

Atom-RAG: Atomic Fact-Centric Retrieval for Multi-Hop Reasoning

Seok Jun Hwang¹, Soohyeong Kim¹, JungHyouon Kim², Jeonghyeon Park³, Yong Suk Choi^{3*}

¹Department of Artificial Intelligence, Hanyang University, Seoul, Korea

²Department of Intelligence and Convergence, Hanyang University, Seoul, Korea

³Department of Computer Science, Hanyang University, Seoul, Korea

{cody628, ksh970404, kkksk1157, shshjhjh4455, cys}@hanyang.ac.kr

Abstract

Retrieval-Augmented Generation (RAG) enhances large language models with external knowledge but often suffers from query–evidence mismatch due to inappropriate retrieval granularity. Chunk-level retrieval can introduce substantial noise, while graph-based retrieval may suffer from overly fine-grained or densely connected structures that hinder effective multi-hop reasoning. To address this issue, we propose Atom-RAG, a retrieval-augmented framework based on atomic fact–centric knowledge representation and granularity-aligned retrieval. Atom-RAG represents documents as collections of atomic facts and aligns retrieval with the fine-grained information needs of complex queries. By decomposing queries into fact-level subqueries, the framework retrieves and aggregates evidence in a manner aligned with the information requirements of multi-hop reasoning, producing a compact and evidence-aligned context for generation. Experiments on benchmarks covering different reasoning depths show that Atom-RAG consistently outperforms strong iterative and graph-based RAG baselines, achieving improved multi-hop QA performance and stronger evidence retrieval. These results highlight the importance of aligning the granularity of queries and retrieval units for reliable multi-hop reasoning within a unified retrieval pipeline. Code and data are available at <https://github.com/RAGcody628/Atom-RAG>.

1 Introduction

Retrieval-Augmented Generation (RAG) enhances large language models (LLMs) by integrating external knowledge retrieval into the generation process (Karpukhin et al., 2020; Lewis et al., 2020). By enabling models to access information beyond their parametric knowledge, RAG has shown strong performance on knowledge-intensive tasks (Lewis

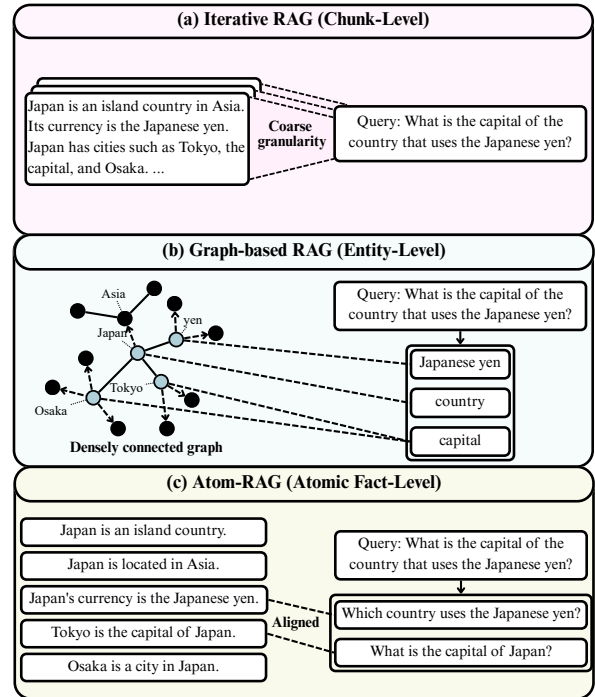


Figure 1: Retrieval granularity comparison: chunk-level, entity-level, and atomic fact–level retrieval.

et al., 2020). In its most common form, often referred to as Naïve RAG, the corpus is decomposed into fixed-length text chunks, and relevant chunks are retrieved based on similarity between the input query and chunk representations (Gao et al., 2023). The retrieved chunks are then provided to the LLM as additional context for answer generation. However, conventional chunk-level retrieval primarily relies on similarity matching and typically treats each chunk independently. As a result, it fails to capture relationships across multiple pieces of evidence, making it ineffective for multi-hop queries that require compositional or sequential reasoning (Trivedi et al., 2023; Yasunaga et al., 2021).

To address these limitations, recent RAG approaches have introduced iterative and graph-based retrieval paradigms. Iterative RAG interleaves reasoning with intermediate retrieval (Khattab et al.,

*Corresponding author.

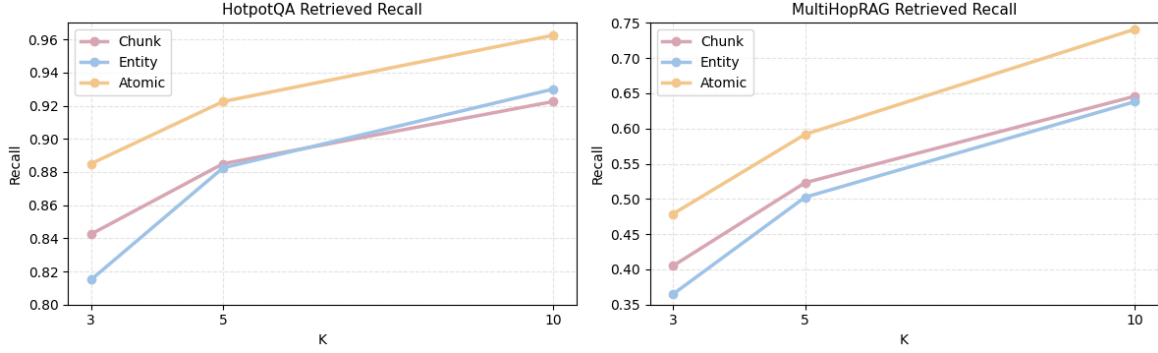


Figure 2: Recall comparison of chunk-, entity-, and atomic fact-level retrieval on HotpotQA and MultiHopRAG.

2022), decomposing complex queries into sequential steps and progressively accumulating evidence for multi-hop reasoning (Press et al., 2023; Shao et al., 2023; Trivedi et al., 2023). In contrast, graph-based RAG organizes knowledge as graphs of entities and relations, enabling structured multi-hop reasoning through node matching and relational traversal (Edge et al., 2024; Guo et al., 2025; Chen et al., 2026; Luo et al., 2025a; Gutiérrez et al., 2025).

Despite their advantages, both iterative RAG and graph-based RAG share a fundamental limitation: a mismatch between the query intent and the granularity of retrieved information. As illustrated in Figure 1(a), iterative RAG typically performs retrieval at the chunk level, where coarse-grained passages often contain mixed or partially irrelevant information, introducing substantial contextual noise (Cuconasu et al., 2024) during multi-step reasoning (Press et al., 2023). As illustrated in Figure 1(b), many graph-based RAG methods represent knowledge through fine-grained entities and relations, which can introduce redundant or loosely relevant information during retrieval (Chen et al., 2026). This mismatch between query requirements and retrieval granularity can negatively affect the quality of retrieved evidence. As shown in Figure 2, retrieval performance varies substantially across different retrieval units. In our preliminary analysis, atomic fact-level retrieval consistently achieves higher recall than chunk- and entity-level retrieval on both HotpotQA and MultiHopRAG, motivating the use of atomic facts as the primary retrieval unit. Atomic facts provide a proposition-level representation of factual information (Min et al., 2023; Chern et al., 2023). Atomic facts provide an intermediate granularity between coarse passages and isolated entity-level representations:

they preserve a complete factual proposition while remaining more focused than chunks. We therefore target fact-centric QA settings requiring fine-grained evidence composition rather than claiming atomic facts as universally optimal retrieval units.

Building on this insight, we introduce Atom-RAG, a framework that aligns retrieval granularity with the information needs of multi-hop reasoning. Atom-RAG represents documents as atomic facts and decomposes queries into fact-aligned sub-queries, enabling more precise evidence retrieval while filtering loosely relevant evidence. Experiments on diverse single-hop and multi-hop benchmarks show that Atom-RAG consistently outperforms strong baselines across different evaluation settings.

