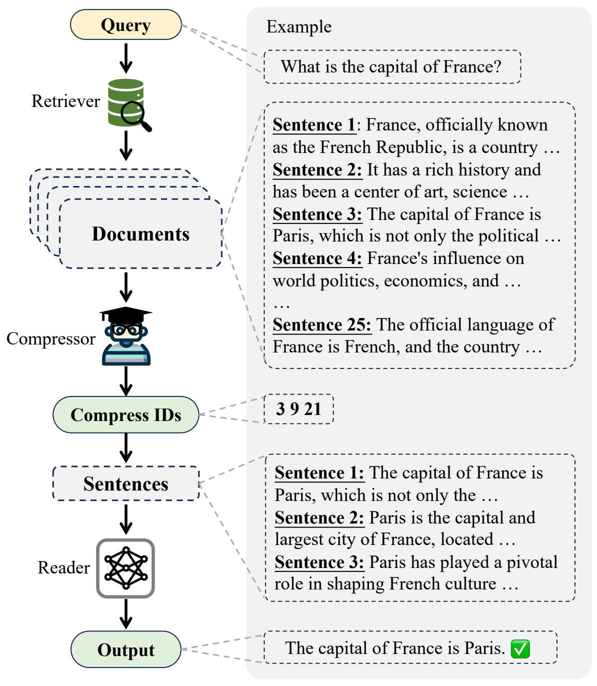
GCR: Generative Compressing for Retrieval Augmented Generation

Anonymous ACL submission

Abstract

Retrieval Augmentation Generation (RAG) has significantly mitigated hallucination issues in Large Language Models (LLMs), with context compressing playing a pivotal role in enhancing the efficiency of the RAG systems. Traditional context compressing approaches include extractive and abstractive methods. Extractive methods often perform poorly due to their independent modeling of sentences, while abstractive methods suffer from high latency and the risk of introducing hallucinations. In this paper, we propose **GCR**, a novel generative compression method that reformulates context compression as sentence index generation, ensuring minimal inference latency. GCR effectively models semantic interactions between sentences, prevents potential hallucinations during compression, and offers adaptive control over the compression rate. Extensive experiments across three knowledge-intensive tasks confirm the effectiveness and efficiency of our method.



Query
Retriever
Documents
Compressor
Compress IDs
Sentences
Reader
Output
Example
What is the capital of France?
Sentence 1: France, officially known as the French Republic, is a country …
Sentence 2: It has a rich history and has been a center of art, science …
Sentence 3: The capital of France is Paris, which is not only the political …
Sentence 4: France's influence on world politics, economics, and …
…
Sentence 25: The official language of France is French, and the country …
3 9 21
Sentence 1: The capital of France is Paris, which is not only the …
Sentence 2: Paris is the capital and largest city of France, located …
Sentence 3: Paris has played a pivotal role in shaping French culture …
The capital of France is Paris. ✅

Figure 1: Illustration of Generative Compressing.

1 Introduction

Recently, Large Language Models (LLMs) (Taylor et al., 2022; Chowdhery et al., 2022; Zhao et al., 2023a) have demonstrated impressive performance across a variety of downstream tasks (Xia et al., 2024; Yamauchi et al., 2023; Imani et al., 2023; Lewkowycz et al., 2022). Despite these advancements, LLMs are still prone to generate responses that contain hallucinated facts and inaccurate information (Ji et al., 2023; Shuster et al., 2021; Zhang et al., 2023a), which raises concerns about their reliability. To mitigate this issue, researchers have adopted Retrieval-Augmented Generation (RAG), which retrieves external documents to enhance response accuracy (Ram et al., 2023; Shi et al., 2023; Rashkin et al., 2021; Gao et al., 2022; Bohnet et al., 2022; Menick et al., 2022). However, directly incorporating retrieved documents into the prompt can be computationally expensive and may introduce irrelevant or noisy information.

A promising solution is to use context compression techniques (Li, 2023; Xu et al., 2024; Wang et al., 2023c; Yoon et al., 2024; Jiang et al., 2023b; Pan et al., 2024) to condense retrieved documents into a more concise and relevant format. Current context compression methods can be broadly classified into two categories: extractive and abstractive methods. Extractive methods (Xu et al., 2024; Jin et al., 2024a; Reimers and Gurevych, 2019) typically utilize retrieval methods to calculate the similarity between queries and sentences, selecting the sentences with the highest similarity as the compressed output. In contrast, abstractive methods generate summaries of the retrieved documents. For example, RECOMP (Xu et al., 2024) trains a compressor to produce summaries of retrieved content, while FILCO (Wang et al., 2023c) first identifies useful context through lexical and information-theoretic approaches before training a context-filtering model. COMPACT (Yoon et al., 2024) employs an active strategy to condense extensive documents without losing critical information, and SKR (Qiao et al., 2024) optimizes the compression process by focusing on supportiveness. LongLLMLingua (Jiang et al., 2023b) filters out less important information based on perplexity.

While these methods have demonstrated promising results, they still face three significant limitations. **First**, extractive methods typically evaluate the similarity between each sentence and the query independently, disregarding the contextual relationships between sentences, which can result in suboptimal compression. **Second**, although abstractive methods offer more flexibility in generating summaries, they often modify the original content, which risks introducing hallucinations or information not present in the retrieved documents. This issue becomes more pronounced when the model's parametric knowledge conflicts with the non-parametric knowledge in the documents (Jin et al., 2024b; Tan et al., 2024; Wang et al., 2023a). **Third**, the generative process in abstractive methods is typically iterative, leading to high latency as the model produces the compressed tokens step by step. This delay poses a significant challenge in real-world applications, particularly in online serving scenarios where low latency is crucial.

