FineFilter: A Fine-grained Noise Filtering Mechanism for Retrieval-Augmented Large Language Models

Anonymous ACL submission

Abstract

Retrieved documents containing noise will hinder Retrieval-Augmented Generation (RAG) from detecting answer clues, necessitating noise filtering mechanisms to enhance accuracy. Existing methods use reranking or summarization to identify the most relevant sentences, but directly and accurately locating answer clues from these large-scale and complex documents remains challenging. Unlike these document-level operations, we treat noise filtering as a sentence-level MinMax optimization problem: first identifying potential clues from multiple documents, then ranking them by relevance, and finally retaining the minimum number of clues through truncation. In this paper, we propose **FineFilter**, a novel fine-grained noise filtering mechanism for RAG, consisting of a clue extractor, a reranker, and a truncator. We optimize each module to tackle complex reasoning challenges: (1) The clue extractor first uses sentences containing the answer and similar ones as fine-tuning targets, aiming to extract **sufficient** potential clues; (2) The reranker is trained to prioritize **effective** clues based on the real feedback from the generation module, with clues capable of generating correct answers as positive samples and others as negative; (3) The truncator takes the minimum number of clues needed to answer the question (truncation point) as fine-tuning targets, and performs truncation on the reranked clues to achieve fine-grained noise filtering. Experiments on three QA datasets demonstrate that FineFilter significantly improves QA performance over baselines on both LLaMA3 and Mistral. Further analysis confirms its effectiveness in complex reasoning, robustness to unreliable retrieval, and generalization to different scenarios[1].

1 Introduction

Retrieval-Augmented Generation (RAG) (Lewis et al., 2020; Gao et al., 2023) has demonstrated impressive performance across various NLP tasks (Chen et al., 2022; Huang et al., 2023; Gao et al., 2023), but its effectiveness heavily depends on the relevance of retrieved documents (Yu et al., 2023; Zhang et al., 2024a). When retrieved documents contain noise or irrelevant information (Shi et al., 2023; Liu et al., 2024), the generation model struggles to detect answer clues because noise interferes with self-attention's ability to reason over the correct context. Therefore, it is crucial to filter out irrelevant and low-value contexts.

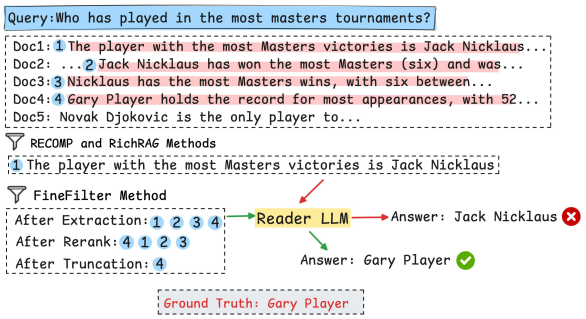

Figure 1: An illustration of the challenge in locating accurate answer clues from retrieved documents. Both baselines RECOMP and RichRAG select an incorrect clue from the 1^{st} document, whereas our FineFilter identifies the correct clue from the 4^{th} document via extraction, reranking, and truncation.

Current noise filtering methods primarily utilize reranking (Wang et al., 2025; Ke et al., 2024; Qin et al., 2024) or summarization (Xu et al., 2024; Zhu et al., 2024) models to identify the most relevant sentences, aiming at increasing the information density for RAG reasoning. The former reranks the retrieval results based on metrics such as answer contribution or user preference (Zhu et al., 2023). The latter retains the query-relevant sentences through summarization models. However, directly and accurately locating answer clues from the retrieved documents remains challenging, especially in complex reasoning scenarios. As shown

[1]Our code is available at https://anonymous.4open.science/r/FineFilter-5BE0

in Figure 1, all the five documents retrieved by the retriever contain query-relevant information. Both baselines RichRAG (Wang et al., 2025) and RECOMP (Xu et al., 2024) select the relevant sentences from the 1^{st} document, yet fail to generate correct answers. This is because these documents contain a multitude of seemingly relevant but unhelpful noisy information. Such document-level filtering is too coarse and struggles to capture effective answer clues precisely. Therefore, a fine-grained operation is required to retain **sufficient** and **effective** context for RAG.

We treat fine-grained noise filtering as a sentence-level MinMax optimization problem. First, we leverage contextual information to identify potential answer clues, which form the maximal subset capable of answering the question. Then, we carefully compare and rerank these clues based on their completeness and relevance to the query in order to move effective clues to the forefront. Finally, we retain only the most essential clues through truncation, with the goal of minimizing the necessary contexts for RAG. As shown in Figure 1, our approach first identifies the potential clues with a red background, then reranks these clues, ultimately placing the correct answer clue at the top. Notably, the last three clues are redundant and should be filtered out to improve the information density of the reasoning clues for RAG.

In this paper, we propose a novel fine-grained noise filtering mechanism for RAG, named FineFilter, consisting of a clue extractor, a reranker, and a truncator. It leverages the clue extractor and the reranker to provide sufficient and effective reasoning clues to the generation model while employing the truncator to filter noise to reduce reasoning contexts. We design three optimization strategies for each module to tackle complex reasoning challenges: (1) The clue extractor uses all sentences containing the answer and their similar sentences based on K-Nearest Neighbors (KNN) clustering (Guo et al., 2003; Peterson, 2009) as fine-tuning targets, since we find that RAG requires more relevant contextual information to reason the correct answer for multi-hop questions. Thus, the fine-tuned extractor can extract **sufficient** potential clues for complex reasoning. (2) The reranker is trained to prioritize **effective** clues based on the real feedback from the generation module, with clues capable of generating correct answers as positive samples and others as negative. (3) The truncator takes the minimal number of clues (truncation point) required for RAG to generate correct answers as the fine-tuning target. Based on the predicted point, the reranked clues are truncated to achieve fine-grained noise filtering.

Experimental results on three open-domain QA datasets (NQ, TriviaQA, and HotpotQA) show that FineFilter, whether based on LLaMA3 or Mistral, consistently outperforms baseline models in terms of performance while significantly reducing the context required for inference. Further analysis confirms its effectiveness in complex reasoning, robustness to unreliable retrieval, and generalization to different scenarios.

The innovations in this paper are as follows:

- We frame noise filtering as a sentence-level MinMax optimization, where the extractor and reranker gather sufficient and effective reasoning clues, while the truncator filters out noise to improve reasoning density.
- Three strategies tackle complex reasoning: KNN-based extractor gathers sufficient relevant context, while reranker and truncator adapt quickly and effectively to RAG systems using generator feedback.
- Experiments on three datasets show that filtering out unimportant noisy sentences enhances inference performance.

2 Related Work

Retrievers often fetch noisy content, reducing output accuracy, while overly long contexts further hinder model efficiency (Xia et al., 2024). To address these challenges, some researchers utilize reranking methods to prioritize more relevant sentences. RichRAG (Wang et al., 2025) uses a generative list-wise ranker to generate and rank candidate documents, ensuring the answer is comprehensive and aligns with the model's preferences. Ke et al. (2024) propose a novel bridge mechanism to optimize the connection between retrievers and LLMs in retrieval-augmented generation, improving performance in question-answering and personalized generation tasks. CPC (Liskavets et al., 2025) ranks sentence relevance with context-aware embeddings. However, reranking sentences may disrupt the original logical structure of the document and generate unfaithful clues.