2 Related Work

2.1 Iterative RAG

Iterative RAG methods have been proposed to overcome the limitations of single-step retrieval in Naïve RAG for multi-hop queries by interleaving reasoning and retrieval across multiple steps. Self-Ask (Press et al., 2023) decomposes complex questions into follow-up queries to progressively retrieve evidence, while ITER-RETGEN (Shao et al., 2023) and IRCOT (Trivedi et al., 2023) further strengthen the interaction between generation and retrieval by using intermediate reasoning outputs to guide subsequent retrieval. Despite these advances, many iterative RAG implementations perform retrieval over passage- or chunk-level units, which can contain irrelevant or mixed information and lead to a mismatch between query intent and retrieved evidence (Mavi et al., 2024). In addition, iterative reasoning can propagate early-stage errors and increase query-time cost due to repeated

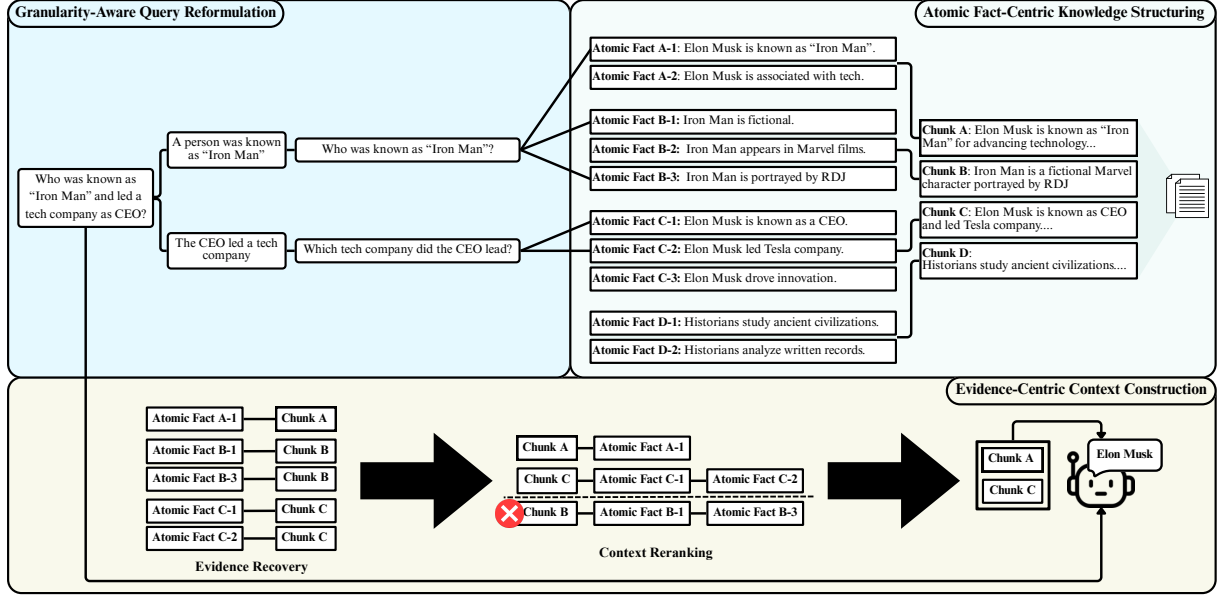


Figure 3: Overview of Atom-RAG, including atomic fact-centric structuring, granularity-aware query reformulation, and evidence-centric context construction.

retrieval and generation (Shao et al., 2023).

2.2 Graph-based RAG

Graph-based RAG approaches aim to support multi-hop reasoning by modeling relational structures between pieces of knowledge. Instead of retrieving isolated text chunks, these methods organize information as structured graphs and perform retrieval using graph-structured representations or traversal-based mechanisms (Han et al., 2024). GraphRAG (Edge et al., 2024) constructs entity–relation graphs from documents to enable structured retrieval over interconnected knowledge. LightRAG (Guo et al., 2025) constructs graph-structured knowledge representations from entities and relations extracted from text, while PathRAG (Chen et al., 2026) extends graph-based retrieval by explicitly searching and ranking relational paths within the graph. HyperGraphRAG (Luo et al., 2025a) introduces hypergraph structures to capture higher-order relationships and alleviate graph fragmentation. HippoRAG2 (Gutiérrez et al., 2025) improves graph construction and retrieval strategies to better capture semantic relationships across documents. Despite these advances, fine-grained graph representations can introduce challenges when the resulting retrieval structure becomes overly dense or contains many weakly relevant connections. In such cases, graph traversal may expose loosely relevant nodes or relations, making it more difficult to identify evidence that is closely aligned with the query

intent (Edge et al., 2024; Chen et al., 2026).

3 Method

We propose Atom-RAG, a retrieval framework designed to mitigate the granularity mismatch between queries and retrieved evidence. Atomic facts are used as proposition-level retrieval units for settings requiring precise composition of factual evidence. Atom-RAG consists of three components: Atomic Fact-Centric Structuring, Granularity-Aware Query Reformulation, and Evidence-Centric Context Construction. Figure 3 illustrates the overall architecture.

3.1 Atomic Fact-Centric Structuring

Atomic Fact Extraction. Given a corpus segmented into fixed-length chunks, we decompose each chunk into a set of atomic facts, where each atomic fact represents a minimal and semantically self-contained verifiable statement. Let $\mathcal{C} = \{c_1, c_2, \dots, c_{|\mathcal{C}|}\}$ denote the set of all chunks in the corpus. For each chunk $c_i \in \mathcal{C}$, we extract a corresponding set of atomic facts, denoted as $\mathcal{F}(c_i) = \{f_{i,1}, f_{i,2}, \dots, f_{i,|\mathcal{F}(c_i)|}\}$. To preserve provenance and enable faithful evidence recovery, each atomic fact is explicitly linked to its source chunk. This chunk–fact association enables fact-level retrieval while preserving access to the original textual context. The extraction process is implemented using an LLM-based prompt that instructs the model to generate single-claim and minimally

overlapping facts; the full prompt is provided in Appendix A.1. We treat the collection of all extracted atomic facts as the core knowledge structure of the system:

$$\mathcal{F} = \bigcup_{i=1}^{|\mathcal{C}|} \mathcal{F}(c_i). \quad (1)$$

Initial retrieval and query–evidence alignment are performed over $\mathcal{F}$, making atomic facts the primary units for fine-grained knowledge access. Importantly, atomic facts serve as the primary alignment units for retrieval. Queries are matched directly with atomic facts, and only their associated source chunks are recovered at later stages to construct an evidence-centric context. This design helps mitigate granularity mismatch in settings where chunk-level retrieval is too coarse or graph-based representations introduce unnecessarily fine-grained evidence units.

Chunk and Atomic Fact Storage. Our framework performs initial retrieval at the atomic fact level, while retaining source chunks for provenance preservation and subsequent context construction. We denote the set of chunks by $\mathcal{C} = \{c_1, \dots, c_{|\mathcal{C}|}\}$ and the set of atomic facts by $\mathcal{F} = \{f_1, \dots, f_{|\mathcal{F}|}\}$. Each atomic fact maintains an explicit link to its source chunk through a mapping $\pi : \mathcal{F} \rightarrow \mathcal{C}$, where $\pi(f_j)$ returns the chunk from which f_j was extracted. For efficient semantic retrieval, we embed only atomic facts and store their representations in a fact-level vector index $\mathcal{V}_F$ using a text embedding model $\text{Enc}(\cdot)$:

$$\mathbf{v}_{f_j} = \text{Enc}(f_j). \quad (2)$$

Once query-relevant atomic facts are identified, their associated source chunks are recovered via $\pi(\cdot)$ to construct an evidence-centric context. By separating fine-grained retrieval from context recovery, this design aims to limit irrelevant information from coarse chunk-level retrieval while preserving access to the original document context. A step-by-step description of the Atomic Fact-Centric Structuring process is provided in Appendix C.1.

3.2 Granularity-Aware Query Reformulation

Query Reformulation. Given an input query q , we first decompose it into a set of atomic fact-level query representations. This decomposition aligns the query representation with the atomic fact-centric knowledge structure introduced in Sec-

tion 3.1. We denote the resulting set as

$$\mathcal{Q}^{AF} = AF(q) = \{q_1^{AF}, q_2^{AF}, \dots, q_{|\mathcal{Q}^{AF}|}^{AF}\}, \quad (3)$$

where each q_k^{AF} represents a minimal and semantically self-contained query expression derived from q . Based on these representations, we generate a corresponding set of subqueries, where each subquery targets a specific information need implied by the original query:

$$\mathcal{S} = \{s_k \mid s_k = g(q_k^{AF}), q_k^{AF} \in \mathcal{Q}^{AF}\}. \quad (4)$$

The number of subqueries is determined dynamically by the LLM during decomposition rather than being fixed in advance, allowing the framework to accommodate queries with different reasoning requirements. The atomic fact decomposition and subquery generation are implemented using an LLM-based prompt provided in Appendix A.2.

Query-to-Fact Retrieval. To retrieve relevant evidence, each subquery $s_k \in \mathcal{S}$ is matched against the atomic fact knowledge base $\mathcal{F}$. Specifically, we compute the cosine similarity between the embedding of subquery s_k and each atomic fact $f_j \in \mathcal{F}$:

$$\text{sim}(s_k, f_j) = \cos(\text{Enc}(s_k), \text{Enc}(f_j)), \quad (5)$$

where $\text{Enc}(\cdot)$ denotes the shared embedding model used for both queries and atomic facts. For each subquery s_k , we select the top- K atomic facts with the highest similarity scores, forming a subquery-specific candidate set

$$\mathcal{F}_k^{(K)} = \text{TopK}_{f_j \in \mathcal{F}} \text{sim}(s_k, f_j). \quad (6)$$

By retrieving evidence directly at the atomic fact level, the framework mitigates the granularity mismatch between queries and retrieved evidence. Because both subqueries and knowledge units are expressed in the same representation, semantic matching can be performed in a uniform and fine-grained manner. This design also generalizes naturally across both single-hop and multi-hop queries. A step-by-step description of the Granularity-Aware Query Reformulation process is provided in Appendix C.2.