To address these challenges, we propose a generative compressor, **GCR**, which redefines the context compression process as a sentence index generation task. Specifically, during inference, GCR first splits the original documents into sentences, which are input to the compressor to generate the indexes of the most relevant sentences. Our approach follows the following three stages. In the **Supervised Distillation** stage, a strong LLM extracts the most relevant sentences from the retrieved documents to create training data. To improve extraction accuracy, we guide the model to follow a Chain-of-Thought process (Wei et al., 2022), where it first analyzes both the query and sentences before outputting the relevant indexes. The compressor is then fine-tuned on this labeled data to develop its basic compression capability. In the **Critic Sampling** stage, the compressor generates multiple compression results for each query, which are ranked by the LLM. To reduce positional bias (Xiong et al., 2023), we apply **permutation ranking**, where the positions of the compression results are randomly shuffled, and the LLM reranks them for each permutation. The results from multiple permutations are then ensembled to produce the final ranking. In the **Preference Alignment** stage, we construct preference pairs from the ranking information and use them to perform the alignment. During inference, to further enhance the quality of the compression results, we introduce **constrained consistency sampling**, which performs multiple top-k samplings (Fan et al., 2018) and ranks sentences based on their appearance frequencies.

GCR offers the following three main advantages: (1) **Lower Latency**: Unlike traditional abstractive methods, GCR produces only a small number of index tokens, significantly reducing latency. Moreover, since it does not modify the original content, it completely eliminates the risk of introducing hallucinated information into compression results. (2) **Enhanced Interaction Modeling**: In contrast to traditional extractive methods, GCR can effectively model the semantic interactions between all sentences simultaneously, leveraging the strong reasoning capabilities of the language model. (3) **Flexible Compression Control**: The novel index generation format of GCR allows seamless integration with the self-consistency sampling technique (Wang et al., 2022), which not only enhances compression quality but also provides flexible control over the compression rate.

To summarize, our contributions are as follows:

- We propose GCR, a novel generative compression method that reformulates context compression as sentence index generation, offering minimal inference latency.
- GCR effectively models semantic interactions between sentences, prevents potential hallucinations during compression, and enables adaptive control of the compression rate.
- We conduct extensive experiments on five datasets across three QA tasks, validating both the effectiveness and efficiency of our method.

2 Methodology

2.1 Preliminary

In Retrieval Augmented Generation (RAG), given query q, a retriever is first employed to retrieve a set of similar documents $\mathcal{D} = \{d_1, d_2, ..., d_m\}$. Then, a reader LLM will answer the question based on these documents. We assume that each retrieved document $d_i = [t_i^1, ..., t_i^n]$ contains n text spans t. The task of context compression aims to select the

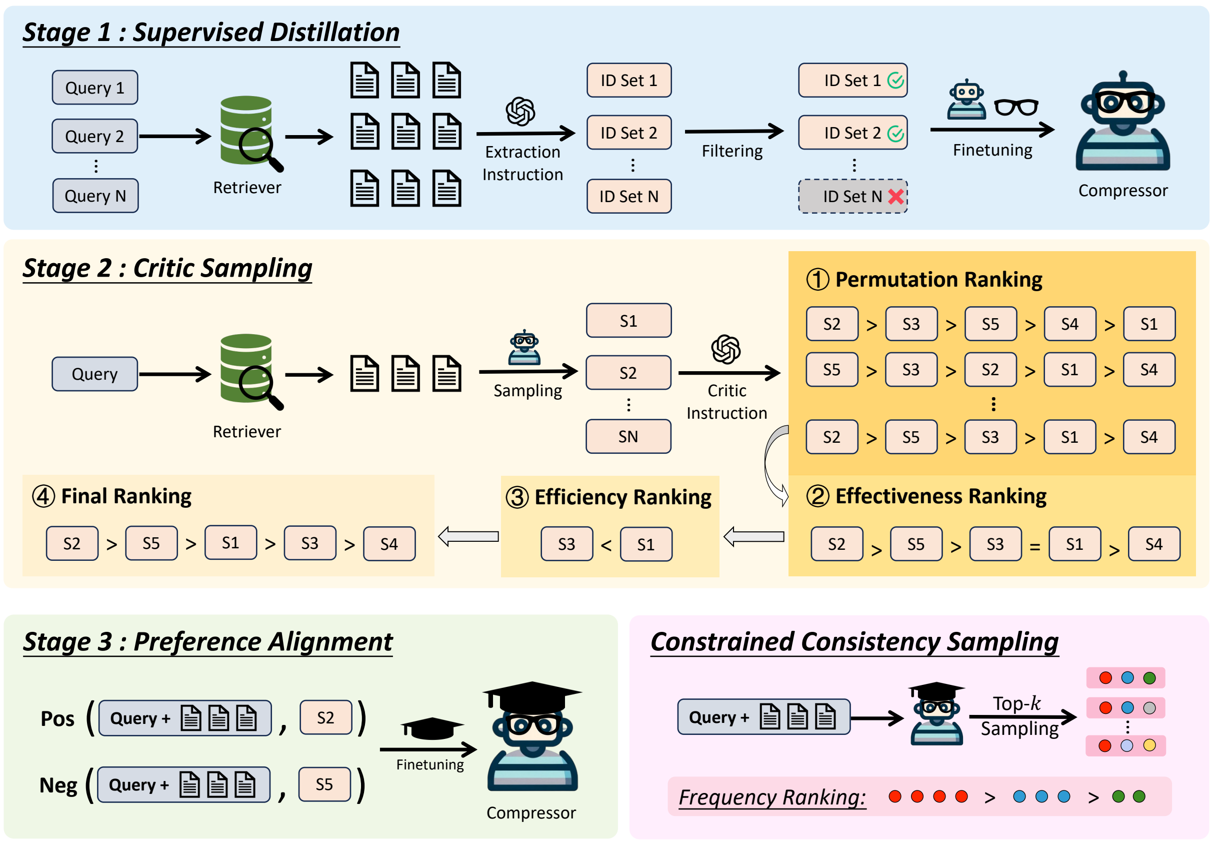


Figure 2: Overview of the GCR framework. 1) Supervised Distillation: A strong LLM selects relevant sentences from retrieved documents to create training data for fine-tuning the compressor. 2) Critic Sampling: The compressor generates multiple compression results for each query, which the LLM ranks to form a ranked list. 3) Preference Alignment: Preference pairs are constructed from ranking information for alignment. During inference, we perform multiple top-k samplings and rank sentences based on their appearance frequencies.

most relevant text spans and output their indexes. The selected spans are then concatenated and fed into the reader LLM for question answering.

