COMPACT: Compressing Retrieved Documents Actively for Question Answering

Anonymous ACL submission

Abstract

Retrieval-augmented generation supports language models to strengthen their factual groundings by providing external context. However, language models often face challenges in locating and integrating extensive information, diminishing their effectiveness in solving complex questions. Query-focused compression tackles this issue by filtering out information irrelevant to the query, but current methods still struggle in realistic scenarios where crucial information may not be located with a single-step approach. To overcome this limitation, we introduce **COMPACT**, a novel framework that employs an active strategy to condense extensive documents without losing key information. COMPACT flexibly operates as a cost-efficient plug-in module with any off-the-shelf retriever or reader model, achieving extremely high compression rates (44x). Our experiments demonstrate that COMPACT brings significant improvements in both compression rate and QA performance on multi-hop question-answering datasets.

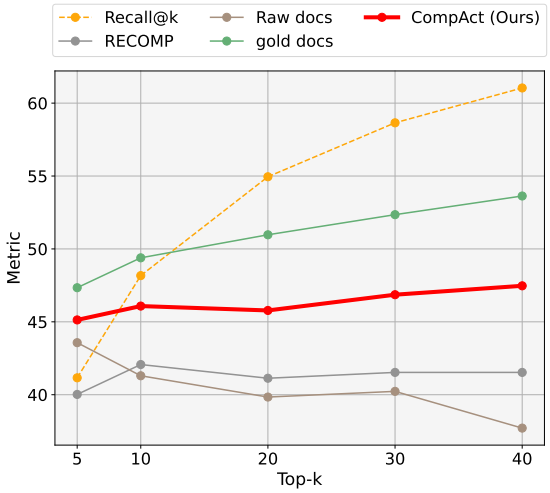

Figure 1: Performance of HotpotQA with different top-k documents. We set the reader as LLaMA3-8B. Our **COMPACT** framework demonstrates solid performance improvements that align with those of gold documents. This highlights that COMPACT effectively leverages the benefits of increased top-k, unlike other methods which struggle to maintain their performance due to increased noisy context.

1 Introduction

Retrieval-augmented generation empowers language models to solidify their factual groundings, presenting relevant contexts to answer questions (Khandelwal et al., 2019; Lewis et al., 2020; Karpukhin et al., 2020a; Izacard et al., 2023). While this approach extends the knowledge scope of language models beyond their inherent capabilities, it also introduces a number of challenges when it comes to handling long contexts (Li et al., 2024; An et al., 2024; Qian et al., 2024). First, models often struggle to find key information from these extensive contexts, which diminishes their abilities to reference documents (Liu et al., 2024). Also, models often fail to integrate information across multiple documents, which is a natural situation in real-world scenarios (Cheng et al., 2024). To this end, there is a growing need for methods that can assist models with handling long contexts.

One way to overcome these challenges is by compressing contexts into more compact forms (Li et al., 2023; Pan et al., 2024). The main goal of compression is to reduce the amount of tokens from the original text without losing too much information. As it focuses on retaining all the crucial contexts from the source documents, the compressed output can be applied to various tasks without compromising the integrity of the information.

However, simply compressing retrieved contexts can be suboptimal for question-answering (QA) tasks (Joshi et al., 2017; Kwiatkowski et al., 2019), where important details may be filtered out during the compression process (Li et al., 2023). Maintaining redundant information without compres-

sion can also harm performance, as they may serve as distractors that can induce models to generate incorrect responses. To handle these limitations, query-focused compression emerges as an effective approach in QA. This approach reduces the context length by focusing on information relevant to the question (Xu et al., 2024; Cao et al., 2024).

However, existing query-focused compressors (Jiang et al., 2023c; Xu et al., 2024) still struggle to take advantage of information located behind lengthy contexts, leaving out potential opportunities for reader models to improve their answers. Figure 1 highlights the difficulty of utilizing retrieved documents of extensive lengths. The increase in retrieval recall parallel to the number of retrieved documents shows that even lower-ranked documents may still include valuable information. Simply using more documents can result in a significant amount of noisy context, making it challenging for language models to effectively leverage additional information.

Furthermore, existing methods lack the ability to integrate information across multiple documents, which is required in real-world scenarios (Gutiérrez et al., 2024). Figure 2 depicts an example: the question is *"What 'Virtual Choir'-noted conductor has created works for the Austin-based ensemble Conspirare?"*. To answer this, not only do we need to retrieve information implied within the question (*"conductors worked for the Austin-based ensemble Conspirare"*), we should also holistically connect and synthesize information across multiple documents (*"'Virtual Choir'-noted conductor"*). In other words, the quality of answers hinges on the ability of models to dynamically integrate information across multiple documents, which is an underexplored area in compression.

To this end, we propose **COMPACT**, a novel framework that can address these challenges by using an active strategy to compress extensive documents and retain crucial information. This approach has two key components: active compression and early termination. During compression, the model actively encapsulates input documents by jointly analyzing previously compressed contexts with newly provided segments. This ensures that only the most relevant information to the question is preserved at each step, creating a dense and compact context. At each step, the model then decides whether to terminate the compression process. This decision is made based on the relevance and completeness of the information gathered to answer the query.

Our approach offers two distinct advantages. First, it effectively captures essential context from long documents by incorporating segments along with the previously compressed context. This is crucial for complex QA tasks that require in-depth reasoning and synthesis of information such as multi-hop QA. Second, it condenses large volumes of documents with a high compression rate, but without missing essential information. We conduct experiments on five question-answering datasets to evaluate our COMPACT framework. The results demonstrate that our framework brings significant improvement in compression rate and end-QA performance in several multi-document benchmarks. This represents the effectiveness of our compression method, as it preserves necessary context without losing critical information.

Our contributions are as follows: (1) We propose **COMPACT**, a novel framework that employs an active strategy for compressing extensive documents. Our framework dynamically filters and preserves relevant information by jointly considering previously compressed contexts with newly provided segments. (2) We address the limitations of existing compression methods by ensuring the integration of information across multiple documents. (3) Our approach effectively manages the challenges associated with handling long contexts, particularly in complex QA tasks that require in-depth reasoning and synthesis of information. (4) Our framework achieves a high compression rate (44x) which demonstrates its cost-efficiency, especially when collaborating with API calls such as GPT-3.5-turbo. (5) We demonstrate the effectiveness of our COMPACT framework through comprehensive experiments on five question-answering QA benchmarks.

