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 re-ranking 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 the potential clues from multiple documents using contextual information, then ranking them by relevance, and finally retaining the least clues through truncation. In this paper, we propose **FineFilter**, a novel fine-grained noise filtering mechanism for RAG consisting of a clue extractor, a re-ranker, and a truncator. We optimize each module to tackle complex reasoning challenges: (1) Clue extractor firstly uses sentences containing the answer and similar ones as fine-tuned targets, aiming at extracting **sufficient** potential clues; (2) Re-ranker is trained to prioritize **effective** clues based on the real feedback from generation module, with clues capable of generating correct answer as positive samples and others as negative; (3) Truncator takes the minimum clues needed to answer the question (truncation point) as fine-tuned targets, and performs truncation on the re-ranked clues to achieve fine-grained noise filtering. Experiments on three QA datasets demonstrate that FineFilter significantly outperforms baselines in terms of performance and inference cost. Further analysis on each module shows the effectiveness of our optimizations for complex reasoning [1].

1 Introduction

Retrieval-Augmented Generation (RAG) has demonstrated impressive performance across various knowledge-intensive NLP tasks (Chen et al., 2022; Huang et al., 2023; Gao et al., 2023), but its effectiveness heavily depends on the relevance of the retrieved documents (Liu et al., 2024). When retrieved documents contain noise or irrelevant information (Zhu 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.

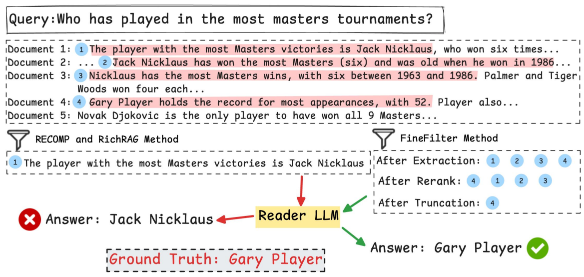

Figure 1: An illustration of the challenge in locating accurate answer clues from retrieved documents. The baseline RECOMP and RichRAG pick an incorrect clue from the 1^{th} document, whereas our FineFilter relies on the extraction, reranking, and truncation to identify the correct clue from the 4^{th} document.

Current noise filtering methods primarily utilize re-ranking (Wang et al., 2025; Ke 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 re-ranks retrieval results based on metrics such as answer contribution or user preference. 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 in Figure 1, all the five documents recalled from a retriever contain query-relevant information. Both baseline RichRAG and RECOMP

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

select the relevant sentences from the 1^{th} document, yet they produce incorrect 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 the 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 re-rank 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 reasoning costs for RAG. As shown in Figure 1, our approach first identifies the potential clues with a red background, then re-ranks 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 re-ranker, and a truncator. It leverages the clue extractor and re-ranker to provide sufficient and effective reasoning clues to the generation model while employing the truncator to filter noise to reduce reasoning costs. We design three optimization strategies for each module to tackle complex reasoning challenges: (1) Clue extractor uses all sentences containing the answer and their similar sentences based on KNN clustering 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) Re-ranker 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 while others as negative. (3) 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 re-ranked clues are truncated to achieve fine-grained noise filtering.

We conduct experiments on three open-domain question answering datasets, i.e., NQ, TriviaQA, and HotpotQA. The experimental results show that, whether based on LLaMA3 or Mistral, FineFilter outperforms the baseline models in terms of performance while significantly reducing the context required for inference. Further analysis of each module demonstrates the effectiveness of our optimization strategies for complex reasoning. The innovations in this paper are as follows:

- We frame noise filtering as a sentence-level MinMax optimization, where the extractor and re-ranker gather sufficient and effective reasoning clues, while the truncator filters out noise to reduce reasoning costs.
- Three strategies tackle complex reasoning: KNN-based extractor gathers sufficient relevant context, while re-ranker 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 and efficiency.