2.2 GCR Framework

As shown in Figure 2, GCR mainly consists of three stages. In the supervised distillation stage, a strong LLM selects the most relevant sentences from the retrieved documents to create training data, which is then used to fine-tune the compressor and build its basic compression ability. In the critic sampling stage, the compressor generates multiple compression results for each query, which are ranked by the LLM to form a ranked list. In the preference alignment stage, we construct preference pairs from the ranking and use them to conduct the alignment.

Supervised Distillation In the first stage, we utilize a strong LLM as a data labeler to extract relevant sentences for each query in the training set. To improve the extraction accuracy of the LLM, we instruct it to decompose the index extraction process into three steps: query analysis, sentence analysis, and index output. Specifically, the LLM first performs query analysis to thoroughly understand the topic and intent of the query. Next, it conducts sentence analysis, summarizing the content related to the query and identifying sentences that are relevant. Finally, the LLM lists the specific indices of the relevant sentences. To further improve the quality of the labeled data, we filter out the extraction results that do not contain the correct answers to the question. Then the compressor model is trained on filtered data, which equips it with the basic ability to generate compressing results for given queries and documents.

Critic Sampling In this stage, we construct preference data to facilitate the preference alignment process. Given a query q and its retrieved documents, the compressor is used to sample multiple compression results for each query, denoted as $S = \{s_1, s_2, \ldots, s_N\}$. Next, we employ a strong LLM to perform a list-wise ranking of these sampled compression results. Specifically, a group of compression results is fed into the LLM, which then outputs their ranking based on their helpfulness in answering the query. To mitigate any positional bias that might influence the LLM's judgment due to the order in which the compression results are presented (Xiong et al., 2023), we introduce **Permutation Ranking**. This process involves randomly shuffling the positions of the compression results in the prompt and requesting the LLM to output the reranked list for each permutation.

During the permutation ensembling process, we calculate the pairwise ranking scores for each compression result s_i by counting the number of times the LLM ranks s_i higher than other compression results across the different permutations. This provides a cumulative pairwise ranking score for each result, reflecting how consistently s_i performs relative to the others. Based on these cumulative scores, we establish a ranking of all the compression results. In instances where two compression results share the same pairwise ranking score, we resolve the tie by considering their compression rates, prioritizing the result with the higher compression rate. This ensures that the final ranking list reflects our preference for both effectiveness and efficiency.

Preference Alignment In this stage, we sample preference pairs from the final ranking list and use them to train the compressor with Direct Preference Optimization (Rafailov et al., 2024). After preference alignment, the compressor not only selects the most relevant sentences from the documents but also prioritizes those with higher compression rates, balancing effectiveness and efficiency. This results in a compression model that extracts useful information while maintaining a compact output.

Constrained Consistency Sampling During inference, we apply a constrained decoding mechanism that restricts the output indices to valid sentence positions within the input documents. To further enhance the robustness and reliability of the compression results, we adopt the self-consistency sampling strategy (Wang et al., 2022). Specifically, for each query, we perform multiple Top-k sampling (Fan et al., 2018) iterations with the compressor to generate a set of possible compression outputs. We then aggregate these results by counting the frequency of each sentence index across all sampling iterations. After ranking the sentences based on their frequencies, we select the top-m ranked sentences to form the final compression result. This approach not only enhances the robustness of the compression but also offers flexible control over the compression rate, allowing the system to adapt to different application requirements.

3 Experiment Setup

3.1 Datasets and Metrics

Datasets We experiment on five datasets across three knowledge-intensive tasks: (1) **Open-domain QA,** including NQ dataset (Kwiatkowski et al., 2019), TriviaQA dataset (Joshi et al., 2017) and SQuAD dataset (Rajpurkar et al., 2016); (2) **Multi-hop QA**, including HotpotQA dataset (Yang et al., 2018). (3) **Ambiguous QA**, including ASQA dataset (Stelmakh et al., 2022).

Metrics We evaluate performance using two key metrics: Exact Match (EM) and F1 Score. A predicted answer is considered correct under the EM metric if its normalized form exactly matches any of the normalized versions of the reference answers in the answer list. The F1 score, on the other hand, measures the word-level overlap between the normalized predicted answer and the reference answers in the provided answer list.

3.2 Baselines

Among the baselines, Closed Book represents no retrieval, and Raw Document represents no compression. Extractive methods include LongLLMLingua (Jiang et al., 2023b) and RECOMP (Xu et al., 2024), while generative methods include FILCO (Wang et al., 2023c) and COMPACT (Yoon et al., 2024). Please refer to Appendix B for detailed introductions to these methods.

3.3 Implementation Details

In our experiments, we initialize the compressor model with Qwen2-7B[1]. For the reader models, we employ Qwen2-7B, Meta-Llama-3-8B[2], and Qwen2.5-14B[3]. We use Qwen-Max as the data labeler in the supervised distillation and critic sampling stage. We use Wikipedia dump from Jan. 27, 2020 as our retrieval corpus and use DPR (Karpukhin et al., 2020) as our dense retriever. For each query, we retrieve the top-5 most similar documents from the retrieval corpus. We plan to open-source the code upon acceptance to enhance the reproducibility of our method.