2 Preliminaries

2.1 Multi-Document Question Answering

Multi-document (or multi-hop) question answering (QA) involves the task of answering questions that require gathering information from multiple documents. (Yang et al., 2018; Ho et al., 2020b; Chen et al., 2020; Trivedi et al., 2022; Mavi et al., 2022) This is more complex than single-document QA, since models must find and combine information from different sources. However, limited context windows hinder performance, in spite of the need for models to reference multiple sources.

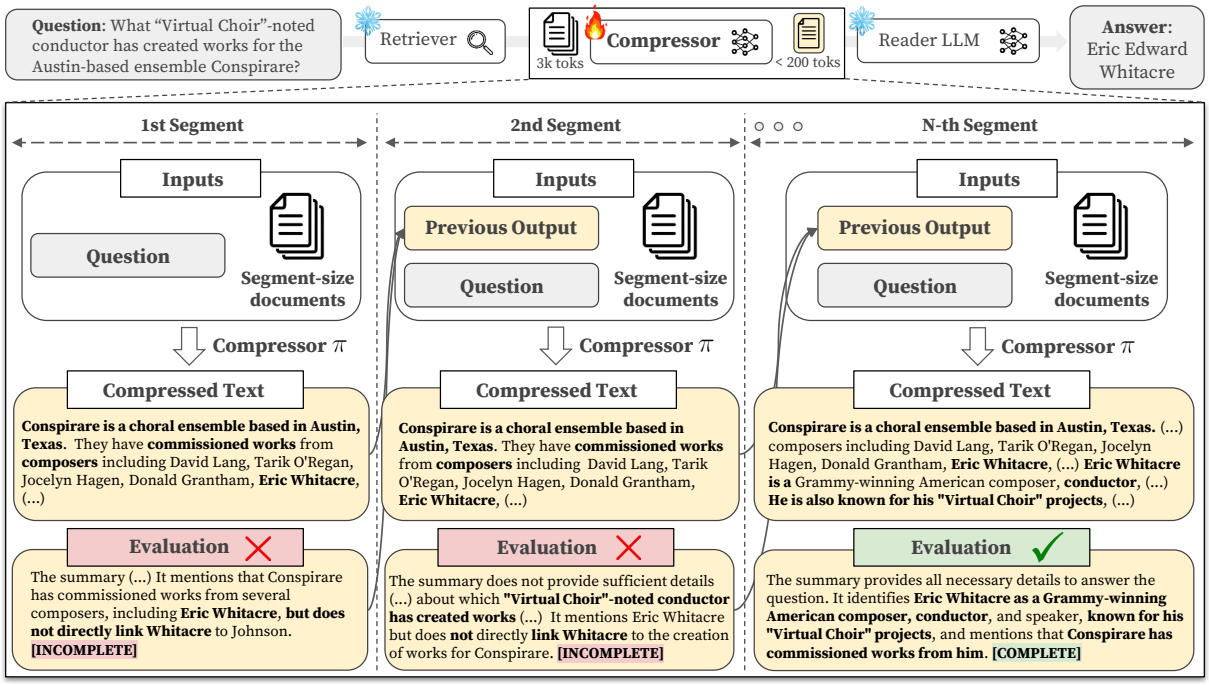

Figure 2: Overall COMPACT framework as a plug-in module between the retriever and the reader LLM. After splitting retrieved documents into segments, our model sequentially compresses these segments into compacted contexts. By checking the termination condition at each step, we actively incorporate the information of the newly provided segment while preserving essential backgrounds in compressed contexts. If the segments do not offer complete information to answer the question (1st, 2nd segments), the model continues to the next step to acquire new information. Once all supporting clues are fully captured (N-th segment), the iteration ends.

2.2 Compression

To alleviate the cost of inference, several studies have proposed compression methods. Mu et al. presents a compression method, called gisting, which allows models to compress prompts into shorter transformer activations. Ge et al. (2024) proposes training objectives related to compression that enable language models to learn to restore contexts. Several works have focused on compressing long context inputs. For example, Chevalier et al. (2023) progressively compress long documents into intermediate summary vectors. Li et al. (2023) and Jiang et al. (2023b) utilize conditional probabilities of LLMs to assess the importance of information. Concurrent with our work, Zhang et al. (2024) have developed an iterative framework using Chain-of-Agents to enable information aggregation and context reasoning over long-context tasks. However, our work focuses on addressing a crucial aspect: capturing pivotal information between segments in retrieved documents while compressing contexts.

2.3 Task Formulation

In retrieval-augmented generation, a model M predicts an output y conditioned on an input x and k retrieved passages $D_k = \{d_1, ..., d_k\}_{i=1}^{k}$. For the task of question answering, the input x typically consists of a question q with an instruction I. Thus, M generates an answer y based on x and the retrieved documents D_k as follows: $M(y|x, D_k)$.

To mitigate the costs of M caused by processing a large number of tokens, several approaches have been recently proposed to compress the documents into a shorter context (Wang et al., 2023; Xu et al., 2024). Building on these approaches, our goal is described as follows:

$$arg\max_{\pi} P_M(y \mid C_\pi, x)$$

and C_π is defined as:

$$C_\pi = \pi(q, D_k) \quad \text{with} \quad l(C_\pi) \ll l(D_k)$$

where l represents the number of tokens and π is a function that compresses documents D_k into a shorter context C_π based on the question q. It is important to note that we do not aim to optimize the model M or the retriever. Instead, our primary focus is on compressing the provided contexts into a concise format, ensuring the essential information is retained to answer the question.

Dataset	[COMPLETE]		[INCOMPLETE]		Total
	Previous Output (O)	Previous Output (X)	Previous Output (O)	Previous Output (X)	
HotpotQA	7.2K	7.2K	7.2K	7.2K	28.8K

Table 1: Statistics of our Dataset Construction.

3 COMPACT

We introduce **COMPACT**, a novel compression framework that actively compresses documents until it finds all necessary evidence for answering a question. To condense a large amount of information from documents, we devise an iterative architecture where the compressed contexts are updated at each iteration. In this section, we initially provide a comprehensive explanation of our framework. Subsequently, we detail the data construction for training our model.