3.3 Evidence-Centric Context Construction

Evidence Recovery. Given the set of atomic facts selected during query-to-fact retrieval, we recover their corresponding source chunks to construct candidate evidence. Let $\mathcal{F}_k^{(K)}$ denote the top- K atomic

facts retrieved for subquery s_k . Using the provenance mapping $\pi(\cdot)$, we recover the associated chunks as

$$\mathcal{C}_k = \{\pi(f) \mid f \in \mathcal{F}_k^{(K)}\}. \quad (7)$$

The recovered chunks from all subqueries are then merged into a unified candidate set, with duplicates removed while preserving the original textual content. This delayed recovery avoids retrieving entire chunks during the initial retrieval stage.

Context Reranking. To prioritize relevant evidence, we rerank the recovered chunks based on their semantic relevance to the generated subqueries. Let $\mathbf{e}_{s_k} = \text{Enc}(s_k)$ and $\mathbf{e}_f = \text{Enc}(f)$ denote the embedding representations of subquery s_k and atomic fact f , respectively. For each chunk c , we compute a representative score using the maximum cosine similarity between any atomic fact $f \in \mathcal{F}(c)$ contained in the chunk and any subquery s_k :

$$\text{score}(c) = \max_k \max_{f \in \mathcal{F}(c)} \cos(\mathbf{e}_{s_k}, \mathbf{e}_f). \quad (8)$$

Chunks are then ranked according to $\text{score}(c)$, and only the top- K_c chunks are retained to construct the final evidence-centric context. This reranking strategy promotes chunks that contain highly relevant atomic facts while filtering out loosely related or noisy evidence.

Evidence-Grounded Answer Generation. Finally, the answer is generated by conditioning a large language model on the reranked chunks and the original query. Let $\mathcal{C}^{\text{final}}$ denote the top- K_c chunks selected after context reranking. The generation input is constructed by concatenating $\mathcal{C}^{\text{final}}$ with the original query q , which is then provided to the LLM through a generation prompt. This design encourages the model to generate answers grounded in the retrieved evidence. The full generation prompt is provided in Appendix A.3. A step-by-step description of the Evidence-Centric Context Construction process is provided in Appendix C.3.

4 Experimental Setup

4.1 Datasets

We evaluate Atom-RAG on a diverse set of question answering benchmarks covering both single-hop and multi-hop settings. These datasets are selected

to examine how effectively the proposed framework aligns queries with relevant evidence under varying reasoning requirements and evidence aggregation patterns.

Single-Hop Datasets. Single-hop benchmarks generally require answering questions without multi-step evidence composition and therefore provide a relatively direct setting for evaluating query–evidence alignment. **TriviaQA** (Joshi et al., 2017) contains factoid questions associated with long and potentially noisy documents, providing a challenging setting for retrieving relevant evidence. **Natural Questions (NQ)** (Kwiatkowski et al., 2019) consists of real search queries with answers grounded in Wikipedia articles, reflecting realistic information-seeking scenarios.

Multi-Hop Datasets. Multi-hop benchmarks require aggregating multiple pieces of evidence across documents, evaluating compositional reasoning and evidence integration. **HotpotQA** (Yang et al., 2018) requires connecting two supporting facts to answer a question and is widely used to evaluate explicit multi-hop reasoning. **2Wiki-MultiHopQA** (Ho et al., 2020) extends this setting to cross-document reasoning across multiple Wikipedia articles. **MuSiQue** (Trivedi et al., 2022) focuses on compositional questions that require reasoning over multiple implicit evidence steps. **MultiHopRAG** (Tang and Yang, 2024) requires evidence aggregation over a larger and more diverse corpus, emphasizing both retrieval accuracy and evidence integration.

Corpus Construction and Preprocessing. We construct the knowledge corpus directly from raw text. For each dataset, all documents are concatenated into a unified corpus and segmented into fixed-length chunks of 1200 tokens. We use the same chunk size across datasets to maintain a consistent preprocessing and atomic fact extraction procedure. Knowledge structuring is applied uniformly across benchmarks. Details of the dataset sampling procedure are provided in Appendix D, and corpus statistics are reported in Appendix E.

4.2 Baselines

We compare Atom-RAG against representative baselines spanning three categories.

Standard RAG. These baselines retrieve relevant text passages and generate answers conditioned on the retrieved context (Lewis et al., 2020).

Method	Single-Hop QA				Multi-Hop QA							
	TriviaQA		NQ		HotpotQA		2Wiki		MuSiQue		MultiHopRAG	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Standard RAG												
Pure LLM	0.678	0.625	0.636	0.526	0.464	0.408	0.331	0.334	0.149	0.139	0.668	0.605
Naïve RAG	0.735	0.670	0.758	0.597	0.627	0.531	0.442	0.408	0.349	0.283	0.739	0.680
Iterative RAG												
Self-Ask	0.729	0.632	0.723	0.562	0.614	0.497	0.587	0.487	0.367	0.311	0.752	0.488
ITER-RETGEN	0.721	0.675	0.756	0.591	0.598	0.501	0.689	0.624	0.308	0.276	0.744	0.696
IRCoT	0.724	0.671	0.746	0.584	0.712	0.542	0.351	0.292	0.341	0.292	0.490	0.372
Graph-Based RAG												
GraphRAG	0.702	0.645	0.625	0.472	0.720	0.668	0.474	0.435	0.273	0.247	0.752	0.699
LightRAG	0.700	0.647	0.660	0.479	0.590	0.496	0.443	0.404	0.265	0.217	0.529	0.464
PathRAG	0.701	0.649	0.684	0.494	0.655	0.551	0.463	0.411	0.310	0.248	0.590	0.468
HyperGraphRAG	0.699	0.644	0.692	0.533	0.775	0.674	0.566	0.499	0.459	0.360	0.629	0.526
HippoRAG2	0.725	0.683	0.720	0.563	0.781	0.663	0.646	0.559	0.431	0.385	0.764	0.708
Atom-RAG (ours)	0.743	0.686	0.760	0.599	0.861	0.700	0.714	0.651	0.498	0.385	0.792	0.754

Table 1: Performance comparison on single-hop and multi-hop QA datasets using Acc (LLM-as-a-judge) and F1.

Pure LLM (Petroni et al., 2019; Brown et al., 2020) answers questions using only the parametric knowledge of the language model, without external retrieval. **Naïve RAG** (Karpukhin et al., 2020; Lewis et al., 2020) represents the standard retrieval-augmented framework that retrieves fixed-length text chunks via dense similarity and conditions generation on the retrieved context.

Iterative RAG. These baselines explicitly perform multi-step reasoning by interleaving retrieval and generation. **Self-Ask** (Press et al., 2023) decomposes complex queries into sequential follow-up questions and retrieves evidence iteratively. **ITER-RETGEN** (Shao et al., 2023) further tightens this interaction by using intermediate generations to guide subsequent retrieval. **IRCoT** (Trivedi et al., 2023) integrates chain-of-thought reasoning with iterative retrieval, using intermediate reasoning steps to guide evidence acquisition.

Graph-based RAG. These baselines leverage structured knowledge representations for multi-hop retrieval. **GraphRAG** (Edge et al., 2024) constructs entity–relation graphs from documents and uses the resulting graph structure to support retrieval and generation. **LightRAG** (Guo et al., 2025) constructs graph-structured representations from extracted entities and relations and retrieves relevant graph-based knowledge for generation. **PathRAG** (Chen et al., 2026) extends graph-based retrieval by explicitly searching and ranking reasoning paths. **HyperGraphRAG** (Luo et al., 2025a) uses hypergraph structures to represent higher-

order, n-ary relationships among multiple entities, while **HippoRAG2** (Gutiérrez et al., 2025) extends memory-inspired graph retrieval using Personalized PageRank, deeper passage integration, and online LLM reasoning. Implementation details are provided in Appendix F, while additional comparisons with recent baselines and embedding configurations are provided in Appendix H.

4.3 Metrics

We evaluate performance using both answer-level and retrieval-level metrics. Answer quality is evaluated using an LLM-as-a-judge protocol, reported as Acc in the main result tables, together with token-level F1 as a complementary lexical overlap metric (Zheng et al., 2023; Rajpurkar et al., 2016). To evaluate retrieval independently of generation, we also report Retrieval Recall@7, measuring the proportion of gold supporting evidence covered by the top-7 retrieved evidence chunks, following standard practice in retrieval-augmented QA (Karpukhin et al., 2020; Lewis et al., 2020). Detailed definitions and evaluation procedures for all metrics are provided in Appendix G.1.