[1] https://huggingface.co/Qwen/Qwen2-7B
[2] https://huggingface.co/meta-llama/Meta-Llama-3-8B
[3] https://huggingface.co/Qwen/Qwen2.5-14B

Methods	NQ		TriviaQA		SQuAD		HotpotQA		ASQA		Avg.	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
QWEN-2-7B												
Closed Book	22.10	29.84	52.30	59.13	13.60	23.84	22.30	30.66	26.37	35.79	27.33	35.86
Raw Document	38.70	46.55	58.75	65.59	18.90	28.52	25.30	33.54	45.70	54.47	37.47	45.73
LongLLMLingua	26.60	35.60	52.60	59.19	13.55	22.86	22.65	29.94	32.74	43.30	29.63	38.18
RECOMP	33.25	41.00	54.15	61.72	17.05	27.33	23.95	32.15	39.22	49.23	33.52	42.29
FILCO	35.35	42.55	58.20	65.15	19.10	28.25	24.10	32.70	42.79	52.40	35.91	44.21
COMPACT	36.65	44.62	57.30	64.85	17.55	26.74	24.75	33.26	43.69	53.80	35.99	44.65
GCR	**40.75**	**49.21**	**61.30**	**68.75**	**20.25**	**29.84**	**26.95**	**35.91**	**47.37**	**57.90**	**39.32**	**48.32**
LLAMA-3-8B												
Closed Book	30.10	37.86	64.05	70.20	16.40	25.92	23.45	31.50	34.86	45.23	33.77	42.14
Raw Document	41.55	49.98	67.30	72.67	22.60	32.09	28.25	36.66	48.38	57.14	41.62	49.71
LongLLMLingua	33.15	41.67	63.60	69.23	17.20	26.92	26.30	34.59	40.67	50.32	36.18	44.55
RECOMP	37.90	45.34	64.35	70.33	20.60	30.17	26.30	34.51	45.25	54.32	38.88	46.93
FILCO	40.30	47.50	65.75	71.15	21.80	30.95	28.70	37.05	44.58	53.63	40.23	48.06
COMPACT	40.95	49.33	65.25	71.44	22.05	31.50	29.90	39.15	47.37	56.91	41.10	49.67
GCR	**42.90**	**50.66**	**67.70**	**73.52**	**22.85**	**32.17**	**30.25**	**39.22**	**49.16**	**58.79**	**42.57**	**50.87**
QWEN-2.5-14B												
Closed Book	28.95	38.04	61.70	67.59	20.60	31.20	26.30	35.00	36.42	45.95	34.79	43.56
Raw Document	42.80	50.44	63.95	69.74	22.60	31.94	27.90	36.45	47.60	55.77	40.97	48.87
LongLLMLingua	30.25	39.00	57.60	64.03	17.80	26.40	25.75	33.68	36.42	45.71	33.56	41.76
RECOMP	36.35	42.56	60.70	66.44	19.65	28.09	26.10	33.89	42.35	50.05	37.03	44.21
FILCO	39.40	46.50	62.95	68.60	21.45	30.40	26.90	35.50	44.13	52.51	38.97	46.70
COMPACT	41.20	48.32	61.95	67.92	21.15	29.67	28.95	37.72	46.93	56.03	40.04	47.93
GCR	**43.40**	**50.85**	**64.65**	**70.55**	**23.50**	**32.50**	**30.35**	**39.00**	**51.40**	**60.00**	**42.66**	**50.58**

Table 1: Performance comparison on five datasets across readers of different parameter sizes.

4 Experimental Results

4.1 Main Results

In this section, we present a comprehensive comparison of the performance of various compressors across five datasets using readers of different sizes. Based on the results shown in Table 1, several observations can be made:

First, our method consistently achieves the best performance across all datasets and readers, demonstrating both its effectiveness and generalizability. This is because, after performing preference alignment, our method effectively extracts the most useful sentences. Additionally, the constrained consistency sampling enhances the method's robustness.

Second, among the baselines, the abstractive method COMPACT performs better than the extractive method RECOMP. This is mainly because the extractive method models each sentence independently, failing to utilize contextual semantic information. It is worth noting that although our method is also extractive, it models all sentences together, enabling it to fully capture semantic information and leading to better results.

Third, our method consistently delivers the best performance across different readers, confirming its superior generalization ability. Furthermore, the training process of our method is reader-agnostic, meaning it can easily compress documents for various readers without requiring additional retraining.

Methods	NQ		ASQA	
	Comp.	EM	Comp.	EM
GCR	10.11	40.75	11.37	47.37
-w/o Sampling	11.37	40.45	13.15	46.93
-w/o Alignment	9.26	40.40	9.93	46.26
-w/o Filtering	9.40	39.75	9.94	45.81

Table 2: Ablation Study. We experiment by gradually removing all components using Qwen2-7B as the reader.

4.2 Ablation Study

In this section, we assess the impact of each component in our model by gradually removing them. Specifically, we conduct experiments on the NQ and ASQA datasets using Qwen2-7B as the reader.

As shown in Table 2, removing any component leads to performance degradation, verifying their importance. Specifically, removing the constrained

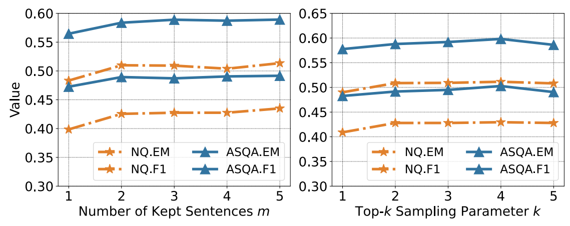


Figure 3: The performance change over different hyper-parameters on NQ and ASQA datasets.