Other researchers utilize abstractive or extractive summarization models to identify query-relevant answer clues. Xu et al. (2024) propose leveraging

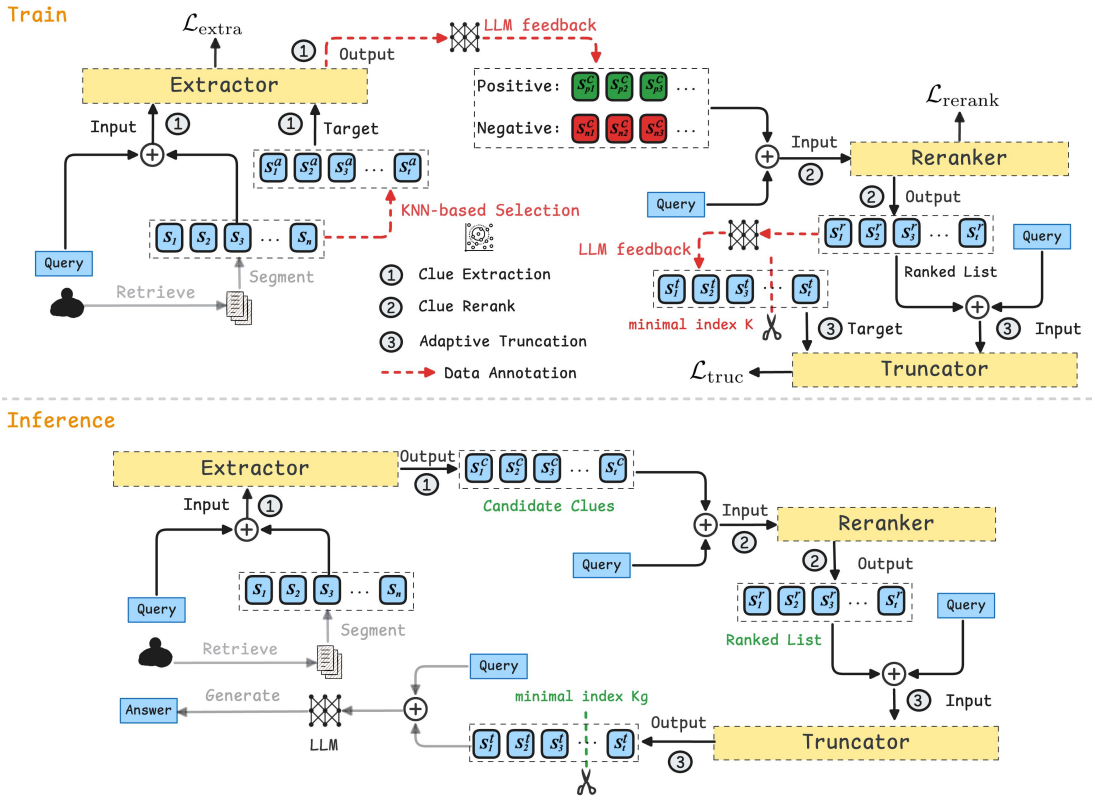

Figure 2: The architecture of FineFilter consists of three modules: clue extractor, reranker, and truncator. The top displays their training strategies and annotated data, while the bottom shows the noise filtering during inference.

LLMs as abstractive filters to compress retrieved text by targeting the most relevant sentences. Zhu et al. (2024) use the information bottleneck principle to balance conciseness and correctness, but this incurs significant training complexity. Xu et al. (2024); Wang et al. (2023) propose extractive filters to select relevant sentences, which reduce irrelevant information but may cause over-compression and harm accuracy. Li et al. (2023) enhance inference efficiency by removing redundant content based on self-information, though this may affect semantic coherence. AdaComp (Zhang et al., 2024b) adaptively adjusts document-level compression but applies relatively mild compression overall.

3 Problem Formulation

Given a query q and a set of retrieved documents $\mathcal{D} = \{d_1, \ldots, d_n\}$, where each document d_i consists of a set of sentences $\mathcal{S}_i = \{s_1^i, \ldots, s_{n_i}^i\}$, n_i is the number of sentences in d_i. The objective of the noise filtering task is to identify an optimal subset $\mathcal{S}^* \subseteq \bigcup_{i=1}^{n} \mathcal{S}_i$ such that a language model f_θ generates the correct answer y for the query q with the highest probability. The optimal subset $\mathcal{S}^*$ can be determined by the following MinMax optimization:

$$\mathcal{S}^* = \arg\min |\mathcal{S}^{'}|,$$
$$\mathcal{S}^{'} = \underset{\mathcal{S} \subseteq \bigcup_{i=1}^{n} \mathcal{S}_i}{\arg\max}\ f_\theta(y|\mathcal{S}, q),$$

where $\mathcal{S}^{'}$ is the subset that is most capable of producing the correct answer, and $|\mathcal{S}^{'}|$ is the number of sentences in $\mathcal{S}^{'}$. The selection of $\mathcal{S}^*$ should dynamically adapt to the real feedback of a RAG system to balance informativeness and conciseness. The problem can be formalized as an NP-hard combinatorial optimization problem (Wu et al., 2023), selecting the smallest, most relevant answer clues from a large set of documents to improve answer accuracy.

4 Methodology

In this section, we introduce FineFilter, a three-stage noise filtering mechanism for RAG, as illustrated in Figure 2. FineFilter comprises three modules: a clue extractor, a clue reranker, and an adaptive truncator. The clue extractor selects potential answer clues from multiple documents by maximizing information gain, reducing the search space and enhancing candidate relevance. The reranker

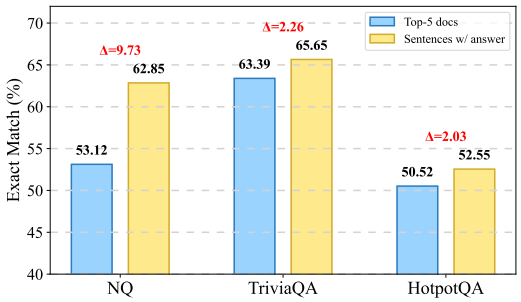

Figure 3: The Exact Match performance of LLaMA3-8B-Instruct on three QA datasets, comparing Top-5 retrieved documents with all sentences containing the ground-truth answer from those documents, regardless of query relevance.

then applies pairwise loss to prioritize the most relevant clues. Finally, the adaptive truncator selects the minimal necessary context to increase inference density, thus improving answer accuracy.

4.1 Clue Extractor

The clue extractor aims to identify potential answer clues from multiple documents to construct a smaller, query-relevant candidate set, thereby reducing the search space. As shown in Figure 3, we compare downstream performance using answer-containing sentences versus full retrieved documents. Filtering out low-value content enhances RAG reasoning. While not all answer-containing sentences are query-relevant, they approximate the maximal subset capable of answering queries and thus serve as a suitable optimization target.

To guide the extraction of informative sentences, we first introduce the concept of information gain. Given a query q and a set of candidate sentences $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$, the information gain $\text{IG}(q, s_i)$ of sentence s_i is defined as:

$$\text{IG}(q, s_i) = H(q) - H(q \mid s_i),$$

where $H(q)$ denotes the entropy of the query q, quantifying its inherent uncertainty; and $H(q \mid s_i)$ represents the uncertainty of the query given the sentence s_i. In question answering tasks, information gain reflects how much a sentence reduces uncertainty about the query. *Typically, sentences containing the answer directly reduce the unresolved part of the query, helping the model better focus on the core intent and improving the accuracy of the downstream generation module.*

Based on the concept of information gain, we first extract sentences from the retrieved document collection that contain the ground-truth answer as targets for extraction. Given a query q, the ground-truth answer y and retrieved sentences $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$, the answer-containing sentences are defined as:

$$\mathcal{S}^a = \{s_j | y \sqsubseteq s_j, s_j \in \mathcal{S}\},$$

where $y \sqsubseteq s_j$ indicates that y is a substring of s_j.

