IterCOMP: Reasoning-aware Adaptive Prompt Compression for Multi-hop Question Answering

Jungmin Yun[1] and **Youngbin Kim**[1, 2]

[1]Department of Artificial Intelligence, Chung-Ang University
[2]Graduate School of Advanced Imaging Sciences, Multimedia and Film, Chung-Ang University
{cocoro357, ybkim85}@cau.ac.kr

Abstract

Multi-hop question answering requires complex reasoning across multiple evidence segments, which often overwhelms retrieval-augmented generation systems with lengthy and noisy contexts, thereby undermining both efficiency and accuracy. While existing prompt compression methods attempt to address this issue, they are typically designed for single-turn queries and fail to capture interdependent reasoning steps. We propose IterCOMP, a unified, training-free prompt compression framework that incorporates multi-hop reasoning within an iterative compression loop. IterCOMP decomposes documents into evidence segments, evaluates question answerability, and generates targeted follow-up questions to iteratively integrate essential evidence, producing a compact, reasoning-oriented prompt. Experiments on MusiQue, 2WikiMultiHopQA, and HotpotQA demonstrate that IterCOMP achieves substantial improvements in Exact Match and F1 scores while reducing the token budget, outperforming existing baselines and exhibiting robustness as reasoning complexity increases.

1 Introduction

Multi-hop question answering (QA) requires reasoning over multiple pieces of evidence to derive the correct answer. Large Language Models (LLMs) have significantly advanced QA performance by enhancing complex query understanding and information integration. Nevertheless, their reliance on static pre-training data imposes inherent limitations on knowledge coverage. Retrieval-Augmented Generation (RAG) (Lewis et al., 2020) mitigates this issue by incorporating dynamic, up-to-date information from external sources, thereby enabling LLMs to generate more diverse, accurate, and contextually grounded responses (Gao et al., 2023).

However, RAG systems face challenges in both efficiency and effectiveness due to the long input sequences induced by retrieved documents. Efficiency declines as input length scales linearly with the number of retrieved documents, leading to higher inference latency and increased computational overhead (Li et al., 2025a; Xu et al., 2024). For API-based commercial LLMs, longer inputs also result in elevated operational costs (Choi et al., 2024). Effectiveness is similarly constrained, as lengthy inputs often include irrelevant content that distracts the model and hinders accurate reasoning (Shi et al., 2023). Furthermore, LLMs exhibit positional biases, such as the lost-in-the-middle phenomenon (Liu et al., 2024). These challenges are especially pronounced in multi-hop QA, which requires linking multiple pieces of evidence through sequential and interdependent reasoning steps to derive the final answer (Tang and Yang, 2024).

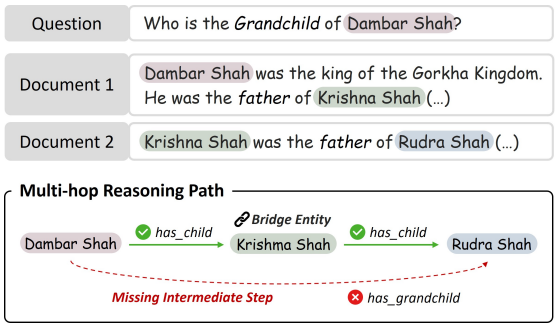

Figure 1: Example of multi-hop reasoning, where answering the question requires integrating evidence from multiple documents through implicit intermediate clues absent from the initial question.

Recently, various prompt compression techniques have been proposed to reduce contextual overhead by eliminating less salient content or condensing information into compact representations (Pan et al., 2024; Jiang et al., 2024; Mu et al., 2023). However, effective compression requires more than simply reducing input length; it fundamentally depends on selectively retaining query-

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 33827–33840

relevant information query-relevant while simultaneously removing irrelevant or distracting content (Cao et al., 2024).

In this context, query-focused compression methods enhance contextual relevance by condensing prompts to retain content pertinent to a given query (Choi et al., 2024; Liskavets et al., 2025; Hwang et al., 2025). However, their predominant reliance on a single-query paradigm, which primarily leverages surface-level query-document relevance, limits their effectiveness in complex scenarios such as multi-hop QA (Trivedi et al., 2023; Press et al., 2023; Shao et al., 2023). As illustrated in Figure 1, these tasks require sequential, interdependent reasoning across multiple documents and often depend on implicit intermediate clues absent from the initial query (Schnitzler et al., 2024; Geva et al., 2021; Trivedi et al., 2022; Ho et al., 2020b). These limitations become more pronounced when queries consist of multiple subcomponents, whose supporting evidence is dispersed across different documents (Levy et al., 2025). Consequently, single-query approaches often fail to capture inter-document dependencies or to reason effectively over linked information, leading to information loss and incomplete contextual understanding that is inadequate for synthesizing multi-source answers.

To this end, we propose IterCOMP, a unified prompt compression framework based on iterative refinement that explicitly incorporates multi-hop reasoning into the compression loop. IterCOMP strategically integrates the reasoning capabilities of LLMs into the compression process to directly address the nuanced evidential demands of complex queries. Specifically, the framework filters relevant evidence segments and employs an LLM to assess whether they are sufficient to answer the question. When the evidence is insufficient, the LLM identifies the informational gap and formulates a targeted follow-up question to bridge it. This initiates a cycle in which new evidence is evaluated against the evolving reasoning path, with only critical segments retained and irrelevant content discarded. Through progressive accumulation and distillation, IterCOMP incrementally constructs a concise yet comprehensive prompt that enables the original multi-hop question to be answered using focused, essential information. Extensive experiments demonstrate that IterCOMP substantially improves both QA performance and efficiency, highlighting the effectiveness of integrating deep reasoning into prompt compression.

2 Related Work

2.1 Prompt Compression

2.1.1 Soft Prompt Compression

Soft prompt compression encodes the original prompt into continuous vector representations (Ge et al., 2024; Cheng et al., 2024). AutoCompressor (Chevalier et al., 2023) segments long prompts into multiple parts, compresses each segment into a soft prompt representation, and concatenates them to form the final prompt. GIST (Mu et al., 2023) generates prefix-style soft prompts for instructional inputs. These methods enable parameter-efficient adaptation of pretrained models while preserving the high-level semantics of the original input. However, soft prompts are typically optimized for specific LLMs, which limits their transferability across different models (Li et al., 2025b). This constraint poses a significant challenge in API-based environments, where direct access to model internals is restricted. Consequently, any update to the underlying LLM requires retraining the soft prompts, reducing their practicality in dynamic or cross-model deployment scenarios.