2 Related Work

Retrievers often fetch noisy content, reducing output accuracy, while overly long contexts further hinder model efficiency. To address these challenges, some researchers utilize re-ranking 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) proposes 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. 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 LLMs as abstractive filters to compress retrieved text by targeting the most relevant sentences. Zhu et al. (2024) apply the information bottleneck principle to filter noise, striving to strike a balance between conciseness and correctness. Despite its potential benefits, this method is associated with

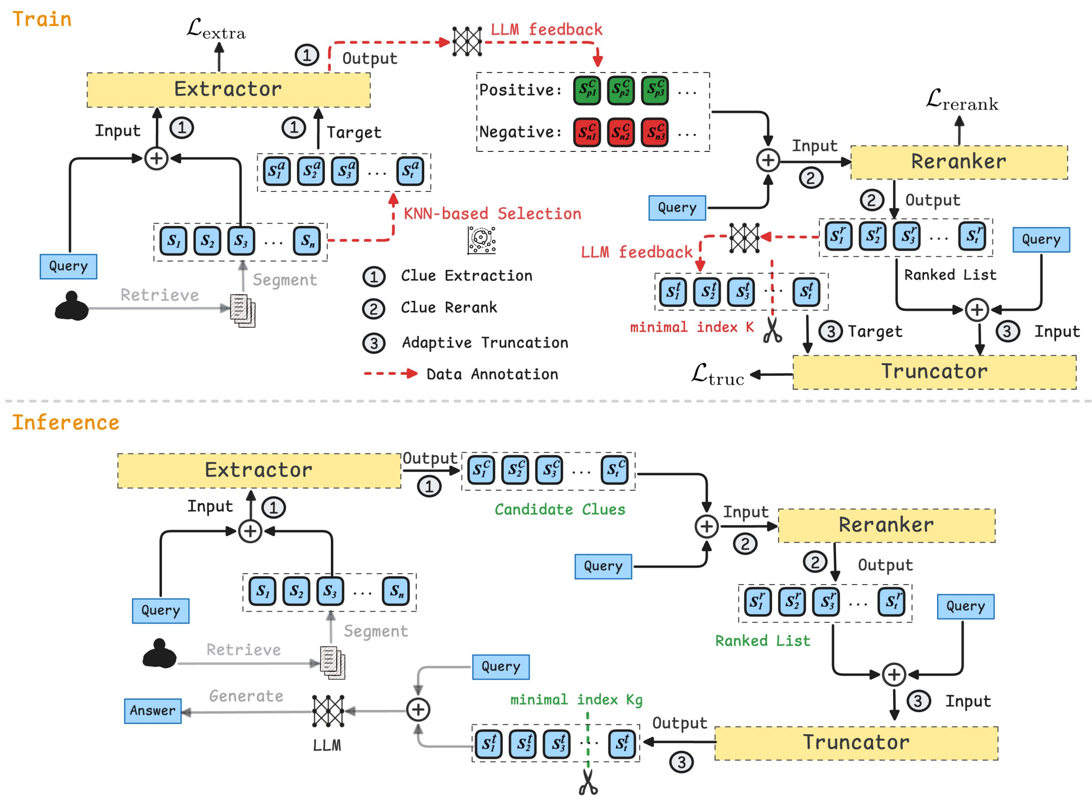

Figure 2: The architecture of FineFilter includes three modules: clue extractor, re-ranker, and truncator. The top displays their training strategies and annotated data, while the bottom shows the noise filtering during inference.

high computational complexity during the training process, posing additional challenges for practical implementation. Xu et al. (2024); Wang et al. (2023) explore extractive filters to select the most relevant sentences. While these methods help eliminate irrelevant information, they also face the risk of over-compression, which may lead to a reduction in output accuracy. In another approach, Li et al. (2023) introduce the concept of Selective Context, which eliminates redundant content based on self-information metrics to enhance the efficiency of LLM inference. However, this technique may compromise the semantic coherence of the context.

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:

$$\mathcal{S}^* = \arg\min |\mathcal{S}^{'}|,$$

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

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, ensuring a trade-off between computational efficiency and answer accuracy. 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 propose a three-stage noise filtering mechanism for RAG, called FineFilter, as shown in Figure 2. FineFilter consists of three modules: the clue extractor, the clue re-ranker, and the adaptive truncator. First, the clue extractor maximizes information gain to select potential answer clues from multiple documents, reducing the search

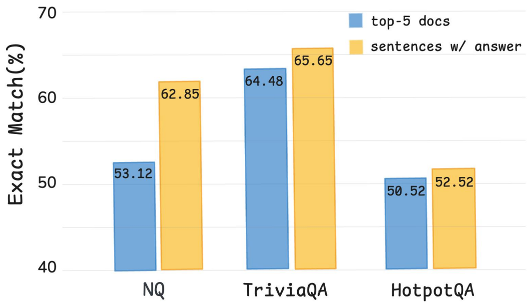

Figure 3: The Exact Match performance of LLaMA3-8B-Instruct on three QA datasets between the top-5 documents and answer-containing sentences. The answer-containing sentences refer to all sentences in the same top-5 documents where the ground-truth answer appears, regardless of their query relevance.

space and improving the relevance of the candidate set. Next, the clue re-ranker optimizes the ranking of sentences using pairwise loss, ensuring the most relevant clues are prioritized. Finally, the adaptive clue truncator truncates the minimal necessary context, ensuring a balance between computational efficiency and answer accuracy.