Then, we fine-tune an LLM as the clue $Extractor$ to generate answer-containing sentences $\mathcal{S}^a$ based on the query q and the retrieved sentences $\mathcal{S}$ with a specific prompt (see Appendix B.2). The loss function of $Extractor$ model is defined as:

$$\mathcal{L}_{\text{extra}} = -\log P_\theta(\mathcal{S}^a | q, \mathcal{S}).$$

Finally, the clue extractor is capable of generating the potential candidate clues based on the user query and retrieved documents during inference:

$$\mathcal{S}^c = Extractor(q, \mathcal{S}).$$

KNN-based Extraction We observe that answer-containing sentences significantly improve performance on simple QA datasets, i.e., NQ, but yield smaller gains on complex ones, i.e., TriviaQA and HotpotQA, as shown in Figure 3. Therefore, we propose a KNN-based strategy for extracting semantically similar sentences in complex reasoning scenarios. For simple questions, we directly select answer-containing sentences as the extractor's optimization targets, as they provide essential information and help reduce uncertainty. For more complex questions, we first select sentences containing the answer and then further select sentences semantically similar to these answer-containing sentences using the KNN method. Although these sentences may not contain the answer directly, they offer contextual clues that help the model better interpret the question and produce a more accurate response. We use both answer-containing and KNN-based similar sentences as the extractor's optimization targets, allowing the extractor to adapt to queries of varying complexity while improving accuracy and reducing necessary contexts.

4.2 Clue Reranker

The clue extractor often selects sentences with multiple relevant clues of varying importance, necessitating reranking. We address this by training a reranker with pairwise loss to prioritize the most relevant sentences.

Training We use the real RAG-generated feedback to annotate the training data for the reranker, as the QA performance on complex questions heavily depends on the characteristics of the generation module. First, we pair each of the extracted clue sentences $s_j^c \in \mathcal{S}^c$ with the query q as (s_j^c, q), where sentence s_j^c that enables the downstream generation module to produce the correct answer for q is considered as positive sample s_{positive}, while other sentences are treated as negative samples s_{negative}[2]. The $Reranker$ aims to minimize the following pairwise loss function (Karpukhin et al., 2020) to improve relevance ranking:

$$\mathcal{L}_{\text{rerank}} = -\log \frac{e^{\text{sim}(q, s_{\text{positive}})}}{e^{\text{sim}(q, s_{\text{positive}})} + e^{\text{sim}(q, s_{\text{negative}})}},$$

where $\text{sim}(q, *)$ represents the semantic similarity between the query q and the sentence $*$ by $Reranker$ model. Minimizing this loss function enables the $Reranker$ model to effectively identify and prioritize the most relevant clues.

Inference Given the query q and the extracted sentences $\mathcal{S}^c$, the $Reranker$ model calculates the relevance score between every sentence $s_j^c \in \mathcal{S}^c$ and query q. The reranked clues are defined as:

$$\mathcal{S}^r = Reranker(q, \mathcal{S}^c).$$

4.3 Adaptive Truncator

The adaptive truncator aims to capture the minimal necessary clues based on the complexity of the question and the documents, ensuring sufficient clues for accurate answer generation.

Training To determine the optimal clues subset $\mathcal{S}^t$ for each query q, we perform data annotation based on the reranked answer clues $\mathcal{S}^r$ obtained from the previous reranking step. Given a query q and its reranked clues $\mathcal{S}^r = \{s_1^r, \ldots, s_n^r\}$, the objective is to identify the smallest subset $\mathcal{S}^t$ such that the RAG system's generation model M can generate the correct answer y based on q and $\mathcal{S}^t$. We define $D_k = \{s_1^r, \ldots, s_k^r\}$, where $1 \leq k \leq n$. The performance on each subset D_k is evaluated by checking if the generation model's output $M(q, D_k)$ matches the ground truth y. The correctness condition is defined as:

$$\text{Correct}(q, D_k) = \begin{cases} 1, & \text{if } M(q, D_k) = y \\ 0, & \text{otherwise} \end{cases}.$$

[2]If no candidate clues can generate the correct answer, or if all samples can generate the correct answer, the sample will be removed from the annotated data.

Since the reranker cannot guarantee that the most relevant sentences are always ranked first, especially for complex questions, we iterate over the subsets from largest to smallest, starting with D_n and continuing to D_1. The optimal subset $\mathcal{S}^t$ is the smallest subset that generates the correct answer:

$$\mathcal{S}^t = \{s_1^r, \ldots, s_K^r\},$$
$$K = \arg\min_k \{k \mid \text{Correct}(q, D_k) = 1\}.$$

If the RAG system cannot generate a correct answer from any subset, then $\mathcal{S}^t = \emptyset$, indicating no subset suffices. This method ensures the use of minimal necessary context $\mathcal{S}^t$.

During the model training stage, we fine-tune an LLM based on the data annotations. The $Truncator$ is trained to predict the smallest index K of $\mathcal{S}^r$ that needed to answer the query:

$$\mathcal{L}_{\text{truc}} = -\log P_\theta(K | q, \mathcal{S}^r).$$

Inference During inference, given a new query q and its reranked sentences $\mathcal{S}^r$, the $Truncator$ predicts the minimal index K_g and truncates $\mathcal{S}^r$ to $\mathcal{S}^t = \{S_1^r, \ldots, S_{K_g}^r\}$. Finally, the generation module of RAG concatenates the query q with the filtered answer clues $\mathcal{S}^t$ as a prompt (see Appendix B.1) to reason the answer.

5 Experiments

5.1 Experimental Setup

Datasets We evaluate our method on three QA benchmark datasets: Natural Questions (NQ) (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017) and HotpotQA (Yang et al., 2018). We utilize the adversarial Dense Passage Retriever (DPR) (Karpukhin et al., 2020) to retrieve the Top-5 passages from the full Wikipedia passages for each question in these datasets.

Evaluation Metrics We evaluate answer quality on three open-domain QA datasets using Exact Match (**EM**) for strict correctness and **F1** score for partial overlap. To assess computational cost, we report compression ratio (**CR**) and inference throughput (**TP**) (Cao et al., 2024; Hwang et al., 2024) on a single A6000-48G GPU. **CR** is defined as the ratio of original to compressed context length, and **TP** refers to the number of examples the generator can process or generate per second during inference.

