section 3.4 are displayed in Table 8 and Table 7. We provide Example 6 as a case to deal with mis-

Algorithm 1 Overall workflow of the proposed framework

Require: Question Q, Sources $\{S_1, S_2, \cdots, S_n\}$, A set of prompt templates P, max_iteration k.

$KB_{text}, KB_{img} \leftarrow$ *KnowledgeBaseConstruction*$(\{S_1, S_2, \ldots, S_n\})$
$V \leftarrow \{Q\}$, $E \leftarrow \{\}$, $G \leftarrow (V, E)$ ▷ Initialize the Adaptive Planning Graph
for $i = 0$ to k **do**
    $\mathcal{I}_{instr}, \mathcal{I}_{\mathcal{C}}, \mathcal{I}_{parent} \leftarrow$ *PlanningModule*$(G, P_{planning})$ ▷ Invoke Planning Module
    $\mathcal{C}_i \leftarrow Decompose(\mathcal{I}_{\mathcal{C}})$
    $v_{p1}, \cdots, v_{pl} \leftarrow \mathcal{I}_{parent}$
    if $\mathcal{C}_i$ is Question **then**
        $V \leftarrow V \cup \{v_i\}$
    else if $\mathcal{C}_i$ is Answer **then**
        $v_i \leftarrow$ *ReasoningModule*$(P_{reason}, \mathcal{I}_{instr}, \mathcal{I}_{parent})$ ▷ Invoke Reasoning module
        $V \leftarrow V \cup \{v_i\}$
    else if $\mathcal{C}_i$ is Retrieval **then**
        $v_i \leftarrow$ *RetrievalModule*$(\mathcal{I}_{instr}, \mathcal{I}_{parent}, KB_{text}, KB_{img}, P_{retrieval})$ ▷ Invoke Retrieval Module
        $V \leftarrow V \cup \{v_i\}$
    else if $\mathcal{C}_i$ is Stop **then**
        break ▷ Terminate the process
    end if
    for $j = 1$ to l **do**
        $E \leftarrow E \cup \{(v_{pj}, v_i)\}$
    end for
    Update the Adaptive Planning Graph G from (V, E)
end for
if the type of last node $\mathcal{C}_k$ is not Answer **then**
    $A \leftarrow$ *ReasoningModule*$(P_{reason}, \mathcal{I}_{instr}, \mathcal{I}_{parent}, G)$
else
    $A \leftarrow$ the content of the last answer node
end if
return A

classification. For example, if $\mathcal{I}_{instr}^{img}$ corresponds to descriptive image retrieval and does not contain a specific target, after inputting into Eq.7, its $\mathcal{Q}'_{text}$ should be returned as an empty list.

A.4 Time Complexity

In this section, we analyze the time complexity of our framework. Here we denote the time for a single LLM operation be t_L, a VLM operation be t_V, a text embedding computation be t_{TE}, a image embedding computation be t_{IE}. Assume there are n sources. During knowledge based construction, since we utilize *sklearn.BallTree* to establish the search space, the complexity will be:

$$O(n \cdot (t_{TE} + t_{IE})) + O(n \log n). \tag{15}$$

Assume q queries are generated, a single retrieval step takes

$$t_L + O(q \cdot \log n) + O(n \cdot (t_L + t_V)). \tag{16}$$

A reasoning step includes a single LLM operation, therefore it takes t_L.

Let the total number of nodes be c, including c_{retr} retrieval steps and c_{reason} reasoning steps. The planning stages would take $c \cdot t_L$ in total. The overall time complexity is displayed as following:

$$\begin{aligned} &O(n \cdot (t_{TE} + t_{IE})) + O(n \log n) \\ &+ c_{retr} \cdot (t_L + O(q \cdot \log n) + O(n \cdot (t_L + t_V))) \\ &+ c_{reason} \cdot t_L + c \cdot t_L. \end{aligned} \tag{17}$$