4.1 Clue Extractor

The goal of clue extractor is to identify potential answer clues from multiple documents and construct a smaller query-relevant candidate set to reduce search space. We compare the performance of answer-containing sentences and the original retrieved documents in downstream tasks, as shown in Figure 3. We can see that filtering out the low-value information from the documents benefits RAG reasoning. Although not all answer-containing sentences are query-relevant, they approximate the maximal subset capable of addressing user queries and can serve as the optimization target for the clue extractor.

To optimize the sentence extraction process, 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 $\mathrm{IG}(q, s_i)$ of sentence s_i is defined as:

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

where $H(q)$ represents the entropy of the query q, measuring the uncertainty of the query; and $H(q \mid s_i)$ represents the uncertainty of the query given the sentence s_i. In question answering tasks, information gain measures the reduction of uncertainty in the query by including a particular sentence. *Typically, sentences that contain the answer directly reduce the unresolved part of the query, helping the model better understand the core of the query and improve the accuracy of the downstream generation module.*

Based on this information gain concept, we first extract sentences from the retrieved document collection that contain the ground-truth answer as extraction targets. Given the query q, the ground-truth answer y and retrieved sentences $\mathcal{S} = \{s_1, s_2, \ldots, s_n\}$, the answer-containing sentences is 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 finetune an LLM model as the clues $Extractor$ 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 A.1). The loss function of $Extractor$ model is defined as:

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

Finally, our clue extractor has the ability to generate the potential candidate clues based on the user query and retrieved documents in the inference:

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

KNN-based Extraction We find that answer-containing sentences significantly improve performance on the simple QA dataset, i.e., NQ, but less so on the complex QA dataset, i.e., TriviaQA and HotpotQA, as shown in Figure 3. Therefore, we propose a KNN-based similar sentence extraction strategy for complex reasoning scenarios. For simple questions, we directly select sentences containing the answer as the extractor's optimization targets, as these sentences provide the key information to answer the question and significantly reduce the uncertainty. For more complex questions, we first select sentences containing the answer and then further select sentences semantically similar to the answer using the K-Nearest Neighbors (KNN) method (Guo et al., 2003). Although these sentences may not directly contain the answer, they provide contextual information related to the question's answer, helping the model better understand the nature of the question and generate a more accurate answer. We utilize both the answer-containing sentences and the KNN-based similar sentences as the extractor's optimization targets. This KNN-based strategy allows the system to respond to queries of varying complexity flexibly, ensuring higher accuracy while minimizing computational overhead.

4.2 Clue Reranker

The sentences selected by the clue extractor often contain multiple relevant clues, but their relevance may vary, requiring further re-ranking. To achieve this, we train a re-ranker using pairwise loss to optimize the ranking model, ensuring that the most relevant sentences are ranked at the front.

Training We use the real RAG-generated feedback to annotate the training data for the re-ranker, 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 goal of $Reranker$ is to minimize the following pairwise loss function (Karpukhin et al., 2020) to improve the 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 $\text{sim}(q, *)$ is the semantic similarity between the query q and the sentence $*$ by $Reranker$ model. By minimizing this loss function, the $Reranker$ model can effectively identify the most relevant clues and prioritize them accordingly.

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

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

4.3 Adaptive Truncator

The goal of the adaptive truncator is to capture the minimal necessary clues based on the complexity of the question and the content of the retrieval documents, retaining sufficient clues to generate an accurate answer while further reducing reasoning overhead.

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 re-ranking step. Given a query q and its re-ranked clues $\mathcal{S}^r = \{s_1^r, \dots, s_n^r\}$, 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 $D_k = \{s_1^r, \dots, s_k^r\}$, 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:

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

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:

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

If the RAG system cannot generate a correct answer for any subset, then $\mathcal{S}^t = \emptyset$, indicating that no subset of the re-ranked sentences suffices to produce the correct answer. This method ensures that the minimal necessary context $\mathcal{S}^t$ is used, optimizing the balance between information relevance and computational efficiency.

