

A QUALITATIVE EXAMPLES

This section presents a qualitative case study illustrating how MI-RAG constructs reasoning records at each iteration and arrives at the final answer. All examples are drawn from the InfoSeek validation set using MI-RAG with Gemini-2.5-Flash. Figures A1–A10 adopt the same layout described below.

- **Upper-left panel:** The text question and the input image. A descriptive caption about its entity is shown to the right of the image for demonstration only.
- **Lower-left panel:** This panel shows MI-RAG’s final prediction, the ground-truth answer, and the evaluation result. Correct predictions are highlighted in green, while incorrect predictions are in red.
- **Right panel:** This panel presents a four-iteration summary of the MI-RAG reasoning process. The color of each step indicates its reasoning quality with respect to the available evidence: green signifies correct reasoning, yellow is for partially valid or inconclusive steps, red marks an error, and blue is used for emphasis only. Furthermore, any steps following an incorrect one are shaded gray to show they are built upon a previous error.

A.1 SUCCESSFUL CASE ANALYSIS

We present several successful cases that demonstrate the strengths of our MI-RAG framework. These examples illustrate how the iterative process of reasoning-guided retrieval and retrieval-augmented reasoning enables the model to correctly answer knowledge-intensive visual questions.

Successful Visual Identification. A common success pattern begins with accurate recognition of visual entities, followed by a search for related cultural and historical knowledge. In Figure A1, the entity is first identified as a submarine sandwich. MI-RAG then attributes its cultural origins to Italian-American communities and identifies its historical emergence in the Northeastern United States.

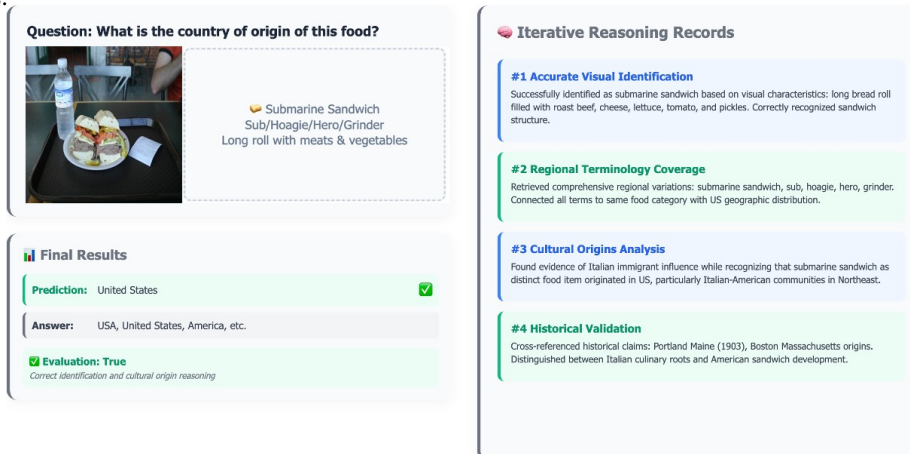


Figure A1: An initial correct visual entity leads to the retrieval of its corresponding knowledge.

Compositional Reasoning with Heterogeneous Knowledge. MI-RAG demonstrates compositional reasoning by integrating visual grounding with textual knowledge. As shown in the Amrum Lighthouse example (Figure A2), it first performs a visual analysis, identifying a lighthouse on an elevated landform. Through iterative retrieval, it grounds the specific entity as the "Amrum Lighthouse" and retrieves textual evidence from the knowledge base stating the structure is built "atop a 25 metres high dune". The final compositional reasoning step involves synthesizing the visual evidence of an "elevated landform" with this specific textual fact to correctly identify the feature as a dune.