3.1 How to Compress

We reconsider compression as sequential updates of compressed contexts based on the previous information. Figure 2 clearly shows the concept of our framework. Given a question and documents $D_k = \{d_1, ..., d_k\}_{i=1}^{k}$ from a retrieval system, we first group the documents as follows:

$$S_t = \{d_{t\times j+1}, d_{t\times j+2}, ..., d_{(t+1)\times j}\}$$

where S_t is a t-th segment consisting of j documents, and j represents the predefined number of documents to be compressed at each iteration. We then begin compressing each segment iteratively until it satisfies the end condition. It can be formulated as follows:

$$C_t, E_t = \pi(q, S_t, C_{t-1})$$

Here, q is a given question to answer. C_t and E_t represent the compressed context and a condition token at step t, respectively. C_t is used as part of the input for the next step. During compression, the model actively integrates information related to the question by analyzing both the previously compressed context and the newly provided segments. This approach ensures that only the most relevant information is preserved at each stage, resulting in a compact context. As the resulting context is designed to retain query-related information, it serves as a comprehensive memory of all iterations up to the current step.

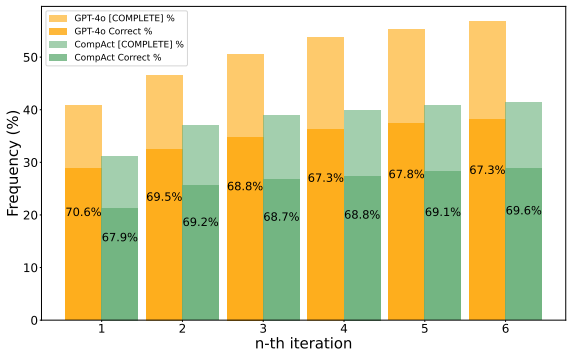

Figure 3: Distribution of iterations where models determine the compressed contexts to be complete. We compare the distribution between GPT-4o (Yellow) and COMPACT (Green). We also measure the percentage of correctness in complete cases.

3.2 Early Termination

Instead of completing all iterations, we introduce a specific end condition that early terminates the iterations. We implement this by including a condition token E in the generation process. The purpose of the condition token E is to assess whether an input segment S_t, combined with the previous context C_{t-1}, provides sufficient details to answer the question. If the token indicates the provided context is sufficient, the iteration terminates; otherwise, the iterations continue to gather lacking information until all missing details are obtained.

This adaptive termination offers three primary benefits. First, we prevent redundant contexts from entering the compressed contexts or acting as a distraction. Second, we can avoid meaningless iterations, thereby drastically lowering the computational burdens of our iterative architecture. Moreover, the adaptive feature allows the model to dynamically adjust to the complexity of the question and the information density of the passages. This flexibility allows the model to be both effective and efficient across a wide range of scenarios, from simple queries to more complex, multi-hop questions requiring extensive context integration.

3.3 Dataset Construction

Our model aims to compress documents into query-compacted contexts while concurrently determining the termination of the iterations. To cultivate this capability, we instruct a superior LLM to follow the three-step processes.

Sentence-Level Selection. We begin by asking the LLM to identify sentences, particularly focus-

ing on relevant clues that may help the answer the question. If certain sentences provide relevant information or implicitly clarify ambiguous points, the LLM is prompted to generate these sentences from the provided documents.

Query-focused Compression. We generate a summary of the selected sentences, emphasizing clues that can help answer the question. we explicitly restrict the LLM from making assumptions or attempting to draw conclusions without supporting evidence, as instructed: *"DO NOT make assumptions or attempt to answer the question; your job is to summarize only."* This restriction is crucial because our main objective is to condense relevant information from the provided documents, not to answer the questions directly. Skipping logical steps to directly answer the question, as if relying on parametric knowledge, can harm the compression performance by increasing the likelihood of missing necessary information.

Determining the Early Termination. We also prompt the LLM to evaluate its own compressed contexts based solely on the provided information, without any additional background context. We direct the LLM to generate a condition token (e.g., [COMPLETE] or [INCOMPLETE]) along with the rationale for its judgment.

Overall, we construct a synthetic dataset for training using the LLM with the instructions that describe the three-step processes. we conduct the data construction in two scenarios: realistic and distractor. In realistic scenarios, the provided documents are the results of a retrieval system. However, it is difficult to collect early termination cases due to the infrequent appearance of gold documents. To address this issue, we also conducted data collection in distractor scenarios which include predefined documents that contain all supporting facts needed to answer the question. After filtering the collected datasets from both scenarios, we build a training dataset consisting of 28k instances categorized into four distinct groups. Table 1 shows the categories of the dataset.

4 Experiment

4.1 Experimental Setup

Dataset Construction We employ GPT-4o API (2024-05-13) as the LLM to collect our dataset. We only use a subset of HotpotQA (Yang et al., 2018) train set for data collection. To retrieve documents, we use Contriever (Izacard et al., 2022), fine-tuned on MS-MARCO (Bajaj et al., 2016), as our retrieval system on the 2018 Wikipedia corpus (Karpukhin et al., 2020b). We set the default number of documents per segment as 5. Since it is rare to find additional evidence beyond the top 30 documents, we set the top-k to 30, allowing for a maximum of 6 iterations per query. To prevent lengthy API responses, the maximum number of generated tokens is limited to 700.

Training & Inference Leveraging the collected dataset, we perform supervised fine-tuning to train our model. Without using specific labeling or methods for particular iterations, we focus on teaching the model to effectively update the previous context based on the question and given documents at the current steps. we use instruction-tuned Mistral-7B (Jiang et al., 2023a) as our base model. At inference, we process the same number of segments and inputs as training. Further information is provided in the Appendix A.2.

4.2 Datasets

We evaluate COMPACT on both single-document and multi-document question-answering (QA) datasets. For single-document QA, we use Natural Question (NQ) (Kwiatkowski et al., 2019) and TriviaQA (TQA) (Joshi et al., 2017). For multi-document QA, we evaluate on HotpotQA (Yang et al., 2018), MuSiQue (Trivedi et al., 2022), and 2WikiMultiHopQA (Ho et al., 2020a). The evaluation is conducted on the dev set of each dataset, except for TriviaQA, which is evaluated on the test set. As mentioned, we comprise the training data only from HotpotQA. Therefore, we conducted zero-shot evaluation on the other datasets without accessing their training set.