Methods	NQ		TriviaQA		ASQA	
	EM	Comp.	EM	Comp.	EM	Comp.
Raw Document	38.70	1.00x	58.75	1.00x	45.70	1.00x
LongLLMLingua	26.60	3.81x	52.60	3.76x	32.74	3.81x
RECOMP	33.25	4.70x	54.15	4.57x	39.22	4.66x
FILCO	35.35	3.07x	58.20	3.12x	42.79	3.09x
COMPACT	36.65	9.35x	57.30	9.99x	43.69	10.09x
GCR	**40.75**	**10.11x**	**61.30**	**11.16x**	**47.37**	**11.37x**

Table 3: Compression Analysis. Comp. refers to the compression rate which is denoted as follows: compression rate $= \frac{\text{\# of tokens in retrieved documents}}{\text{\# of tokens in compressed text}}$.

consistency sampling mechanism decreases the compression rate but increases accuracy. This occurs because single sampling cannot capture all the information needed to answer the question. By sampling multiple times, we improve the recall rate of useful sentences, which justifies the increased accuracy. Moreover, removing the preference alignment leads to a significant decrease in the compression rate. This is because, during preference alignment, the compressor is trained to choose compression results that balance both effectiveness and efficiency. Therefore, when preference alignment is removed, both the effectiveness and efficiency of the compressor decrease. Finally, removing the filtering mechanism introduces noise into the training data of supervised distillation, which can confuse the compressor and lead to inferior performance.

4.3 Hyper-parameter Study

In this section, we analyze the impact of two important hyperparameters on our model's performance: the number of kept sentences m and the Top-k sampling parameter k. Specifically, we experiment on the NQ and ASQA datasets, using Meta-Llama-3-8B as the reader LLM. Based on the result shown in Figure 3, several observations can be made.

First, as the number of kept sentences m increases, the performance gradually improves. This is expected, as keeping more sentences in the compressed result provides the reader with more information. Then, the reader is more likely to absorb useful content to answer the question, leading to enhanced performance. However, retaining more sentences reduces the compression rate, leading to higher inference costs for the reader. Therefore, we recommend tuning this parameter according to available computational resources and the desired trade-off between performance and efficiency.

Second, as the Top-k sampling parameter k increases, the performance initially improves but eventually declines. This is because when k is low, the compressor generally generates the same result across multiple sampling iterations, making the consistency sampling mechanism ineffective. However, when k is too high, the compressor may generate outputs randomly, introducing noise into the compression and decreasing performance. Therefore, selecting an optimal value for k is crucial to ensure the robustness of the compressed results.

4.4 Analysis

Compression Analysis In this section, we analyze the effectiveness of the compressors by comparing their compression rates. Specifically, we conduct the experiments on NQ, TriviaQA and ASQA datasets using Qwen2-7B as the reader.

As shown in Table 3, all compressor models significantly reduce the number of tokens in the retrieved documents, thereby dramatically decreasing the inference cost for the reader LLMs. Among the compressor methods, our approach achieves the highest compression rate while maintaining the best model performance. Although other compressor methods also reduce cost, they fail to capture all the important information, resulting in inferior model performance compared with the raw document. In contrast, our method not only outperforms the uncompressed raw document method but also achieves a lower cost. This is because our method effectively extracts important information from the retrieved documents, preventing noisy information from influencing the model's performance.

Latency Analysis In this section, we compare the inference latency of our framework with other baselines. Specifically, we measure the GPU time taken to compress documents and read the compressed texts on the HotpotQA dataset using Qwen2.5-14B as the reader.

As shown in Table 4, all compressor methods reduce the inference time for the reader. Among them, the extractive method RECOMP achieves the lowest compression latency due to its parallel pipeline. However, it struggles to extract the

Methods	**Compress**	**Read**	**Throughput**	**EM**
Raw Document	-	309.0 ms	3.2 Iter/s	27.9
LongLLMLingua	189.8 ms	210.9 ms	2.5 Iter/s	25.8
RECOMP	31.4 ms	222.3 ms	3.9 Iter/s	26.1
FILCO	2322.5 ms	236.3 ms	0.4 Iter/s	26.9
COMPACT	3518.6 ms	209.2 ms	0.3 Iter/s	29.0
GCR	673.3 ms	203.1 ms	1.1 Iter/s	30.4
-w/o Sampling	209.2 ms	208.4 ms	2.4 Iter/s	30.0

Table 4: Latency Analysis. We measure the GPU time taken to compress documents and read the compressed texts. We also report the throughput (examples per second) and the corresponding performance (EM).

most relevant sentences, resulting in lower performance. The abstractive method COMPACT improves model performance but exhibits higher compression latency, mainly due to its iterative generation process. In contrast, our method not only enhances model performance but also achieves significantly lower compression latency, primarily due to its shorter generation length. Additionally, it's worth noting that the compression latency of our method can be further reduced by removing the consistency sampling mechanism.

Critic Analysis In this section, we evaluate the effectiveness of Permutation Ranking in the critic sampling stage by comparing it to Single Ranking, which only samples the results once. Our experiments, using Qwen2-7B as the reader LLM, are presented in Table 5. In these results, "Win" and "Lose" refer to using the highest-ranked and lowest-ranked compressor outputs, respectively, to answer the questions. "Raw" represents the use of uncompressed documents to answer the questions.

The results show that the top-ranked compressed output significantly outperforms the bottom-ranked output, with the performance of uncompressed documents falling in between. This indicates that the data labeler LLM is effective in identifying the most useful compression results. Additionally, when comparing Single Ranking with Permutation Ranking, we observe that the top-ranked output from Permutation Ranking performs better than that of Single Ranking, while the bottom-ranked output performs worse. This suggests that Permutation Ranking is more effective at distinguishing valuable compression results, ranking high-quality outputs higher and lower-quality ones lower, thereby validating its effectiveness.