Progressive Reasoning from Generic to Specific. This case demonstrates the model’s ability to progressively refine its understanding through iterative refinement. When asked for the creator of the object in Figure A3, the model’s initial reasoning record correctly notes that a specific creator

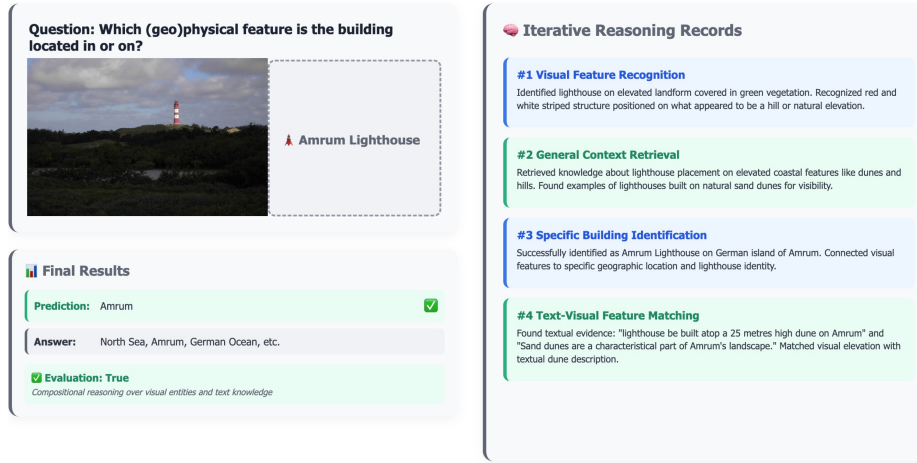


Figure A2: A visual entity is linked with corresponding textual knowledge.

of the telescope cannot be determined from the image alone. However, it successfully classifies the object as a Newtonian telescope in subsequent iterations. This crucial intermediate step enables a subsequent query generation focused on the inventor. This refined query retrieves the correct historical fact, showcasing how MI-RAG leverages reasoning records to guide the progression from a generic classification to a specific answer.

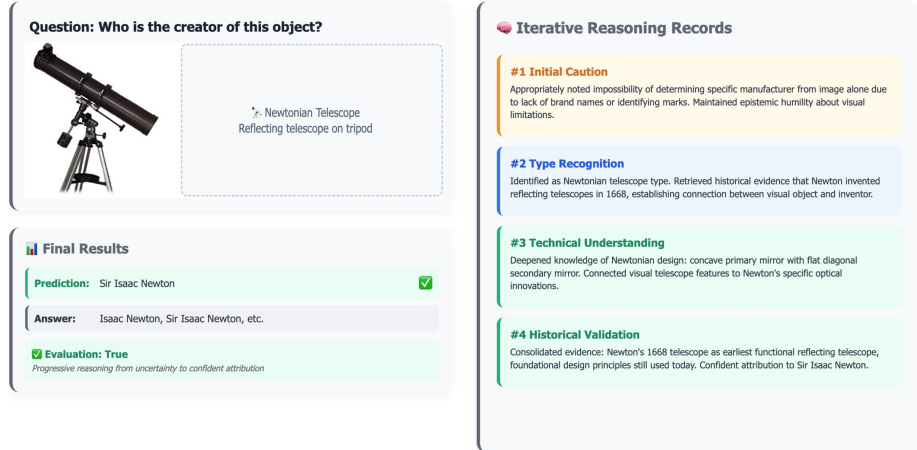


Figure A3: Progressively refine its understanding from generic object recognition (Telescope) to specific knowledge of an entity (Newtonian telescope, which was invented by Newton).

Compose Multi-Faceted Knowledge. This case demonstrates how MI-RAG answers questions that require linking multiple facets of knowledge. In Figure A4, it identifies architectural details (Russian Orthodox), the specific building's identity (Vysokopetrovsky Monastery), and historical facts about its founder through iterations. MI-RAG then performs compositional reasoning over this diverse evidence, correctly synthesizing the facts to conclude that the monastery is dedicated to Saint Peter of Moscow.

A.2 FAILURE CASE ANALYSIS

To better understand the limitations of our model, we present a qualitative analysis of representative failure cases. Each case provides insights into areas for future improvement by pairing the visual example directly with its analysis.