4.4 Implementation Details

We use GPT-4o-mini (gpt-4o-mini-2024-07-18) for all LLM-dependent components across Atom-RAG and the baselines, including construction, intermediate reasoning, query reformulation, and generation when applicable. We use text-embedding-3-small for all embedding-based retrieval components. Retrieval budgets are aligned as closely as possible while preserving each

Method	HotpotQA	2Wiki	MuSiQue	MultiHop
				RAG
Standard RAG				
Naïve RAG	0.825	0.610	0.624	0.641
Iterative RAG				
Self-Ask	0.893	0.825	0.654	0.646
ITER-RETGEN	0.835	0.894	0.569	0.743
IRCoT	0.835	0.644	0.765	0.532
Atom-RAG (ours)	0.964	0.914	0.792	0.771

Table 2: Retrieval performance across multi-hop QA benchmarks, measured by Recall@7.

method’s original retrieval structure. Atom-RAG retrieves the top-10 atomic facts per subquery and retains the top-7 source chunks after reranking. Detailed settings are provided in Appendix F.

5 Main Results

5.1 QA Performance

Table 1 reports QA performance on both single-hop and multi-hop QA benchmarks. Overall, Atom-RAG achieves the best or tied-best performance across all datasets in terms of Acc and F1. On single-hop datasets, Atom-RAG performs competitively with strong baselines. For example, Atom-RAG achieves an Acc of 0.743 on TriviaQA and 0.760 on NQ, outperforming Naïve RAG and other retrieval-based baselines while also achieving the highest F1 scores. These results indicate that atomic fact-level retrieval remains competitive on simple factoid questions despite using finer-grained retrieval units. The advantages of Atom-RAG become more pronounced on multi-hop benchmarks. On HotpotQA, Atom-RAG achieves 0.861 Acc and 0.700 F1, outperforming HippoRAG2 (0.781 Acc, 0.663 F1) and other graph-based baselines. Similarly, on 2Wiki, Atom-RAG achieves an Acc of 0.714, outperforming HyperGraphRAG (0.566) and HippoRAG2 (0.646). On MuSiQue and MultiHopRAG, Atom-RAG also achieves the best or tied-best performance across both metrics. These improvements are consistent with the benefits of Atom-RAG’s atomic fact-centric retrieval and evidence construction design. By retrieving evidence at a finer granularity, Atom-RAG can avoid introducing entire passages during initial retrieval and does not require traversal over potentially dense graph structures. Aligning queries with atomic facts enables more targeted evidence selection, which is associated with improved performance on multi-hop QA tasks. Overall, these results are con-

Method	2Wiki		MultiHopRAG	
	Acc	F1	Acc	F1
Atom-RAG (ours)	0.714	0.651	0.792	0.754
w/o Atomic Fact	0.418	0.420	0.692	0.689
w/o Query Reformulation	0.624	0.617	0.723	0.718
w/o Context Reranking	0.576	0.572	0.704	0.698

Table 3: Ablation results of Atom-RAG on 2Wiki and MultiHopRAG.

sistent with the benefit of aligning complex queries with fine-grained atomic fact retrieval, particularly on multi-hop QA. Additional experimental results using alternative evaluation metrics are provided in Appendix G.2.

5.2 Retrieval Performance

Table 2 reports retrieval performance across multiple multi-hop QA benchmarks measured by Recall@7. Graph-based RAG methods are excluded because their retrieved graph units cannot be reliably mapped to source chunks for chunk-level Recall@7 evaluation. Naïve RAG shows relatively lower retrieval performance with its flat chunk-level retrieval strategy. Iterative RAG methods improve retrieval on some datasets but exhibit inconsistent performance across benchmarks. In contrast, Atom-RAG achieves the best retrieval performance among the evaluated methods on all datasets, reaching 0.964 on HotpotQA, 0.914 on 2Wiki, 0.792 on MuSiQue, and 0.771 on MultiHopRAG. These results indicate that atomic fact-guided retrieval improves evidence alignment and enables more reliable multi-hop retrieval across diverse QA benchmarks.

6 Discussions

6.1 Ablation Study

We conduct an ablation study on the 2Wiki and MultiHopRAG datasets to examine the contribution of each component in Atom-RAG. Table 3 shows that the full model consistently outperforms all ablated variants across both datasets. Removing atomic fact modeling (*w/o Atomic Fact*) results in the largest performance drop, especially on 2Wiki. This suggests that atomic fact-level representations provide a more effective retrieval unit than the chunk-level alternative used in the ablated variant. Eliminating granularity-aware query reformulation (*w/o Query Reformulation*) also degrades performance, indicating that reformulating

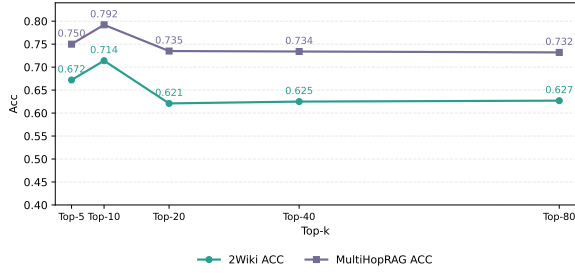


Figure 4: Sensitivity analysis on subquery-based atomic fact retrieval with varying top- K .

the query helps better align the query with the information to be retrieved. Similarly, removing context reranking (*w/o Context Reranking*) leads to lower scores, indicating that prioritizing more relevant candidate evidence improves the quality of the final context. Notably, removing atomic fact modeling causes the largest degradation, particularly on 2Wiki (0.714→0.418 Acc and 0.651→0.420 F1), indicating that the atomic fact representation itself is the core contributor. Query reformulation and context reranking provide additional, complementary gains.

6.2 Sensitivity on Subquery-based Atomic Retrieval

We analyze the sensitivity of Atom-RAG to the number of atomic facts retrieved per subquery by varying top- K . Figure 4 shows the results on 2Wiki and MultiHopRAG. Performance improves up to $K = 10$ on both datasets, where the best results are observed (0.714 Acc on 2Wiki and 0.792 Acc on MultiHopRAG). Beyond $K = 10$, performance declines but remains relatively stable. On 2Wiki, Acc is 0.621, 0.625, and 0.627 at $K = 20, 40$, and 80 , respectively; on MultiHopRAG, the corresponding scores are 0.735, 0.734, and 0.732. Overall, $K = 10$ provides the best observed performance.

6.3 Sensitivity to Final Context Size

We study the sensitivity of Atom-RAG to the final context size by varying the number of source chunks retained for answer generation, denoted as K_c . Figure 5 reports performance measured by Acc. Across datasets, performance improves or remains stable as K_c increases from 3 to 7, indicating that a moderate increase in context size helps include sufficient supporting evidence. In particular, $K_c = 7$ achieves the best performance on HotpotQA, MuSiQue, and MultiHopRAG, while maintaining competitive performance on 2Wiki. How-

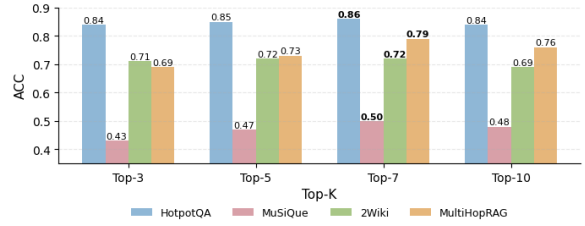


Figure 5: Sensitivity analysis of final context size (Top- K_c) using Acc (LLM-based accuracy).

ever, further increasing the context size to $K_c = 10$ results in a slight performance drop across most benchmarks, suggesting diminishing benefits from adding more context. Overall, these results show that Atom-RAG is robust to reasonable variations in K_c , with $K_c = 7$ providing the best observed overall performance. Accordingly, we use $K_c = 7$ as the default final context size in all experiments.

6.4 Efficiency, Cost Performance, and Stability Analysis

Table 4 compares construction and inference efficiency across methods. Although Atom-RAG requires atomic fact extraction, this cost is incurred once during offline corpus construction and reused across queries. Its construction overhead remains comparable to graph-based RAG methods, while achieving the lowest inference latency among the evaluated iterative and graph-based baselines. Atom-RAG also achieves the highest Acc/\$ and Acc/s within this group, demonstrating a favorable cost-performance trade-off. Table 5 further shows only small variations across three runs, indicating stable performance under the same configuration.