A.5 Implementation

In our training-free framework, models can be easily incorporated as required, without the need for fine-tuning. Here, we utilize *Llama3-70B-Instruct* as the LLM and *LLaVA-13B* as the visual-language model. As outlined in Section 3.2, the text knowledge base supports search within a text embedding space, while the image knowledge base expects cross-modal retrieval between text queries and images. To facilitate these retrieval processes, we employ CLIP to generate embeddings for both texts and images. During knowledge base construction, we utilize *sklearn.BallTree* structure to efficiently store and organize the embeddings. To retrieve candidates, we employ *query_radius* method, which returns all neighbors within a predefined distance threshold. We choose this method over top-k retrieval as top-k assumes a fixed number of relevant

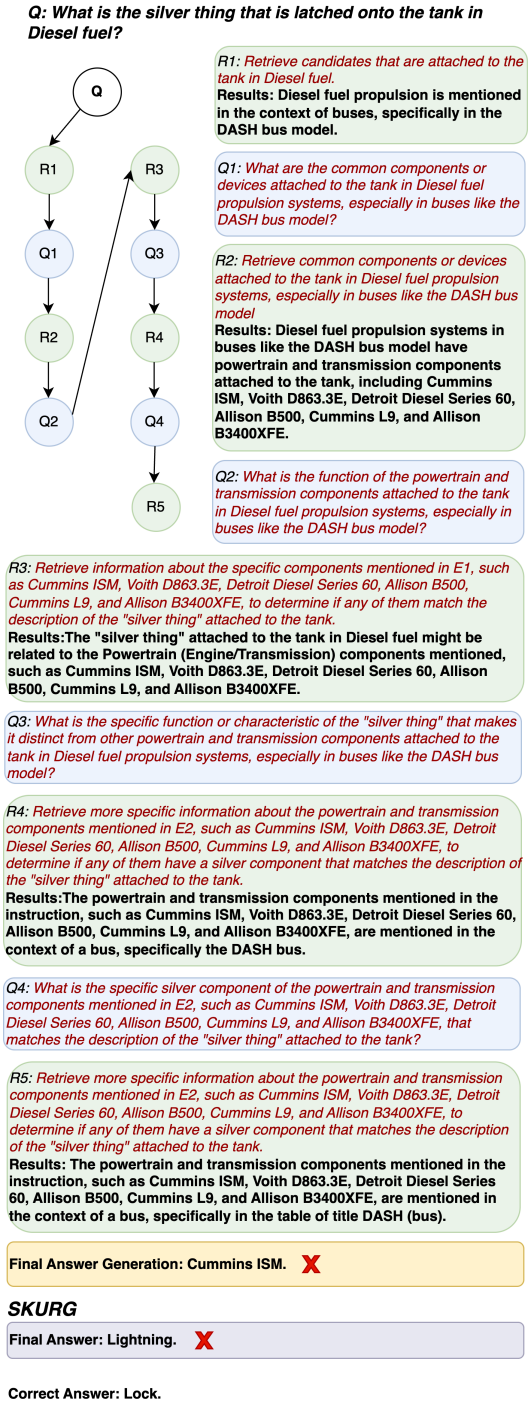

Figure 8: Comparison of similarity score distributions

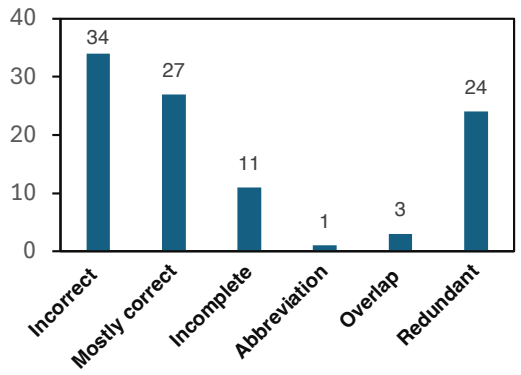

Figure 9: The error analysis of cases where the EM score is zero.

items, which is unsuitable for multimodal multi-hop question answering tasks, where the number of relevant sources can vary depending on the question.

For datasets that include the table modality, we preprocess tables by converting the structured data from each row into a sentence. They are then treated similarly to text modality for subsequent processing.