Intent Misinterpretation. A primary failure mode involves misunderstanding the user's query, even with correct evidence. In Figure A5, the model is asked for the "closest parent taxonomy" of

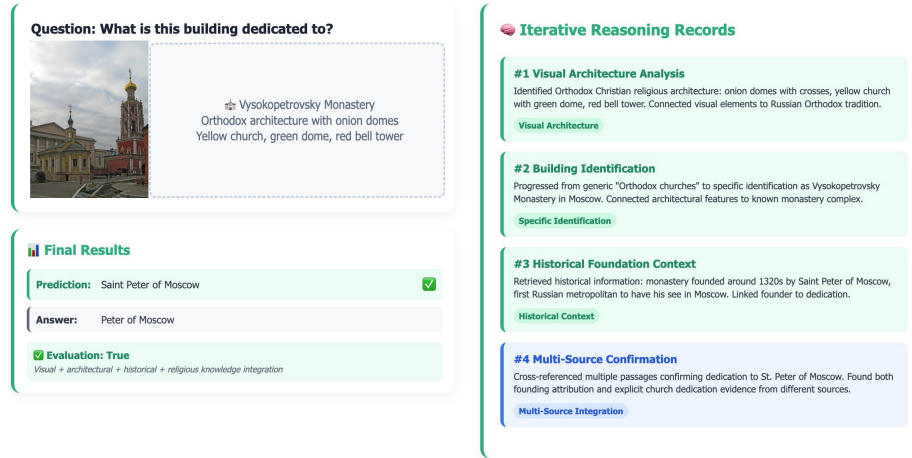


Figure A4: Multi-Faceted Knowledge Synthesis: Diverse aspects of knowledge, including architectural and historical facts, are retrieved and synthesized to answer.

a peccary. Despite correctly identifying the animal and its full taxonomic hierarchy, it misinterprets the query's intent, providing the suborder (Suina) rather than the more immediate genus (Pecari).

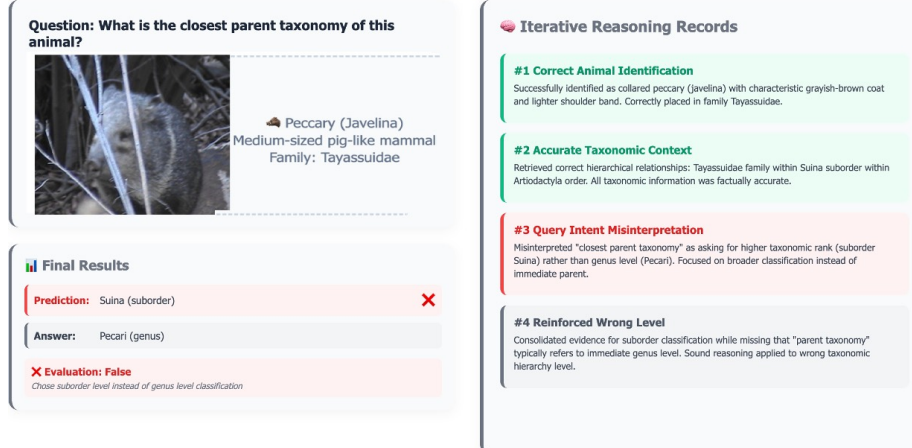


Figure A5: Intent Misinterpretation: The model misunderstands the required taxonomic specificity.

Entity Misidentification. Failures can also arise from the initial visual analysis stage, such as incorrectly identifying the primary subject. In Figure A6, the model correctly identifies the general scene as a Swiss lake, but incorrectly identifies the specific body of water as Lake Walen instead of Lake Lucerne. This initial error leads to a wrong subsequent retrieval and concludes with wrong answer.

Subject Selection Error. Another form of visual grounding error occurs when the model focuses on a non-primary subject in the scene. Figure A7 shows a scene with two aircraft. The model correctly identifies both but erroneously selects the second aircraft as the subject of the query, derailing the entire subsequent reasoning process.

Incomplete Knowledge Retrieval. This class of errors arises from limitations in the knowledge base or the retrieval process. In Figure A8, the model identifies a bird as a New World Oriole but fails to retrieve knowledge on the specific genus (Pitangus). Its final answer is therefore based on incomplete evidence, leading to an incorrect conclusion.