Method	NQ				TriviaQA				HotpotQA			
	EM	F1	CR	TP	EM	F1	CR	TP	EM	F1	CR	TP
Closed-book	26.98	62.51	-	-	30.54	68.86	-	-	19.96	55.84	-	-
Retrieval without Filtering												
Top-1 document	36.81	69.21	5.17×	2.17	42.74	77.13	5.32×	3.11	25.54	60.09	4.83×	3.11
Top-5 documents	40.21	70.95	1.0×	1.69	48.32	80.16	1.0×	2.90	25.07	59.57	1.0×	1.82
LLAMA3-8B-INSTRUCT												
Retrieval with Filtering												
RECOMP(Xu et al., 2024)	37.12	69.43	11.97×	3.54	43.41	77.61	10.91×	3.25	24.59	59.26	12.95×	4.97
LongLLMLingua(Jiang et al., 2024)	36.96	69.25	4.56×	1.97	47.56	79.15	4.18×	3.04	24.31	58.93	4.45×	3.39
FILCO (Wang et al., 2023)	32.43	64.78	17.43×	**3.82**	38.96	74.14	13.93×	3.47	20.12	56.03	11.77×	5.39
BottleNeck (Zhu et al., 2024)	39.72	70.14	14.32×	3.36	48.16	79.83	**21.26×**	4.32	25.64	60.23	13.21×	5.51
Ours	**42.17**	**71.31**	**19.56×**	3.72	**48.81**	**80.33**	20.77×	**4.91**	**26.47**	**61.15**	**14.37×**	**5.73**
MISTRAL-7B-INSTRUCT												
Retrieval with Filtering												
RECOMP(Xu et al., 2024)	36.95	69.25	13.83×	3.25	43.39	77.51	10.91×	3.17	24.34	59.16	7.24×	4.35
LongLLMLingua(Jiang et al., 2024)	37.45	69.67	4.09×	1.58	47.84	79.23	4.31×	3.01	24.05	58.75	4.22×	3.36
FILCO (Wang et al., 2023)	32.59	64.83	16.35×	3.09	38.47	73.87	12.83×	3.31	21.34	56.91	13.00×	4.73
BottleNeck (Zhu et al., 2024)	39.48	70.05	12.53×	3.01	48.03	79.97	15.24×	4.28	25.47	59.97	11.06×	4.75
Ours	**41.93**	**71.12**	**17.43×**	**3.47**	**48.64**	**80.21**	**16.49×**	**4.49**	**26.03**	**60.78**	**14.89×**	**4.77**

Table 1: Experimental results on NQ, TriviaQA, and HotpotQA using two base models: LLaMA3-8B-Instruct and Mistral-7B-Instruct.

Implementation Details We use LLaMA3-8B-Instruct (Dubey et al., 2024) and Mistral-7B-Instruct (Jiang et al., 2023) as backbone LLMs. We fine-tune the two models with LORA (Hu et al., 2021) as the clue extractor and adaptive truncator for 16 epochs on a single A6000-48G GPU. The initial learning rate is set to 5e-4, and the batch size is set to 4. We select the best model based on the performance of the validation set. For clue reranker, we implement Sentence-BERT (Reimers and Gurevych, 2020) using distilbert-base-uncased[3]. In the final generation phase, we utilize the LLaMA2-7B (Touvron et al., 2023) model. More details can be seen in Appendix A.

5.2 Baselines

We consider three baseline strategies:

Without Filtering (i) **Closed-book**, which relies solely on the generator's parametric knowledge; (ii) **Top-1**, which uses only the highest-ranked document for generation; (iii) **Top-5**, which concatenates the top five retrieved documents as input.

Extractive Methods (i) **RECOMP** (Xu et al., 2024), which employs a fine-tuned cross-encoder to select salient sentences through dense retrieval; (ii) **LongLLMLingua**(Jiang et al., 2024), which prunes irrelevant tokens in long contexts via a dynamic programming algorithm guided by question-aware perplexity scores.

Abstractive Methods (i) **FILCO** (Wang et al., 2023), which trains a context filtering model to dynamically select key sentences and jointly learns with the generator for end-to-end distillation; (ii) **BottleNeck** (Zhu et al., 2024), which employs reinforcement learning and information bottleneck theory to improve both filtering and generation.

5.3 Main Results

The comparison results on three QA datasets are shown in Table 1. The results indicate the following: (i) **RAG improves downstream task performance across all datasets.** Using Top-5 documents generally outperforms Top-1, indicating that incorporating more contextual information improves model performance; (ii) **Noise filtering is crucial for further improving performance.** Across multiple datasets, filtering methods significantly reduce context length while preserving performance close to that of Top-5 documents, effectively removing irrelevant information and enhancing accuracy; (iii) **FineFilter outperforms baselines across multiple models and datasets.** FineFilter consistently outperforms all filtering baselines across LLaMA3 and Mistral, i.e., it improves EM by 6% and CR by 37% over BottleNeck on NQ with LLaMA3; (iv) **FineFilter performs remarkably better than Top-5 documents on complex multi-hop tasks.** FineFilter shows the largest im-

[3] https://huggingface.co/distilbert/distilbert-base-uncased

Method	LLaMA2-7B (NQ)		Flan-T5-Large (HotpotQA)	
	EM	F1	EM	F1
FineFilter	**42.17**	**71.31**	**23.79**	**58.44**
w/o clue extractor	39.70	70.13	20.35	56.31
w/o clue reranker	41.64	71.05	22.94	57.79
w/o adaptive truncator	42.03	71.17	22.62	57.41

Table 2: Ablation study on NQ and HotpotQA test set. We use LLaMA2-7B and Flan-T5-Large as the generators, respectively.

Method	EM ↑	F1 ↑	Latency (s.) ↓
LongLLMLingua	36.96	69.25	3.43
BottleNeck	39.72	70.14	3.63
RECOMP	37.12	69.43	**0.97**
FILCO	32.43	64.78	2.64
FineFilter (Ours)	**42.17**	**71.31**	3.18

Table 3: Latency Analysis. We report end-to-end inference latency and QA performance (EM, F1) on NQ test set using LLaMA3-8B-Instruct. Experiments are conducted on a single A6000 GPU.

provement on HotpotQA, with a 5.4% EM gain over Top-5 using LLaMA3, followed by a 4.8% on NQ and 1.0% on TriviaQA, underscoring its superior performance in handling multi-hop QA.

6 Analysis

In this section, we will conduct the ablation study, system latency evaluation, robustness analysis and generalization of FineFilter. Additional analysis can be seen in Appendix C and D.

6.1 Ablation Study

To explore the impact of different components, we use LLaMA3-8B-Instruct as the base LLM and introduce the following variants of FineFilter for ablation study: 1) *w/o clue extractor.* directly uses LLaMA3-8B-Instruct without fine-tuning for clue extraction; 2) *w/o clue reranker.* kips reranking and retains the original sentence order; 3) *w/o adaptive truncator.* disables adaptive truncation of the reranked clues. As shown in Table 2, removing any single component leads to a performance drop in both EM and F1, confirming that all modules are essential to the overall effectiveness. Additional ablation results are presented in Appedix C.

6.2 System Latency Evaluation

To assess the overall system efficiency, we measure the average total inference time (Latency, in seconds) required for all components in each method to process a single sample on the NQ test set. As shown in Table 3, RECOMP has the lowest latency

Dataset	Method	EM	F1
NQ	Direct Extraction	39.41	69.98
	Fine-tuned Extraction	**41.43**	**71.01**
TriviaQA	Direct Extraction	45.54	78.01
	Fine-tuned Extraction	**48.36**	**80.16**
HotpotQA	Direct Extraction	24.71	59.29
	Fine-tuned Extraction	**26.01**	**60.73**

Table 4: Performance comparison between Direct Extraction and Fine-tuned Extraction across three QA datasets using LLaMA3-8B-Instruct.

but worse EM. FineFilter achieves the highest EM with only 0.52s more latency than the baselines, owing to its lightweight reranker and truncator that add minimal overhead, and the clue extractor with comparable complexity to existing methods. Thus, FineFilter offers a better trade-off between accuracy and efficiency.