2.1.2 Hard Prompt Compression

Hard prompt compression reduces prompt length by preserving essential information through extractive or abstractive methods (Li et al., 2025b; Chuang et al., 2024; Jung and Kim, 2024). This approach improves computational efficiency and enhances the quality of generated outputs by eliminating redundant or uninformative content.

Query-Agnostic Compression. Query-agnostic methods operate independently of the query, leveraging the statistical or structural properties of the prompt. Selective-Context (Li et al., 2023) discards low-information tokens based on self-information metrics. LLMLingua (Jiang et al., 2023b) removes tokens with low perplexity, while LLMLingua-2 (Pan et al., 2024) formulates compression as a token classification task using knowledge distillation. Despite their broad applicability, these approaches overlook query-specific context. As a result, compressed prompts may retain irrelevant content or omit critical information, thereby degrading retrieval effectiveness and response relevance (Cao et al., 2024).

Query-Aware Compression. Incorporating query information into the compression process is essential for retaining relevant and critical content for reasoning. LongLLMLingua (Jiang et al., 2024)

employs a coarse-to-fine strategy, first estimating document-level importance via query-conditioned perplexity, followed by refining token selection. COMPACT (Yoon et al., 2024) compresses context by jointly analyzing previously selected content and newly introduced segments. RECOMP (Xu et al., 2024) performs sentence-level compression by measuring the similarity between query and sentence embeddings to identify key content. R2C (Choi et al., 2024) encodes the query alongside each context chunk, enabling the decoder to dynamically preserve relevant information.

Despite their advantages, existing query-aware methods typically assess relevance by matching a single query against individual documents or sentences. This one-to-one paradigm is insufficient for complex multi-hop QA, where a query often comprises multiple subcomponents that must be jointly resolved using evidence distributed across documents (Tang and Yang, 2024). Such tasks require capturing inter-document dependencies and reasoning over linked information segments (Trivedi et al., 2023). However, current methods struggle to model these multi-hop relationships, thereby limiting their effectiveness in synthesizing information from multiple sources (Zhu et al., 2025).

2.2 Self-Ask Mechanisms in LLMs

Recent studies highlight the potential of LLMs to generate and respond to follow-up questions, thereby enhancing performance across various tasks. Follow-up questioning improves the coherence and informativeness of document generation (Tix, 2024) and increases user satisfaction in conversational search by supporting deeper exploration (Kim et al., 2024). A prominent approach is the self-ask method, in which LLMs generate and answer sub-questions to decompose complex queries, yielding substantial gains in compositional reasoning (Press et al., 2023). This line of work has been extended to interleave reasoning, retrieval, and self-reflection, allowing LLMs to dynamically control queries during problem solving (Jiang et al., 2023c; Yao et al., 2023; Asai et al., 2024). Domain-specific adaptations demonstrate that iterative RAG can effectively address complex clinical scenarios (Xiong et al., 2024). In broader contexts, additional work explores anticipating follow-up questions in information search (Wilcock, 2024), refining outputs through targeted questioning (Shridhar et al., 2024), and augmenting LLMs with search engines to improve factuality (Vu et al., 2024).

3 Preliminary Analysis

Our proposed approach is grounded in two key assumptions: (1) LLMs can reliably judge whether a question is answerable given a set of evidence; and (2) if the question is unanswerable, LLMs can identify the additional information required to derive the correct answer. To empirically validate these assumptions, we conduct two preliminary analyses guided by the following research questions: ***RQ1: Can LLMs determine whether the provided evidence is sufficient to answer a given question? RQ2: If the evidence is insufficient, can LLMs identify the missing information needed to derive the correct answer?***

3.1 Settings

To address these questions, we employ the MuSiQue dataset (Trivedi et al., 2022), which consists of multi-hop QA pairs requiring two to four reasoning steps. For each hop length, we randomly sample 400 QA pairs. Following prior work (Wang et al., 2025), we use `GPT-4o` to generate sub-question and sub-answer pairs based on the decomposition annotations provided in the dataset. Table 7 presents examples of the prompts and resulting generated data.

Using the original question along with the generated sub-QA pairs, we conduct experiments with several LLMs, including `LLaMA-3.1-8B-Instruct` (Dubey et al., 2024), `Mistral-7B-Instruct` (Jiang et al., 2023a), `GPT-3.5-Turbo` (Brown et al., 2020), and `GPT-4o` (Achiam et al., 2023). Each model is evaluated on two tasks: (1) determining whether the provided evidence alone is sufficient to answer the original question, and (2) identifying the additional information required when the evidence is deemed insufficient.

3.1.1 Answerability Judgment

To evaluate answerability judgment, we define two experimental conditions based on the completeness of the provided evidence:

- *Full*: The evidence includes sub-answers for all hops required to answer the original question (e.g., all three sub-answers in a 3-hop QA pair) → *answerable*.
- *Partial*: The evidence includes sub-answers only up to an intermediate hop (e.g., only the 1st and 2nd hops in a 3-hop QA pair) → *unanswerable*.

	Full				*Partial*			
	2-hop	**3-hop**	**4-hop**	***Avg.***	**2-hop**	**3-hop**	**4-hop**	***Avg.***
`Llama-3.1-8B`	86.5	58.0	47.3	63.9	71.3	93.5	96.5	87.1
`Mistral-7B`	92.0	80.3	78.5	83.6	71.8	95.5	97.6	88.3
`GPT-3.5`	68.3	63.3	58.0	63.2	88.3	96.0	96.9	93.7
`GPT-4o`	81.8	72.5	67.8	74.0	95.5	98.3	99.5	97.8

Table 1: Results on answerability judgment for information sufficiency assessment.

	2-hop		**3-hop**		**4-hop**		***Avg.***	
	BLEU	**Rel.**	**BLEU**	**Rel.**	**BLEU**	**Rel.**	**BLEU**	**Rel.**
`Llama-3.1-8B`	0.40	81.3	0.31	75.8	0.27	70.3	0.30	73.8
`Mistral-7B`	0.52	92.0	0.43	88.1	0.37	82.1	0.41	85.6
`GPT-3.5`	0.44	88.5	0.37	87.8	0.30	78.7	0.35	85.4
`GPT-4o`	0.61	97.9	0.58	91.9	0.41	87.2	0.50	90.6

Table 2: Results on missing information identification.