Insufficient Visual Evidence. Sometimes, a query is unanswerable from the image, since it provides generic entities. The image in Figure A9 shows a generic classroom, which lacks sufficient

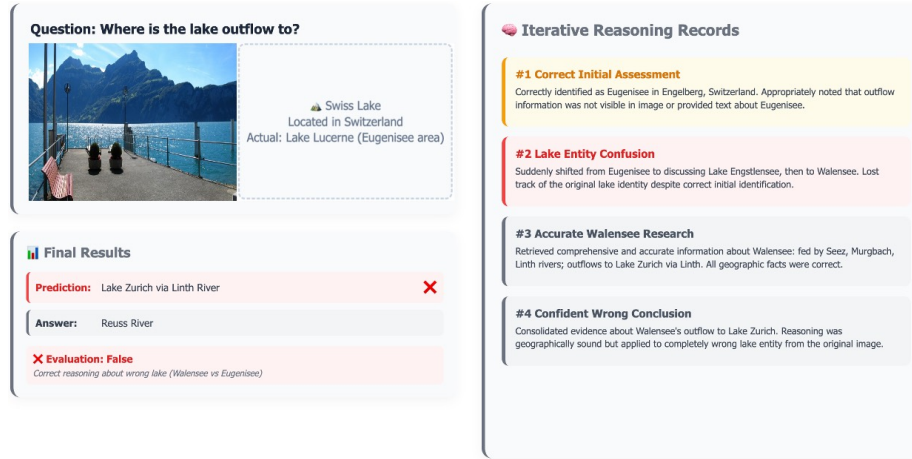


Figure A6: Entity Misidentification: An initial, incorrect identification of the primary subject.

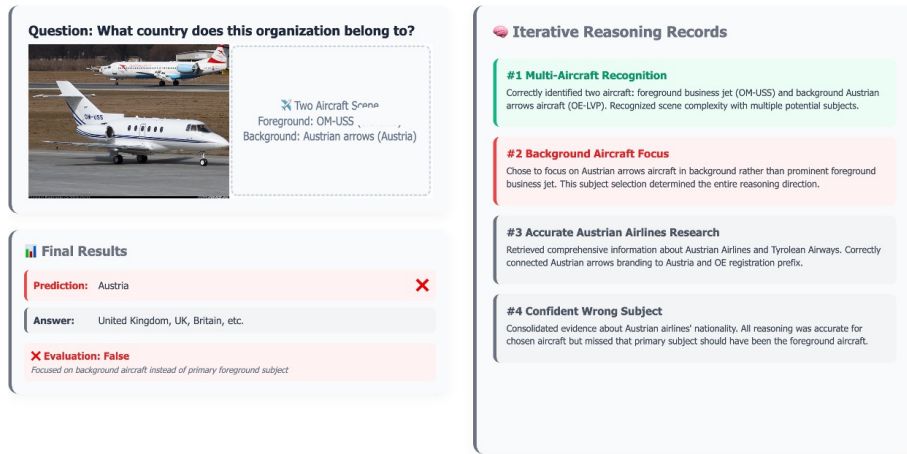


Figure A7: Subject Selection Error: The model focuses on a background subject instead of the primary subject.

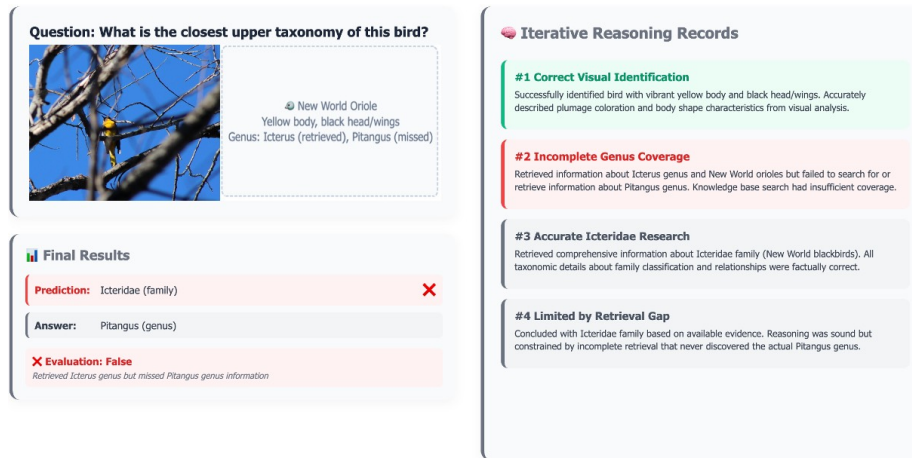


Figure A8: Incomplete Knowledge Retrieval: Fails to retrieve knowledge about the correct entity.

visual features to determine a specific location. The model correctly identifies the generic context but fails to recognize the query’s unanswerable nature, producing a non-specific answer.