During the model training stage, we fine-tune a LLM based on the data annotations as the adaptive truncator. The $Truncator$ is trained to predict the smallest index K of $\mathcal{S}^r$ that needed to answer each 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 $\mathcal{S}^t = \{S_1^r, \dots, S_{K_g}^r\}$. This ensures efficient utilization of computational resources while maintaining answer accuracy. Finally, the generation module of RAG concatenates the query q with the filtered answer clues $\mathcal{S}^t$ as a prompt(see Appendix A.3) to reason the answer.

[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.

Methods	NQ EM	NQ F1	NQ CR	NQ TP	TriviaQA EM	TriviaQA F1	TriviaQA CR	TriviaQA TP	HotpotQA EM	HotpotQA F1	HotpotQA CR	HotpotQA TP
Closed-book	26.98	62.51	-	-	30.54	68.86	-	-	19.96	55.84	-	-
Retrieval without Filtration												
Top-1 document	36.81	69.21	5.17x	2.17	42.74	77.13	5.32x	3.11	25.54	60.09	4.83x	3.11
Top-5 documents	40.21	70.95	1.0x	1.69	48.32	80.16	1.0x	2.90	25.07	59.57	1.0x	1.82
LLaMA3-8B-Instruct												
Retrieval with Filtration												
RECOMP	37.12	69.43	11.97x	3.54	43.41	77.61	10.91x	3.25	24.59	59.26	12.95x	4.97
FILCO	32.43	64.78	17.43x	**3.82**	38.96	74.14	13.93x	3.47	20.12	56.03	11.77x	5.39
LongLLMLingua	36.96	69.25	4.56x	1.97	47.56	79.15	4.18x	3.04	24.31	58.93	4.45x	3.39
BottleNeck	39.72	70.14	14.32x	3.36	48.16	79.83	**21.26x**	4.32	25.64	60.23	13.21x	5.51
Ours	**42.17**	**71.31**	**19.56x**	3.72	**48.81**	**80.33**	20.77x	**4.91**	**26.47**	**61.15**	**14.37x**	**5.73**
Mistral-7B-Instruct												
Retrieval with Filtration												
RECOMP	36.95	69.25	13.83x	3.25	43.39	77.51	10.91x	3.17	24.34	59.16	7.24x	4.35
FILCO	32.59	64.83	16.35	3.09	38.47	73.87	12.83x	3.31	21.34	56.91	13.00x	4.73
LongLLMLingua	37.45	69.67	4.09x	1.58	47.84	79.23	4.31x	3.01	24.05	58.75	4.22x	3.36
BottleNeck	39.48	70.05	12.53x	3.01	48.03	79.97	15.24x	4.28	25.47	59.97	11.06x	4.75
Ours	**41.93**	**71.12**	**17.43x**	**3.47**	**48.64**	**80.21**	**16.49x**	**4.49**	**26.03**	**60.78**	**14.89x**	**4.77**

Table 1: Experimental results on NQ, TriviaQA, and HotpotQA datasets. **EM** = exact match, **F1** = F1 score, **CR** = compression ratio, **TP** = throughput (examples/second). We compare our FineFilter with Closed-book, Top-1, Top-5, and various filtering methods (RECOMP, FILCO, LongLLMLingua, BottleNeck) on two basement models LLaMA3-8B-Instruct and Mistral-7B-Instruct.

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 For the three open-domain QA datasets, we evaluate end-task performance using Exact Match (**EM**) and **F1** score for the answer strings. EM measures exact correctness, while F1 evaluates answers that are close to but not necessarily exact, offering a more nuanced view of how well-predicted answers overlap with the correct ones. To assess the computational cost of downstream tasks, we introduce two metrics (Cao et al., 2024; Hwang et al., 2024): compression ratio (**CR**) and inference throughput (**TP**) on a single A6000-48G GPU. The **CR** is defined as the ratio of the original context length to the compressed context length. **TP** refers to the number of examples the model can process or generate per second during inference.

Implementation Details We use LLaMA3-8B-Instruct (Dubey et al., 2024) and Mistral-7B-Instruct (Jiang et al., 2023) as the backbone large language models. 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 for the three QA datasets.

5.2 Baselines

We select three types of baselines, including no filtration, extractive and abstractive filtration.