6.3 Direct vs. Fine-tuned Extraction

We analyze the effect of fine-tuning on Clue Extraction by comparing two approaches: 1) *Direct Extraction*. Using LLM without fine-tuning to extract clue sentences from retrieved documents based on the given prompt (see Appendix B.2); 2) *Fine-tuned Extraction*. Using fine-tuned LLM to select answer-containing sentences. As shown in Table 4, Fine-tuned Extraction consistently outperforms Direct Extraction by leveraging task-specific knowledge to identify more relevant sentences.

6.4 Threshold of KNN-Based Extraction

We assess the KNN-based extraction by varying the threshold, which controls the cosine similarity to answer-containing sentences. A threshold of 0 selects only the answer sentences, while higher thresholds allow semantically similar sentences to expand the context with additional relevant information. As shown in Figure 4, we compare the model's performance at different threshold values. For simple questions such as NQ, the KNN extraction strategy does not improve performance, as answers can typically be obtained directly from sentences containing the answers. For more complex questions such as HotpotQA and TriviaQA, the KNN strategy improves performance at lower thresholds but declines at higher thresholds due to increased noise. It is worth noting that KNN-based extraction plays a supplementary role. While the threshold setting impacts FineFilter's performance, it does not change the experimental result that FineFilter outperforms the best noise filtering baseline.

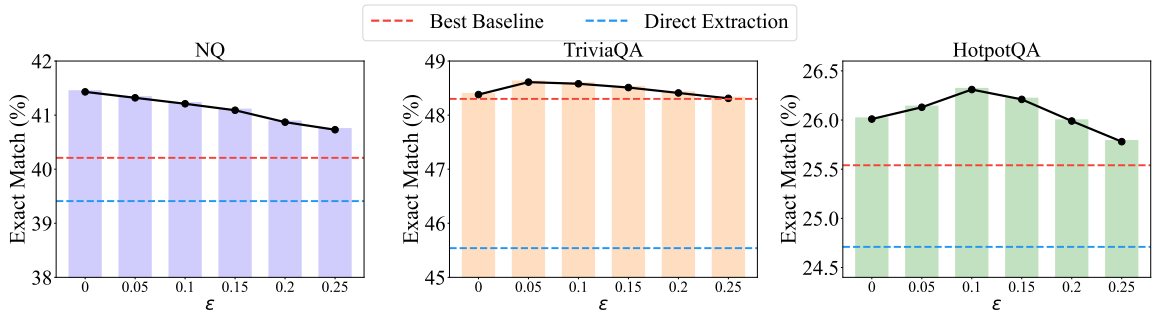

Figure 4: An illustration of clue extraction performance using LLaMA3-8B-Instruct on NQ, TriviaQA, and HotpotQA. The x-axis shows the KNN threshold and higher values introduce more contextual sentences.

Dataset	Recall-1 (%)	Hit-2@1 (%)	Recall-3 (%)
NQ	81.96	77.16	80.17
TriviaQA	91.94	83.25	85.28
HotpotQA	84.54	77.90	80.54

Table 5: Cascading errors analysis of extractor (Recall-1), reranker (Hit-2@1) and truncator (Recall-3) on three datasets using LLaMA3-8B-Instruct.

Method	EM	F1
RECOMP	15.51	51.33
FILCO	14.47	50.46
Ours (*w/o Truncator*)	15.63	51.42
Ours (*w/ Truncator*)	**16.20**	**51.94**

Table 6: Results under unreliable retrieval on HotpotQA test set using LLaMA3-8B-Instruct.

Method	ROUGE-1	ROUGE-2	ROUGE-L
RECOMP	0.3051	0.1022	0.1668
FILCO	0.2743	0.0976	0.1428
BottleNeck	0.3708	0.1529	0.2158
LongLLMLingua	0.4082	0.1775	0.2369
Top-1	0.3532	0.1275	0.2103
Top-5	0.3769	0.1546	0.2234
FineFilter (Ours)	**0.4211**	**0.1842**	**0.2556**

Table 7: Comparison of cross-task generalization ability across different baselines, evaluated using ROUGE-1, ROUGE-2, and ROUGE-L.

6.5 Robustness Analysis

Cascading Errors Resilience Analysis For high-quality retrieval scenarios, we conduct experiments on NQ test set with gold-standard answers to evaluate cascading errors. **Recall-1** measures whether the extractor can retrieve the gold-standard answer. **Hit-2@1** measures the probability that the reranker places the gold-standard answer as Top-1. **Recall-3** measures whether the truncator can continue to retain the gold-standard answer. As shown in Table 5, FineFilter maintains a low error rate in the filtering components, indicating that cascading errors are kept within a controllable range.

Performance under Unreliable Retrieval The truncator not only shortens context but also helps filter out unreliable contents. We conduct experiments on samples from HotpotQA test set without gold-standard answers to simulate unreliable retrieval. As shown in Table 6, FineFilter's truncator effectively suppresses noise and prevents answer degradation. This is because FineFilter is trained to allow empty outputs, whereas baseline models perform poorly under unreliable retrieval conditions, as they must choose an output answer clue.

6.6 Cross-Task Generalization

To evaluate FineFilter's generalization, we test it on 1,200 random samples from the Conversational Multi-Doc QA dataset [4], after training on HotpotQA. As shown in Table 7, FineFilter maintains strong performance in this new dataset, demonstrating its generalization across diverse QA tasks.

7 Conclusion

We propose FineFilter, a fine-grained noise filtering mechanism to enhance performance and efficiency in RAG. By framing noise filtering as a sentence-level MinMax optimization problem, it effectively identifies relevant clues in complex reasoning scenarios. Its three optimized modules use KNN clustering to gather sufficient context and retain key clues based on generator feedback. Experiments show FineFilter outperforms baselines in performance and efficiency on three QA datasets. Future work can explore adaptive noise filtering that dynamically adjusts based on query complexity or retrieval quality for complex reasoning tasks.

[4] https://sites.google.com/view/wsdm24-docqa

Limitations

Although FineFilter has made significant progress in clue extraction and computational efficiency, there is still the issue of system transferability. FineFilter fine-tunes the LLM based on downstream generator feedback, and if a new generative LLM is adopted, the filtering modules need to be retrained. This tight coupling results in increased transfer costs for the system.

Ethics Statement

This paper presents FineFilter, a fine-grained noise filtering mechanism that formulates RAG noise reduction as a sentence-level MinMax optimization problem. It extracts sufficient supporting clues, reranks them based on answerability, and adaptively truncates redundancy to enhance both answer accuracy and inference efficiency. Throughout this research, we have adhered to ethical guidelines to ensure the integrity and fairness of our work. All experiments are conducted using publicly available datasets, and the retrieval corpus is based on open-domain sources such as Wikipedia, ensuring transparency and reproducibility.

FineFilter does not involve any personally identifiable information or private user data. The proposed methods are designed to enhance model efficiency and robustness without reinforcing harmful biases. Nevertheless, as with all RAG systems, the quality and neutrality of retrieved content can influence outputs. Future work may incorporate bias detection and content verification modules to further improve fairness and reliability.

References

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

Jiangui Chen, Ruqing Zhang, Jiafeng Guo, Yixing Fan, and Xueqi Cheng. 2022. Gere: Generative evidence retrieval for fact verification. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2184–2189.