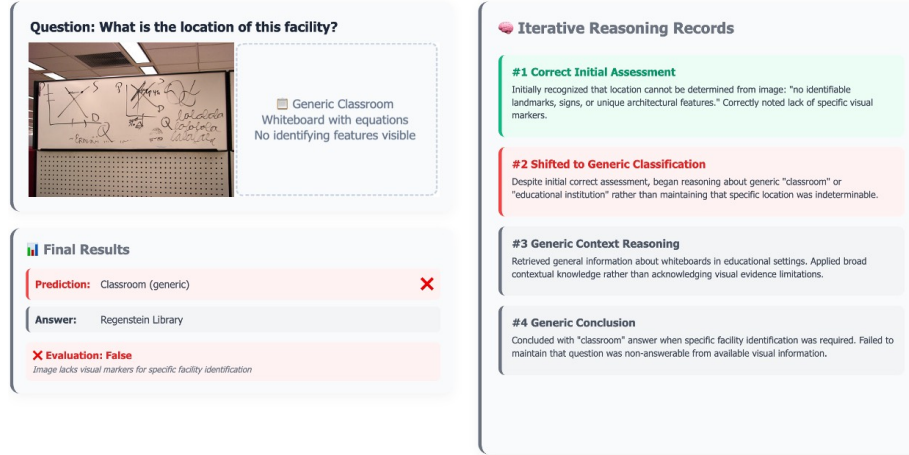


Figure A9: Insufficient Visual Evidence: The query is unanswerable from the image provided.

Reasoning-Answer Discrepancy. Finally, some failures occur when the reasoning is sufficient, but the final answer does not align with the expected format. In Figure A10, the model correctly deduces that the Hawker 800 is a development of the British Aerospace 125. However, its prediction is marked as incorrect because the ground truth expects a different but related entity name, highlighting a format mismatch issue rather than a fundamental reasoning error.

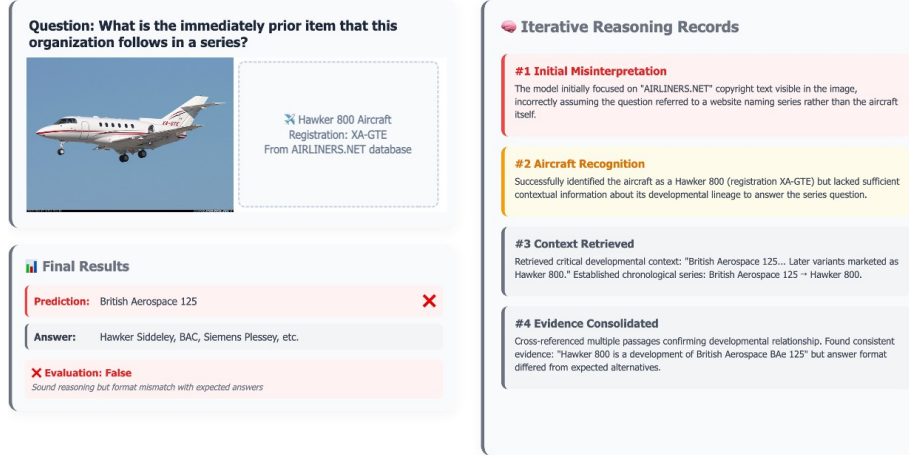


Figure A10: Format Mismatch: The model’s reasoning is sufficient, but the output format differs from the expected answer.