3.1.2 Missing Information Identification.

For missing information identification, we adopt a *partial* evidence setting. Specifically, the sub-answers up to a given hop are concatenated and provided as evidence, while the sub-question at the subsequent hop is treated as the reference missing information that the model is required to identify.

To evaluate model predictions, we employ two metrics. BLEU score measures surface-level similarity between the generated follow-up question and the reference sub-question by capturing n-gram overlap. Relatedness score (Rel.) is computed through binary classification using `GPT-4o`, assessing whether the predicted and reference questions are topically or conceptually aligned.

3.2 Results

Answerability Judgment. Table 1 presents the results of the answerability judgment task across various LLMs under two conditions: *Full*, where all required evidence (i.e., sub-answers for each reasoning step) is provided, and *Partial*, where only a subset of evidence is available.

The models exhibit contrasting trends across the two conditions. In the *Full* setting, accuracy consistently declines as the number of hops increases; for example, `Mistral-7B` drops from 92% at 2-hop to 78.5% at 4-hop. This indicates that, even when all relevant evidence is available, increasing hop length amplifies both the volume of evidence and the complexity of their interconnections. Accordingly, answerability judgment requires more than surface-level binary classification and instead demands advanced reasoning capabilities to integrate multiple evidence segments. In contrast, under the *Partial* setting, all models show consistent improvements as hop length grows. High-performing models such as `GPT-4o` maintain strong performance with minimal variance even at longer hop lengths. For a more balanced evaluation, we also report overall F1 scores: `Llama-3.1-8B` (70.51), `Mistral-7B` (84.27), `GPT-3.5-Turbo` (73.12), and `GPT-4o` (83.57).

Although these scores reveal an asymmetry in detecting answerable and unanswerable cases, both misclassification types pose distinct risks. Misclassifying an unanswerable case as answerable causes the compression process to terminate prematurely, leaving the reasoning chain with insufficient clues and thus compromising the reliability of the final answer. Conversely, misclassifying an answerable case as unanswerable triggers unnecessary iterations; while this allows the reasoning chain to incorporate new evidence, prolonged iterations may introduce irrelevant noise that degrades compression effectiveness. Given the relative degradation observed in the *Full* setting, we bound the maximum number of iterations to prevent excessive refinement. Building on this analysis, we design IterCOMP's iterative compression mechanism with an iteration budget calibrated to these tendencies, reliably using answerability as a core signal to guide evidence collection in complex multi-hop QA.

Missing Information Identification. Table 2 presents the experimental results for the missing information identification task, which evaluates the quality of follow-up questions generated by LLMs when the initial evidence is insufficient to answer the original question. The analysis shows that `GPT-4o` achieves the best performance, with an average relatedness score of 90.6% and an average BLEU score of 0.50. These results highlight `GPT-4o`'s strong capability to pinpoint missing information and generate contextually appropriate follow-up questions within multi-hop reasoning contexts. However, all models exhibit a consistent decline in both relatedness and BLEU scores as the number of hops increases. This trend reflects the growing difficulty of accurately identifying knowledge gaps and formulating effective follow-up questions as reasoning chains become longer and more complex. While LLMs clearly demonstrate the capability to generate meaningful follow-up questions, the observed performance degradation suggests that precisely locating and articulating missing information remains a challenging subtask in multi-hop QA.

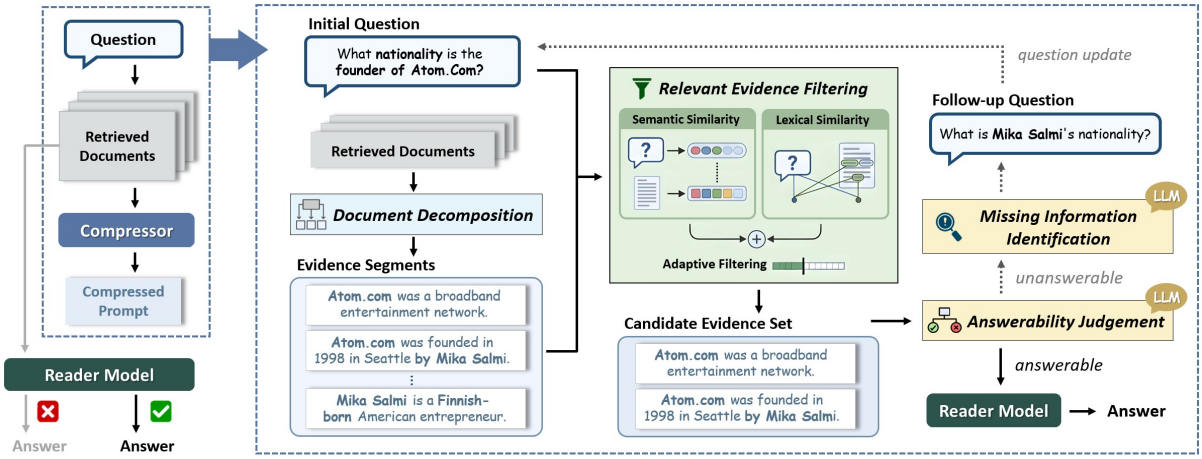

Figure 2: Overall pipeline of IterCOMP. Retrieved documents are decomposed into evidence segments and filtered via dual-aspect relevance scoring. An LLM then judges whether the candidate evidence set is sufficient to answer the question. If answerable, the evidence is finalized as the compressed prompt and passed to the reader model. If unanswerable, the LLM identifies missing information and generates a targeted follow-up question, which serves as a new query for the filtering stage to enable iterative evidence accumulation.

4 Methodology

4.1 Problem Formulation

We define the task of prompt compression as follows. Given an initial question q and a large corpus of retrieved documents $D = \{d_1, \ldots, d_N\}$, the primary objective is to synthesize a compressed prompt P_{comp}. The prompt P_{comp} consists of a concise yet information-sufficient set of evidence from D, satisfying the constraint $L(P_{\text{comp}}) \ll \sum_{i=1}^{N} L(d_i)$, where $L(\cdot)$ denotes the token length. The principal aim is to construct a P_{comp} such that a downstream reader model M can generate a high-fidelity output, $y = M(P_{\text{comp}}, q)$, while preserving essential reasoning information from corpus D.