A.6 Dataset Statistics

In this section, we present the statistics of the datasets used. WebQA dataset comprises 34K training samples, 5K development samples and 7.5K test samples. MultimodalQA dataset includes 23K training samples, 2.4K development samples and 3.6K test samples.

A.7 Ablation Study Setup

Firstly, to evaluate the performance without planning module, we modify the framework by omitting the Adaptive Planning Graph construction. In this setting, the actions are planned in a sequential manner, where each step is built directly upon previous one, similar to an iterative framework. Consequently, the system is constrained to a single-path approach, alternating between retrieval and reasoning without the flexibility to explore other paths. Secondly, we evaluated the performance without our retrieval module by replacing it with a simplified version. It retrieves information based solely on similarity score between the embeddings of instruction queries and sources. The key components of our retrieval design, including query construction and candidate examination, are eliminated.

A.8 Case Study

We demonstrate an example for failure cases in Figure 8. The potential failure points arise because nodes continue providing seemingly relevant information at starting point, leading the system to follow this path even though it does not capture the truely relevant sources. It repeatedly extracts the same context, mistakenly believing it may contain the necessary details. Additionally, the graph state summary gradually becomes too long, making it harder for the LLM to produce the correct action, as handling long contexts remains a challenge for LLMs. The complexity of such questions also applies to the baseline method (Yang et al., 2023a), highlighting the inherent challenges posed by indistinguishable distractors.

A.9 Error analysis

To gain a deeper understanding of the causes of errors, we conduct an error analysis by randomly sampling instances from the results where the exact match (EM) score is zero. We categorize the reasons for these errors into six distinct categories, with detailed descriptions provided in Table 4. As depicted in Figure 9, we observe that 27% of the results, despite being mostly correct, were classified as inaccurate due to the strict criteria of the exact match metric. Another significant source of error was the presence of redundant results, which generate more outputs than expected.

Category	Description
Incorrect	The prediction does not match with ground truth answer at all.
Mostly Correct	The prediction has the same meaning as the ground truth answer but receives a zero EM score due to minor discrepancies such as number formatting, date representation, or the use of symbols.
Incomplete	The prediction provides a partial response, failing to fully capture the complete content of the ground truth answer.
Abbreviation	The prediction reflects the same meaning as the ground truth answer but is presented in an abbreviated form.
Overlap	The prediction partially aligns with the ground truth, sharing some overlapping content but lacking a complete match.
Redundant	The prediction includes the complete ground truth answer but is characterized by the presence of extraneous or redundant components.

Table 4: Descriptions of the categories in error analysis.

Notation	Prompt Template
$\mathcal{P}_{plan\_gen}$	Given a multi-hop question, break the question into key components to identify all necessary information.
$\mathcal{P}_{parent}$	Specify which existing node(s) will be used as the foundation for generating a new node, ensuring a clear and logical progression in the reasoning process.
$\mathcal{P}_{\mathcal{C}}$	Please analyze the content of existing nodes step by step, and then decide what new node to generate. Here are some options: 1. Stop: when the answer to question Q is already found given the current nodes. 2. Retrieval: to retrieve candidates and extract useful information from candidates based on existing nodes. 3. Answer: to produce the instruction to generate an answer based on an existing question and other nodes.
$\mathcal{P}_{state}(G)$	Here is the graph: ... Your task is to determine the next step in deriving the final answer of Q based on provided input. Please think step by step and consider the current reasoning graph and overall plan.

Table 5: Prompts for Planning Module.

Notation	Prompt Template
$\mathcal{P}_{decomp}$	Given the following instruction, your task is to identify and extract the text-related part and the image-related part for retrieval. The instruction may contain references to both textual and visual content. For image-related parts, determine if the task is Targeted Image Retrieval (specific images named) or Descriptive Image Retrieval (search based on description).
$\mathcal{P}_{extract}$	Given the following text, your task is to extract the keywords. The keywords should be specific and helpful for retrieval.
$\mathcal{P}_{tgt}$	For targeted image retrieval, please list specific image if mentioned, and craft a precise question based on the image-related request to guide the assistant on what to identify or analyze in the image.
$\mathcal{P}_{descr}$	For descriptive image retrieval, please extract descriptive phrase, and craft a precise question based on the image-related request to guide the assistant on what to identify or analyze in the image.
$\mathcal{P}_{exam}^{text}$	Your task is to analyze the input text and check if it is related to the instruction: ...
$\mathcal{P}_{exam}^{img}$	Given the image, provide a brief description of its content and answer the question based on the provided instruction: ...
$\mathcal{P}_{retr}$	These are the instruction and retrieval results: ... Please extract only the relevant information from the result and rephrase into valid description as the corresponding answer to the instruction for later analysis.