6.5 Atomic Fact Extraction Quality and Error Analysis

Atomic fact extraction errors mainly affect retrieval because atomic facts are used to identify relevant source chunks, while the original chunks are provided to the generator. We inspect atomic facts from 100 sampled HotpotQA chunks against their source chunks and observe three recurring errors: omitted modifiers, unresolved references, and missing relation details, all of which can weaken query-fact matching. We evaluate extraction quality using support precision, which measures source support; decontextualization, which measures whether a fact is self-contained; and incomplete fact rate, which measures missing information needed to preserve the original claim. As shown in Table 6, improving the extraction prompt and applying evaluation-

Method	Construction		Generation Efficiency			HotpotQA Cost-Effectiveness	
	Build Time /1k Tokens	Build Cost /1k Tokens	Latency /Query	Cost /1k Queries	Acc (HotpotQA)	Acc/\$	Acc/s
Pure LLM	0 s	0 \$	0.970 s	0.059 \$	0.464	7.864	0.478
Naïve RAG	0 s	0 \$	1.089 s	1.016 \$	0.627	0.617	0.576
Self-Ask	0 s	0 \$	3.267 s	3.048 \$	0.614	0.201	0.188
ITER-RETGEN	0 s	0 \$	2.478 s	2.032 \$	0.598	0.294	0.241
IRCoT	0 s	0 \$	2.913 s	2.441 \$	0.712	0.292	0.244
GraphRAG	63.18 s	0.0108 \$	3.482 s	3.241 \$	0.720	0.222	0.207
LightRAG	46.34 s	0.0079 \$	2.659 s	2.959 \$	0.590	0.199	0.222
PathRAG	46.34 s	0.0079 \$	3.230 s	3.116 \$	0.655	0.210	0.203
HyperGraphRAG	27.651 s	0.0058 \$	1.896 s	2.724 \$	0.775	0.285	0.409
HippoRAG2	41.73 s	0.0072 \$	2.214 s	2.681 \$	0.781	0.291	0.353
Atom-RAG (ours)	37.22 s	0.0063 \$	1.717 s	2.114 \$	0.861	0.407	0.501

Table 4: Efficiency and cost–performance comparison across RAG methods. Acc/\$ and Acc/s are computed using HotpotQA Acc and the corresponding generation cost and inference latency, respectively.

	HotpotQA		2Wiki	
	Acc	F1	Acc	F1
Ours (run1)	0.861	0.700	0.714	0.651
Ours (run2)	0.859	0.710	0.713	0.639
Ours (run3)	0.855	0.696	0.715	0.644

Table 5: Run stability of Atom-RAG across multiple executions.

Setting	Support Precision	Decontext.	Incomplete Rate
Original Prompt	0.9346	0.9419	0.0409
Improved Prompt	0.9651	0.9702	0.0206
+ Refinement	0.9726	0.9803	0.0145

Table 6: Atomic fact extraction quality. Higher is better except for incomplete fact rate.

guided refinement before indexing improve all three dimensions. Refinement checks source support, self-containedness, and relation completeness, increasing support precision from 0.9346 to 0.9726 and decontextualization from 0.9419 to 0.9803, while reducing incomplete fact rate from 0.0409 to 0.0145. These improvements also yield downstream gains on a 100-example HotpotQA subset in Acc (LLM-as-a-judge) (0.830→0.840), F1 (0.672→0.677), indicating that higher-quality facts improve query–fact matching and source-chunk recovery. Additional extractor comparisons are provided in Appendix I.

6.6 Failure Analysis of Query Reformulation

Atom-RAG performs query reformulation once before retrieval, avoiding sequential dependence but making the initial decomposition critical. We ob-

serve three failure modes: *aspect-shift*, where a subquery targets the wrong aspect; *missing-step*, where a required information need is omitted; and *over-broad* subqueries, which retrieve diffuse or weakly relevant evidence. For example, a question asking what *Kát'a Kabanová* and *Der ferne Klang* have in common was decomposed around their themes rather than work type, causing retrieval to miss that both are operas. Thus, even accurate fact-level retrieval can fail when the reformulated subqueries do not preserve the original information need. Lightweight feedback after initial retrieval may mitigate such errors while retaining one-shot reformulation efficiency.

7 Conclusion

We proposed Atom-RAG, an atomic fact–centric retrieval framework that aligns retrieval granularity with multi-hop reasoning requirements. Atom-RAG decomposes documents into atomic facts and retrieves evidence at finer granularity, enabling precise query–evidence alignment. Experiments on single-hop and multi-hop QA benchmarks show that Atom-RAG consistently achieves strong performance, outperforming or matching iterative and graph-based RAG baselines across evaluated settings. These results highlight the effectiveness of aligning query and retrieval granularity for improving evidence retrieval and multi-hop QA performance in retrieval-augmented generation systems.

Limitations

Atom-RAG has several limitations. First, atomic fact extraction introduces offline construction cost

and may produce incomplete or context-dependent facts, although Section 6.5 shows that refinement can reduce these errors. Second, Atom-RAG primarily targets fact-centric QA; tasks requiring discourse-level, narrative, procedural, or long-context reasoning may benefit from hybrid or larger retrieval units. Third, one-shot query factorization can miss required reasoning steps, motivating future lightweight corrective feedback. Finally, comparison with concurrent RAG methods remains useful as the field evolves.

Acknowledgments

This work was supported by the Institute of Information and communications Technology Planning and evaluation (IITP) grant (No.RS-2025-25422680, No. RS-2020-II201373), and the National Research Foundation of Korea (NRF) grant (No. RS-2025-00520618) funded by the Korean Government (MSIT).

References

- Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.
- Boyue Chen, Zirui Guo, Zidan Yang, Yuluo Chen, Junze Chen, Zhenghao Liu, Chuan Shi, and Cheng Yang. 2026. *PathRAG: Pruning graph-based retrieval augmented generation with relational paths*. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 30183–30191.
- I-Chun Chern, Steffi Chern, Shiqi Chen, Weizhe Yuan, Kehua Feng, Chunting Zhou, Junxian He, Graham Neubig, and Pengfei Liu. 2023. FacTool: Factuality detection in generative AI – a tool augmented framework for multi-task and multi-domain scenarios. *arXiv preprint arXiv:2307.13528*.
- Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonello, and Fabrizio Silvestri. 2024. The power of noise: Redefining retrieval for RAG systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 719–729.
- Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitan, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph RAG approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.
- Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.
- Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2025. *LightRAG: Simple and fast retrieval-augmented generation*. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 10746–10761.
- Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to memory: Non-parametric continual learning for large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 21497–21515. PMLR.
- Haoyu Han, Yu Wang, Harry Shomer, Kai Guo, Jiayuan Ding, Yongjia Lei, Mahantesh Halappanavar, Ryan A. Rossi, Subhabrata Mukherjee, Xianfeng Tang, Qi He, Zhigang Hua, Bo Long, Tong Zhao, Neil Shah, Amin Javari, Yinglong Xia, and Jiliang Tang. 2024. Retrieval-augmented generation with graphs (GraphRAG). *arXiv preprint arXiv:2501.00309*.
- Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625.
- Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611.
- Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.
- Omar Khattab, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. 2022. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive NLP. *arXiv preprint arXiv:2212.14024*.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob

- Uszkoreit, Quoc Le, and Slav Petrov. 2019. [Natural questions: A benchmark for question answering research](#). *Transactions of the Association for Computational Linguistics*, 7:452–466.
- Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474.
- Haoran Luo, Haihong E, Guanting Chen, Yandan Zheng, Xiaobao Wu, Yikai Guo, Qika Lin, Yu Feng, Zemin Kuang, Meina Song, Yifan Zhu, and Anh Tuan Luu. 2025a. HyperGraphRAG: Retrieval-augmented generation via hypergraph-structured knowledge representation. In *Advances in Neural Information Processing Systems*, volume 38, pages 152206–152234.
- Linhao Luo, Zicheng Zhao, Reza Haffari, Dinh Phung, Chen Gong, and Shirui Pan. 2025b. [GFM-RAG: Graph foundation model for retrieval augmented generation](#). In *Advances in Neural Information Processing Systems*, volume 38, pages 36371–36405.
- Yizhuo Ma, Shuang Liang, Rongzheng Wang, Jiakai Li, Qizhi Chen, Muquan Li, and Ke Qin. 2026. Mitigating KG quality issues: A robust multi-hop GraphRAG retrieval framework. *arXiv preprint arXiv:2603.14828*.
- Vaibhav Mavi, Anubhav Jangra, and Adam Jatowt. 2024. Multi-hop question answering. *Foundations and Trends in Information Retrieval*, 17(5):457–586.
- Sewon Min, Kalpesh Krishna, Xinxu Lyu, Mike Lewis, Wen-tau Yih, Pang Koh, Mohit Iyyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2023. FActScore: Fine-grained atomic evaluation of factual precision in long form text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12076–12100.
- Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473.
- Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.
- 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. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9248–9274.
- Yixuan Tang and Yi Yang. 2024. MultiHop-RAG: Benchmarking retrieval-augmented generation for multi-hop queries. In *First Conference on Language Modeling*.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. [MuSiQue: Multi-hop questions via single-hop question composition](#). *Transactions of the Association for Computational Linguistics*, 10:539–554.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037.
- 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.
- Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, pages 46595–46623.

A Prompts Used in Ours

A.1 Atomic Fact-Centric Structuring Prompt

To extract atomic facts from document chunks, we employ an LLM-based prompt designed for fact-centric knowledge decomposition. The prompt instructs the model to decompose an input text chunk into a list of minimal, self-contained, and verifiable atomic facts, with instructions for single-claim decomposition, minimizing content overlap, explicit pronoun resolution, and consistent output formatting. The full prompt template used in our framework is shown in Figure 6.