In multi-hop QA, where synthesizing information across multiple sources is indispensable, prompt compression plays a pivotal role. Its utility extends beyond filtering irrelevant content and selecting evidence that directly addresses the initial question q. Multi-hop reasoning frequently requires resolving latent intermediate questions not explicitly stated in q. Consequently, an effective compression strategy must support progressive evidence accumulation, enabling the model to incrementally integrate and build upon relevant information as the reasoning process unfolds. This capability is fundamental to maintaining logical coherence across complex inferential chains and mitigating performance degradation due to attentional dilution or contextual drift in long inputs.

4.2 Proposed Framework

4.2.1 Document Decomposition

Each document $d_i \in D$ is segmented into a set of smaller units, termed *evidence segments* and denoted by $E_i = \{e_{i,1}, \ldots, e_{i,R_i}\}$, where R_i indicates the number of segments for d_i. Although segments can be defined at various levels of granularity, such as paragraphs or sentences, we adopt sentence-level decomposition in this work. A sentence serves as a basic unit that preserves the local semantic context of the original input while enabling effective compression (Xu et al., 2024; Choi et al., 2024). These evidence segments, along with the initial question q, are then passed to the relevant evidence filtering module.

4.2.2 Relevant Evidence Filtering

The relevant evidence filtering module systematically identifies and retains salient evidence segments by evaluating the relevance of each candidate segment $e_{i,j}$ with respect to a given question q. To establish a robust foundation for evidence selection, our approach combines semantic and lexical signals in a unified scoring scheme.

Semantic similarity captures contextual alignment between the question q and a candidate evidence segment $e_{i,j}$ to encode deeper semantic correspondences. We transform both q and $e_{i,j}$ into dense vectors representations via a text encoder $E(\cdot)$. The semantic similarity score, $S_{\text{sem}}(q, e_{i,j})$, is then computed as the inner product between the

corresponding embeddings:

$$S_{\text{sem}}(q, e_{i,j}) = E(q)^\top E(e_{i,j}), \tag{1}$$

where $E(q)$ and $E(e_{i,j})$ denote the embeddings of the question q and the evidence segment $e_{i,j}$, respectively.

While semantic similarity effectively captures broader contextual relevance, it may overlook critical lexical cues. In multi-hop settings, sensitivity to precise keyword matches improves coverage and accuracy (Zhang et al., 2025). To complement this, we incorporate a *lexical similarity* component that quantifies token-level relevance based on the importance of overlapping terms between q and $e_{i,j}$. Following M3-Embedding (Chen et al., 2024), we define the lexical importance weight of each token t in the question q as follows:

$$w_t^q = \text{ReLU}(\mathbf{w}_{\text{lex}}{}^\top E(t)), \tag{2}$$

where $\mathbf{w}_{\text{lex}}$ is a projection vector that maps a contextualized token representation to a scalar importance score, and $E(t)$ denotes the contextualized embedding of token t. An identical operation is applied to compute $w_t^{e_{i,j}}$ for tokens in the evidence segment $e_{i,j}$. The lexical similarity score is then defined as:

$$S_{\text{lex}}(q, e_{i,j}) = \sum_{t \in q \cap e_{i,j}} w_t^q \cdot w_t^{e_{i,j}}, \tag{3}$$

which computes a weighted sparse inner product over co-occurring tokens, capturing their joint lexical salience under respective contexts.

For both the encoder and the projection vector $\mathbf{w}_{\text{lex}}$, we adopt the pretrained bge-m3 model (Chen et al., 2024) without any additional fine-tuning. All parameters remain frozen throughout the entire IterCOMP pipeline, in line with our training-free design principle.

Finally, we combine the semantic and lexical signals into a single *dual-aspect relevance score*:

$$\begin{aligned} S_{\text{dual}}(q, e_{i,j}) = \lambda \cdot S_{\text{sem}}(q, e_{i,j}) \\ + (1 - \lambda) \cdot S_{\text{lex}}(q, e_{i,j}), \end{aligned} \tag{4}$$

where the hyperparameter $\lambda \in [0, 1]$ balances semantic and lexical relevance.

To prune less relevant segments, we adopt percentile-based filtering that adaptively selects evidence according to relative importance within the retrieved context. Let $\mathcal{S} = \{S_{\text{dual}}(q, e_{i,j}) \mid \forall i, j\}$ denote the multiset of relevance scores for all candidates. The filtered candidate set E_{cand} is constructed by retaining segments whose scores exceed the k-th percentile threshold over the global score distribution:

$$E_{\text{cand}} = \{e_{i,j} \mid S_{\text{dual}}(q, e_{i,j}) \geq \text{Percentile}(\mathcal{S}, k)\}. \tag{5}$$

The segments retained in E_{cand} form the basis for constructing the final compressed prompt P_{comp}. During multi-hop inference, this adaptive filtering is applied at each reasoning step, enforcing a dynamic cutoff that suppresses noise propagation and guides the model to attend to salient evidence throughout the reasoning chain.

4.2.3 Answerability Judgment & Missing Information Identification

The answerability judgment module serves as a reasoning controller, establishing a dynamic feedback loop for iterative evidence accumulation. At iteration h, the currently accumulated candidate evidence set is denoted as $E_{\text{cand}}^{(h)}$. The module employs an LLM as a binary classifier to determine whether $E_{\text{cand}}^{(h)}$ contains sufficient information to answer the original question $q^{(0)}$.

If the judgment is *answerable*, the framework triggers an *early termination*. The evidence set is finalized and used as the compressed prompt, $P_{\text{comp}} = E_{\text{cand}}^{(h)}$, and passed to the reader model M to generate the final answer. This principled stopping condition prevents unnecessary iterations once sufficient information has been gathered. Conversely, if the judgment is *unanswerable*, the framework proceeds with *iterative refinement*. The LLM explicitly identifies the missing information that prevents a complete answer to the original question $q^{(0)}$. To bridge the gap, it formulates a targeted follow-up question $q^{(h)}$ to retrieve complementary evidence. Relevant evidence filtering is subsequently re-applied with respect to $q^{(h)}$, producing an updated set $E_{\text{cand}}^{(h+1)}$ progressively accumulated across iterations. This cycle repeats until the sufficiency condition is satisfied or a predefined maximum hop limit is reached.