No Filtration We evaluate the following three baselines: (i) **Closed-book** generation relying solely on parametric knowledge, (ii) **Top-1** retrieval using the highest-ranked document, and (iii) **Top-5** retrieval with direct concatenation of all retrieved documents.

Extractive Methods We choose **RECOMP** (Xu et al., 2024) and **LongLLMLingua** (Jiang et al., 2024). RECOMP employs a fine-tuned cross-encoder to identify salient sentences through dense retrieval. LongLLMLingua utilizes question-aware

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

Methods	EM
FineFilter	**42.17**
w/o clue extractor	39.70
w/o clue reranker	41.64
w/o adaptive truncator	42.03

Table 2: The Exact Match of abalation study on NQ test set based on LLaMA3-8B-Instruct.

perplexity scoring with a dynamic programming algorithm to prune irrelevant tokens progressively in long contexts.

Abstractive Methods We choose **FILCO** (Wang et al., 2023) and **BottleNeck** (Zhu et al., 2024). FILCO learns a context filtering model to dynamically identify key sentences and jointly optimizes with the generator for end-to-end content distillation. Bottleneck leverages reinforcement learning and information bottleneck theory to optimize filtering and generation.

5.3 Main Results

The comparison results on NQ, TrivialQA, and HotpotQA datasets are shown in Table 1. From the results, we can see that: **RAG improves downstream task performance across three datasets.** Using the top-5 documents generally outperforms using the top-1 document, indicating that incorporating more contextual information improves model performance. **Noise Filtering are crucial for further improving performance.** Across multiple datasets, applying filtering methods significantly reduces context length while maintaining performance close to that of the top-5 documents, effectively filtering out irrelevant information and enhancing accuracy. **FineFilter outperforms baselines across multiple models and datasets.** FineFilter consistently surpasses all filtration baselines across LLaMA3 and Mistral, achieving superior EM and F1 performance and compression efficiency, i.e., 6% and 37% improvement of EM and CR than BottleNeck on NQ dataset with LLaMA3. **FineFilter performs remarkably better than Top-5 documents on complex multi-hop tasks.** Compared to other datasets, FineFilter shows a larger improvement on the HotpotQA dataset, i.e., the EM improvement than Top-5 with LLaMA3 is 5.4% on HotpotQA, 4.8% on NQ and 1.0% on TriviaQA, highlighting its exceptional ability in handling complex multi-hop reasoning tasks.

Dataset	Method	EM
NQ	Direct	39.41
	Fine-tuned	**41.43**
TriviaQA	Direct	44.37
	Fine-tuned	**48.58**
HotpotQA	Direct	24.71
	Fine-tuned	**26.12**

Table 3: Performance comparison between Direct Extraction and Fine-tuned Extraction across three datasets based on LLaMA3-8B-Instruct.

5.4 Analysis

5.4.1 Ablation Study

To explore the impact of different components on FineFilter, we use LLaMA3-8B-Instruct as the base LLM and introduce the following variants of FineFilter for ablation study:1) *w/o clue extractor.* LLaMA3-8B-Instruct, without fine-tuning, is used to directly extract answer clues; 2) *w/o clue reranker.* The original sentence ranking given by the fine-tuned extractor is used; 3) *w/o adaptive truncator.* No longer performs adaptive truncation on the re-ranked clues.

As shown in Table 2, removing the clue extractor and reranker leads to a significant drop in accuracy, highlighting the critical role of these components in the performance of FineFilter, with the clue extractor having a more pronounced impact on subsequent steps. In contrast, removing the truncator has a smaller impact on performance, as its contribution primarily lies in improving RAG reasoning efficiency.

5.4.2 Impact of Fine-tuning Clue Extractor

We analyze the impact of fine-tuning on the performance of the Clue Extraction module by comparing the following two approaches: 1) *Direct Extraction*. Using LLM without fine-tuning to extract clue sentences from retrieved documents based on extraction prompts (see Appendix A.1). 2) *Fine-tuned Extraction*. Fine-tune the extraction model to select the answer-containing sentences.

As shown in Table 3, the experimental results demonstrate that fine-tuned extraction outperforms direct extraction across all three datasets. Fine-tuning allows the model to select more relevant sentences by incorporating task-specific knowledge, leading to significant performance improvements.

5.4.3 Impact of KNN-Based Extraction

We evaluate the impact of the KNN-based extraction strategy by adjusting its threshold, which de-

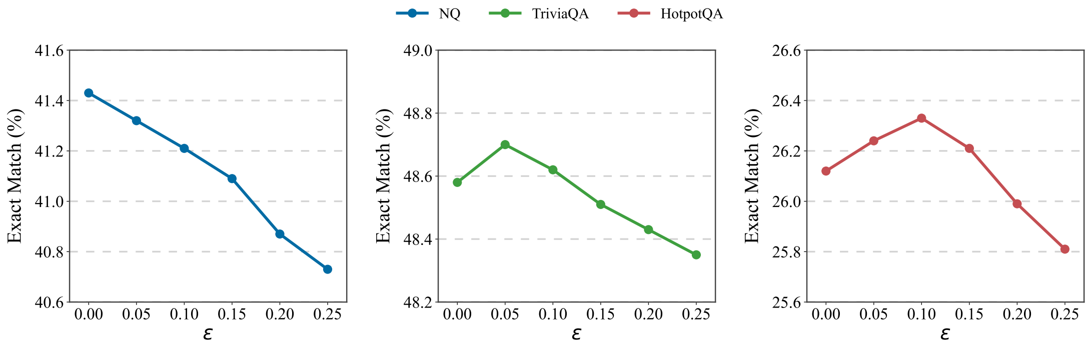

Figure 4: An illustration of clue extraction model performance using LLaMA3-8B-Instruct on NQ, TriviaQA, and HotpotQA datasets. The x-axis represents the KNN-clustering threshold, with the model incorporating more context as the threshold increases.

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

Table 4: Comparison of different reranking methods on NQ test set based on LLaMA3-8B-Instruct.

Dataset	Method	EM	F1
NQ	Random	41.23	71.01
	Adaptive(Ours)	**42.17**	**71.31**
TriviaQA	Random	48.67	80.23
	Adaptive(Ours)	**48.81**	**80.33**
HotpotQA	Random	26.28	61.06
	Adaptive(Ours)	**26.47**	**61.15**

Table 5: Comparison of Random and our Adaptive Truncation methods across NQ, TriviaQA, and HotpotQA based on LLaMA3-8B-Instruct.

fines the distance of cosine similarity to the answer-containing sentences. When set to 0, the model selects sentences only containing the answer without KNN. As the threshold increases, the model gradually selects more semantically similar sentences to the answer, expanding the context and providing additional relevant information.

As shown in Figure 4, we compare the performance of the model 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 initially improves performance at lower thresholds but declines at higher thresholds due to increased noise.

5.4.4 Impact of Different Reranker

To select a more effective re-ranking 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 4, the results demonstrate that Sentence-BERT outperforms all baseline models in both EM and F1 scores, so we chose it as the re-ranking base model.

5.4.5 Adaptive vs. Random Truncation

To validate the effectiveness of our adaptive truncation strategy, we compare it with random truncation. As shown in Table 5, our adaptive truncation method outperforms random truncation in all metrics. This is because adaptive truncation dynamically selects context based on the results feedback from the downstream generator, retaining the most relevant and informative content to improve model performance.

6 Conclusion

In this paper, we introduce FineFilter, a novel fine-grained noise filtering mechanism aimed at improving performance and reducing cost in RAG systems. By framing noise filtering as a sentence-level MinMax optimization problem, FineFilter effectively tackles the challenge of identifying relevant clues in complex reasoning scenarios. Its three optimized modules leverage KNN clustering to obtain sufficient relevant context and retain effective clues based on generator feedback. Experiments show that FineFilter outperforms baselines on three QA datasets in both performance and efficiency. Future work can explore more adaptive noise filtering that dynamically adjusts based on query complex or retrieval quality for complex reasoning tasks.

7 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.

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.

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.

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.

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.

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.

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.

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.

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.

A Prompt

A.1 Prompt for Clue Extractor

We show our prompt for clue extraction in Table 6, 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.

A.2 Prompt for Adaptive Truncator

We show our prompt for adaptive truncation in Table 7. 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.

A.3 Prompt for 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 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.

C 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 9 and Table 10 for the NQ dataset, and Table 11 and Table 12 for the HotpotQA dataset, these examples will demonstrate the advantages and effectiveness of the FineFilter method in handling question answering tasks of varying complexity.

Prompt for 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 6: Prompt for Clue Extractor.

Prompt for 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 7: Prompt for Adaptive Truncator.

Prompt for 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 Generator.

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 9: 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 10: 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 11: 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 12: Case study based on an example from HotpotQA.