Hanxing Ding, Shuchang Tao, Liang Pang, Zihao Wei, Liwei Chen, Kun Xu, Huawei Shen, and Xueqi Cheng. 2025. Revisiting robust rag: Do we still need complex robust training in the era of powerful llms? *arXiv preprint arXiv:2502.11400*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

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

Gongde Guo, Hui Wang, David Bell, Yaxin Bi, and Kieran Greer. 2003. Knn model-based approach in classification. In *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings*, pages 986–996. Springer.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Qiushi Huang, Shuai Fu, Xubo Liu, Wenwu Wang, Tom Ko, Yu Zhang, and Lilian Tang. 2023. Learning retrieval augmentation for personalized dialogue generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2523–2540, Singapore. Association for Computational Linguistics.

Taeho Hwang, Sukmin Cho, Soyeong Jeong, Hoyun Song, SeungYoon Han, and Jong C Park. 2024. Exit: Context-aware extractive compression for enhancing retrieval-augmented generation. *arXiv preprint arXiv:2412.12559*.

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. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

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

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)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

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.

Zixuan Ke, Weize Kong, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. 2024. Bridging the preference gap between retrievers and LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10438–10451, Bangkok, Thailand. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

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

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

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

Leif E Peterson. 2009. K-nearest neighbor. *Scholarpedia*, 4(2):1883.

Zhen Qin, Rolf Jagerman, Kai Hui, Honglei Zhuang, Junru Wu, Le Yan, Jiaming Shen, Tianqi Liu, Jialu Liu, Donald Metzler, Xuanhui Wang, and Michael Bendersky. 2024. Large language models are effective text rankers with pairwise ranking prompting. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1504–1518, Mexico City, Mexico. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

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

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Shuting Wang, Xin Yu, Mang Wang, Weipeng Chen, Yutao Zhu, and Zhicheng Dou. 2025. RichRAG: Crafting rich responses for multi-faceted queries in retrieval-augmented generation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 11317–11333, Abu Dhabi, UAE. Association for Computational Linguistics.

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*.

Zhiyong Wu, Yaoxiang Wang, Jiacheng Ye, and Lingpeng Kong. 2023. Self-adaptive in-context learning: An information compression perspective for in-context example selection and ordering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1423–1436, Toronto, Canada. Association for Computational Linguistics.

Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. 2024. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7655–7671, Bangkok, Thailand. Association for Computational Linguistics.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. 2024. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pages 641–649.

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

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, S Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. 2023. Generate rather than retrieve: Large language models are strong context generators. In *International Conference on Learning Representations*.

Jiahao Zhang, Haiyang Zhang, Dongmei Zhang, Liu Yong, and Shen Huang. 2024a. End-to-end beam retrieval for multi-hop question answering. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1718–1731, Mexico City, Mexico. Association for Computational Linguistics.

Qianchi Zhang, Hainan Zhang, Liang Pang, Hongwei Zheng, and Zhiming Zheng. 2024b. Adacomp: Extractive context compression with adaptive predictor for retrieval-augmented large language models. *arXiv preprint arXiv:2409.01579*.

Kun Zhu, Xiaocheng Feng, Xiyuan Du, Yuxuan Gu, Weijiang Yu, Haotian Wang, Qianglong Chen, Zheng Chu, Jingchang Chen, and Bing Qin. 2024. An information bottleneck perspective for effective noise filtering on retrieval-augmented generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1044–1069, Bangkok, Thailand. Association for Computational Linguistics.

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen. 2023. Large language models for information retrieval: A survey. *arXiv preprint arXiv:2308.07107*.

A More Details of Experimental Settings

We utilize LLaMA3-8B-Instruct (Dubey et al., 2024) and Mistral-7B-Instruct (Jiang et al., 2023) as the backbone language models, both of which demonstrate excellent performance across various tasks and exhibit high flexibility during fine-tuning. We apply the LORA method (Hu et al., 2021) for fine-tuning, which is an efficient low-rank adaptation technique that significantly reduces the computational cost of parameter updates while maintaining model performance. The LORA method is applied to the clue extractor and adaptive truncator. All training is conducted on a single A6000-48G GPU, with 16 training epochs. The initial learning rate is set to 5e-4, and the batch size is 4. During training, we employ gradient accumulation to handle smaller batch sizes and improve training stability. The best model is selected based on the performance of the validation set.

During the fine-tuning phase, we train the models on the three QA datasets, i.e., NQ, TriviaQA, and HotpotQA. We employ the KNN-based sentence selection method across all datasets. The maximum number of samples is limited to 10000, and the KNN-based sentence selection varies with the ϵ value for each dataset: for NQ, ϵ is set to 0; for TriviaQA, ϵ is set to 0.05; and for HotpotQA, ϵ is set to 0.1. These adjustments ensure flexibility and accuracy in clue extraction for different datasets. Additionally, data preprocessing is accelerated during each training epoch using 16 parallel workers. The maximum input length is set to 7168 to accommodate large-scale context information.

For the clue reranker, we use Sentence-BERT (Reimers and Gurevych, 2020) with the distilbert-base-uncased model, which is effective for generating high-quality sentence embeddings to compute sentence similarity. During training, we apply the Adam optimizer with a batch size of 64, a learning rate of 2e-5, and 1000 warm-up steps. The training lasts for 4 epochs.

B Prompt

B.1 Prompt for the Generator

We use the LLaMA2-7B (Touvron et al., 2023) model as the final generator. During the generation phase, we design a specialized generator prompt to ensure that the generated answers are highly relevant to the questions. We show our prompt in Table 8. The prompt guides the model in generating accurate and concise responses based on the given question and context.

B.2 Prompt for the Clue Extractor

We show our prompt for clue extraction in Table 9, which plays a crucial role in identifying and selecting relevant information from the input documents. This prompt is designed to guide the model in extracting the most informative sentences, those most likely to contain the answer to the given question.

B.3 Prompt for the Adaptive Truncator

We show our prompt for adaptive truncation in Table 10. The prompt is designed to guide the model in optimizing context truncation based on the complexity of the question and the quality of the document, thereby improving the efficiency of the language model. Specifically, given a question and a ranked list of sentences, the model's task is to identify and retain the most relevant sentences while truncating those that are irrelevant to the question. Through this process, the model is able to maintain answer accuracy while reducing unnecessary information, thus enhancing processing efficiency.

Prompt for the Generator

```
[INST]
<<SYS>>
You are a helpful, respectful, and honest assistant. Please use the documents provided to answer the query.
Documents:
{Documents}
<</SYS>>

{Question}
[/INST]
```

Table 8: Prompt for the Generator.

C Additional Experimental Analysis

C.1 Impact of Different Rerankers

To select a more effective reranking base model, we compare BM25 (Robertson et al., 2009), BGE-rerank (Xiao et al., 2024), and Sentence-BERT(Reimers and Gurevych, 2020). As shown in Table 11, the results demonstrate that Sentence-BERT outperforms all baseline models in both EM

Prompt for the Clue Extractor

You are a highly skilled assistant specializing in extracting relevant information from provided documents. Your task is to identify and extract sentences from the documents as much as possible that are most directly useful for answering the given question. Rank the sentences in order of relevance, with the most relevant sentence listed first. Preface each sentence with its sequence number as follows:
Sentence 1:
.....
Sentence n:

Question:
`{Question}`

Documents:
`{Documents}`

Table 9: Prompt for the Clue Extractor.