	MuSiQue			2WikiMultiHopQA			HotpotQA		
	EM	**F1**	**Ratio**	**EM**	**F1**	**Ratio**	**EM**	**F1**	**Ratio**
Oracle	21.53	37.51	0.15	36.6	51.3	0.42	41.40	58.95	0.17
Raw Documents	9.69	19.92	(1.0)	24.06	36.75	(1.0)	31.60	43.63	(1.0)
Selective-Context	3.02	9.96	0.15	18.87	29.30	0.36	17.74	28.82	0.17
RECOMP (extractive)	8.32	16.76	0.16	22.26	33.89	0.45	28.09	40.78	0.19
LLMLingua	2.61	8.96	0.15	18.02	26.58	0.43	18.54	28.66	0.21
LLMLingua-2	6.62	14.72	0.17	18.70	33.35	0.46	24.01	37.76	0.19
LongLLMLingua	10.95	19.66	0.19	23.19	34.83	0.50	31.68	46.49	0.26
R2C	12.29	22.44	0.16	19.41	32.09	0.36	33.63	47.80	0.22
IterCOMP (ours)	**16.67**	**27.36**	0.14	**27.39**	**39.69**	0.37	**37.65**	**51.78**	0.19

Table 3: Experimental results on three multi-hop QA benchmark datasets. **Ratio** denotes the compression ratio, defined as the proportion of tokens in the compressed prompt relative to the those in the raw documents. The best performance is highlighted in **bold**, and the second-best is highlighted with an underline. The full experimental results are presented in Appendix B.3.

5 Experiments

5.1 Experimental Setup

We evaluate our proposed prompt compression method, IterCOMP, in a zero-shot setting on three widely used multi-hop QA datasets: MuSiQue (Trivedi et al., 2022), 2WikiMultiHopQA (Ho et al., 2020a), and HotpotQA (Yang et al., 2018). All experiments are conducted on dev sets of these datasets. For MuSiQue, we specifically use the `musique_ans_v1.0_dev` subset, which contains only answerable questions. Further detailed statistics and descriptions of the datasets are provided in the Appendix B.1.

We compare IterCOMP against several baselines. To ensure a fair and meaningful comparison, we focus our evaluation on hard prompt compression and extractive compression strategies, which are most relevant to our approach. These include representative prompt compression baselines such as LLMLingua (Jiang et al., 2023b), LongLLMLingua (Jiang et al., 2024), LLMLingua-2 (Pan et al., 2024), RECOMP (Xu et al., 2024), Selective-Context (Li et al., 2023), and R2C (Choi et al., 2024). We additionally report the performance of *Oracle*, which serves as an upper bound for document-level compression by providing the reader model only with the gold supporting documents. In contrast, *Raw Document* concatenates all documents in datasets without any filtering, representing a no-compression scenario. Detailed explanations of these baselines are available in Appendix B.2.

We employ `LLaMA-3-8B` (Dubey et al., 2024) as the reader model across all evaluated methods and baselines. For IterCOMP, we bound the maximum number of iterations at 5 and set the hyperparameter λ, which balances the dual-aspect relevance score, to 0.6. The percentile threshold k for relevance filtering is empirically determined and aligned with the *Oracle* compression setting for fair comparison: $k = 90$ for MuSiQue and HotpotQA, and $k = 85$ for 2WikiMultiHopQA.

5.2 Experimental Results

Main Results. To evaluate the effectiveness of our proposed IterCOMP, we employ Exact Match (EM) and F1 scores as metrics for QA performance, alongside the compression ratio (Ratio) to measure efficiency gains. As shown in Table 3, IterCOMP consistently achieves the highest EM and F1 scores across all benchmark datasets, significantly outperforming the baselines. Notably, it yields substantial improvements over the *Raw Documents* setting, in which no compression is applied. For instance, on the MuSiQue dataset, the F1 score increases from 19.92 to 27.36, while on HotpotQA it rises from 43.63 to 51.78. These results demonstrate that our proposed adaptive compression framework enhances downstream QA performance while reducing input length (e.g., a 7× reduction on MuSiQue) by effectively filtering out irrelevant content from lengthy raw documents.

Compared to existing compression methods, IterCOMP demonstrates notable performance gains. On MuSiQue, it attains an F1 score of 27.36, sur-

	EM	F1	# Iteration	# Tokens
2-hop	19.33	30.19	1.98	228
3-hop	14.87	26.16	2.45	348
4-hop	11.60	20.72	3.08	403

Table 4: Experimental results across different hop lengths. **# Iterations** indicates the average number of iteration loops per question, and **# Tokens** denotes the token lengths of the compressed prompt.

passing the second-best method, R2C, by 4.92 points. Importantly, these results are achieved in a training-free manner, underscoring the generalizability and practicality of our approach. The proposed framework is designed to bridge missing-information gaps in multi-hop QA reasoning. This iterative refinement design dynamically retains only the most relevant information and ensures high performance even under strict input length constraints.

To further contextualize the results, we compare performance against the *Oracle* setting, which serves as an upper bound based on ideal document selection. On HotpotQA, the performance gap between *Raw Documents* and *Oracle* is 15.32 points. IterCOMP attains an F1 score of 51.79, effectively closing 53.2% of this gap. These results highlight the effectiveness of our framework in identifying and retaining compact yet highly informative evidence that supports complex multi-hop reasoning.

Reasoning Complexity. To assess the robustness of IterCOMP under varying reasoning complexity, we evaluate its performance on the MuSiQue dataset with respect to hop length. As shown in Table 4, performance declines as reasoning paths become longer. Specifically, the F1 score drops from 30.19 at 2-hop to 20.72 at 4-hop. This decline trend reflects the inherent difficulty of maintaining a coherent evidence chain and further indicates that increasing reasoning complexity imposes additional challenges on the reader model.

The average number of iterations also increases with hop length, rising from 1.98 for 2-hop to 3.08 for 4-hop, indicating that IterCOMP dynamically allocates additional refinement steps when confronted with more complex questions. Similarly, the length of the compressed prompt expands from 228 to 403 tokens, reflecting that the framework adaptively retains a larger evidence base to support longer reasoning chains. Overall, rather than operating as a static filter, IterCOMP adjusts

	EM	F1	Ratio
-	**16.67**	**27.36**	**0.14**
(without)			
Relevant Evidence Filtering	9.69	19.92	(1.0)
Answerability Judgment	11.67	23.63	0.21
Iterative Refinement	8.50	17.39	0.06
Semantic Similarity Score	15.91	27.03	0.21
Lexical Similarity Score	14.57	25.19	0.27

Table 5: Ablation study of IterCOMP, evaluating the role of each component and comparing different similarity measures for evidence filtering.