B THE USE OF LARGE LANGUAGE MODELS

In accordance with ICLR 2026 policy, we disclose the use of a large language model (LLM) as an assistive tool in the preparation of this manuscript. The LLM’s role was confined to that of a writing and formatting assistant. Specific tasks included improving grammar, clarity, and style, as well as providing support for formatting LaTeX code, such as adjusting the layout and style of figures and captions. All conceptual contributions, research ideation, experimental design, data analysis, and the final written conclusions were generated exclusively by the human authors. The authors take full responsibility for all content presented in this paper, including its scientific accuracy and integrity.

C PROMPT TEMPLATES FOR MI-RAG

```

1026 1 # 1. Initial description generation
1027 2 prompt = f"Question: {question}\n Concisely describe image which is
1028 3 ↪ relevant to question.\n"
1029 4
1030 5 -----
1031 6 # 2. Query Expansion
1032 7 prompt = f"Question: {question}\n{reasoning_record}\n"
1033 8 -----
1034 9 # 3. Query Generation Prompt
1035 10 Question: {question}
1036 11 Knowledge: {reasoning_records}
1037 12
1038 13 Please first analyze all the information in a section named Analysis
1039 14 ↪ (## Analysis).
1040 15 Generate two follow-up questions to search for additional information
1041 16 ↪ and helpful to confirm knowledge, in a section named Queries (##
1042 17 ↪ Queries).
1043 18 Your output should be in the following format:
1044 19
1045 20 ## Analysis
1046 21 Analysis question and knowledge to ask context-specific queries that
1047 22 ↪ helps to address question.
1048 23 ## Queries
1049 24 Question 1: question 1.
1050 25 Question 2: question 2.
1051 26
1052 27 -----
1053 28 # 4. Reasoning Record Generation Prompt
1054 29 # This prompt instructs the model to synthesize information into a
1055 30 ↪ coherent summary.
1056 31
1057 32 Question: {question}
1058 33 Knowledge: {knowledge}
1059 34
1060 35 Based on image, description and knowledge, summarize correct and
1061 36 ↪ relevant information with image and question.
1062 37 -----
1063 38 # 5. Final answer prompt
1064 39 Please answer the following question using the provided information and
1065 40 ↪ image.
1066 41
1067 42 Question: {question}
1068 43 Relevant Knowledge: {reasoning_records}
1069 44
1070 45 Based on the information, provide a detailed answer to the question.

```

Figure A11: Pseudo-code for prompt templates used in our MI-RAG framework. The Query Generation prompt drives the iterative retrieval process, while the Reasoning Record Generation prompt synthesizes information at each step.

Our Multimodal Iterative RAG (MI-RAG) framework employs structured prompts to dynamically refine its understanding and gather relevant knowledge. To generate initial reasoning records, we use a query-focused description of the input image to retrieve an initial set of knowledge. The core iterative loop then utilizes two prompts. First, the query generation component of multi-query uses the prompt to instruct the model to analyze the accumulated reasoning records and generate two questions. This step is crucial for progressively searching for more specific and helpful information. For the query expansion component of multi-query, we simply concatenate the query with the previous reasoning record. Second, the reasoning record generation prompt directs the model to synthesize the image and newly retrieved knowledge into a query-focused summarization. This rea-

soning record is the result of compositional reasoning and improves subsequent retrieval iterations, and is used to derive the final answer.

D MULTIMODAL RETRIEVAL IMPLEMENTATIONS

To build our knowledge base, we processed the Wikipedia dataset from the Encyclopedic VQA benchmarks, which contains 2 million image-section pairs. The construction process involves generating a multimodal embedding for each pair and indexing them for efficient retrieval. For each sample, we extracted a normalized image embedding using a pretrained SigLIP model and a normalized text embedding using a pretrained ModernBert model, which is trained with the GTE method. These two vectors are then concatenated to form a multimodal embedding. Finally, all embeddings are indexed using FAISS with an IndexFlatIP (Inner Product) to create a knowledge base. A corresponding metadata file was also created to map the index entries back to their original image paths and Wikipedia text sections. When encoding the text embedding, we use a summary of each Wikipedia section. For the large retriever configuration of SigLIP2-g and Qwen3-Embedding-0.6B, we employ the image-to-mixture similarity proposed in RA-CM3 Yasunaga et al. (2023) and encode mixed embedding instead of image embedding.