Prompt for the Adaptive Truncator

You are a highly skilled assistant specializing in optimizing language model efficiency by truncating context based on question complexity and document quality. Given a question and a ranked list of sentences, identify and retain the most relevant ones while truncating the irrelevant sentences.

Question:
`{Question}`

Ranked List:
`{Ranked List}`

Table 10: Prompt for the Adaptive Truncator.

Method	EM	F1
BM25	41.51	71.03
BGE-rerank	41.73	71.06
Sentence-BERT(Ours)	**42.03**	**71.21**

Table 11: Comparison of different reranking methods on NQ test set based on LLaMA3-8B-Instruct.

	NQ		TriviaQA		HotpotQA	
Method	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**
LongLLMLingua	34.77	65.28	46.32	78.43	22.19	57.33
FILCO	31.96	64.05	37.71	73.24	19.76	55.74
RECOMP	35.12	68.71	42.96	77.39	23.91	58.19
BottleNeck	38.49	69.88	46.15	78.37	24.27	58.83
Top-1	36.81	69.21	42.74	77.13	25.54	60.09
Top-5	**40.21**	**70.95**	**48.32**	**80.16**	**25.07**	**59.57**
FineFiler (Ours)	**39.23**	**69.98**	**46.88**	**78.85**	**24.65**	**59.26**

Table 12: Performance comparison on NQ, TriviaQA, and HotpotQA under a weakened setup where FineFilter's clue extractor and adaptive truncator are replaced with Flan-T5-Large, to simulate limited clue processing capabilities.

and F1 scores, so we chose it as the reranking base model.

C.2 Weaker Extractor and Truncator

To better assess the impact of clue extractor and adaptive truncator quality on the end-to-end performance of RAG systems, we conduct an experiment in which FineFilter's clue extractor and truncator are replaced with a weaker model, Flan-T5-Large. This setup simulates scenarios where the clue processing pipeline exhibits limited reasoning or compression capabilities, enabling us to examine the extent of performance degradation under such constraints.

As shown in Table 12, FineFilter consistently outperforms the baselines across all three datasets, although it still slightly underperforms compared to the Top-5 setting. This suggests that when the downstream generator is sufficiently powerful (e.g., LLaMA2-7B), relying solely on a simple or less capable clue processing module is insufficient to fully exploit the potential of the overall system. Instead, only upstream components with strong extraction and compression capabilities can interact effectively with a powerful generator to achieve high-quality question answering.

C.3 Weaker Downstream Generator

To assess the impact of generator capacity on the overall performance of our RAG system, we re-

Method	NQ		TriviaQA		HotpotQA	
	EM	F1	EM	F1	EM	F1
LongLLMLingua	35.71	68.89	46.28	78.31	22.06	57.13
FILCO	31.79	63.88	34.59	71.77	20.79	56.71
RECOMP	36.28	69.11	41.32	76.44	22.58	57.34
BottleNeck	37.94	69.83	45.91	77.94	23.26	58.02
Top-5	34.85	68.04	46.03	78.16	22.86	57.59
FineFiler (Ours)	**38.95**	**69.91**	**47.03**	**78.98**	**23.79**	**58.51**

Table 13: Performance comparison on NQ, TriviaQA, and HotpotQA using Flan-T5-Large as the weaker generator, while keeping the clue processing pipeline unchanged.

Method	NQ	TriviaQA	HotpotQA
FineFilter	**38.95**	**47.03**	**23.79**
w/o clue extractor	36.18	44.72	20.35
w/o clue reranker	36.91	45.33	22.94
w/o adaptive truncator	38.09	46.21	22.62

Table 14: Ablation study on NQ, TriviaQA, and HotpotQA test set using EM scores. The generator is Flan-T5-Large, and the clue processing pipeline is kept unchanged.

Method	EM	CR	Latency (s.)
Ours (*Truncator: LLaMA3-8B-Instruct*)	**42.17**	**19.56**×	3.18
Ours (*Truncator: Flan-T5-Large*)	42.09	18.79×	**3.05**

Table 15: Performance of adaptive truncators with different parameter sizes on the NQ dataset.

place the stronger generator (LLaMA2-7B) with a weaker model (Flan-T5-Large), while keeping all other components and configurations unchanged. We further conduct an ablation study on three datasets, as presented in Table 13 and Table 14.

Experimental results demonstrate that replacing a strong generator with a weaker model results in performance degradation across all datasets. Nevertheless, our method consistently outperforms the Top-5 baseline. Notably, the relative contribution of the adaptive truncator becomes more pronounced under weaker generation conditions (Ding et al., 2025), particularly on reasoning-intensive datasets such as TriviaQA and HotpotQA. These findings indicate that, when the downstream generator is less capable, upstream components like the adaptive truncator play a more critical role. Therefore, in scenarios constrained by limited computational resources or smaller model sizes, well-designed filtering and compression strategies can effectively mitigate the limitations of weaker generators.

C.4 Truncators with Different Parameters

To investigate the impact of truncator size on overall system performance, we replace the default truncator with a smaller Flan-T5-Large model and evaluate the system on the NQ dataset.

As shown in Table 15, using a smaller truncator leads to a slight drop in EM and CR, but provides a marginal improvement in inference speed (0.13s faster). These results suggest that smaller truncators can offer a reasonable trade-off between performance and efficiency. Therefore, we recommend choosing the truncator model based on application-specific requirements and resource constraints.

D Case Study

We select examples from the NQ and HotpotQA datasets, covering two typical question-answering scenarios: one involving simple single-answer questions and the other involving complex multi-answer questions requiring reasoning. As shown in Table 16 and Table 17 for the NQ dataset, and Table 18 and Table 19 for the HotpotQA dataset, these examples will demonstrate the advantages and effectiveness of the FineFilter method in handling question answering tasks of varying complexity.

Question: what kind of beast is the beast from beauty and the beast
Correct Answer: a chimera

Retrieved Documents

Document 1:
Beast (Beauty and the Beast) The Beast is a fictional character who appears in Walt Disney Animation Studios' 30th animated feature film "Beauty and the Beast" (1991). He also appears in the film's two direct-to-video followups "" and "Belle's Magical World". Based on the hero of the French fairy tale by Jeanne-Marie Leprince de Beaumont, the Beast was created by screenwriter Linda Woolverton and animated by Glen Keane. A pampered prince transformed into a hideous beast as punishment for his cold-hearted and selfish ways, the Beast must, in order to return to his former self, earn the love of a

Document 2:
the arms and body of a bear, the eyebrows of a gorilla, the jaws, teeth, and mane of a lion, the tusks of a wild boar and the legs and tail of a wolf. He also bears resemblance to mythical monsters like the Minotaur or a werewolf. He also has blue eyes, the one physical feature that does not change whether he is a beast or a human. As opposed to his original counterpart, Disney gave him a more primal nature to his personality and mannerisms, which truly exploited his character as an untamed animal (i.e. alternating between walking and

Document 3:
the Beast to resemble a creature that could possibly be found on Earth as opposed to an alien. The initial designs had the Beast as humanoid but with an animal head attached as per the original fairy tale, but soon shifted towards more unconventional forms. The earlier sketches of the Beast2019s character design are seen as gargoyles and sculptures in the Beast's castle. Inspired by a buffalo head that he purchased from a taxidermy, Keane decided to base the Beast's appearance on a variety of wild animals, drawing inspiration from the mane of a lion, head of a buffalo, brow