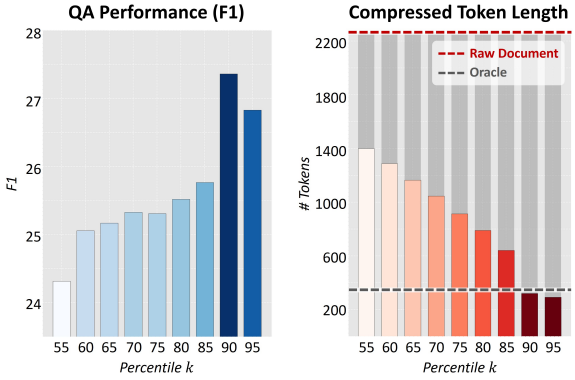

Figure 3: Comparison of *(left)* QA performance and *(right)* compressed token length across different percentile values k.

both its reasoning depth and evidence size in accordance with task complexity, thereby facilitating more effective handling of challenging multi-hop reasoning scenarios.

5.3 Ablation Study & Analysis

As shown in Table 5, we conduct an ablation study on the MuSiQue dataset to assess the contribution of each component in IterCOMP. Removing *Iterative Refinement* leads to the largest performance drop, confirming the necessity of evidence accumulation in multi-hop reasoning. Excluding *Relevant Evidence Filtering* (no-compression) also results in poor performance (19.92 F1), underscoring the importance of discarding irrelevant context. The absence of the *Answerability Judgment* module degrades performance to 23.63 F1 and produces less compact prompts, validating the role of early stopping for both accuracy and efficiency. For the similarity measure, relying solely on semantic or lexical signals is suboptimal, whereas their combination achieves the best performance, demonstrating the synergistic complementarity of semantic and lexical cues.

	Cost Reduction (%)	Speedup
GPT-3.5-Turbo	79.4	1.13×
GPT-4o	75.5	1.19×
Gemini-2.5-Flash	76.5	1.40×
Gemini-2.5-Pro	79.4	1.87×
Claude-4.5	76.0	2.01×

Table 6: Efficiency analysis with API-based LLMs. The reported cost and latency correspond to the final QA step performed by the reader LLM with the compressed prompt.

We evaluate different percentile values k for relevant evidence filtering on the MuSiQue dataset, as shown in Figure 3. *(Left)* The F1 score increases steadily with larger k, peaking at 27.36 when $k = 90$. This indicates that a stricter relevance threshold effectively isolates salient information and reduces noise, thereby enhancing the reader model's reasoning. However, performance drops at $k = 95$, suggesting that an overly stringent cutoff discards supplementary evidence necessary for completing the reasoning chain. *(Right)* As k increases, compressed prompts become shorter due to more aggressive filtering. Notably, when $k \geq 90$, the compressed prompt is even more concise than that of the *Oracle*.

To assess the practical utility and economic viability of our method, we analyze 500 random instances from the MuSiQue dataset using prominent API-based LLMs. As shown in Table 6, applying our framework for prompt compression substantially improves inference efficiency. Across all models, compression reduces costs by 75.5%–79.4%, making multi-hop QA with large-scale commercial LLMs considerably more affordable. In addition, compressed prompts consistently accelerate inference, highlighting prompt compression as a practical enabler of scalable, time-efficient, and cost-effective deployment of advanced complex reasoning in commercial LLMs.

6 Conclusion

This paper introduces IterCOMP, a unified prompt compression framework for multi-hop QA. IterCOMP embeds multi-step reasoning into the compression loop through iterative refinement, combining answerability judgment with targeted follow-up question generation. This process progressively constructs a concise, information-rich prompt by retaining only the evidence essential for complex reasoning. Extensive experiments on three multi-hop QA benchmarks demonstrate that IterCOMP significantly enhances QA performance while reducing token usage. Its training-free and model-agnostic design ensures broad applicability, including integration with commercial black-box LLMs for which fine-tuning is infeasible. By addressing both the performance degradation in long contexts and the economic burden of large-scale inference, IterCOMP offers a robust and scalable solution for complex multi-document LLM applications.

Limitations

While IterCOMP demonstrates improvements in multi-hop QA, several aspects remain open for refinement. First, its effectiveness depends on the reasoning capability of the underlying LLM for answerability judgment and missing information detection, introducing the risk of error propagation such as premature termination or accumulation of irrelevant evidence. Second, the iterative design incurs additional compression overhead compared to single-pass methods, which could be alleviated by lightweight controllers or parallel filtering. Third, performance is sensitive to hyperparameters such as the relevance percentile and iteration limit; adaptive mechanisms that dynamically adjust these settings based on question complexity or intermediate retrieval quality could further enhance robustness. Fourth, our lexical-semantic scoring module adopts a relatively simple combination of signals, and integrating more sophisticated scoring schemes may yield additional performance gains. Finally, our evaluations focus on general-domain multi-hop QA benchmarks to ensure fair comparison with existing baselines; although IterCOMP is modular and domain-agnostic by design, validating its generalization to diverse reasoning types and non-Wikipedia domains remains an important direction for future work.

Acknowledgments

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) [RS-2021-II211341, Artificial Intelligence Graduate School Program (Chung-Ang University)] and by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2025-00556246).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*.

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. *Preprint*, arXiv:2005.14165.

Zhiwei Cao, Qian Cao, Yu Lu, Ningxin Peng, Luyang Huang, Shanbo Cheng, and Jinsong Su. 2024. Retaining key information under high compression ratios: Query-guided compressor for LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12685–12695, Bangkok, Thailand. Association for Computational Linguistics.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*, Bangkok, Thailand. Association for Computational Linguistics.

Xin Cheng, Xun Wang, Xingxing Zhang, Tao Ge, Si-Qing Chen, Furu Wei, Huishuai Zhang, and Dongyan Zhao. 2024. xrag: Extreme context compression for retrieval-augmented generation with one token. *Advances in Neural Information Processing Systems*, 37:109487–109516.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. 2023. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3829–3846, Singapore. Association for Computational Linguistics.

Eunseong Choi, Sunkyung Lee, Minjin Choi, Jun Park, and Jongwuk Lee. 2024. From reading to compressing: Exploring the multi-document reader for prompt compression. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 14734–14754, Miami, Florida, USA. Association for Computational Linguistics.