A.2 Granularity-Aware Query Reformulation Prompt

To perform granularity-aware query reformulation, we employ an LLM-based prompt that decomposes an input query into atomic reasoning units and generates one independent sub-query for each unit. The prompt specifies minimal and self-contained decomposition, explicit resolution of ambiguous references, and strict JSON-formatted output to improve output consistency and facilitate downstream retrieval. The full prompt template is shown in Figure 7.

A.3 Evidence-Grounded Answer Generation Prompt

To generate the final answer, we employ a reasoning-oriented prompt that conditions a large language model on the retrieved context and the input query. The prompt explicitly specifies a structured response format consisting of a reasoning process (Thought) followed by a concise final answer (Answer), encouraging step-by-step inference while maintaining a consistent answer format. A representative example of the answer generation prompt is shown in Figure 8.

B Pipeline Example

Figure 9 illustrates an example execution of Atom-RAG, showing how query factorization and answer-generation prompts are applied within the pipeline. These prompts support structured decomposition of input queries and structured reasoning during answer generation.

C Algorithm Detail

C.1 Atomic Fact-Centric Structuring

Algorithm 1: Atomic Fact-Centric Structuring

Input: Document corpus segmented into chunks $\mathcal{C} = \{c_1, \dots, c_{|\mathcal{C}|}\}$
Output: Atomic fact set $\mathcal{F}$, provenance mapping π , fact-level vector index $\mathcal{V}_F$

Initialize atomic fact set $\mathcal{F} \leftarrow \emptyset$;
Initialize provenance mapping π ;
for each chunk $c_i \in \mathcal{C}$ **do**
 Extract atomic facts $\mathcal{F}(c_i)$ from c_i using an LLM-based extraction prompt;
 for each atomic fact $f_{i,j} \in \mathcal{F}(c_i)$ **do**
 Add $f_{i,j}$ to $\mathcal{F}$ and set $\pi(f_{i,j}) \leftarrow c_i$;
for each atomic fact $f_j \in \mathcal{F}$ **do**
 Compute embedding $\mathbf{v}_{f_j} \leftarrow \text{Enc}(f_j)$ and insert it into $\mathcal{V}_F$;
return $\mathcal{F}, \pi, \mathcal{V}_F$;

C.2 Granularity-Aware Query Reformulation

Algorithm 2: Granularity-Aware Query Reformulation

Input: Input query q , atomic fact knowledge base $\mathcal{F}$
Output: Subquery set $\mathcal{S}$ and subquery-specific atomic fact sets $\{\mathcal{F}_k^{(K)}\}_{k=1}^{|\mathcal{S}|}$

Query Decomposition:
Decompose the input query q into atomic fact-level query representations $\mathcal{Q}^{AF} = \{q_1^{AF}, \dots, q_{|\mathcal{Q}^{AF}|}^{AF}\}$ using an LLM-based prompt;
Generate subqueries $\mathcal{S} = \{s_k = g(q_k^{AF}) \mid q_k^{AF} \in \mathcal{Q}^{AF}\}$;
for each subquery $s_k \in \mathcal{S}$ **do**
 Compute $\text{sim}(s_k, f_j) = \cos(\text{Enc}(s_k), \text{Enc}(f_j))$ for all $f_j \in \mathcal{F}$;
 Select top- K atomic facts to form $\mathcal{F}_k^{(K)}$;
return $\mathcal{S}, \{\mathcal{F}_k^{(K)}\}_{k=1}^{|\mathcal{S}|}$;

C.3 Evidence-Centric Context Construction

Algorithm 3: Evidence-Centric Context Construction

Input: Subquery set $\mathcal{S}$, subquery-specific atomic fact sets $\{\mathcal{F}_k^{(K)}\}_{k=1}^{|\mathcal{S}|}$, provenance mapping $\pi(\cdot)$, atomic fact set $\mathcal{F}$, input query q

Output: Final evidence-centric context $\mathcal{C}^{\text{final}}$ and generated answer a

Evidence Recovery::

for each subquery $s_k \in \mathcal{S}$ **do**

Retrieve associated chunks

$\mathcal{C}_k = \{\pi(f) \mid f \in \mathcal{F}_k^{(K)}\};$

Merge all recovered chunks to form a unified candidate set $\mathcal{C}^* = \bigcup_k \mathcal{C}_k$;

Remove duplicate chunks while preserving original content;

Context Reranking::

for each chunk $c \in \mathcal{C}^*$ **do**

Identify the atomic facts associated with c as $\mathcal{F}(c) = \{f \in \mathcal{F} \mid \pi(f) = c\};$

Compute $\text{score}(c) = \max_k \max_{f \in \mathcal{F}(c)} \cos(\text{Enc}(s_k), \text{Enc}(f));$

Rank all chunks according to $\text{score}(c)$ and select the top- K_c chunks to form $\mathcal{C}^{\text{final}}$;

Evidence-Grounded Answer Generation::

Generate answer $a = \text{LLM}(q, \mathcal{C}^{\text{final}})$ using the evidence-grounded generation prompt;

return $\mathcal{C}^{\text{final}}, a$;

D Dataset Construction

We sample datasets to evaluate Atom-RAG across different QA paradigms, including single-hop factual QA and multi-hop QA.

TriviaQA. We randomly sample 1,000 questions from TriviaQA for evaluation. TriviaQA is typically regarded as a single-hop open-domain QA benchmark, where questions generally do not require explicit multi-step evidence composition.

Natural Questions (NQ). For single-hop factual QA evaluation, we randomly sample 1,000 queries from the Natural Questions (NQ) dataset. The sampled questions cover a wide range of real user information needs and are used without any additional filtering. We retain the original Wikipedia-derived documents associated with the sampled queries and construct the retrieval corpus from these documents.

HotpotQA. For multi-hop QA evaluation, we randomly sample 1,000 queries from HotpotQA. These questions require reasoning over multiple supporting passages. We follow the sampling protocol used in HippoRAG, retaining the Wikipedia-based corpus and gold supporting documents associated with each sampled question.

2WikiMultiHopQA. From the 2WikiMultiHopQA dataset, we randomly select 1,000 queries. These questions require reasoning across multiple Wikipedia articles, and we preserve the original retrieval corpus and gold evidence annotations.

MuSiQue. We randomly sample 1,000 queries from the MuSiQue dataset. MuSiQue is specifically designed for multi-hop reasoning through compositional question construction, making it suitable for evaluating compositional evidence retrieval and reasoning. No additional constraints are imposed during sampling, and the original corpus is retained.

MultiHopRAG. We randomly sample 1,000 queries from the MultiHopRAG dataset. These questions require retrieving and integrating evidence across multiple passages to derive the final answer.

E Statistics of Knowledge Construction

Table 7 reports statistics of the datasets and the structured knowledge representations constructed by different RAG frameworks. LightRAG and PathRAG use the same underlying entity–relation (triple-based) graph in our experiments, while employing different retrieval strategies. Iterative RAG methods such as Self-Ask and ITER-RETGEN do not rely on explicitly structured knowledge representations; instead, they retrieve evidence as fixed-length text chunks and perform step-wise reasoning over these chunks. HyperGraphRAG introduces hyperedges to capture higher-order relations, resulting in a larger structured knowledge representation in terms of the reported knowledge units. In contrast, Atom-RAG represents knowledge directly as atomic facts, using fine-grained factual statements as its fundamental knowledge units. Across all datasets, Atom-RAG constructs more atomic fact units than the combined number of entities and relations in the triple-based representations used by LightRAG and PathRAG. This difference reflects the finer granularity of Atom-RAG’s knowledge representation, which preserves factual statements

	TriviaQA	NQ	HotpotQA	2WikiMultiHopQA	MuSiQue	MultiHopRAG
Dataset Statistics						
# Queries	1,000	1,000	1,000	1,000	1,000	1,000
# Chunks	653	432	1,176	606	1,207	909
Graph-based Knowledge Statistics						
<i>GraphRAG</i>						
# Entities	10,842	5,973	49,812	12,906	22,681	12,094
# Relations	5,219	2,941	57,335	4,728	9,402	7,201
<i>LightRAG / PathRAG (shared graph)</i>						
# Entities	12,328	6,851	57,235	14,718	27,043	14,253
# Relations	6,507	3,534	69,203	5,512	11,511	8,900
<i>HyperGraphRAG</i>						
# Entities	64,969	35,322	107,705	53,535	107,697	73,392
# Hyperedges	54,511	28,305	84,503	39,872	91,176	56,716
<i>HippoRAG2</i>						
# Entities	14,105	7,812	61,427	16,504	29,816	15,604
# Relations	7,012	3,968	72,551	6,193	12,983	9,764
<i>Atom-RAG (ours)</i>						
# Atomic Facts	67,067	40,640	112,031	60,085	121,336	62,201

Table 7: Statistics of datasets and constructed knowledge structures across RAG frameworks.

as independent retrieval units rather than decomposing them solely into entities and relations. Despite its finer granularity, Atom-RAG maintains a knowledge representation of a scale comparable to HyperGraphRAG in terms of the reported number of knowledge units. Overall, these statistics highlight the distinct granularity of Atom-RAG’s fact-centric representation compared with entity–relation and hypergraph-based knowledge structures.