Document 4:
the villagers. Beast (Beauty and the Beast) The Beast is a fictional character who appears in Walt Disney Animation Studios' 30th animated feature film "Beauty and the Beast" (1991). He also appears in the film's two direct-to-video follow-ups and "Belle's Magical World." Based on the hero of the French fairy tale by Jeanne-Marie Leprince de Beaumont, the Beast was created by screenwriter Linda Woolverton and animated by Glen Keane. A pampered prince transformed into a hideous beast as punishment for his cold-hearted and selfish ways, the Beast must, in order to return to his former self, earn the love of a person

Document 5:
of a gorilla, tusks of a wild boar, legs and tail of a wolf, and body of a bear. However, he felt it important that the Beast's eyes remain human. In fear that Glen Keane would design the Beast to resemble voice actor Robby Benson, Walt Disney Studios chairman Jeffrey Katzenberg did not allow Keane to see Benson during production of the film. The Beast is not of any one species of animal, but a chimera (a mixture of several animals), who would probably be classified as a carnivore overall. He has the head structure and horns of a buffalo

Table 16: An example from NQ, including Question, Correct Answer, and Top-5 Retrieved Documents.

Method	Summary	Answer
Closed-book:	-	a bear
Top-5 Documents	-	a bear
Top-1 Document	Beast (Beauty and the Beast) The Beast is a fictional character who appears in Walt Disney Animation Studios' 30th animated feature film "Beauty and the Beast" (1991). He also appears in the film's two direct-to-video followups "" and "Belle's Magical World". Based on the hero of the French fairy tale by Jeanne-Marie Leprince de Beaumont, the Beast was created by screenwriter Linda Woolverton and animated by Glen Keane. A pampered prince transformed into a hideous beast as punishment for his cold-hearted and selfish ways, the Beast must, in order to return to his former self, earn the love of a	a bear
RECOMP	Beast (Beauty and the Beast) The Beast is a fictional character who appears in Walt Disney Animation Studios' 30th animated feature film "Beauty and the Beast" (1991).	a bear
FILCO	the arms and body of a bear, the eyebrows of a gorilla, the jaws, teeth, and mane of a lion, the tusks of a wild boar and the legs and tail of a wolf.	a bear
Ours	Sentence1:The Beast is not of any one species of animal, but a chimera (a mixture of several animals), who would probably be classified as a carnivore overal Sentence2:of a gorilla, tusks of a wild boar, legs and tail of a wolf, and body of a bear	a chimera

Table 17: Case study based on an example from NQ.

Question: What writer worked on both The Ice Cream Man and and a 2007 fantasy comedy loosely based on a Donald Henkel poem?
Correct Answer: David Dobkin

Retrieved Documents

Document 1:
Ice Cream Man (film) Ice Cream Man is a 1995 American horror comedy film produced and directed by Norman Apstein, a director of pornographic films. In his first and only attempt at mainstream filmmaking, and written by Sven Davison and David Dobkin (who later wrote and directed the films "Wedding Crashers" and "Fred Claus"), and starring Clint Howard, Olivia Hussey, and David Naughton. The plot follows a deranged man recently released from a psychiatric institution who opens an ice cream factory where he begins using human flesh in his recipes. The film had an estimated 2 million budget and was

Document 2:
“Water Tower and the Turtle” won the 39th Kawabata Yasunari Prize. The Japanese Ministry of Education, Culture, Sports, Science and Technology recognized Tsumura’s work with a New Artist award in 2016. Tsumura’s writing often employs Osaka-ben, a distinctive Japanese dialect spoken in Osaka and surrounding cities. Kikuko Tsumura was born in Osaka, Japan in 1978. While commuting to school, she read science fiction novels, especially the work of William Gibson, Philip K. Dick, and Kurt Vonnegut, and began writing her own novel, “Manīta” (“Maneater”), while still a third-year university student. “Manīta” won the 21st Dazai Osamu Prize and was

Document 3:
Sentai-style shows called “Go Sukashi!” based on a character by Shoko Nakagawa (who appears in the films), and starring John Soares and Brooke Brodack. He has also published an online superhero-genre-spoofing webcomic titled “Ratfist.” In September 2012, Fox Animation optioned TenNapel’s published Graphix novel “Cardboard”, with plans for actor Tobey Maguire’s Material Pictures, graphic novelist Doug TenNapel, and the Gotham Group to be executive producers. Fox plans to have the picture developed under its WedgeWorks subsidiary. WedgeWorks director Chris Wedge (“Ice Age”) is producing, and is considering directing the film as well. TenNapel has used Kickstarter to produce a bound

Document 4:
The film industry, and his interest particularly in contemporary animated film from Eastern Europe — particularly the work of Jan Lenica, Daniel Szczechura and Walerian Borowczyck — as well as the Brothers Quay has been a marked influence on his work. He has published three novels. Weiner’s 1993 debut novel “The Museum of Love” was published by Bloomsbury UK and subsequently by Kodansha in Japan, The Overlook Press in the United States and Canada, and Belfond in France. It earned comparisons to William S. Burroughs, Céline, Jean Genet, David Lynch and Todd Haynes for its blend of surrealism and dark

Document 5:
See her idol, Eudora Welty, Flagg won first prize in the writing contest for a short story told from the perspective of an 11-year-old girl, spelling mistakes and all—a literary device that she figured was ingenious because it disguised her own pitiful spelling, later determined to be an outgrowth of dyslexia. An editor at Harper & Row approached her about expanding the story into a full-length novel. “I just burst into tears and said, ‘I can’t write a novel,’” she told “The New York Times” in 1994. “‘I can’t spell. I can’t diagram a sentence.’ He took my hand and

Table 18: An example from HotpotQA, including Question, Correct Answer, and Top-5 Retrieved Documents.

Method	Summary	Answer
Closed-book:	-	Quentin Tarantino
Top-5 Documents	-	Grady Hendrix
Top-1 Document	Ice Cream Man (film) Ice Cream Man is a 1995 American horror comedy film produced and directed by Norman Apstein, a director of pornographic films, in his first and only attempt at mainstream filmmaking, and written by Sven Davison and David Dobkin (who later wrote and directed the films "Wedding Crashers" and "Fred Claus"), and starring Clint Howard, Olivia Hussey, and David Naughton. The plot follows a deranged man recently released from a psychiatric institution who opens an ice cream factory where he begins using human flesh in his recipes. The film had an estimated 2 million budget and was	David Dobkin
RECOMP	Ice Cream Man (film) Ice Cream Man is a 1995 American horror comedy film produced and directed by Norman Apstein, a director of pornographic films.	Norman Apstein
FILCO	Ice Cream Man (film) Ice Cream Man is a 1995 American horror comedy film produced and directed by Norman Apstein, a director of pornographic films.	Norman Apstein
Ours	Sentence 1:Ice Cream Man (film) Ice Cream Man is a 1995 American horror comedy film produced and directed by Norman Apstein, a director of pornographic films. Sentence 2:in his first and only attempt at mainstream filmmaking, and written by Sven Davison and David Dobkin (who later wrote and directed the films "Wedding Crashers" and "Fred Claus"), and starring Clint Howard, Olivia Hussey, and David Naughton.	David Dobkin

Table 19: Case study based on an example from HotpotQA.