At query time, we encode the input image and refined text query into the same multimodal embedding space as our knowledge base. We first generate normalized embeddings for the image and the text query separately. These vectors are then concatenated to form a single multimodal query vector. Finally, we perform a maximum inner product search against the knowledge base to retrieve the top-k entries with the highest multimodal similarity.

Table A1: Performance analysis of multimodal similarity on the InfoSeek validation split and Encyclopedia test split.

Retriever	Dataset	Query Modality	R@1	R@5	R@10
SigLIP2-SO400m+ ModernBert-GTE	InfoSeek	image	52.5	60.2	68.3
		image+text	57.8	65.3	72.3
	Encyclopedia	image	30.8	36.6	41.8
		image+text	34.9	40.5	45.3
SigLIP2-g+ Qwen3-Embedding-0.6B	InfoSeek	image	66.7	72.7	77.5
		image+text	69.4	76.2	81.1
	Encyclopedia	image	36.2	41.9	46.4
		image+text	43.1	48.7	54.5

As shown in Table A1, we evaluated the retriever performance on the InfoSeek validation and Encyclopedia test splits. The results confirm two key points. First, using a multimodal query (image+text) that considers both visual and textual similarity consistently yields higher recall than using a unimodal (image) query alone. Second, we observed that our scaled retriever configuration (SigLIP2-g + Qwen3-Embedding-0.6B) performs better than the more lightweight setup (SigLIP2-SO400m + ModernBert-GTE). Despite the performance gap, we used the lightweight setup for our main experiments. This choice ensures a fair comparison and emphasizes that our framework’s primary improvements stem from its iterative reasoning process, not from relying on a stronger base retriever.

E FEW-SHOT PROMPTING FOR ANSWER EXTRACTION

To evaluate performance on the Exact Match metric for the InfoSeek dataset, we employed a few-shot prompting strategy to guide the model toward generating an exact answer. For each question in the validation set, we provided the model with three demonstrations drawn from the training split, selected based on question similarity. These demonstrations consist of an image, a context of reasoning records, a question, and the ground-truth answer. By presenting these structured examples, we condition the model to generate responses in a specific format, which minimizes extraneous text and focuses on producing the exact entity or fact required by the question. This in-context learning approach helps to align the model’s output with the strict requirements of the EM metric.

```

1134 1 Answer the knowledge-intensive question based on the provided image and
1135 2 ↪ context.
1136 3 Generate a concise and accurate answer grounded in the retrieved
1137 4 ↪ information.
1138 5 Use the context to support reasoning, and directly output the final
1139 6 ↪ answer.
1140 7
1141 8 Three examples are shown below:
1142 9
1143 10 ##Example 1:
1144 11 [Image 1 Content]
1145 12 ##Context: {few_shot_context_1}
1146 13 ##Question: {few_shot_question_1}
1147 14 ##Best Answer: {few_shot_answer_1}
1148 15
1149 16 ##Example 2:
1150 17 [Image 2 Content]
1151 18 ##Context: {few_shot_context_2}
1152 19 ##Question: {few_shot_question_2}
1153 20 ##Best Answer: {few_shot_answer_2}
1154 21
1155 22 ##Example 3:
1156 23 [Image 3 Content]
1157 24 ##Context: {few_shot_context_3}
1158 25 ##Question: {few_shot_question_3}
1159 26 ##Best Answer: {few_shot_answer_3}
1160 27
1161 28 Now, answer this question
1162 29 [Main Image Content]
1163 30 ##Context: {reasoning_records}
1164 31 ##Question: {question}
1165 32 ##Best Answer:

```

Figure A12: The few-shot prompt template used to guide the model for the Exact Match evaluation.