4.3 Baselines

In Table 2, we compare COMPACT to several baseline methods. To ensure a fair comparison, we feed compressed contexts from each baseline to the same reader model, LLaMA3-8b (AI@Meta, 2024). We consider the following baselines:

- *Oracle.* We provide the reader with documents that contain an answer of questions. if such documents are not available, we include five documents as a default.
- *Raw Document.* We simply concatenate the top-k retrieved documents.

Baselines	HotpotQA			MuSiQue			2WikiMQA			NQ			TriviaQA		
	Comp.	EM	F1	Comp.	EM	F1	Comp.	EM	F1	Comp.	EM	F1	Comp.	EM	F1
Oracle	10.8x	39.9	51.2	10.3x	14.21	23.66	11.0x	37.4	43.2	-	-	-	-	-	-
Raw Document	1x	32.5	43.1	1x	6.8	16.0	1x	**31.6**	**37.2**	1x	40.0	52.1	1x	70.7	**77.5**
Long-Context LLM															
InternLM2-chat-7B	1x	8.0	20.3	1x	1.0	6.8	1x	9.3	19.5	1x	7.6	22.6	1x	12.1	31.5
Mistral-7B-Instruct-v0.2	1x	9.5	22.6	1x	1.0	7.9	1x	1.2	15.4	1x	4.3	20.9	1x	35.3	50.4
FILM	1x	32.4	43.7	1x	6.9	15.7	1x	26.4	31.7	1x	38.2	50.8	1x	62.7	71.7
GPT-3.5-turbo	1x	32.8	43.8	1x	7.3	16.1	1x	28.6	33.9	1x	**40.8**	**54.6**	1x	**69.9**	**77.4**
Compressor															
AutoCompressors	35.4x	18.4	28.4	34.7x	3.9	11.9	36.2x	19.0	24.5	34.4x	17.3	31.8	34.5x	55.3	64.3
LongLLMLingua	3.4x	25.6	35.3	3.4x	4.8	13.5	3.6x	27.9	32.9	3.5x	27.7	40.6	3.3x	64.0	70.8
RECOMP (extractive)	34.3x	29.7	39.9	32.7x	6.7	15.7	35.9x	29.9	34.9	32.7x	34.6	45.1	39.2x	67.6	74.1
CompAct (Ours)	**44.7x**	**35.0**	**46.5**	**36.0x**	**8.4**	**17.7**	**48.5x**	30.1	35.9	**44.3x**	37.8	49.6	**46.9x**	65.6	74.8

Table 2: Main results. We set the reader as LLaMA3-8b (AI@Meta, 2024) for a fair comparison. We retrieve top-30 documents to compute the scores. We use three Multi-hop and two single-hop question-answering datasets. Since our training datasets consist of HotpotQA dataset, we perform zero-shot evaluation on the rest of the datasets. Comp. refers to the compression rate which is denoted as follows: compression rate $= \frac{\text{\# of tokens in retrieved documents}}{\text{\# of tokens in compressed text}}$.

- *Long-Context LLM.* We select a number of LLMs that support long context windows, including Mistral-7B-Instruct-v0.2 (Jiang et al., 2023a), GPT-3.5-turbo (OpenAI, 2023), InternLM2-chat-7B (Cai et al., 2024), and FILM (An et al., 2024).

- *Compressor.* We compare CompAct with three compression-based methods. AutoCompressors (Chevalier et al., 2023) process segments of long context into soft prompts, which are prepended to the next segment as summary vectors. RECOMP (Xu et al., 2024) suggests extractive/abstractive methods to compress documents into textual summaries. LongLLMLingua (Jiang et al., 2023c) takes a perplexity-based approach to filter out tokens with less importance.

4.4 Results

We assess the performance of CompAct using three metrics: Compression rate (Comp.), Exact Match (EM), and F1 score (F1). Overall, CompAct exhibits strong performance across all benchmarks, while achieving the highest compression rate across all baselines. Specifically, CompAct outperforms other compression-based methods in all three metrics, demonstrating its ability to process abundant information efficiently. Compared to long-context LLMs, CompAct shows comparable performance in NQ and TriviaQA, while outperforming all three multi-document question answering benchmarks. This shows how CompAct excels at tasks that require integrating information within multiple documents.

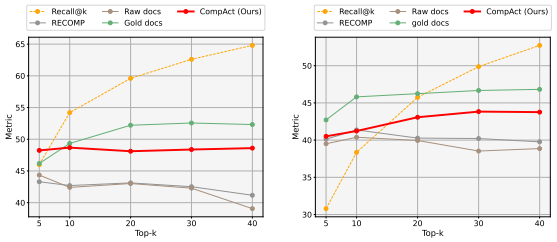

Figure 4: End QA performance using diverse retriever setups. We use BM25 (Left) and Contriever (Right) to retrieve top-k documents.

5 Analysis

We also analyze the quality of our compressed text, both qualitatively and quantitatively, to evaluate its usefulness through diverse reader LLMs. Finally, we investigate ways to facilitate the usage of compressors by reducing the inference time of CompAct in both realistic and benchmark experimental setups.

5.1 Compressor as a Plug-in Module

In Figure 2, we depict the compressor as a plug-in module. Our design highlights the ease of replacing it with new models as the retriever or reader evolves. Our goal is to determine if our CompAct can flexibly perform compressing the context provided by diverse retrievers and efficiently preserve useful information regardless of various readers.

Generalizability across Retrievers. In Figure 4, we describe the overall results. We use BM25 (Robertson et al., 2009) and Contriever (Izacard et al., 2022), the most common and useful setups to replace our retrieved documents.