F Baseline Details

Table 8 summarizes the method-specific retrieval and generation settings. All LLM-dependent stages use GPT-4o-mini, and all embedding-based retrieval components use text-embedding-3-small. Because retrieval units differ across methods, Top- K values are method-specific and should not be interpreted as directly equivalent retrieval capacities. We preserve each method’s original retrieval structure while aligning the retrieval budget as closely as possible.

G Evaluation Details

This section describes the evaluation protocols used in our experiments. We first present the primary evaluation metrics used in the main results, fol-

lowed by additional metrics and results reported for supplementary analysis.

G.1 Primary Evaluation Metrics

Acc (LLM-Judged Accuracy) In addition to automatic string-based metrics, we evaluate answer correctness using an LLM-as-a-judge protocol, which serves as the primary accuracy metric in the main results. Specifically, we employ GPT-4o-mini as an external evaluator to assess whether a predicted answer correctly conveys the essential factual information required by the question. The evaluation prompt instructs the model to classify each prediction as either CORRECT or INCORRECT, allowing for paraphrasing and minor wording variations as long as the core fact is preserved. For questions with multiple acceptable gold answers, a prediction is considered correct if it is judged semantically consistent with any one of the acceptable answers. A representative example of the evaluation prompt is shown in Figure 10. To reduce variability in the judgments, the decoding temperature is set to zero. The final LLM-based accuracy is computed as the proportion of predictions labeled CORRECT by the judge model.

F1 Score We additionally report the token-level F1 score following the standard SQuAD-style eval-

Method	Retrieval Type	Top- K Units	Final Generation Input	Generation Model
Pure LLM	None	–	–	GPT-4o-mini
Naïve RAG	Chunk	7 chunks	7 chunks	GPT-4o-mini
Self-Ask	Chunk	7 chunks	7 chunks	GPT-4o-mini
ITER-RETGEN	Chunk	7 chunks	7 chunks	GPT-4o-mini
IRCoT	Chunk	7 chunks	7 chunks	GPT-4o-mini
GraphRAG	Entity + Relation Graph	60	Graph Context (Entities/Relations)	GPT-4o-mini
LightRAG	Entity + Relation	60	Entity/Relation Info	GPT-4o-mini
PathRAG	Graph Path	60	Path Info	GPT-4o-mini
HyperGraphRAG	Entity + Hyperedge	60	Entity/Hyperedge Info	GPT-4o-mini
HippoRAG2	Entity Graph (Memory-based)	60	Entity Memory Context	GPT-4o-mini
Atom-RAG (ours)	Atomic Fact	10	7 chunks	GPT-4o-mini

Table 8: Comparison of retrieval and generation configurations across different RAG frameworks. Top- K units indicate the number of knowledge units retrieved at inference time, while the final generation input specifies the information provided to the language model for answer generation.

uation. Both predicted and gold answers are first normalized by lowercasing, removing articles (*a*, *an*, *the*), punctuation, and extra whitespace. Let P and G denote the multisets of tokens in the predicted and gold answers, respectively. Token overlap is computed using multiset intersection. Precision and recall are defined as:

$$\text{Precision} = \frac{|P \cap G|}{|P|}, \quad \text{Recall} = \frac{|P \cap G|}{|G|}. \quad (9)$$

The F1 score is then computed as:

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \quad (10)$$

For datasets with multiple gold answers, we report the maximum F1 score across all gold candidates for each question.

Recall@7 We additionally report Recall@7 to evaluate retrieval effectiveness. Recall@7 measures whether the retrieved evidence set contains the gold-standard chunks within the top-7 retrieved results. For each query, let G denote the set of gold evidence chunks and R_7 denote the set of retrieved chunks truncated to the top 7. Recall@7 is computed as:

$$\text{Recall@7} = \frac{|G \cap R_7|}{|G|}. \quad (11)$$

This metric captures the coverage of required evidence under a fixed retrieval budget and is particularly relevant for multi-hop question answering, where missing a single critical evidence chunk can lead to reasoning failure.

G.2 Additional Evaluation Metrics and Results

To provide a more comprehensive analysis of QA performance, we additionally report Accuracy and Exact Match (EM) metrics across all datasets.

Accuracy (Substring Match) Accuracy measures whether any normalized gold answer string is contained within the normalized predicted answer. A prediction is considered correct if at least one acceptable gold answer appears as a substring of the normalized prediction. This metric captures cases where the model produces a correct answer embedded within a longer explanatory output. The final accuracy is computed as the ratio of correct predictions over all evaluated questions.

Exact Match (EM) Exact Match (EM) evaluates whether the normalized predicted answer exactly matches a normalized gold answer. Let $\hat{y}$ and y denote the normalized predicted and gold answer strings, respectively. Formally, EM is defined as:

$$\text{EM} = \begin{cases} 1, & \text{if } \hat{y} = y, \\ 0, & \text{otherwise.} \end{cases} \quad (12)$$

For datasets with multiple gold answers, we take the maximum EM score over all gold candidates.

Additional QA Performance Results Table 9 summarizes the performance of all methods on both single-hop and multi-hop QA benchmarks using Accuracy and EM metrics. These results provide complementary perspectives on answer cor-

Method	Single-Hop QA				Multi-Hop QA							
	TriviaQA		NQ		HotpotQA		2Wiki		MuSiQue		MultiHopRAG	
	Accuracy	EM	Accuracy	EM	Accuracy	EM	Accuracy	EM	Accuracy	EM	Accuracy	EM
Standard RAG												
Pure LLM	0.618	0.562	0.512	0.412	0.351	0.287	0.311	0.273	0.089	0.038	0.623	0.593
Naïve RAG	0.677	0.601	0.596	0.432	0.481	0.401	0.399	0.336	0.223	0.161	0.704	0.658
Iterative RAG												
Self-Ask	0.682	0.577	0.532	0.404	0.456	0.345	0.549	0.376	0.254	0.196	0.705	0.447
ITER-RETGEN	0.661	0.614	0.580	0.436	0.428	0.372	0.625	0.533	0.197	0.132	0.699	0.688
IRCoT	0.668	0.609	0.561	0.433	0.533	0.411	0.362	0.229	0.281	0.241	0.558	0.374
Graph-Based RAG												
GraphRAG	0.651	0.607	0.492	0.363	0.553	0.515	0.423	0.338	0.184	0.145	0.691	0.689
LightRAG	0.655	0.583	0.504	0.340	0.439	0.370	0.392	0.335	0.165	0.112	0.526	0.412
PathRAG	0.654	0.581	0.508	0.340	0.488	0.414	0.401	0.347	0.199	0.134	0.525	0.422
HyperGraphRAG	0.652	0.577	0.544	0.412	0.599	0.523	0.505	0.403	0.310	0.211	0.621	0.446
HippoRAG2	0.683	0.615	0.587	0.432	0.592	0.512	0.556	0.472	0.331	0.234	0.702	0.697
Atom-RAG (ours)	0.699	0.617	0.640	0.460	0.675	0.524	0.658	0.585	0.332	0.247	0.757	0.747

Table 9: Additional QA performance comparison using Accuracy and Exact Match (EM) metrics.

Method	Dataset	EM	F1
HippoRAG2 (NV-Embed-v2)	HotpotQA	0.627	0.755
	2Wiki	0.650	0.710
	MuSiQue	0.372	0.486
GFM-RAG (all-mpnet-v2)	HotpotQA	0.516	0.669
	2Wiki	0.698	0.777
	MuSiQue	0.302	0.404
C2RAG (all-MiniLM-L6-v2)	HotpotQA	0.553	0.728
	2Wiki	0.659	0.739
	MuSiQue	0.332	0.461
Atom-RAG (NV-Embed-v2)	HotpotQA	0.631	0.762
	2Wiki	0.710	0.781
	MuSiQue	0.373	0.488

Table 10: Additional comparison with recent graph-based RAG methods. Embedding models are shown below each method.

rectness beyond the primary LLM-based evaluation reported in the main results.

H Additional Comparison with Recent Baselines

To further position Atom-RAG against recent graph-based RAG methods, we provide an additional comparison with HippoRAG2, GFM-RAG (Luo et al., 2025b), and C2RAG (Ma et al., 2026) on three multi-hop QA benchmarks. Unlike the controlled comparison in Table 1, these results use the embedding backbone associated with each setting, as indicated in Table 10. We additionally evaluate Atom-RAG with NV-Embed-v2 to facilitate comparison with the original HippoRAG2 configuration. Atom-RAG with NV-

Embed-v2 achieves competitive or stronger performance across the three benchmarks. In particular, it improves over the NV-Embed-v2 HippoRAG2 setting on both EM and F1 across all three datasets, while also remaining competitive with GFM-RAG and C2RAG. These results provide additional evidence that the effectiveness of Atom-RAG is maintained under a stronger embedding backbone.