Table 6: Prompts for Retrieval Module.

Notation	Few-shot Examples
$\mathcal{F}_{text}$	Example 1: Instruction: Retrieve the NHL team played against the Pittsburgh Penguins in the playoff series. Key Phrase:["NHL team played against the Pittsburgh Penguins in the playoff series"] Example 2: Instruction: Retrieve the 1977 Seattle Seahawks Kingdome regular season opponent that has the most Super Bowl losses in NFL history. Key Phrase: ["1977 Seattle Seahawks Kingdome opponent", "Super Bowl losses in NFL history"] Example 3: Instruction: Retrieve the role Peppe Lanzetta played in the 2009 film. Key Phrase: ["role Peppe Lanzetta played in the 2009 film"] Example 4: Instruction: Retrieve the Magazine that had Caroline Miller as Editor in Chief and the year it won a National Magazine Award. Key Phrase: ["Magazine that had Caroline Miller as Editor", "year of Magazine won a National Magazine Award"] Example 5: Instruction: Retrieve the song performed in episode 3 of season 1 of The Clash on July 14. Key Phrase: ["song performed in The Clash on July 14", "episode 3 of season 1 of The Clash on July 14"].

Table 7: Few-shot examples for $\mathcal{F}_{text}$.

Notation	Few-shot Examples
$\mathcal{F}_{tgt}$	Example 1: Instruction: Retrieve the structure at the top of the Stockport County F.C.'s logo. Question: What is the structure at the top of logo? Target: ["the Stockport County F.C.'s logo"] Example 2: Instruction: Retrieve the colors that make up the flag for Denmark. Question: What are the colors that make up the flag? Target: ["flag for Denmark"] Example 3: Instruction: Retrieve the number of leaves are on the clover of the Celtic F.C.'s logo. Question: How many leaves are on the clover of the logo? Target: ["Celtic F.C.'s logo"] Example 4: Instruction: Retrieve the hair style of the man in blue in American football. Question: What hair style does the man in blue have in American football? Target: ["American football"] Example 5: Instruction: Retrieve whether there is a fence around the outside of the Cotton Bowl (stadium). Question: Is there a fence around the outside of the Cotton Bowl? Target: ["the Cotton Bowl (stadium)"] Example 6: Instruction: Retrieve the movie poster that has a woman with a green dress on it. Question: Is there a woman with a green dress on it? Target: []
$\mathcal{F}_{descr}$	Example 1: Instruction: Retrieve the team whose logo has a bird on it. Question: Is there a bird in the logo? Key Phrase: ["logo with a bird"] Example 2: Instruction: Retrieve the movie poster that has a woman with a green dress on it. Question: Is there a woman with a green dress on it? Key Phrase: ["movie poster that has a woman with a green dress"] Example 3: Instruction: Retrieve the television title with a car on its poster. Question: Is there a car on the poster? Key Phrase: ["television poster with a car on it"] Example 4: Instruction: Retrieve opponent that has a football helmet on its logo. Question: Is there a football helmet on the logo? Key Phrase: ["logo with a football helmet"] Example 5: Instruction: Retrieve the poster that has a man reaching out with his hand. Question: Is there a man reaching out with his hand? Key Phrase: ["poster with a man reaching out with his hand."] Example 6: Instruction: Retrieve the number of leaves are on the clover of the Celtic F.C.'s logo. Question: How many leaves are on the clover of the logo? Key Phrase: []

Table 8: Few-shot examples for $\mathcal{F}_{tgt}$ and $\mathcal{F}_{descr}$.