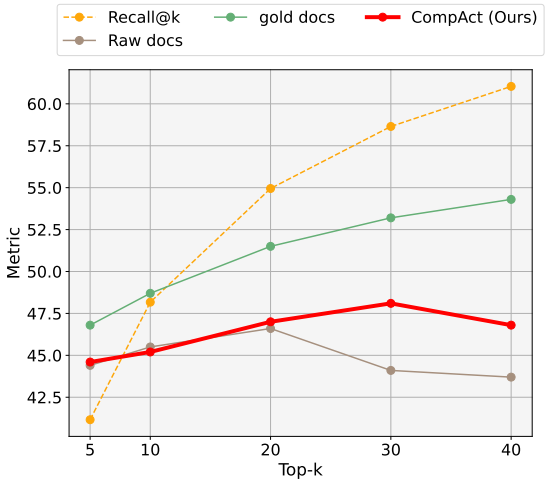

Figure 5: Performance of HotpotQA with different top-k documents. We set the reader as GPT-3.5-Turbo.

Components	HotpotQA			MuSiQue		
	Comp.	EM	F1	Comp.	EM	F1
Eval.	124.8x	30.8	41.5	119.8x	6.8	14.8
CT	44.7x	35.1	46.6	36.0x	8.3	17.7
CT + Eval.	31.9x	34.8	46.3	27.1x	8.1	17.2

Table 3: Results of each component effectiveness. Eval. refers to the evaluation text containing rationale and condition token. CT refers to the compressed text.

Datasets	N-th Iterations					
	1	2	3	4	5	6
HotpotQA	78.1	114.1	128.5	126.5	135.9	147.5
MuSiQue	77.5	110.6	135.2	91.6	145.6	124.0

Table 4: Average Length of Compressed Text per Iteration for HotpotQA and Musique.

We use the HotpotQA (Yang et al., 2018) dev set to compute the recall and F1 performances.

Surprisingly, in the top-5 documents, we observe that our COMPACT framework achieves a higher score compared to the gold documents given BM25 retrieved documents. Our framework also shows a saturated performance while retrieving up to 40 documents. However, we want to highlight that the score (48.7) shows significantly higher performance compared to other compressor models. Additionally, for the Contriever setup, where the retriever initially fails to retrieve useful documents, increasing the top-k leads to performance improvements. As we intended, these observations demonstrate that our COMPACT framework shows robustness across various retriever setups.

Generalizability across Readers. We try to figure out that our COMPACT framework truly provides useful compressed text to solve the multi-hop question answering. Thus, we assess our compressed text with diverse reader LLMs to prove it provides useful information regardless of specific reader models such as LLaMA2 13B (Touvron et al., 2023), LLaMA-3-8b (AI@Meta, 2024), and GPT-3.5-Turbo (OpenAI, 2023). We randomly sample 500 instances from the HotpotQA dataset and describe the reader performance of using top-30 retrieved documents. We compare our results with several baselines: providing pivotal information to solve questions (gold documents), prepending top-30 documents (raw documents), and other compression methods such as RECOMP (Xu et al., 2024) and LongLLMLingua (Jiang et al., 2023c).

In Figure 5, we demonstrate that our COMPACT framework sufficiently provides high-quality compressed text to solve multi-hop questions. Furthermore, our framework proves its effectiveness on the widely used top-k retrieved documents such as $k \in \{5, 10, 20, 30, 40\}$. Notably, until providing top-20 documents, there is little difference between the raw documents and our score. However, at the top-30 and top-40, performance degradation occurs as more documents are included as input, increasing irrelevant context. In contrast, our COMPACT framework shows lower performance degradation even with increased context. Additionally, COMPACT framework achieves a higher compression rate (44x) signifies that the number of input tokens is significantly reduced resulting in cost-efficiency using API calls. We provide LLaMA2-13B and LLaMA3-8B performance in Table 6.

5.2 Ablation Studies

Component Effectiveness. Our key point of COMPACT framework is using the compressed text (CT) and evaluation (Eval.) from previous outputs. The evaluation consists of the rationale for the compressed text and the condition token that determines early termination based on this rationale. We describe the detailed performance in Table 3. If we only provide the evaluation text, the compression rate increases dramatically, but the end performance (F1) significantly drops (Row 1). Additionally, when comparing cases where both compressed text and evaluation are provided versus compressed text only, there is no significant difference in performance (Row 2 & 3).

We also identify that as the reader LLM advances, the rationale provided by our COMPACT framework can negatively impact the performance. For instance, when using the LLaMA2-7B (Touvron et al., 2023) as the reader LLM, the performance achieves 45.0 given the previous output. However, when only the compressed text is provided, the performance drops to 44.2. To accurately analyze the trend of these findings, a detailed exploration through comparison with different reader LLMs will be reserved for future work.

Average Length of Compressed Text per Iteration. In Table 4, we provide the detailed length information of compressed text per iteration. From a compression perspective, our COMPACT framework compresses text from 30 retrieved documents into under 200 tokens. We observe that it maintains a high compression rate on average throughout iterations. To ensure the practicality of providing context with fewer tokens, we also provide an additional point. Among the models with over 1 million downloads on Huggingface[1], 102 out of 154 are language models. Of these, 77.5% can feed inputs of 512 tokens or fewer. Despite ongoing research on LLMs capable of handling long contexts, it is evident that many users still frequently employ models with smaller token inputs. Therefore, it seems like a positive direction to examine how the compressed text and evaluation provided by our framework can enhance the performance of classification models like BERT (Kenton and Toutanova, 2019), which accept inputs with fewer tokens.

5.3 Inference Latency

While COMPACT offers a significant cost-saving advantage by reducing the token usage in the reader, we also consider a potential increase in inference latency due to the active iteration of our framework. To investigate this, we measure the time taken to answer the question with our framework and other baselines. Given that the inference speed can vary depending on the composition of retrieved documents and types of queries, we assess the time on the multi-hop (e.g., HotpotQA (Yang et al., 2018)) question-answering dataset.

In Table 5, we measure diverse inference time: inference GPU time, compression GPU time, total GPU time (inference + compression), throughput (examples per second), and corresponding F1 score. We agree that our COMPACT framework has a crucial limitation for inference time in processing the compression. However, our COMPACT framework provides high-quality compressed text regardless of retriever and reader. Furthermore, we prove its effectiveness through the trained HotpotQA dataset and other zero-shot evaluations.