I Additional Atomic Fact Extraction Analysis

We additionally examine whether atomic fact extraction behavior is sensitive to the choice of extractor LLM. Specifically, we use Qwen/Qwen3.5-27B, google/gemma-4-31B, and GPT-4o-mini as extractor LLMs. Table 11 shows a qualitative comparison of atomic facts generated by the three models from the same source passage. Across this example, all three extractors preserve the major factual content of the source. Qwen3.5-27B produces mostly correct and source-supported facts but omits some role-specific details. Gemma-4-31B generates concise and well-grounded facts, while GPT-4o-mini provides more complete and self-contained statements with greater consistency in preserving relational details. This qualitative result suggests that atomic fact extraction is not restricted to a single LLM, although extraction completeness can vary with model capability.

Extractor LLM	Example Atomic Facts	Observation
Qwen3.5-27B	(1) Family Joules is the fourteenth studio album by Foghat. (2) Family Joules was released in 2003. (3) Family Joules was Foghat's first album without Dave Peverett.	Mostly correct and source-supported, but some role information is less detailed.
Gemma-4-31B	(1) Family Joules is Foghat's fourteenth studio album. (2) The album was released in 2003. (3) It was the first Foghat album without founding member Dave Peverett. (4) The album first featured Charlie Huhn and Bryan Bassett.	Concise and well-grounded; sufficiently reliable for practical indexing.
GPT-4o-mini	(1) Family Joules is the fourteenth studio album by Foghat. (2) Family Joules was released in 2003. (3) Family Joules is the first Foghat album without founding member Dave Peverett. (4) Family Joules is the first Foghat album to feature Charlie Huhn and Bryan Bassett.	More complete and self-contained, with high consistency.

Table 11: Qualitative comparison of atomic fact extraction across different extractor LLMs. Qwen3.5-27B and Gemma-4-31B correspond to Qwen/Qwen3.5-27B and google/gemma-4-31B, respectively.

<p>You are an assistant specialized in decomposing text chunks into atomic facts.</p> <p>Given a text chunk, your task is to: Decompose the input into a list of atomic facts. Each atomic fact should be a concise, indivisible unit of meaning that captures a basic proposition or event.</p> <p>Requirements:</p> <ol style="list-style-type: none"> Atomic facts must be decomposed into independent units of meaning that are explicitly supported by the chunk, ensuring that no meaningful unit is duplicated or omitted. If multiple topics appear in the chunk, separate their atomic facts clearly. Replace pronouns with specific noun references when applicable (e.g., "he" → "Einstein"). Ensure that atomic facts are presented in {language}. Each output line must follow the format: ("Atomic fact"){record_delimiter} When finished, output {completion_delimiter} <p>#####</p> <p>-Examples-</p> <p>#####</p> <p>{examples}</p> <p>#####</p> <p>-Real Data-</p> <p>#####</p> <p>User: {input_text}</p> <p>#####</p> <p>Assistant:</p>

Figure 6: Atomic fact extraction prompt used to decompose document chunks into fact-centric knowledge units.

```

---Role---
You are a query decomposition assistant.
Your task is to take a given query, break it down into atomic facts, and then generate independent sub-queries based on those atomic facts.

---Goal---
1. First, decompose the input query into a list of atomic facts.
2. From each atomic fact, generate one sub-query.
3. The final output must be in JSON format with only one top-level key: 'sub_queries'.
4. 'sub_queries' should map to a list of strings.
5. Do not include extra text, explanations, or fields outside the JSON.

---Instructions---
- Always preserve the original meaning of the query.
- Atomic facts must be concise, indivisible units of meaning.
- Replace pronouns or ambiguous references (e.g., "this director") with explicit descriptive phrases from the query so that each sub-query is self-contained.
- Sub-queries must collectively cover the full scope of the original query.
- Keep the sub-queries in the same language as the input query.

#####
-Example-
#####
{examples}

#####
-Real Data-
#####
Current Query: {query}
#####
Output:

```

Figure 7: Prompt used for reasoning-aware query decomposition and sub-query generation. The prompt decomposes an input query into atomic facts and generates corresponding independent sub-queries in a structured JSON format.

```

SYSTEM:
As an advanced reading comprehension assistant, your task is to analyze text passages and corresponding questions meticulously.
Your response start after "Thought: ", where you will methodically break down the reasoning process, illustrating how you arrive at conclusions.
Conclude with "Answer: " to present a concise, definitive response, devoid of additional elaborations.

USER:
The Last Horse
The Last Horse (Spanish:El último caballo) is a 1950 Spanish comedy film directed by Edgar Neville starring Fernando Fernán Gómez.
Southampton
The University of Southampton, which was founded in 1862 ...
Stanton Township, Champaign County, Illinois
...

Question: When was Neville A. Stanton's employer founded?
Thought: The employer of Neville A. Stanton is University of Southampton. The University of Southampton was founded in 1862.
Answer: 1862.

USER:
{retrieved_context}

Question: {question}
Thought:
Answer:

```

Figure 8: Answer generation prompt used in Atom-RAG. The prompt conditions the language model on retrieved evidence and the input query, and enforces a structured reasoning format with explicit Thought and Answer fields.

Fact-Centric Knowledge Extraction Example	
Chunk: ... Charlie Nicholas Charles "Charlie" Nicholas (born 30 December 1961 in Cowcaddens, Glasgow, Scotland) is a former professional footballer from Scotland who played as a forward. His playing career is best known for his spells at Celtic and Arsenal. He was also capped by Scottish national team, and played for them at the 1986 FIFA World Cup. ...	
Atomic Fact: ("Charlie Nicholas was born on December 30, 1961.") ("Charlie Nicholas was born in Cowcaddens, Glasgow, Scotland.") ("Charlie Nicholas is a former professional footballer from Scotland.") ("Charlie Nicholas played as a forward.") ("Charlie Nicholas played for Celtic.") ("Charlie Nicholas played for Arsenal.") ("Charlie Nicholas was capped by the Scottish national team.") ("Charlie Nicholas played at the 1986 FIFA World Cup.")	
Granularity-Aware Query Reformulation Example	
Query: A man who played in the 1986 FIFA World Cup played for what team during the 1982 Scottish League Cup Final?	
Atomic Fact: ("A man played in the 1986 FIFA World Cup.") ("The same man played for a team during the 1982 Scottish League Cup Final.")	
Sub-Query: "Who played in the 1986 FIFA World Cup?" "Which team did the man play for during the 1982 Scottish League Cup Final?"	
Evidence-Centric Context Construction Example	
Sub-Query: "Who played in the 1986 FIFA World Cup?"	Sub-Query: "Which team did the man play for during the 1982 Scottish League Cup Final?"
Extracted Atomic Facts: ("A man played in the 1986 FIFA World Cup.") ...	Extracted Atomic Facts: ("Charlie Nicholas played for Celtic.") ("The 1982 Scottish League Cup Final was contested by Celtic.") ...
Extracted Source Chunk: ... Charles "Charlie" Nicholas (born 30 December 1961 in Cowcaddens, Glasgow, Scotland) is a former professional footballer from Scotland who played as a forward. His playing career is best known for his spells at Celtic and Arsenal. He was also capped by Scottish national team, and played for them at the 1986 FIFA World Cup. ...	Extracted Source Chunk: ... The 1982 Scottish League Cup Final was played on 4 December 1982, at Hampden Park in Glasgow and was the final of the 37th Scottish League Cup competition. The final was contested by the Old Firm rivals, Celtic and Rangers. Celtic won the match 2–1 thanks to goals by Charlie Nicholas and Murdo MacLeod. Rangers goal was scored by Jim Bett. ...

Figure 9: Example execution of Atom-RAG, illustrating atomic fact extraction, query reformulation, evidence retrieval, and source-chunk recovery for a multi-hop question.

Instruction: You are an expert evaluator for question-answering systems. Your task is to determine whether the predicted answer correctly conveys the essential factual information required to answer the given question.
Input: <ul style="list-style-type: none"> • Question: {The input question} • Gold Answer(s): {One or more reference answers} • Predicted Answer: {The answer generated by the model}
Evaluation Guidelines <ul style="list-style-type: none"> • Label the prediction as CORRECT if it contains the core factual information needed to answer the question. • Differences in wording, paraphrasing, or additional explanatory context are acceptable as long as the essential fact is preserved. • Label the prediction as INCORRECT if it contradicts the gold answer or fails to include the key factual content. • When multiple gold answers are provided, matching any one of them is sufficient to be considered correct.
Output Format (JSON only) {"verdict": "CORRECT or INCORRECT", "confidence": "HIGH or MEDIUM or LOW", "reason": "Brief explanation of the judgment"}

Figure 10: LLM-as-a-judge evaluation prompt used to assess answer correctness.