4.5 Case Study

In this section, we analyze the effectiveness of our method by examining several cases from the ASQA

Datasets	**Win**		**Raw**		**Lose**	
	EM	**F1**	**EM**	**F1**	**EM**	**F1**
			SINGLE RANKING			
NQ	37.20	45.85	33.43	42.05	29.27	37.83
ASQA	46.08	56.77	44.83	55.41	42.32	53.11
SQuAD	20.87	30.05	18.26	27.41	14.78	21.77
TriviaQA	56.88	64.84	53.27	61.36	49.06	57.76
HotpotQA	28.57	37.97	27.89	36.03	24.49	32.46
			PERMUTATION RANKING			
NQ	37.99	46.53	33.43	42.05	29.18	37.73
ASQA	47.34	59.04	44.83	55.41	41.07	51.63
SQuAD	21.16	30.63	18.26	27.41	13.33	21.00
TriviaQA	57.01	65.09	53.27	61.36	48.29	56.88
HotpotQA	28.91	38.49	27.89	36.03	22.79	30.00

Table 5: Critic Analysis. We compare the performance between Single Ranking and Permutation Ranking by utilizing the top-ranked (Win) and bottom-ranked (Lose) compression results to answer the question.

datasets, which is shown in Table 6.

As we can see, our model demonstrates several notable advantages: (1) Noise Filtering: In Case 1, for the query "When did Breaking Dawn Part 2 come out?", our model isolated the precise sentence "Part 2 was released on November 16, 2012," effectively filtering out irrelevant information. (2) Cross-Examination: In Case 2, regarding the current sheriff of Maricopa County, Arizona, the model correctly identified "Paul Penzone" by synthesizing information across multiple sentences. (3) Comprehensive Coverage: In Case 3, concerning the production timeline of the first Fast and Furious film, the compression results not only confirmed the year 2000 as the start of production but also provided its release date, offering comprehensive coverage. Overall, these cases exemplify how our compression model efficiently filters noisy information, extracts relevant information from multiple sentences, and provides comprehensive coverage to answer questions accurately and reliably.

5 Related Work

5.1 Augmented Generation

Despite advancements, Large Language Models (LLMs) can generate responses containing hallucinated facts and inaccurate information (Ji et al., 2023; Shuster et al., 2021; Zhang et al., 2023a), which undermines their reliability. To address this issue, researchers have adopted Retrieval-Augmented Generation (RAG), integrating external knowledge to enhance response accuracy (Ram et al., 2023; Shi et al., 2023; Rashkin et al., 2021;

Case 1: Noise Filtering
Original Query: When did breaking dawn part 2 come out? Retrieved Documents: **Sentence 2**: Part 2 was released on November 16, 2012. Compressor result: 2 Answer: November 16, 2012 [CORRECT]
Case 2: Cross Examination
Original Query: Who is the current sheriff of maricopa county arizona? Retrieved Documents: **Sentence 5**: Paul Penzone is the current Sheriff of Maricopa. **Sentence 8**: Paul Penzone (born March 29, 1967) is the sheriff of Maricopa County, Arizona, United States. **Sentence 9**: Penzone was elected sheriff in 2016, defeating longtime incumbent Joe Arpaio. Compressor result: 5 8 9 Answer: Paul Penzone [CORRECT]
Case 3: Comprehensive Coverage
Query: When was the first fast and furious film made? Retrieved Documents: **Sentence 4**: The film was shot in various locations within Los Angeles and southern California, from July to October 2000. **Sentence 7**: Production began in 2000, as part of an international co-production between the United States and Germany, and is set and filmed across California. **Sentence 9**: Upon its release on June 22, 2001, The Fast and the Furious grossed \$207 million from a \$38 million budget. Compressor result: 4 7 9 Answer: 2000 [CORRECT]

Table 6: Case studies of context compressing. Blue text indicates the stem, pink text indicates the effective hint, [CORRECT] indicates the judgment of whether the answer is correct.

Gao et al., 2022; Bohnet et al., 2022; Menick et al., 2022). Among existing studies, some studies propose retrieving information only once at the beginning of the generation process (Shi et al., 2023; Wang et al., 2023c; Zhang et al., 2023b; Yu et al., 2023a,c). Other works (Qian et al., 2023; Yu et al., 2023b) suggest retrieving multiple times during generation, offering flexibility in when and what to search. For example, Jiang et al. (2023c) propose retrieving when the generation contains low-confidence tokens. Ram et al. (2023) recommend refreshing the retrieved documents every n tokens, which is more effective than retrieving only once. Furthermore, Wang et al. (2023b); Asai et al. (2023); Zhao et al. (2023b) propose retrieving only when the LLM deems it necessary.

5.2 Context Compressing

Context compressing techniques (Chevalier et al., 2023; Ge et al., 2023; Jiang et al., 2023b,a; Pan et al., 2024) aims to condense retrieved documents into a more concise and relevant format. Current context compressing methods can be broadly classified into two categories: extractive approaches and abstractive approaches. Extractive methods (Xu et al., 2024; Jin et al., 2024a; Reimers and Gurevych, 2019) typically utilize retrieval methods to calculate the similarity between queries and sentences, selecting the sentences with the highest similarity as the compressed output. In contrast, abstractive methods generate summaries of the retrieved documents. For example, RECOMP (Xu et al., 2024) trains a compressor to produce summaries of retrieved content, while FILCO (Wang et al., 2023c) first identifies useful context through lexical and information-theoretic approaches before training a context-filtering model. COMPACT (Yoon et al., 2024) employs an active strategy to condense documents without losing critical information, and SKR (Qiao et al., 2024) optimizes the compression by focusing on supportiveness. LongLLMLingua (Jiang et al., 2023b) filters out less important information based on perplexity.

6 Conclusion

In this work, we propose GCR, a novel generative compression method that reformulates context compression as sentence index generation, ensuring minimal inference latency. GCR effectively models semantic interactions between sentences, prevents hallucinations during compression, and offers adaptive control over the compression rate. We conduct extensive experiments on five datasets across three knowledge-intensive tasks and the results demonstrate that GCR outperforms other compression methods, achieving both high compression rates and minimal inference latency.

Limitations

In this paper, we propose a generative compression method for retrieval-augmented generation. We acknowledge two limitations of our method:

(1) The compression operates at the fixed sentence-level granularity, which may limit its applicability in scenarios requiring finer or coarser levels of detail.

(2) Our method incurs a small amount of additional computational cost due to the constrained consistency sampling mechanism.