[1] https://huggingface.co/Models

Baselines	Inference GPU Time	Compression GPU Time	Total GPU Time	Throughput (example/sec)	F1
No Documents	1.5m	-	1.5m	5.58	31.7
Raw Documents	11.5m	-	11.5m	0.72	42.5
LongLLMLingua	3.3m	32.3m	35.6m	0.23	35.5
RECOMP	1.9m	2.1m	4.0m	2.10	41.5
COMPACT (5 docs)	1.4m	178.7m	180.1m	0.05	47.3
COMPACT (10 docs)	1.9m	92.9m	94.8m	0.09	45.4

Table 5: Inference time for HotpotQA dataset. Our iterative inference lets us consider the trade-off between massive inference time and high performance (compression rate and end QA performance).

How to Speed up Inference in COMPACT Framework (Varying Segment Size). Extending the segment size is a way to improve inference speed efficiently. Instead of retraining the model to handle more documents, we simply increase the number of documents provided per iteration. Specifically, we apply 10 documents to COMPACT, which was originally trained to compress only 5 documents. We observe performance degradation when looking at segments of different sizes than those seen during training, but we still observe a high level of performance. Adopting this approach can yield advantages during inference time in our framework.

6 Conclusion

We introduce **COMPACT**, a novel framework that employs an active strategy to compress extensive documents. Our framework effectively captures important context and compresses documents of large volumes without losing pivotal information. COMPACT can serve as a convenient plug-in module that can fully collaborate with advanced off-the-shelf retrievers and readers. Our framework achieves a high compression rate (44x), which significantly increases the cost-efficiency when collaborating with external API calls. Our experiments show that COMPACT shows significant improvements in compression rates and QA performance on multi-hop question-answering datasets such as HotpotQA and MuSiQue.

Limitations

Our main concern is about the inference time required to compress top-k retrieved documents.

We acknowledge that our COMPACT framework spends considerable time compressing the retrieved documents. However, with the combination of a strong retrieval system, there is also potential for significant time savings if sufficient contexts are provided earlier. Also, our effort to address complex question types through compression is pioneering in this field. This aspect makes our research valuable, as it sets the foundation for future work to build upon and potentially resolve these issues. We hope that subsequent research will continue to refine these methods, further enhancing the efficiency of inference latency of our framework.

Additionally, we found that there are non-trivial errors when judging the completeness of compressed contexts. We use GPT-4o API to collect the training data with our custom instructions. For example, during data collection, even when the documents provide oracle evidence to answer the question, GPT-4o outputs a [COMPLETE] condition token at a rate of 39.88%. This indicates that even GPT-4o, which we believed to perform the best in our situation, struggles with accurately determining completeness of contexts. Although we attempt to address the issue by filtering the error cases, there may still be instances where the model incorrectly judges completeness.

Moreover, we only train our model in Mistral-7B-Instruct-v0.2 due to resource limitations. We need to verify whether our COMPACT framework works well across a range of model sizes, both smaller (< 7B) and larger (> 7B). It is challenging to assert that our framework operates efficiently when the compression model is significantly larger than the reader LLM used afterward. It would be beneficial to conduct experiments during the rebuttal period to confirm these aspects.

Ethics Statement

In environmental cost, our training process can use a significant amount of energy as the process is computationally expensive. In our manuscript, we attempt to minimize these effects by pre-training on one Mistral model and only do the necessary supervised fine-tuning to minimize the computation cost. Furthermore, a potential risk of this work is that the generated dataset can contain biases of API calls such as stereotypes of racism and gender. To our knowledge, there haven't been significant issues reported when creating datasets related to question answering. However, it would be beneficial to apply methods that robustly train or validate against such concerns.

References

AI@Meta. 2024. Llama 3 model card.

Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, and Jian-Guang Lou. 2024. Make your llm fully utilize the context. *arXiv preprint arXiv:2404.16811*.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, Xiaoyi Dong, Haodong Duan, Qi Fan, Zhaoye Fei, Yang Gao, Jiaye Ge, Chenya Gu, Yuzhe Gu, Tao Gui, Aijia Guo, Qipeng Guo, Conghui He, Yingfan Hu, Ting Huang, Tao Jiang, Penglong Jiao, Zhenjiang Jin, Zhikai Lei, Jiaxing Li, Jingwen Li, Linyang Li, Shuaibin Li, Wei Li, Yining Li, Hongwei Liu, Jiangning Liu, Jiawei Hong, Kaiwen Liu, Kuikun Liu, Xiaoran Liu, Chengqi Lv, Haijun Lv, Kai Lv, Li Ma, Runyuan Ma, Zerun Ma, Wenchang Ning, Linke Ouyang, Jiantao Qiu, Yuan Qu, Fukai Shang, Yunfan Shao, Demin Song, Zifan Song, Zhihao Sui, Peng Sun, Yu Sun, Huanze Tang, Bin Wang, Guoteng Wang, Jiaqi Wang, Jiayu Wang, Rui Wang, Yudong Wang, Ziyi Wang, Xingjian Wei, Qizhen Weng, Fan Wu, Yingtong Xiong, Chao Xu, Ruiliang Xu, Hang Yan, Yirong Yan, Xiaogui Yang, Haochen Ye, Huaiyuan Ying, Jia Yu, Jing Yu, Yuhang Zang, Chuyu Zhang, Li Zhang, Pan Zhang, Peng Zhang, Ruijie Zhang, Shuo Zhang, Songyang Zhang, Wenjian Zhang, Wenwei Zhang, Xingcheng Zhang, Xinyue Zhang, Hui Zhao, Qian Zhao, Xiaomeng Zhao, Fengzhe Zhou, Zaida Zhou, Jingming Zhuo, Yicheng Zou, Xipeng Qiu, Yu Qiao, and Dahua Lin. 2024. Internlm2 technical report. *Preprint*, arXiv:2403.17297.

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. *arXiv preprint arXiv:2406.02376*.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

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. *arXiv preprint arXiv:2405.13792*.

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*.

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*.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. *arXiv preprint arXiv:2405.14831*.

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*, 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. In *Proceedings of the 28th International Conference on Computational Linguistics*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023a. Mistral 7b. *arXiv 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 language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Singapore. Association for Computational Linguistics.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023c. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. *arXiv preprint arXiv:2310.06839*.