Yu-Neng Chuang, Tianwei Xing, Chia-Yuan Chang, Zirui Liu, Xun Chen, and Xia Hu. 2024. Learning to compress prompt in natural language formats. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7756–7767, Mexico City, Mexico. Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, Haofen Wang, and 1 others. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1):32.

Tao Ge, Hu Jing, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. 2024. In-context autoencoder for context compression in a large language model. In *The Twelfth International Conference on Learning Representations*.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020a. 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, Barcelona, Spain (Online). International Committee on Computational Linguistics.

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

Taeho Hwang, Sukmin Cho, Soyeong Jeong, Hoyun Song, SeungYoon Han, and Jong C. Park. 2025. EXIT: Context-aware extractive compression for enhancing retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 4895–4924, Vienna, Austria. Association for Computational Linguistics.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023a. Mistral 7b. *Preprint*, arXiv:2310.06825.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023b. LLMLingua: Compressing prompts for accelerated inference of large lan-

guage models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13358–13376, Singapore. Association for Computational Linguistics.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. LongLLMLingua: Accelerating and enhancing LLMs in long context scenarios via prompt compression. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677, Bangkok, Thailand. Association for Computational Linguistics.

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023c. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992, Singapore. Association for Computational Linguistics.

Hoyoun Jung and Kyung-Joong Kim. 2024. Discrete prompt compression with reinforcement learning. *IEEE Access*, 12:72578–72587.

Hyunwoo Kim, Yoonseo Choi, Taehyun Yang, Honggu Lee, Chaneon Park, Yongju Lee, Jin Young Kim, and Juho Kim. 2024. Using llms to investigate correlations of conversational follow-up queries with user satisfaction. *arXiv preprint arXiv:2407.13166*.

Shahar Levy, Nir Mazor, Lihi Shalmon, Michael Hassid, and Gabriel Stanovsky. 2025. More documents, same length: Isolating the challenge of multiple documents in RAG. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 19539–19547, Suzhou, China. Association for Computational Linguistics.

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

Yuankai Li, Jia-Chen Gu, Di Wu, Kai-Wei Chang, and Nanyun Peng. 2025a. BRIEF: Bridging retrieval and inference for multi-hop reasoning via compression. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 5449–5470, Albuquerque, New Mexico. Association for Computational Linguistics.

Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. 2023. Compressing context to enhance inference efficiency of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6342–6353, Singapore. Association for Computational Linguistics.

Zongqian Li, Yinhong Liu, Yixuan Su, and Nigel Collier. 2025b. Prompt compression for large language models: A survey. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7182–7195, Albuquerque, New Mexico. Association for Computational Linguistics.

Barys Liskavets, Maxim Ushakov, Shuvendu Roy, Mark Klibanov, Ali Etemad, and Shane K Luke. 2025. Prompt compression with context-aware sentence encoding for fast and improved llm inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24595–24604.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Jesse Mu, Xiang Li, and Noah Goodman. 2023. Learning to compress prompts with gist tokens. *Advances in Neural Information Processing Systems*, 36:19327–19352.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. 2024. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 963–981, Bangkok, Thailand. Association for Computational Linguistics.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah 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, Singapore. Association for Computational Linguistics.

Julian Schnitzler, Xanh Ho, Jiahao Huang, Florian Boudin, Saku Sugawara, and Akiko Aizawa. 2024. Morehopqa: More than multi-hop reasoning. *arXiv preprint arXiv:2406.13397*.

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, Singapore. Association for Computational Linguistics.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR.

Kumar Shridhar, Koustuv Sinha, Andrew Cohen, Tianlu Wang, Ping Yu, Ramakanth Pasunuru, Mrinmaya Sachan, Jason Weston, and Asli Celikyilmaz. 2024. The ART of LLM refinement: Ask, refine, and trust.

In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5872–5883, Mexico City, Mexico. Association for Computational Linguistics.

Yixuan Tang and Yi Yang. 2024. Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries. *arXiv preprint arXiv:2401.15391*.

Bernadette J Tix. 2024. Follow-up questions improve documents generated by large language models. *arXiv preprint arXiv:2407.12017*.

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, Toronto, Canada. Association for Computational Linguistics.

Tu Vu, Mohit Iyyer, Xuezhi Wang, Noah Constant, Jerry Wei, Jason Wei, Chris Tar, Yun-Hsuan Sung, Denny Zhou, Quoc Le, and Thang Luong. 2024. Freshllms: Refreshing large language models with search engine augmentation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 13697–13720, Bangkok, Thailand. Association for Computational Linguistics.

Keheng Wang, Feiyu Duan, Peiguang Li, Sirui Wang, and Xunliang Cai. 2025. LLMs know what they need: Leveraging a missing information guided framework to empower retrieval-augmented generation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 2379–2400, Abu Dhabi, UAE. Association for Computational Linguistics.

Graham Wilcock. 2024. Anticipating follow-up questions in exploratory information search. In *Proceedings of the 25th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 103–109, Kyoto, Japan. Association for Computational Linguistics.

Guangzhi Xiong, Qiao Jin, Xiao Wang, Minjia Zhang, Zhiyong Lu, and Aidong Zhang. 2024. Improving retrieval-augmented generation in medicine with iterative follow-up questions. In *Biocomputing 2025: Proceedings of the Pacific Symposium*, pages 199–214. World Scientific.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2024. RECOMP: Improving retrieval-augmented LMs with context compression and selective augmentation. In *The Twelfth International Conference on Learning Representations*.

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, Brussels, Belgium. Association for Computational Linguistics.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. 2024. CompAct: Compressing retrieved documents actively for question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21424–21439, Miami, Florida, USA. Association for Computational Linguistics.

Nan Zhang, Prafulla Kumar Choubey, Alexander Fabbri, Gabriel Bernadett-Shapiro, Rui Zhang, Prasenjit Mitra, Caiming Xiong, and Chien-Sheng Wu. 2025. Sirerag: Indexing similar and related information for multihop reasoning. In *The Thirteenth International Conference on Learning Representations*.

Rongzhi Zhu, Xiangyu Liu, Zequn Sun, Yiwei Wang, and Wei Hu. 2025. Mitigating lost-in-retrieval problems in retrieval augmented multi-hop question answering. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22362–22375, Vienna, Austria. Association for Computational Linguistics.

A Prompts

Prompt for Sub-QA Pair Generation

Given a question, its final answer, and evidences, please generate a sub-question and sub-answer for each evidence. Each pair must be reflect the content of its corresponding evidence.

Question: When was the creator of The Painter's Studio born?
Answer: 10 June 1819
Evidence: ["question": "The Painter's Studio » creator", "answer": "Gustave Courbet", "question": "The date of birth of #1 is?", "answer": "10 June 1819"]

(generated sub-QA example)
Sub-question 1: Who is the creator of The Painter's Studio?
Sub-answer 1: The creator of The Painter's Studio is Gustave Courbet.
Sub-question 2: What is the date of birth of Gustave Courbet?
Sub-answer 2: The date of birth of Gustave Courbet is 10 June 1819.

Answerability Judgment & Missing Information Identification

Given the following Question and Information, your task is to determine whether the Information alone is sufficient to answer the Question.

If the Information provides a clear and direct answer, of if answer to the Question can be logically derived by combining multiple statements within the Information, then output: "answerable".

If the Information lacks the necessary content to fully answer the Question, then output: "unanswerable", and generate a follow-up question that identifies the most specific and essential piece of missing information required to answer the Question.

Your response must strictly follow this format: ["answer": "answerable or unanswerable", "follow_up_question": "a specific and detailed question that would help retrieve the missing information"]

Do not include any explanation or reasoning outside of this format.

Table 7: Prompts and generated examples used for sub-QA pair generation (top) and answerability judgment with missing information identification (bottom).

B Implementation Details

B.1 Datasets

- **MuSiQue** (Trivedi et al., 2022) consists of questions constructed by combining multiple single-hop queries, requiring 2 to 4 reasoning hops, and is specifically designed to prevent answer derivation from superficial clues.
- **2WikiMultiHopQA** (Ho et al., 2020a) is a Wikipedia-based dataset featuring questions that require up to five reasoning steps, with each question annotated with its explicit reasoning path and supporting evidence.
- **HotpotQA** (Yang et al., 2018) requires models to retrieve and synthesize evidence from multiple documents, containing explainable questions explicitly designed to encourage transparent and interpretable reasoning paths.

Table 8 presents the statistics of the benchmark multi-hop QA datasets used in our experiments. We employ three widely used standard datasets: MusiQue, 2WikiMultiHopQA, and HotpotQA. For each dataset, we summarize the number of instances in the train, dev, and test splits.

	Train	Dev	Test	# Context
MuSiQue	39876	4834	4918	20
(MuSiQue-Ans)	(19938)	2417	(2459)	(20)
2WikiMultiHopQA	167454	12576	12576	10
HotpotQA	90447	7405	-	10

Table 8: Statistics of multi-hop QA datasets used in our experiments.

B.2 Baselines

To ensure reproducibility, all experiments were conducted using officially released codebases and publicly available models.

- For **RECOMP** (Xu et al., 2024), we implement an extractive compressor based on a dual-encoder model that identifies and selects relevant sentences, representing a sentence-level extractive strategy.
- **Selective-Context** (Li et al., 2023) is a token-level compression approach that removes low-information lexical units using a compact language model.

	MuSiQue			2WikiMultiHopQA			HotpotQA		
	EM	**F1**	**Ratio**	**EM**	**F1**	**Ratio**	**EM**	**F1**	**Ratio**
Oracle	21.53	37.51	0.15	36.6	51.3	0.42	41.40	58.95	0.17
Raw Documents	9.69	19.92	(1.0)	24.06	36.75	(1.0)	31.60	43.63	(1.0)
Selective-Context	3.02	9.96	0.15	18.87	29.30	0.36	17.74	28.82	0.17
RECOMP (extractive)	8.32	16.76	0.16	22.26	33.89	0.45	28.09	40.78	0.19
RECOMP (abstractive)	4.18	11.57	0.03	13.34	33.08	0.07	21.74	38.44	0.05
LLMLingua	2.61	8.96	0.15	18.02	26.58	0.43	18.54	28.66	0.21
LLMLingua-2	6.62	14.72	0.17	18.70	33.35	0.46	24.01	37.76	0.19
LongLLMLingua	10.95	19.66	0.19	23.19	34.83	0.50	31.68	46.49	0.26
CompAct	8.27	18.77	0.07	13.32	33.1	0.16	26.12	42.98	0.12
R2C	12.29	22.44	0.16	19.41	32.09	0.36	33.63	47.80	0.22
IterCOMP (ours)	**16.67**	**27.36**	0.14	**27.39**	**39.69**	0.37	**37.65**	**51.78**	0.19

Table 9: Full experimental results on three multi-hop QA benchmarks, including abstractive RECOMP and CompAct.

- **LLMLingua** (Jiang et al., 2023b) leverages a smaller language model to eliminate low-perplexity tokens from the original prompt to satisfy a predefined compression ratio.
- **LLMLingua-2** (Pan et al., 2024) is an extension of LLMLingua that incorporates a refined budget controller and data distillation mechanisms to more effectively preserve essential information.
- **LongLLMLingua** (Jiang et al., 2024) extends LLMLingua, introducing a question-aware, coarse-to-fine compression strategy to retain critical information from lengthy documents.
- **R2C** (Choi et al., 2024) compresses prompts by leveraging cross-attention scores from a Fusion-in-Decoder (FiD) model to score the relevance of each chunk and sentence, retaining only those identified as most important.

B.3 Additional Experiments Results

RECOMP (Xu et al., 2024) also supports abstractive compression, which generates summaries by synthesizing information across retrieved documents. Moreover, CompAct (Yoon et al., 2024) compresses retrieved documents by jointly analyzing previously selected content alongside newly introduced segments, enabling context-aware compression across the document set.

However, a fair comparison with these methods is challenging due to inconsistencies in length control. While neither our method nor most baselines enforce strict token-level constraints, both abstractive RECOMP and CompAct exhibit substantially larger deviations from the target token budget. These methods rely on free-form generation without an explicit length-control mechanism and fail to reliably regulate the length of the compressed prompt. Consequently, we exclude these methods from the main comparison and instead report their full results in Table 9, explicitly noting the fairness limitations arising from these inconsistent compression budgets. As shown in Table 9, IterCOMP still consistently outperforms all baselines, including abstractive RECOMP and CompAct, across the three benchmarks.