F COMPUTATIONAL COST ANALYSIS

Table A2: Computational cost per sample for each iteration of the MIRAG framework

Iteration	API Time (s)	Retrieval Time (s)
Iteration 0	7.96	15.23
Iteration 1	19.89	46.36
Iteration 2	32.29	79.74
Iteration 3	41.71	105.384

We analyze the per-sample computational cost of MI-RAG on a commodity server (NVIDIA RTX 3090, Intel Xeon Gold 6248R), performing retrieval locally and using the OpenRouter API OpenRouter (2025) for MLLM inference. As detailed in Table A2, both API and retrieval times scale linearly with each iteration, an expected characteristic of our design. Specifically, API time may increase as accumulated reasoning records lengthen the context for query generation, while retrieval time grows due to the increased processing required to encode these context-enriched queries. Crucially, this latency is not a fundamental limitation of our framework. The overhead stems primarily from the serialized execution of queries using vanilla index configurations and network overhead from successive API calls with multiple images and texts.

G SUBSET DOWNSAMPLING PROCEDURES

To support an efficient analysis and design of our model, MI-RAG, we perform ablation studies on a compact and diverse subset of each dataset. We generate this subset using the iterative downsampling procedure detailed in Figure A13, which aims to eliminate redundancy.


```

1188 1 # D: Input Dataset of questions
1189 2 # Q: A single question from the dataset
1190 3 # Emb_Q: The text embedding of a question Q
1191 4 # t: The similarity threshold
1192 5 # subset: Downsampled dataset
1193 6
1194 7 function DownSampleDataset(D, t):
1195 8     subset = []
1196 9     subset_emb = []
1197 10
1198 11     for Q in D:
1199 12         is_redundant = False
1200 13         Emb_Q = Encode(Q)
1201 14
1202 15         for emb in subset_emb:
1203 16             if Similarity(Emb_Q, emb) > t:
1204 17                 is_redundant = True
1205 18                 break
1206 19
1207 20         if not is_redundant:
1208 21             subset.append(Q)
1209 22             subset_emb.append(Emb_Q)
1210 23
1211 24     return subset

```

Figure A13: Pseudo-code for downsampling: discard questions whose similarity to any in the subset exceeds threshold, yielding a non-redundant subset.

The algorithm greedily constructs a subset by adding a new question Q from the original dataset D only if it is sufficiently dissimilar to all questions already selected. Specifically, a question is considered redundant and discarded if the similarity of its embedding, Emb_Q , exceeds a predefined threshold with any embedding already in the subset. This threshold is adjusted for each dataset to yield a final subset containing approximately 500 diverse question-answer pairs.

H QUERY STRATEGY ANALYSIS

Table A3: Analysis of each query formulation’s contributions to retrieval recall from each KB on the InfoSeek validation subset.

Method	Multimodal KB Recall (%)	Textual KB Recall (%)	Hetero. KB Recall (%)
Query Expansion	90.02	87.44	93.08
Query Generation	88.24	80.03	90.80
Multi-Query	90.50	90.02	94.36

In this section, we analyze the contributions of each query strategy to retrieval performance on the heterogeneous KB. Query expansion proves highly effective at retrieving directly related information, achieving 93.08% recall. In contrast, query generation is designed to explore complementary reasoning paths. Because its retrieval budget is divided between two generated sub-queries, its recall is slightly lower than that of the query expansion. However, its primary value lies in finding a diverse set of factual links that a direct expansion might overlook.

I QUERY TRANSFORMATION VS. REASONING

To assess which component of our framework benefits from increased model capacity, we performed an analysis by switching a lightweight model (Gemma-3-4B) and a large model (Gemini-2.5-Flash) between query transformation and reasoning generation. As shown in Table A4, allocating a more capable model to query transformation yields higher recall, likely due to improved query precision

Table A4: Impact of allocating a larger model (Gemini-2.5-Flash) to either the query transformation or reasoning records generation, while a lighter model (Gemma-3-4B) handles the other. Evaluated on the InfoSeek validation subset.

Query Transformation	Reasoning Generation	Acc (%)	Recall (%)
Gemma-3-4B	Gemini-2.5-Flash	56.36	91.63
Gemini-2.5-Flash	Gemma-3-4B	54.59	93.56

and diversity. However, assigning the large model to reasoning records generation results in higher overall accuracy, indicating that this configuration provides a benefit to the final accuracy.