Mandar Joshi, Eunsol Choi, Daniel 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)*. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020a. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020b. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2019. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA.

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

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*.

Tianle Li, Ge Zhang, Quy Duc Do, Xiang Yue, and Wenhu Chen. 2024. Long-context llms struggle with long in-context learning. *arXiv preprint arXiv:2404.02060*.

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*, Singapore. Association for Computational Linguistics.

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*.

Vaibhav Mavi, Anubhav Jangra, and Adam Jatowt. 2022. A survey on multi-hop question answering and generation. *arXiv preprint arXiv:2204.09140*.

Jesse Mu, Xiang Li, and Noah Goodman. Learning to compress prompts with gist tokens. In *Advances in Neural Information Processing Systems*, pages 19327–19352. Curran Associates, Inc.

OpenAI. 2023. Chatgpt.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, et al. 2024. Llmlingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. *arXiv preprint arXiv:2403.12968*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*.

Hongjin Qian, Zheng Liu, Peitian Zhang, Kelong Mao, Yujia Zhou, Xu Chen, and Zhicheng Dou. 2024. Are long-llms a necessity for long-context tasks? *arXiv preprint arXiv:2405.15318*.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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*.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Shengyi Huang, Kashif Rasul, Alexander M. Rush, and Thomas Wolf. 2023. The alignment handbook. https://github.com/huggingface/alignment-handbook.

Zhiruo Wang, Jun Araki, Zhengbao Jiang, Md Rizwan Parvez, and Graham Neubig. 2023. Learning to filter context for retrieval-augmented generation. *arXiv preprint arXiv:2311.08377*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

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*.

Yusen Zhang, Ruoxi Sun, Yanfei Chen, Tomas Pfister, Rui Zhang, and Sercan Ö. Arik. 2024. Chain of agents: Large language models collaborating on long-context tasks. *Preprint*, arXiv:2406.02818.

A Example Appendix

A.1 HuggingFace Models Statistics

Sequence Length	Language Models (%)
128	14.7
512	62.8
$\geq$ 1024	22.5

Table 6: Huggingface Models Statistics. 77.5% of models cannot receive at least top-5 documents as input. We select frequently-used models downloaded at least 1M in https://huggingface.co/Models.

A.2 Training & Inference Details

We use 8 Nvidia A100 with 80GB memory to train our COMPACT framework. Our code is written in PyTorch (Paszke et al., 2019) and HuggingFace (Wolf et al., 2019). We use supervised fine-tuning through published alignment-handbook (Tunstall et al., 2023). We train the model with Adam optimizer (Kingma and Ba, 2015), using a learning rate of 2e-6, a batch size of 128, and 0.1 warm up ratio for 4 epochs. For inference, we use batch decoding to speed up our inference time.

A.3 Details of Baselines

Long-context LLMs. (1) InternLM2-chat-7B (Cai et al., 2024) has shown near-perfect performance on the Needle-in-the-Haystack task, which tests how well a model utilizes information within a long context. (2) Mistral-7B-Instruct-v0.2 (Jiang et al., 2023a) has recently shown

Dataset	Train	Dev	Test	Avg. # of Supporting Documents	# of Pre-defined Context
NaturalQuestions (Kwiatkowski et al., 2019)	79,168	8,757	3,610	-	-
TriviaQA (Joshi et al., 2017)	78,785	8,837	11,313	-	-
HotpotQA (Yang et al., 2018)	90,447	7,405	-	2	10
MuSiQue (Trivedi et al., 2022)	39,876	4,834	4,918	1.89 (Dev)	20
2WikiMultiHopQA (Ho et al., 2020a)	167,454	12,576	12,576	2.44 (Dev)	10

Table 7: Statistics of multi-hop and single-hop question answering datasets.

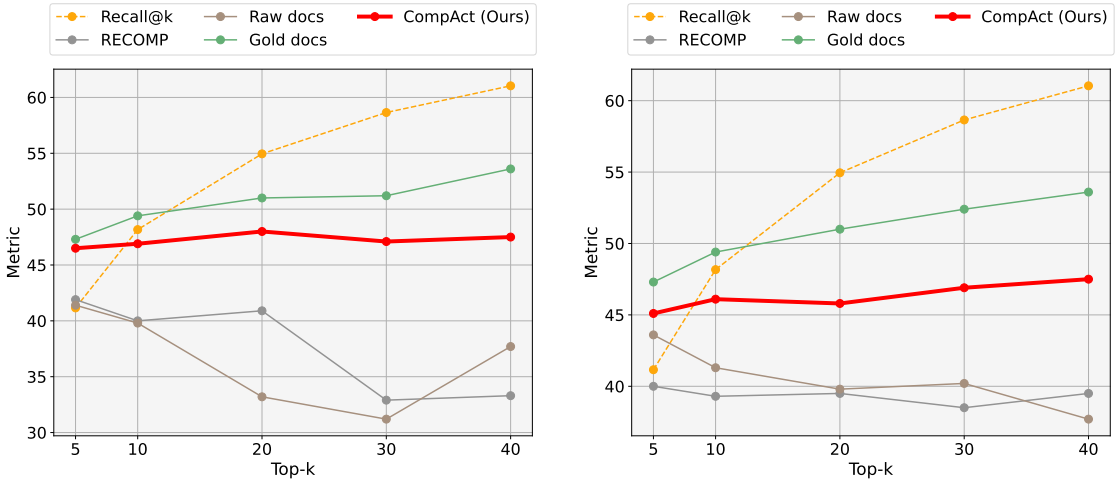

Figure 6: Performance of HotpotQA with different top-k documents. We set the reader as LLaMA2-13B (Left) and LLaMA3-8B (Right).

strong performance across various benchmarks and supports a 32k context window. (3) FILM-7B (An et al., 2024), trained with a synthetic long-context question-answering dataset, has shown strong performance on tasks that require information awareness in the long context. (4) We also experiment with GPT-3.5-turbo, a popular proprietary LLM that supports a 16k context window.

Compressors. (5) AutoCompressors (Chevalier et al., 2023) process segments of long context into soft prompts, which are prepended to the next segment as summary vectors. We use 50 summary tokens for every 2,048 tokens, following the setup from the original paper. (6) LongLLMLingua (Jiang et al., 2023c) takes a perplexity-based approach to filter out tokens with less importance. (7) RECOMP (Xu et al., 2024) suggests an extractive compressor that extracts relevant sentences using a dual encoder model, and an abstractive compressor that summarizes documents using an encoder-decoder model. We experiment with the extractive compressor setting.

First Iteration:

1. Generate a summary of source documents to answer the question. Ensure the summary is under 200 words and does not include any pronouns. DO NOT make assumptions or attempt to answer the question; your job is to summarize only.

2. Evaluate the summary based solely on the information of it, without any additional background context: if it lacks sufficient details to answer the question, print [INCOMPLETE]. If it provides all necessary details, print [COMPLETE]. You should provide the reason of the evaluation.

Question: [QUESTION]

Source documents: [SOURCE DOCUMENTS]

Summary:

Subsequent Iterations:

1. Generate a summary of the source documents and the previous summary to answer the question based on the evaluation of the previous summary. The evaluation indicates the missing information needed to answer the question. Ensure the summary is under 200 words and does not include any pronouns. DO NOT make assumptions or attempt to answer the question; your job is to summarize only.

2. Evaluate the summary based solely on the information of it, without any additional background context: if it lacks sufficient details to answer the question, print [INCOMPLETE]. If it provides all necessary details, print [COMPLETE]. You should provide the reason of the evaluation.

Question: [QUESTION]

Evaluation of previous summary: [EVALUATION OF PREVIOUS SUMMARY]

Previous summary: [PREVIOUS SUMMARY]

Source documents: [SOURCE DOCUMENTS]

Summary:

Table 8: Prompts used in CompAct

Sentence Selection (First Step):

Source sentences: [SOURCE SENTENCES]

Reference sentences: [REFERENCE SENTENCES]

Question: [QUESTION]

1. Follow instructions below. Choose 0 to 3 sentences that directly address the critical points needed to answer the question. Additionally, include 0 to 3 sentences that provide useful context, even if they do not directly answer the question. Ensure that you avoid selecting multiple sentences with overlapping content. (prefix: Sentences)

2. Generate a summary of chosen sentences (prefix: Summary). Ensure the summary is under 200 words and does not include any pronouns. DO NOT make assumptions or attempt to answer the question; your job is to summarize only.

3. Evaluate the summary based solely on the information of it, without any additional background context: if it lacks sufficient details to answer the question, print [INCOMPLETE]. If it provides all necessary details, print [COMPLETE]. You should provide the reason of evaluation (prefix: Evaluation)

Sentences:

Summary Generation (Second Step):

Source sentences: [SOURCE SENTENCES]

Question: [QUESTION]

1. Generate a summary of the source documents and the previous summary to answer the question based on the evaluation of the previous summary. The evaluation indicates the missing information needed to answer the question. Ensure the summary is under 200 words and does not include any pronouns. DO NOT make assumptions or attempt to answer the question; your job is to summarize only.

2. Evaluate the summary based solely on the information of it, without any additional background context: if it lacks sufficient details to answer the question, print [INCOMPLETE]. If it provides all necessary details, print [COMPLETE]. You should provide the reason of evaluation.

Sentences:

Table 9: Prompts for dataset construction

Question: Who was the eldest brother of the Mexican drug trafficker born 12 March 1952?

Model: COMPACT (Ours)

Iter 1

Document (Rank 3): Benjamín Arellano Félix (born 12 March 1952) is a Mexican drug trafficker and former leader of the Mexican criminal organization known as the Tijuana Cartel or "Arellano-Félix Organization". Benjamín Arellano Félix, who worked closely with his brothers, was one of Mexico's most powerful drug lords and the supplier of one-third of the U.S.'s cocaine. Benjamín had six brothers: He also has four sisters. Two of them, Alicia and Enedina, are most active in the cartel's affairs. Benjamín was first arrested on 18 June 1982, in Downey, California, for receiving 100 kilos of cocaine smuggled through the San

Compressed Text: Benjamín Arellano Félix, born on 12 March 1952, is a Mexican drug trafficker and former leader of the Tijuana Cartel. He had six brothers, including He, and four sisters. He also has four sisters.

Evaluation: The summary provides information about Benjamín Arellano Félix and his siblings but does not specify who the eldest brother is. **[INCOMPLETE]**

Iter 2

Document (Rank 7): Francisco Rafael Arellano Félix (24 October 1949 - 18 October 2013) was a Mexican drug lord and former leader of the Tijuana Cartel, a drug trafficking organization. He was the oldest of seven brothers and headed the criminal organization early in the 1990s alongside them. Through his brother Benjamín Francisco Rafael joined the Tijuana Cartel in 1989 following the arrest of Miguel Ángel Félix Gallardo, one of the most prominent drug czars in Mexico during the 1980s. When the Arellano Félix took control of the organization in the early 1990s, tensions with the rival Sinaloa Cartel

Compressed Text: Francisco Rafael Arellano Félix, born on 24 October 1949, was a Mexican drug lord and former leader of the Tijuana Cartel. He was the oldest of seven brothers and headed the criminal organization early in the 1990s alongside them. Benjamín Arellano Félix, born on 12 March 1952, is a Mexican drug trafficker and former leader of the Tijuana Cartel. He had six brothers, including He, and four sisters. He also has four sisters.

Evaluation: The summary provides all necessary details to answer the question. It identifies Francisco Rafael Arellano Félix as the oldest brother of Benjamín Arellano Félix, who was born on 12 March 1952. **[COMPLETE]**

Answer: Francisco Rafael Arellano Félix (**Correct**)

Model: RECOMP (Xu et al., 2024)

Summary: Miguel Rodríguez Orejuela He is the younger brother of Gilberto Rodríguez Orejuela.
Roberto de Jesús Escobar Gaviria Roberto de Jesús Escobar Gaviria Roberto de Jesús Escobar Gaviria (born January 13, 1947), nicknamed El Osito (Little Bear), was the brother of the drug kingpin, Pablo Escobar, and the former accountant of the Medellín Cartel, which was responsible for up to 80 percent of the cocaine smuggled into the United States.

Answer: Fabio Ochoa Vásquez (**Wrong**)

Table 10: A compression example of COMPACT and comparison with another compressor (RECOMP)