Ethics Statement

This work was conducted in strict compliance with the ACL Ethics Policy. All datasets and large language models (LLMs) used for evaluation are publicly available. Furthermore, our work aims to explore a context-compressing method, which can lower the inference cost of the reader LLM. We do not foresee any negative ethical impacts arising from our work.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*.

Bernd Bohnet, Vinh Q Tran, Pat Verga, Roee Aharoni, Daniel Andor, Livio Baldini Soares, Jacob Eisenstein, Kuzman Ganchev, Jonathan Herzig, Kai Hui, et al. 2022. Attributed question answering: Evaluation and modeling for attributed large language models. *arXiv preprint arXiv:2212.08037*.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. 2023. Adapting language models to compress contexts. *arXiv preprint arXiv:2305.14788*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*.

Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Y Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, et al. 2022. Rarr: Researching and revising what language models say, using language models. *arXiv preprint arXiv:2210.08726*.

Tao Ge, Jing Hu, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. 2023. In-context autoencoder for context compression in a large language model. *arXiv preprint arXiv:2307.06945*.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023. Mathprompter: Mathematical reasoning using large language models. *arXiv preprint arXiv:2303.05398*.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023a. Llmlingua: Compressing prompts for accelerated inference of large language models. *arXiv preprint arXiv:2310.05736*.

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

Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023c. Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983*.

Jiajie Jin, Yutao Zhu, Yujia Zhou, and Zhicheng Dou. 2024a. Bider: Bridging knowledge inconsistency for efficient retrieval-augmented llms via key supporting evidence. *arXiv preprint arXiv:2402.12174*.

Zhuoran Jin, Pengfei Cao, Yubo Chen, Kang Liu, Xiaojian Jiang, Jiexin Xu, Qiuxia Li, and Jun Zhao. 2024b. Tug-of-war between knowledge: Exploring and resolving knowledge conflicts in retrieval-augmented language models. *arXiv preprint arXiv:2402.14409*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL 2017*, pages 1601–1611.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

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. *TACL 2019*, pages 452–466.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo

Gutman-Solo, et al. 2022. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857.

Yucheng Li. 2023. Unlocking context constraints of llms: Enhancing context efficiency of llms with self-information-based content filtering. *arXiv preprint arXiv:2304.12102*.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*.

Jacob Menick, Maja Trebacz, Vladimir Mikulik, John Aslanides, Francis Song, Martin Chadwick, Mia Glaese, Susannah Young, Lucy Campbell-Gillingham, Geoffrey Irving, et al. 2022. Teaching language models to support answers with verified quotes. *arXiv preprint arXiv:2203.11147*.

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*.

Hongjin Qian, Zhicheng Dou, Jiejun Tan, Haonan Chen, Haoqi Gu, Ruofei Lai, Xinyu Zhang, Zhao Cao, and Ji-Rong Wen. 2023. Optimizing factual accuracy in text generation through dynamic knowledge selection. *arXiv preprint arXiv:2308.15711*.

Zile Qiao, Wei Ye, Yong Jiang, Tong Mo, Pengjun Xie, Weiping Li, Fei Huang, and Shikun Zhang. 2024. Supportiveness-based knowledge rewriting for retrieval-augmented language modeling. *arXiv preprint arXiv:2406.08116*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *arXiv preprint arXiv:2302.00083*.

Hannah Rashkin, Vitaly Nikolaev, Matthew Lamm, Lora Aroyo, Michael Collins, Dipanjan Das, Slav Petrov, Gaurav Singh Tomar, Iulia Turc, and David Reitter. 2021. Measuring attribution in natural language generation models. *arXiv preprint arXiv:2112.12870*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2023. Replug: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. *arXiv preprint arXiv:2104.07567*.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. 2022. ASQA: factoid questions meet long-form answers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 8273–8288. Association for Computational Linguistics.

Hexiang Tan, Fei Sun, Wanli Yang, Yuanzhuo Wang, Qi Cao, and Xueqi Cheng. 2024. Blinded by generated contexts: How language models merge generated and retrieved contexts when knowledge conflicts? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6207–6227.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. *CoRR*, abs/2211.09085.

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

Yike Wang, Shangbin Feng, Heng Wang, Weijia Shi, Vidhisha Balachandran, Tianxing He, and Yulia Tsvetkov. 2023a. Resolving knowledge conflicts in large language models. *arXiv preprint arXiv:2310.00935*.

Yile Wang, Peng Li, Maosong Sun, and Yang Liu. 2023b. Self-knowledge guided retrieval augmentation for large language models. *arXiv preprint arXiv:2310.05002*.

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

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

Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. 2024. Evaluating mathematical reasoning beyond accuracy. *arXiv preprint arXiv:2404.05692*.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. 2023. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. *arXiv preprint arXiv:2306.13063*.

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*.

Ryutaro Yamauchi, Sho Sonoda, Akiyoshi Sannai, and Wataru Kumagai. 2023. Lpml: llm-prompting markup language for mathematical reasoning. *arXiv preprint arXiv:2309.13078*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. 2024. Compact: Compressing retrieved documents actively for question answering. *arXiv preprint arXiv:2407.09014*.

Wenhao Yu, Hongming Zhang, Xiaoman Pan, Kaixin Ma, Hongwei Wang, and Dong Yu. 2023a. Chain-of-note: Enhancing robustness in retrieval-augmented language models. *arXiv preprint arXiv:2311.09210*.

Wenhao Yu, Zhihan Zhang, Zhenwen Liang, Meng Jiang, and Ashish Sabharwal. 2023b. Improving language models via plug-and-play retrieval feedback. *arXiv preprint arXiv:2305.14002*.

Zichun Yu, Chenyan Xiong, Shi Yu, and Zhiyuan Liu. 2023c. Augmentation-adapted retriever improves generalization of language models as generic plug-in. *arXiv preprint arXiv:2305.17331*.

Jiaxin Zhang, Zhuohang Li, Kamalika Das, Bradley Malin, and Sricharan Kumar. 2023a. Sac3: Reliable hallucination detection in black-box language models via semantic-aware cross-check consistency: Reliable hallucination detection in black-box language models via semantic-aware cross-check consistency. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15445–15458.

Yunxiang Zhang, Muhammad Khalifa, Lajanugen Logeswaran, Moontae Lee, Honglak Lee, and Lu Wang. 2023b. Merging generated and retrieved knowledge for open-domain qa. *arXiv preprint arXiv:2310.14393*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023a. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Xinran Zhao, Hongming Zhang, Xiaoman Pan, Wenlin Yao, Dong Yu, and Jianshu Chen. 2023b. Thrust: Adaptively propels large language models with external knowledge. *arXiv preprint arXiv:2307.10442*.

A Dataset Statistics

The dataset statistics used in this paper are shown in Table 7.

Settings	**NQ** (Kwiatkowski et al., 2019)	**TriviaQA** (Joshi et al., 2017)	**SQuAD** (Mallen et al., 2022)	**HotpotQA** (Yang et al., 2018)	**ASQA** (Stelmakh et al., 2022)
			Dataset statistics		
Task	Open-domain QA	Open-domain QA	Open-domain QA	Multi-hop QA	Ambiguous QA
Train Data	87,925	61,888	0	0	0
Test Data	2,000	2,000	2,000	2,000	895
			Evaluation settings		
Metrics	EM, F1	EM, F1	EM, F1	EM, F1	EM, F1
			Retrieval settings		
Corpus	Wikipedia	Wikipedia	Wikipedia	Wikipedia	Wikipedia
Retriever	DPR	DPR	DPR	DPR	DPR

Table 7: Statistics and experimental settings of different tasks/datasets.

B Baseline Details

We compare our methods with the following baselines:

- **Closed Book**: Directly use the LLM to answer the question without external documents.
- **Raw Document**: Use the original context of retrieved documents to answer the question.
- **LongLLMLingua** (Jiang et al., 2023b): A method that filters out tokens with low importance based on perplexity.
- **RECOMP** (Xu et al., 2024): A method that employs a dual encoder to select the most similar sentences from the retrieved documents.
- **FILCO** (Wang et al., 2023c): A method that removes distracting content partially supporting and irrelevant to the queries.
- **COMPACT** (Yoon et al., 2024): A method that iteratively compresses documents by actively summarizing relevant information.

C Training Details

Training Data We fine-tuned the model on the NQ and TQA datasets and then used the fine-tuned model to evaluate performance on all test datasets. Specifically, we combined the NQ (87,925 queries) and TQA (61,888 queries) datasets into a single training set.

Training Process We conducted full parameter fine-tuning during both stages.

- **Supervised Distillation Stage**: We randomly sampled 50,000 queries from the combined training set for supervised fine-tuning. The model (Qwen2-7B) was fully fine-tuned for 1 epoch with a learning rate of 3e-6 and a batch size of 8.
- **Preference Alignment Stage**: In this stage, we performed critic sampling using the remaining 99,813 queries, resulting in 81,592 valid preference pairs. The model was further fine-tuned for 2 epochs with a learning rate of 7e-7 and a batch size of 2 using Direct Preference Optimization (Rafailov et al., 2024).

Model Inference After fine-tuning on this combined training set, we tested the model on all the datasets. Therefore, we only need to conduct the data labeling once.

D Prompts

Prompt: Extraction Instruction

Instruction:
Given a list of sentences and a specific query, identify and list all sentences that are relevant to the query. Output the following three steps, without any additional information.

Analysis:
1. Analyze the Query: [Carefully analyze the query to understand what information or topic is being asked about.].
2. Analyze the Sentences: [Summarize the content related to the query. Then, identify sentences that are related to the query in any way, even if they don't answer it directly.]
3. Relevant Sentences: [List the specific indices of the relevant sentences. Format your output like: [1, 2, 3, 4]. If no sentences are relevant, output: [No relevant sentences]].

Here is an example:
Sentences:...
Query: which mode is used for short wave broadcast service?
Analysis:
1. Analyze the Query:
The query is asking for the broadcasting mode used in shortwave broadcast services. We need to identify sentences mentioning the mode, method, or anything related to shortwave broadcasting.
2. Analyze the Sentences:
The sentences discuss different aspects of radio communication, including various frequencies and broadcasting methods. Relevant information includes:
- Sentence 9: Discusses the role of shortwave frequencies in global communication, which is related to shortwave broadcasting.
- Sentence 11: Mentions shortwave bands and their applications in broadcasting and communication.
- Sentence 15: Specifies that most international broadcasters use amplitude modulation (AM) for shortwave broadcast services, which directly relates to the query.
3. Relevant Sentences: [9, 11, 15]

Sentences: {sentences}
Query:{query}
Analysis:

Prompt: Critic Instruction

Instruction:
You are tasked with evaluating multiple documents in relation to a given query and its corresponding answer. Your goal is to rank the documents based on how much valuable support they provide for addressing the query and arriving at the given answer. Focus solely on whether the documents provide the most critical information needed to answer the query, disregarding any extraneous details or context not directly relevant.

Requirements:
- Analyze the key information in each document that directly assists in answering the query.
- Compare the documents based on the relevance and significance of their content concerning the query.
- If the differences in usefulness between two or more documents are negligible, consider them equal in usefulness.
- Provide a clear and concise justification for your rankings in the analysis and provide the complete ranking list without additional strings in the result.

Output Format:
- Analysis: [Briefly explain your reasoning for the rankings, noting the key information each document provides.]
- Result: [Provide a ranked list of the documents using '>' to denote greater usefulness and '=' to denote similar usefulness. The format should be: Doc i > Doc j. If two documents are equally useful, represent it as: Doc i = Doc j.]

Input:
- Query: {query}
- Given Answer: {answer}
- Documents: {documents}

- Analysis: