Efficient Dynamic Clustering-Based Document Compression for Retrieval-Augmented-Generation

Weitao Li[1,2*], **Xiangyu Zhang**[1], **Kaiming Liu**[1,2], **Xuanyu Lei**[1,2], **Weizhi Ma**[2,†], **Yang Liu**[1,2,†]
[1] Dept. of Comp. Sci. & Tech., Institute for AI, Tsinghua University, Beijing, China
[2] Institute for AI Industry Research (AIR), Tsinghua University, Beijing, China

Abstract

Retrieval-Augmented Generation (RAG) has emerged as a widely adopted approach for knowledge injection during large language model (LLM) inference in recent years. However, due to their limited ability to exploit fine-grained inter-document relationships, current RAG implementations face challenges in effectively addressing the retrieved noise and redundancy content, which may cause error in the generation results. To address these limitations, we propose an **E**fficient **D**ynamic **C**lustering-based document **C**ompression framework (**EDC^2-RAG**) that utilizes latent inter-document relationships while simultaneously removing irrelevant information and redundant content. We validate our approach, built upon GPT-3.5-Turbo and GPT-4o-mini, on widely used knowledge-QA and Hallucination-Detection datasets. Experimental results show that our method achieves consistent performance improvements across various scenarios and experimental settings, demonstrating strong robustness and applicability. Our code and datasets are available at https://github.com/Tsinghua-dhy/EDC-2-RAG.

1 Introduction

In recent years, large language models (LLMs) have advanced rapidly, excelling in natural language processing (NLP) tasks such as question answering, code generation, and even medical diagnosis (Yasunaga et al., 2021; He et al., 2025; Yue et al., 2023; Singhal et al., 2023; Li et al., 2024a). Despite their success, LLMs face two key challenges: expensive knowledge updates due to the large number of learnable parameters, and hallucinations that lead to misleading content (Honovich et al., 2023; Hu et al., 2023; Lin et al., 2024; Xu et al., 2024). These issues impact the availability, reliability and consistency of LLMs (Zhou et al., 2024). Retrieval-augmented generation (RAG) (Lewis et al., 2020; Borgeaud et al., 2022; Izacard et al., 2022) addresses these problems by integrating retrieval with generation, allowing LLMs to access external knowledge without parameter updates, reducing hallucinations, and improving reliability.

However, the implementation of RAG methods in real-world settings presents significant challenges. From a structural perspective, the effectiveness of RAG frameworks derives from the information augmentation of integrated databases(Lewis et al., 2020). In practical applications, the databases are often of limited quality due to the scarcity of high-quality data and the high cost of data cleaning. Therefore, the candidate documents faced by retrievers tend to exhibit the following frequently-encountered quality flaws:

- **Noise**: irrelevant content to the query, which may result in errors during generation.
- **Redundancy**: highly similar content between documents, which will consume more tokens and time in inference.

These issues can significantly reduce the effectiveness of retrieval and compromise the quality of the final generated output. Faced with these practical challenges, it is increasingly significant to build a reliable RAG system. However, current RAG frameworks predominantly rely on query-document similarity for retrieval, without explicitly addressing prevalent issues such as noise and redundancy in real-world document corpora. To solve the problems, we propose an efficient dynamic clustering-based compression method for a reliable document retrieval.

Specifically, we first encode the documents to get a denser content representation, then perform clustering to aggregate semantically similar documents, mitigating content repetition. Subsequently, we use prompt-based techniques to guide the LLMs

*Email: liwt23@mails.tsinghua.edu.cn
† corresponding authors.

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 9833–9849
November 4-9, 2025

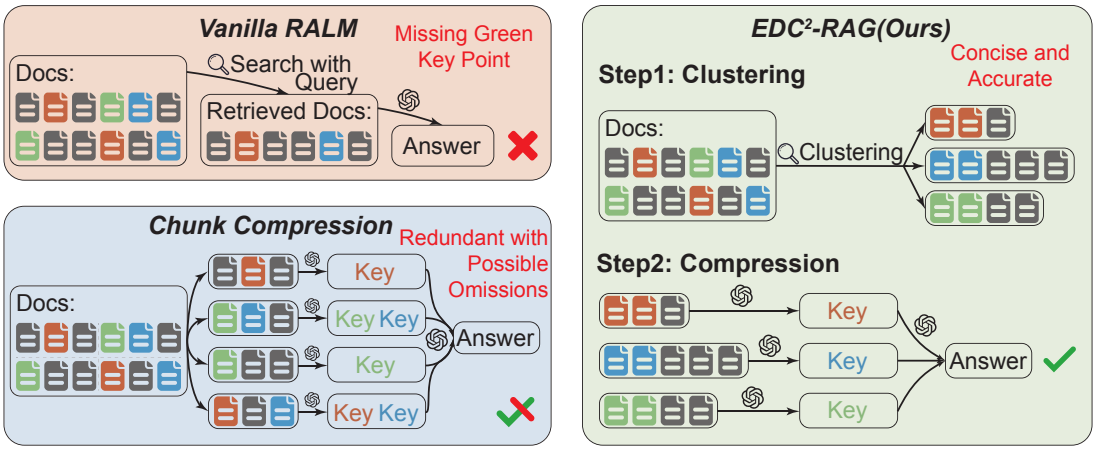

Figure 1: Comparison between our method and prior approaches. Unlike Vanilla RAG, which misses key information, and Chunk Compression, which is redundant and incomplete, our method clusters and compresses documents to extract concise and accurate answers.

in query-specific compression to improve information density and eliminate noise. Finally, we concatenate the compressed content into the prompts for response generation. In summary, our method leverages the latent relationships between documents to reduce noise and redundant content.

To validate the effectiveness of our approach, we selected two types of widely used datasets: KQA tasks and hallucination detection tasks. Systematic experiments conducted on GPT-3.5-Turbo demonstrate that our method achieves significant performance improvements across different settings. Meanwhile, our method also exhibits strong robustness and generalization potential to other scenarios. These findings indicate that by deeply exploring and utilizing fine-grained relationships among documents, RAG methods can reach new performance heights, providing a novel direction for addressing the hallucination problem and knowledge update challenges in LLMs.

The main contributions of our work are:

- To the best of our knowledge, we are the first to apply similarity-based semantic clustering in the post-retrieval stage to address practical challenges in in-the-wild RAG systems.
- Our method effectively improves the performance and robustness of the RAG systems and also enhances their long context capability.
- As a post-retrieval method, our approach is plug-and-play, requiring no additional training, and can be integrated into various pipelines.

2 Related Works

Reranking and Compression. Post-retrieval methods for frozen large language models (LLMs) can be categorized into reranking and compression approaches (Gao et al., 2023b). Reranking refines the order of retrieved documents to improve LLMs-generation performance. Re3val (Song et al., 2024) uses reinforcement learning (RL) and targeted queries, while REAR (Wang et al., 2024) utilizes LLaMA 2 (Touvron et al., 2023) for reranking, enhancing response quality. Compression methods condense retrieved content, primarily through fine-tuned models(Xu et al., 2023; Liu et al., 2023; Yu et al., 2024) or LLMs native capabilities. For instance, SURE (Kim et al., 2023) generates and selects the best answer by summarizing multiple responses. However, existing methods **rarely address document noise and redundancy issues**, whereas our approach tackles them with dynamic clustering and prompt-guided compression.

Retrieval Semantic Relation Modeling. Beyond post-retrieval methods, some studies focus on refining relationships between documents, chunks or entities. Recent approaches frame RAG as a multi-agent collaboration, where each agent processes a subset of retrieved content. Long Agent (Zhao et al., 2024) supports large contexts through chunk-level conflict resolution, while MADAM-RAG (Wang et al., 2025) uses agents to address conflicting responses. Multi-agent RAG is also applied to data integration (Salve et al., 2024), but these methods increase inference costs and latency, limiting real-world applicability. Knowledge Graphs (KGs) structure document information by

Phase 1: Initialization

Input: Document set $V = \{d_1, d_2, \ldots, d_n\}$, query q, similarity function $\text{sim}(\cdot, \cdot)$, embedding model $f(\cdot)$, initial cluster size τ, threshold Λ
Output: Clusters $\{C_1, C_2, \ldots, C_k\}$
Compute query embedding: $\mathbf{v}_q \leftarrow f(q)$
for all $d_j \in V$ **do**
    Compute embedding: $\mathbf{v}_j \leftarrow f(d_j)$
end for
Select initial cluster root: $C.R_1 \leftarrow \arg\max_{d \in V} \text{sim}(\mathbf{v}_q, \mathbf{v}_j)$
for all $d_j \in V$ **do**
    Compute similarity: $s_j \leftarrow \text{sim}(\mathbf{v}_{C.R_1}, \mathbf{v}_j)$
end for
Form C_1 with top-τ documents from V sorted by s_j
Remove C_1 members from V

Phase 2: Iterative Subgraph Formation

$k \leftarrow 2$
while $V \neq \emptyset$ **do**
    Select new root: $C.R_k \leftarrow \arg\max_{d \in V} \text{sim}(\mathbf{v}_q, \mathbf{v}_j)$
    for all $d_j \in V$ **do**
        Compute similarity: $s_j \leftarrow \text{sim}(\mathbf{v}_{C.R_k}, \mathbf{v}_j)$
    end for
    Determine cluster size: $size \leftarrow \min(2 \times |C_{k-1}|, \Lambda)$
    Form C_k with top-$size$ documents from V sorted by s_j
    Remove C_k members from V
    $k \leftarrow k + 1$
end while

Algorithm 1: Efficient Dynamic Graph-based Document Clustering

providing contextual relationships (Ji et al., 2021). KAPING builds a KG for retrieval (Baek et al., 2023), while G-Retriever queries subgraphs (He et al., 2025). Despite their effectiveness in entity-rich tasks, KG-based methods **face scalability and adaptability challenges** and often **require substantial resources on the corpus processing side** (Peng et al., 2023; Li et al., 2024b), and so does RAPTOR (Sarthi et al., 2024). Our method dynamically constructs semantic relationships post-retrieval, avoiding multi-agent systems and pre-built graphs, thereby improving retrieval quality by reducing redundancy and noise.

3 Problem Definition

Consider a set of retrieved documents $V = \{d_1, d_2, \ldots, d_n\}$, where each document d_i is associated with a query q. These documents are retrieved based on their relevance to q, but their exact utility in answering q is initially unknown. Furthermore, there may exist potential overlaps and redundancies among the documents in V, as some documents may share similar or identical information, while others may provide complementary or conflicting details.

Let $E = \{e_{ij}\}$ represent the relationships between pairs of documents d_i and d_j, where $i, j \in \{1, 2, \ldots, n\}$. These relationships can be categorized as:

- **Overlapping**: e_{ij} = Overlap, indicating that d_i and d_j share redundant or highly similar content.
- **Complementary**: e_{ij} = Complementary, indicating that d_i and d_j provide distinct but relevant information to q.

Additionally, let $U = \{u_1, u_2, \ldots, u_n\}$ denote the utility scores of the documents, where u_i represents the degree to which d_i contributes to answering q. These scores are initially unknown and must be inferred based on the relationships E and the content of the documents.

The goal is to effectively utilize the retrieved documents V, their relationships E, and their inferred utilities U to construct a comprehensive and accurate response to the query q. This involves addressing the challenges of redundancy, inconsistency, and varying utility among the documents, while ensuring that the final output maximizes relevance and minimizes noise.

4 Method

4.1 Overview

The core of our approach involves clustering documents using embedding models guided by predefined rules, followed by applying compression techniques to eliminate noise. These refined documents are then integrated into the prompt, enabling the LLM to more effectively utilize the information and enhance its performance. Our methodology is presented in accordance with the processing workflow, and Figure 1 provides a comparative visualization of our method against current RAG frameworks.

4.2 Efficient Dynamic Clustering of Documents

In RAG frameworks, retrieved documents often contain redundancy and noise, which can negatively impact the reasoning quality of LLMs. Traditional post-retrieval methods primarily rely on reranking or compression strategies to refine retrieved results, but they often fail to fully utilize the fine-grained relationships between documents.

To address this, we propose an efficient dynamic clustering-based approach to structure the retrieved documents before further processing. By organizing documents into clusters based on similarity, we aim to reduce redundancy and group related content together, creating a more coherent input for downstream tasks. Specifically, we prioritize documents with high similarity to the query, as these are most likely to contribute valuable information. Additionally, we adopt a dynamically expanding clustering strategy, where the cluster size increases iteratively, ensuring efficient grouping while keeping computational costs manageable. In our experiments, **we set** $\tau = 3$ **and** $\Lambda = 20$.

4.3 Query-Aware Compression

After constructing the subgraphs $C_1, C_2, \ldots, C_k$, it is essential to further refine the retrieved content by eliminating redundancy and distilling key information. While clustering helps organize documents based on similarity, it does not inherently resolve the issue of overlapping or extraneous details.

To address this, we introduce a compression step that leverages a large language model (LLM) to generate concise yet informative summaries. Specifically, we concatenate each C_i ($i \in [1, k]$) with the query q and prompt the LLM to produce a query-aware summary, ensuring that only the most relevant and essential content is preserved. The goal of this step is to maximize the information density of retrieved documents while removing redundant or marginally relevant details, preparing a high-quality input for final generation.

Importantly, this summarization process is highly efficient as **all summaries can be generated in parallel**, allowing the system to scale effectively with the number of clusters while maintaining low latency. An example prompt is as follows:

Compression Prompt

```
--------------------------------
```
Few-shots:
`{example 1}`
`{example 2}`
`{...}`

Instruction:
Given a question and a set of reference documents, extract only the **verifiable, relevant** information that directly supports the question.
Avoid inferences or conclusions.
If nothing is relevant, output: `"No content to extract"`.

Question:
`{query}`

Documents:
`{docs}`

Extracted Summary:
`{to be filled}`

```
--------------------------------
```

4.4 Generation

After clustering and compression refine the documents, the system generates a contextually relevant response. Our query-aware integration ensures the output is based on coherent, information-rich content tailored to the query. To accommodate diverse dataset characteristics, our method flexibly adapts the generation process. In scenarios where compression may risk omitting critical details due to LLM limitations (such as in KQA tasks), we strategically integrate response generation with the compression phase, allowing the system to dynamically refine answers. This approach enhances the retention of essential information and improves response accuracy, particularly in complex question-answering tasks. If compression yields poor summaries, the system falls back to original documents, ensuring robustness.

Unlike traditional RAG methods, which often rely on loosely structured retrieved documents, our approach enhances the informativeness of retrieved content by distilling critical insights in a query-driven manner. This structured input enables the LLM to reason more effectively, reducing hallucinations and improving response precision. Moreover, our method efficiently balances computational costs and performance by limiting the number of API calls required for summarization, ensuring practical deployment feasibility.

By optimizing the input for the final response generation step, our method improves both the precision and efficiency of the system, leading to more

reliable and contextually relevant outputs while reducing computational overhead.

5 Experimental Settings

5.1 Overview

To validate the effectiveness of our method, we employe three types of datasets in the experiments: Knowledge-QA datasets, Hallucination-Detection datasets, and Redundancy dataset built by us. The retrieval settings and implementation details for these datasets vary slightly, which are presented in Appendix B.

We utilize GPT-3.5-Turbo-1106 and GPT-4o-mini-2024-07-18 as the backbone LLMs. For simplicity, we refer to GPT-3.5-Turbo-1106 as "ChatGPT" and GPT-4o-mini-2024-07-18 as "GPT-4o-mini". The decoding temperature is fixed at 0 for reproducibility, with the exception of Long Agent and KQA sampling steps in our methods, where 0.7 is used to enhance output diversity.

5.2 Datasets

Knowledge-QA Datasets: Knowledge Question Answering (KQA) datasets assess a LLM's ability to reason over retrieved external knowledge sources from knowledge graphs or textual corpora. We use three common KQA datasets (Yu et al., 2024; Lv et al., 2024; Song et al., 2025): WebQ (Berant et al., 2013) (single-hop), and 2WikiMultiHopQA (Ho et al., 2020) (hereafter referred to as 2Wiki) plus Musique (Trivedi et al., 2022) (both multi-hop). To analyze noise robustness, following prior work (Lv et al., 2024; Yu et al., 2024), we employ DPR retrieval and its reader to identify noisy documents, constructing cases with varying noise proportions by filtering samples from these three datasets. Details are in the Appendix B.1.

Redundancy dataset: To evaluate the capability of our method in handling redundancy, we used DPR to retrieve Top-20 documents per question from the WebQ dataset. The redundancy rate r is defined as:

$$r = \frac{\text{number of rewritten documents}}{20}$$

Implementation details are provided the in Appendix B.1.

Hallucination-Detection Datasets: Hallucination Detection is an NLP task that verifies whether generated or stated content—like summaries or answers—is factual or nonfactual by checking against available information sources. We conducte experiments on three widely used fact-checking tasks (Li et al., 2024c; Lv et al., 2024): the FELM World Knowledge Subset (Chen et al., 2023), the WikiBio GPT-3 Dataset (Manakul et al., 2023), and the HaluEval Dataset (Li et al., 2023). Details are in the Appendix B.2.

5.3 Baselines and Evaluation Metrics

We compare with several baselines: 1) **Vanilla RALM** (Borgeaud et al., 2022), the standard RAG process; 2) **Chunk Compression** (Jiang et al., 2024), which compresses documents using an LLM; 3) **Long Agent** (Zhao et al., 2024), which divides long documents among collaborating agents with a leader agent aggregating outputs; 4) **CEG** (Li et al., 2024c), a strong post-hoc RAG baseline for hallucination detection; 5) **Raptor**, which leverages recursive abstractive processing for tree-organized retrieval; and 6) task-specific methods including **HalluDetector** (Wang et al., 2023), **Focus** (Zhang et al., 2023), **SelfCheckGPT** w/NLI (Manakul et al., 2023), CoT-augmented prompting (Kojima et al., 2022), and prompts augmented with hyperlinks to reference documents and with human-annotated reference documents (Chen et al., 2023). Full details are in Appendix B.3.

We use F1 score as the evaluation metric for the Knowledge-QA task, Balanced_Acc for the FELM and WikiBio GPT-3 datasets, and Acc for the HaluEval dataset.

6 Experimental Results

6.1 Main Results on Knowledge-QA Datasets

6.1.1 Results on Varying Top-k

Experimental results in Table 1 demonstrate the effectiveness and robustness of our method across multiple datasets and LLM backends.

On Musique, our approach achieves the highest average F1-scores with both ChatGPT and GPT-4o-mini, consistently outperforming all baselines. Notably, while Long Agent performs well with ChatGPT, its performance drops significantly with GPT-4o-mini, indicating possible overfitting or reduced adaptability. In contrast, our method maintains strong performance across both models.

On WebQ, our method also achieves the best average performance with ChatGPT and GPT-4o-mini, showing improvements over Vanilla RALM and other compression-based methods. The results highlight the generalizability of our approach to both simple and diverse question types.

Dataset	Method	Top-k							
		5	10	20	30	50	70	100	Avg
		gpt-3.5-turbo-1106							
Musique	Vanilla RALM	71.05	71.73	74.75	76.93	75.16	80.25	77.04	75.27
	Chunk Compression	74.45	81.01	74.15	76.49	69.57	74.53	67.17	73.91
	Long Agent	**83.07**	**85.83**	82.04	**84.84**	81.87	80.65	83.67	83.14
	Ours	81.66	83.31	**82.55**	80.17	**86.60**	**86.10**	**84.68**	**83.58**
WebQ	Vanilla RALM	88.84	90.14	90.07	90.30	91.13	90.74	91.38	90.89
	Chunk Compression	90.52	**91.15**	90.77	91.18	91.24	90.98	90.38	90.26
	Long Agent	89.79	91.03	90.49	90.25	89.01	90.21	91.03	90.26
	Ours	**92.01**	90.98	**90.79**	**91.74**	**92.97**	**91.51**	**92.45**	**91.78**
2Wiki	Vanilla RALM	69.90	**74.68**	**77.51**	71.36	78.25	76.88	79.17	75.39
	Chunk Compression	67.38	67.14	72.41	68.98	72.08	72.99	72.66	70.52
	Long Agent	69.30	75.39	76.06	78.36	77.16	**83.22**	**83.45**	77.56
	Ours	**73.09**	**74.68**	76.20	**78.64**	**80.90**	80.45	82.06	**78.00**
		gpt-4o-mini-2024-07-18							
Musique	Vanilla RALM	74.43	78.85	77.78	74.95	78.55	76.24	78.20	77.00
	Chunk Compression	77.12	73.59	75.67	**76.02**	75.17	75.35	79.42	76.05
	RAPTOR	75.14	69.40	72.07	73.49	78.65	70.61	74.89	73.46
	Long Agent	73.29	75.25	80.43	72.52	**80.03**	**80.85**	77.38	77.11
	Ours	**78.33**	**79.80**	**81.71**	73.13	78.21	77.95	**80.07**	**78.46**
WebQ	Vanilla RALM	85.92	89.14	88.05	85.10	89.32	**91.92**	87.42	88.12
	Chunk Compression	85.64	84.99	85.07	83.98	88.66	90.79	90.94	87.15
	Long Agent	89.35	89.16	90.77	**91.08**	91.82	90.91	91.52	90.66
	Ours	**90.01**	**90.77**	**91.89**	90.30	**91.51**	91.25	**92.02**	**91.11**
2Wiki	Vanilla RALM	64.81	**73.38**	**73.84**	77.08	78.04	**78.01**	77.89	74.72
	Chunk Compression	62.38	65.76	69.24	67.62	72.45	73.26	74.06	69.25
	Long Agent	66.00	70.04	71.33	**77.68**	**79.98**	77.13	**83.45**	**75.09**
	Ours	**68.67**	69.79	72.86	73.73	75.82	77.43	79.28	73.94

Table 1: Performance comparison of different methods on MusiQue, WebQ, and 2Wiki Datasets Using GPT-3.5-turbo-1106 and GPT-4o-mini-2024-07-18 across various Top-k values.

For 2Wiki, a dataset requiring deeper reasoning, our method achieves the highest average with ChatGPT again, and shows competitive performance with GPT-4o-mini. Moreover, our approach exhibits more stable behavior across top-k values, unlike some baselines that fluctuate significantly—especially Chunk Compression, whose performance is inconsistent across different k.

Overall, these results confirm that our clustering-based compression method is not only effective in preserving essential information and reducing redundancy, but also exhibits strong model-agnostic adaptability and stability across retrieval depths, making it a reliable choice for RAG pipelines.

6.1.2 Results on Noise Resistence

Tables 2 and 11 summarize performance under varying noise levels with Top-k set to 100 and 20, respectively. Our method consistently yields the highest average F1 scores across all datasets and both model backends (ChatGPT and GPT-4o-mini). As noise increases, the performance gap over baselines widens, highlighting the robustness of our approach in noisy retrieval settings.

For instance, on MusiQue with ChatGPT at Top-k=100, our method exceeds the best baseline by over 3.4 F1 points on average and ranks first across all noise levels. Even at 100% noise—when all retrieved documents are distractors—it achieves 84.54 F1, far surpassing the next-best score of 80.47. This demonstrates our compression strategy's ability to suppress irrelevant content and recover useful signals from fully corrupted inputs.

Results on 2Wiki reveal similar strengths. While other methods degrade sharply with noise, our approach sustains relatively high performance, maintaining a 5–7 point margin under heavy noise. This shows its robustness in multi-hop reasoning even with deeply buried evidence.

GPT-4o-mini results show greater overall stabil-

Dataset	Method	Noise Rates (%) at Top-k=100						
		0	20	40	60	80	100	Avg
		gpt-3.5-turbo-1106						
MusiQue	Vanilla RALM	77.04	82.48	79.32	76.49	79.45	75.86	78.44
	Chunk Compression	67.17	77.83	75.62	79.79	77.20	75.81	75.57
	Long Agent	80.54	79.52	79.29	**84.08**	77.20	80.47	80.18
	Ours	**84.68**	**85.06**	**85.43**	81.84	**80.32**	**84.54**	**83.65**
WebQ	Vanilla RALM	91.38	88.88	88.28	88.85	87.54	81.61	87.76
	Chunk Compression	90.38	88.07	88.73	89.73	87.10	82.87	87.81
	Long Agent	91.03	90.79	90.07	88.39	90.17	88.56	89.84
	Ours	**92.45**	**92.04**	**92.40**	**90.67**	**91.08**	**90.20**	**91.47**
2Wiki	Vanilla RALM	79.17	71.76	71.48	71.26	64.81	58.95	69.57
	Chunk Compression	72.66	65.74	66.76	69.96	66.20	59.03	66.73
	Long Agent	**83.45**	**81.41**	**82.52**	**78.88**	71.79	70.92	**78.16**
	Ours	82.06	77.78	74.69	78.14	**76.71**	**75.65**	77.51
		gpt-4o-mini-2024-07-18						
MusiQue	Vanilla RALM	78.20	76.55	72.70	67.36	76.49	64.94	72.71
	Chunk Compression	79.42	76.90	75.62	71.98	70.85	69.66	74.07
	Long Agent	77.38	75.93	74.76	73.44	**76.58**	**78.84**	76.16
	Ours	**80.07**	**82.17**	**77.49**	**74.43**	75.62	78.70	**78.08**
WebQ	Vanilla RALM	87.42	87.08	89.67	85.13	**90.31**	84.89	87.42
	Chunk Compression	90.94	90.06	89.30	89.64	88.68	84.41	88.84
	Long Agent	91.77	90.37	**90.70**	**90.42**	87.84	86.67	89.63
	Ours	**92.02**	**91.42**	89.31	88.97	89.82	**86.83**	**89.73**
2Wiki	Vanilla RALM	77.89	77.83	75.79	**77.15**	**72.69**	66.67	**74.67**
	Chunk Compression	74.06	75.19	75.58	73.88	70.65	63.54	72.15
	Long Agent	**83.45**	**81.13**	**76.97**	73.99	64.06	59.64	73.21
	Ours	79.28	76.27	75.35	71.96	70.64	**68.67**	73.70

Table 2: Comparison of F1 scores under different noise levels at Top-k=100 on MusiQue, WebQ, and 2Wiki datasets for multiple retrieval methods.

ity than ChatGPT, but our method remains consistently superior. On MusiQue, it achieves 79.11 average F1, compared to 76.55 by Long Agent, again outperforming strong long-context baselines.

Under the Top-k=20 setting, where retrieval is constrained and noise more impactful, our method remains highly resilient. On WebQ and MusiQue, it sustains strong performance even under 80–100% noise, while baselines drop sharply—demonstrating that our compression mechanism works effectively not only for large retrieval sets but also in low-budget scenarios where every document matters.

6.1.3 Results on Redundancy Resistence

Table 3 reports performance under varying redundancy rates. Our method achieves the highest average F1 on WebQ, outperforming RALM in high-redundancy settings with a peak gain of +6.18 at 95% redundancy. This demonstrates its effectiveness in handling redundant information while preserving retrieval quality.

In summary, our method's consistent advantage across noise levels, datasets, and LLM backends highlights the generalizability and robustness of the compression strategy. By filtering irrelevant content and distilling key evidence, it boosts downstream performance and offers a reliable solution for noisy retrieval in RAG pipelines.

6.2 Main Results on Hallucination Detection

Table 5 presents a performance comparison of our proposed method against baseline approaches across three Hallucination-Detection datasets: FELM, WikiBio, and HaluEval. Results are reported as Maximum and Average accuracy over Top-k predictions (k from 1 to 10), with balanced accuracy used for FELM and WikiBio, and standard accuracy for HaluEval. Improvements over the best baseline are highlighted in green.

In the FELM dataset, our method achieves the highest maximum accuracy, surpassing baselines

Dataset	Method	Redundancy Rates (%) at Top-k=20						
		0	20	40	60	80	95	Avg
WebQ	Vanilla RALM	90.07	87.67	89.76	89.00	88.17	83.04	87.95
	Chunk Compression	90.77	89.74	90.21	90.96	90.90	87.01	89.93
	Long Agent	90.25	**92.31**	88.75	88.98	**90.95**	**89.89**	90.19
	Ours	**92.01**	91.33	**90.96**	**91.07**	90.93	89.22	**90.92**

Table 3: Performance on WebQ under different redundancy rates (Top-k=20). Values in parentheses indicate differences from Vanilla RALM. Green indicates improvement, red indicates decline.

Dataset	Method	Noise Rates (%) at Top-k=20						
		0	20	40	60	80	100	Avg
WebQ	Dynamic	90.79	91.87	90.75	91.00	89.23	87.87	**90.25**
	Avg	88.94	89.07	89.92	86.80	86.53	86.96	88.04
	Random	90.40	86.84	85.81	86.81	87.78	88.19	87.64

Table 4: Ablation study on clustering strategies under varying noise rates on WebQ.

like Vanilla, CoT, Link. Our method performs only slightly below Doc, which benefits from manually annotated golden documents. Its average accuracy reflects a modest improvement over the CEG baseline, demonstrating robustness across varying k values. For WikiBio GPT-3, our method performs competitively, slightly improving average accuracy over CEG and outperforming HalluDetector, Focus, and SelfCheckGPT, indicating consistent detection in biographical data. In HaluEval, our method records the highest performance, with a notable improvement over CEG, showcasing its effectiveness in open-domain settings.

Dataset	**Methods**	**Accuracy (Top-k, k=1∼10)**
FELM	Vanilla	58.18
	CoT	61.32
	Link	56.78
	Doc	**65.18**
	CEG	63.35 / 61.89
	Ours	64.03 / 62.26$^{+0.37}$
WikiBio	HalluDetector	74.82
	Focus	74.08
	SelfCheckGPT	70.55
	CEG	**76.58** / 74.14
	Ours	75.89 / 74.29$^{+0.15}$
HaluEval	CEG	78.10 / 76.93
	Ours	**78.85** / 77.87$^{+0.94}$

Table 5: Performance comparison on Hallucination-Detection datasets. Each entry shows Max / Avg accuracy over Top-k. Metric: Accuracy for HaluEval; Balanced Accuracy for WikiBio GPT-3 and FELM.

Overall, our method consistently outperforms or matches the best baselines across all datasets, with improvements in average accuracy. These results highlight its stability and generalizability, making it a promising approach for reducing hallucinations in applications like automated fact-checking.

6.3 Effectiveness of Clustering Strategies

To validate the effectiveness of our clustering method, we compare it with two alternative strategies—Average Clustering and Random Clustering—that match our dynamic clustering in both the number of clusters and the overall document compression ratio for a controlled comparison. Average Clustering groups documents by their similarity rank to the query and distributes them evenly across clusters, while Random Clustering assigns documents randomly from the top-k pool, maintaining the same number and size of clusters as dynamic clustering.

Table 4 compares these strategies on WebQ under different noise rates. Our method achieves highest average F1, outperforming baselines. Average Clustering and Random Clustering obtain lower F1, and degrade more under high noise. These results highlight the effectiveness of our entropy-guided dynamic clustering in document compression.

Further validation is provided by evaluating clustering consistency on the Musique dataset using GPT-4o-mini-2024-07-18 for document classification. We measure the intra-class clustering probability for documents labeled as "useful" or "noise,"

Top-$k = 20$											
τ	1	2	**3**	4	5	6	7	8	9	10	20
API Calls	5	4	3	3	3	2	2	2	2	2	1
F1 (%)	$72.86_{\pm 1.62}$	$73.07_{\pm 2.40}$	$76.85_{\pm 1.98}$	$77.15_{\pm 2.89}$	$74.70_{\pm 0.09}$	$76.69_{\pm 3.74}$	$76.51_{\pm 2.11}$	$74.88_{\pm 1.82}$	$77.63_{\pm 0.82}$	$73.55_{\pm 3.42}$	$73.71_{\pm 1.93}$
Top-$k = 100$											
τ	1	2	**3**	4	5	6	7	8	9	10	20
API Calls	7	6	6	5	5	5	4	4	4	4	3
F1 (%)	$78.54_{\pm 3.74}$	$78.33_{\pm 1.70}$	$80.73_{\pm 3.90}$	$80.21_{\pm 3.12}$	$76.66_{\pm 3.32}$	$76.86_{\pm 2.14}$	$76.80_{\pm 2.02}$	$77.41_{\pm 2.69}$	$77.85_{\pm 1.17}$	$77.78_{\pm 2.14}$	$77.97_{\pm 2.09}$

Table 6: Ablation study results on Musique dataset (GPT-4o-mini-2024-07-18) for varying τ at top-$k = 20$ and top-$k = 100$ (noise = 40%).

defined as:

$$\frac{\sum_{i,j \in \text{same-class}, i<j} \mathbb{1}[\text{cluster}(i) = \text{cluster}(j)]}{\binom{N_{\text{same-class}}}{2}}$$

Table 7 summarizes these metrics under varying top-k and noise levels, with random baselines using the same number of clusters. Our method exhibits probabilities exceeding random baselines, demonstrating significant semantic consistency and robustness, particularly under high noise.

Noise Rates (%) at Top-$k = 20$				
Metric	20	40	60	80
Useful Prob. (%)	35.87	36.59	36.43	39.37
Rand. Useful (%)	33.33	33.33	33.33	33.33
Noise Prob. (%)	31.43	34.97	35.22	35.05
Rand. Noise (%)	33.33	33.33	33.33	33.33
Noise Rates (%) at Top-$k = 100$				
Metric	20	40	60	80
Useful Prob. (%)	19.09	20.49	18.80	19.11
Rand. Useful (%)	14.56	14.62	14.29	14.29
Noise Prob. (%)	20.31	20.19	17.35	17.03
Rand. Noise (%)	14.56	14.62	14.29	14.29

Table 7: Clustering consistency metrics on Musique dataset (GPT-4o-mini-2024-07-18 classification) under varying top-k and noise levels, displayed for Top-$k = 20$ and Top-$k = 100$.

The modest gains over baselines stem from (i) the lightweight, dated nature of SimCSE-BERT (circa 2021), which constrains fine-grained semantic capture, and (ii) the binary "useful"/"noise" labels inadequately capturing nuanced real-world document interrelations.

6.4 Ablation Studies on τ

We conduct ablation studies on the Musique dataset with GPT-4o-mini-2024-07-18 (top-k = 20 and 100, noise = 40%), evaluating the initial cluster count (τ) across three independent trials. We report the mean and unbiased standard deviation of F1 scores and API call counts, with Λ fixed for consistency. The results, presented in Table 6, demonstrate stable performance across a wide range of τ, affirming the robustness of our design.

7 Conclusion

In this study, we design an efficient dynamic clustering algorithm and apply compression techniques to exploit fine-grained relationships between documents. Our method **EDC^2-RAG** enhances evidence quality by filtering noise and capturing detailed document relationships, achieving consistent performance improvements on three Hallucination-Detection datasets and three KQA datasets, thus demonstrating the strong robustness and broad applicability of our method. Extensive evaluations show that our approach outperforms competitive baselines across multiple metrics and model backbones.

Limitations

Our study has several limitations: 1) Due to time constraints, we did not validate the generalization ability of our method on more datasets and base models. 2) Using compression technique incurs some API consumption, but these costs are within an acceptable range. See Appendix A for details.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (62372260, 62276152), and Wuxi Research Institute of Applied Technologies, Tsinghua University. Weizhi Ma is also supported by Beijing Nova Program.

References

Jinheon Baek, Alham Fikri Aji, and Amir Saffari. 2023. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. In *Proceedings of the First Workshop on Matching From Unstructured and Structured Data (MATCHING 2023)*, pages 70–98, Toronto, ON, Canada. Association for Computational Linguistics.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, and 1 others. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Shiqi Chen, Yiran Zhao, Jinghan Zhang, I Chern, Siyang Gao, Pengfei Liu, Junxian He, and 1 others. 2023. Felm: Benchmarking factuality evaluation of large language models. *arXiv preprint arXiv:2310.00741*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023a. Enabling large language models to generate text with citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6465–6488, Singapore. Association for Computational Linguistics.

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

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2025. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *Advances in Neural Information Processing Systems*, 37:132876–132907.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2023. Unnatural instructions: Tuning language models with (almost) no human labor. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14409–14428.

Xuming Hu, Junzhe Chen, Xiaochuan Li, Yufei Guo, Lijie Wen, Philip S Yu, and Zhijiang Guo. 2023. Do large language models know about facts? *arXiv preprint arXiv:2310.05177*.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2022. Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S Yu. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE transactions on neural networks and learning systems*, 33(2):494–514.

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.

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.

Jaehyung Kim, Jaehyun Nam, Sangwoo Mo, Jongjin Park, Sang-Woo Lee, Minjoon Seo, Jung-Woo Ha, and Jinwoo Shin. 2023. Sure: Improving open-domain question answering of llms via summarized retrieval. In *The Twelfth International Conference on Learning Representations*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

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

Junkai Li, Yunghwei Lai, Weitao Li, Jingyi Ren, Meng Zhang, Xinhui Kang, Siyu Wang, Peng Li, Ya-Qin

Zhang, Weizhi Ma, and 1 others. 2024a. Agent hospital: A simulacrum of hospital with evolvable medical agents. *arXiv preprint arXiv:2405.02957*.

Junyi Li, Xiaoxue Cheng, Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2023. HaluEval: A large-scale hallucination evaluation benchmark for large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6449–6464, Singapore. Association for Computational Linguistics.

Mufei Li, Siqi Miao, and Pan Li. 2024b. Simple is effective: The roles of graphs and large language models in knowledge-graph-based retrieval-augmented generation. *arXiv preprint arXiv:2410.20724*.

Weitao Li, Junkai Li, Weizhi Ma, and Yang Liu. 2024c. Citation-enhanced generation for LLM-based chatbots. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1451–1466, Bangkok, Thailand. Association for Computational Linguistics.

Zichao Lin, Shuyan Guan, Wending Zhang, Huiyan Zhang, Yugang Li, and Huaping Zhang. 2024. Towards trustworthy llms: a review on debiasing and dehallucinating in large language models. *Artificial Intelligence Review*, 57(9):243.

Junyi Liu, Liangzhi Li, Tong Xiang, Bowen Wang, and Yiming Qian. 2023. TCRA-LLM: Token compression retrieval augmented large language model for inference cost reduction. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9796–9810, Singapore. Association for Computational Linguistics.

Qitan Lv, Jie Wang, Hanzhu Chen, Bin Li, Yongdong Zhang, and Feng Wu. 2024. Coarse-to-fine highlighting: Reducing knowledge hallucination in large language models. *arXiv preprint arXiv:2410.15116*.

Potsawee Manakul, Adian Liusie, and Mark Gales. 2023. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9004–9017, Singapore. Association for Computational Linguistics.

Ciyuan Peng, Feng Xia, Mehdi Naseriparsa, and Francesco Osborne. 2023. Knowledge graphs: Opportunities and challenges. *Artificial Intelligence Review*, 56(11):13071–13102.

Aniruddha Salve, Saba Attar, Mahesh Deshmukh, Sayali Shivpuje, and Arnab Mitra Utsab. 2024. A collaborative multi-agent approach to retrieval-augmented generation across diverse data. *arXiv preprint arXiv:2412.05838*.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D Manning. 2024. Raptor: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations*.

Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Le Hou, Kevin Clark, Stephen Pfohl, Heather Cole-Lewis, Darlene Neal, and 1 others. 2023. Towards expert-level medical question answering with large language models. *arXiv preprint arXiv:2305.09617*.

EuiYul Song, Sangryul Kim, Haeju Lee, Joonkee Kim, and James Thorne. 2024. Re3val: Reinforced and reranked generative retrieval. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 393–409, St. Julian's, Malta. Association for Computational Linguistics.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Han Wang, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. 2025. Retrieval-augmented generation with conflicting evidence. *arXiv preprint arXiv:2504.13079*.

Xiaohua Wang, Yuliang Yan, Longtao Huang, Xiaoqing Zheng, and Xuanjing Huang. 2023. Hallucination detection for generative large language models by Bayesian sequential estimation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15361–15371, Singapore. Association for Computational Linguistics.

Yuhao Wang, Ruiyang Ren, Junyi Li, Xin Zhao, Jing Liu, and Ji-Rong Wen. 2024. REAR: A relevance-aware retrieval-augmented framework for open-domain question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5613–5626, Miami, Florida, USA. Association for Computational Linguistics.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2023. Recomp: Improving retrieval-augmented lms with compression and selective augmentation. *arXiv preprint arXiv:2310.04408*.

Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. 2024. Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv:2401.11817*.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546, Online. Association for Computational Linguistics.

Wenhao Yu, Hongming Zhang, Xiaoman Pan, Peixin Cao, Kaixin Ma, Jian Li, Hongwei Wang, and Dong Yu. 2024. Chain-of-note: Enhancing robustness in retrieval-augmented language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14672–14685, Miami, Florida, USA. Association for Computational Linguistics.

Shengbin Yue, Wei Chen, Siyuan Wang, Bingxuan Li, Chenchen Shen, Shujun Liu, Yuxuan Zhou, Yao Xiao, Song Yun, Wei Lin, and 1 others. 2023. Disc-lawllm: Fine-tuning large language models for intelligent legal services. *arXiv preprint arXiv:2309.11325*.

Tianhang Zhang, Lin Qiu, Qipeng Guo, Cheng Deng, Yue Zhang, Zheng Zhang, Chenghu Zhou, Xinbing Wang, and Luoyi Fu. 2023. Enhancing uncertainty-based hallucination detection with stronger focus. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 915–932, Singapore. Association for Computational Linguistics.

Jun Zhao, Can Zu, Xu Hao, Yi Lu, Wei He, Yiwen Ding, Tao Gui, Qi Zhang, and Xuanjing Huang. 2024. LONGAGENT: Achieving question answering for 128k-token-long documents through multi-agent collaboration. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16310–16324, Miami, Florida, USA. Association for Computational Linguistics.

Lexin Zhou, Wout Schellaert, Fernando Martínez-Plumed, Yael Moros-Daval, Cèsar Ferri, and José Hernández-Orallo. 2024. Larger and more instructable language models become less reliable. *Nature*, 634(8032):61–68.

Appendix

A API costs and Latency Control

API Cost Evaluation. To better understand the overhead introduced by different RAG compression strategies, we evaluate API token consumption using the tiktoken.encoding_for_model("gpt-3.5-turbo") tokenizer, which closely approximates OpenAI's official billing. Costs are computed based on the pricing of `gpt-4o-mini-2024-07-18`: \$0.15 per million input tokens and \$0.60 per million output tokens. We report results on the Musique dataset with $k = 10$ and $k = 100$ under the noise-free setting, and compare our method against RALM, Long Agent, and Chunk Compression. The key metric is the total API usage cost (input + output) across the full pipeline, including both document processing and final answering.

	RALM	Chunk C.	Long Agent	**Ours**
$k = 10$, **noise=0**				
Avg Input	1388.45	2233.03	1843.42	2155.10
Avg Output	34.97	740.70	223.73	553.29
API Cost	2.29	7.79	4.11	6.55
Rel. Cost	1.00	3.40	1.79	2.86
$k = 100$, **noise=0**				
Avg Input	13542.94	20317.25	14406.18	14926.17
Avg Output	38.89	6026.16	395.58	1212.89
API Cost	20.55	66.63	23.98	30.12
Rel. Cost	1.00	3.24	1.17	1.46

Table 8: API cost ($\times 10^{-4}$) comparison on Musique under different k settings.

Cost Analysis. Our method achieves strong cost control, especially in large k settings, for two main reasons: (1) one-time document access ensures bounded input-token cost, and (2) query-aware cluster-based compression balances relevance and brevity, avoiding the excessive output tokens incurred by Chunk Compression. In low-k or noise-free settings, our cost is slightly higher than RALM and Long Agent. However, in such scenarios the total token usage is inherently small and noise is minimal (thus outside the target scenario of our method), making the overhead acceptable.

Efficiency Analysis. Our method is also efficient in runtime. We employ `SimCSE-BERT` (110M) as a lightweight encoder, and each document is encoded only once. The clustering step adds negligible overhead, and all summarization steps are **fully parallelizable**. In practice, this leads to wall-clock latency even lower than a single RALM query. These characteristics are consistent with our design goal of being **efficient**, as emphasized in the paper title.

B Implementation Details

B.1 Knowledge-QA Datasets and Retrieval Setup

Knowledge Question Answering (KQA) datasets are essential resources for evaluating a model's ability to perform knowledge reasoning and question-answering tasks. These datasets typically rely on external knowledge bases (e.g., knowledge graphs or text corpora) and design questions to test the

model's ability to retrieve information from the knowledge base and perform reasoning. In this work, we used three widely adopted datasets (Yu et al., 2024; Lv et al., 2024): WebQ (Berant et al., 2013) (single-hop), and 2WikiMultiHopQA (Ho et al., 2020) (hereafter referred to as 2Wiki) plus Musique (Trivedi et al., 2022) (both multi-hop).

WebQ is constructed by collecting questions posed by users in Google Suggest, with answers primarily based on the Freebase knowledge graph. The dataset is designed to test the model's ability to retrieve answers from structured knowledge bases while understanding natural language questions.

2WikiMultiHopQA is a multi-hop question answering dataset automatically constructed from Wikipedia. Each question requires reasoning over two or more Wikipedia articles to arrive at the correct answer. It is designed to test a model's ability to perform compositional reasoning and handle longer context chains compared to single-hop datasets.

Musique is a multi-hop QA dataset with complex, natural questions decomposed into multiple factoid subquestions. It is built from real queries and aligned with Wikipedia paragraphs, making it suitable for evaluating models on realistic multi-hop reasoning tasks that require integrating information across multiple documents.

In this setting, we follow prior work on retrieval-augmented generation (RAG) (Lv et al., 2024; Yu et al., 2024; Gao et al., 2023a), using the DPR retriever (Karpukhin et al., 2020) with the 2018 Wikipedia snapshot as the retrieval corpus, where each document contains approximately 100 words. For the three KQA datasets—WebQ, 2Wiki, and MuSiQue—we retrieve the top 1000 relevant documents for each test question. We apply string matching to identify whether each document contains the gold answer. A question is included in our final test set only if it has at least 100 documents with the answer (*has_answer*) and 100 without. This filtering yields test sets of approximately 400, 400, and 100 queries for WebQ, 2Wiki, and MuSiQue, respectively.

To build noisy retrieval scenarios, we inject the retrieved irrelevant documents into the retrieved set at controlled noise ratios. Document order is determined by similarity to the query. We vary the number of retrieved documents (top-k) from 5 to 100 and evaluate performance across different noise levels (0% to 100%) using the F1 score as the metric. The clustering threshold τ is set to 3 to balance document compression quality and API cost.

To evaluate the capability of our method in handling redundancy, we selected the k documents when each question was associated with top-20 documents. The remaining $20 - k$ documents were rewritten using ChatGPT. We define the redundancy rate as

$$r = \frac{20 - k}{20}$$

and construct datasets with redundancy rates of $r = 0.2,\ 0.4,\ 0.6,\ 0.8,$ and 0.95 , corresponding to $k = 16,\ 12,\ 8,\ 4,$ and 1 respectively.

B.2 Hallucination Detection Datasets and Retrieval Setup

Fact-checking (Hallucination Detection) is a natural language processing task aimed at verifying the truthfulness and accuracy of generated or stated content. Specifically, it involves determining whether a given piece of generated text (often machine-generated, such as summaries, answers, translations, etc.) or statement is truthful, partially truthful, or false based on available information sources (i.e., containing "hallucinations" or erroneous content). We conducted experiments on three widely used fact-checking tasks: the FELM World Knowledge Subset (Chen et al., 2023), the WikiBio GPT-3 Dataset (Manakul et al., 2023), and the HaluEval Dataset (Li et al., 2023).

These datasets were constructed leveraging the generative capabilities of large language models. Researchers design a series of tasks or scenarios, collected model-generated content, and annotate it using domain-specific background knowledge. Specifically, the datasets include various examples of model outputs, which are manually labeled to classify their truthfulness. Labels indicate whether the content is truthful, partially truthful, or entirely false (in this work, partially truthful and false are treated as false). This method not only captures potential issues in model-generated content but also provides high-quality benchmark datasets for evaluating models' fact-checking capabilities. Below is a sample question.

For the FELM World Knowledge Subset and WikiBio GPT-3 Dataset, the queries are statements. The retrieval corpus consisted of an October 2023 snapshot of Wikipedia from CEG (Li et al., 2024c), and the retriever used is SimCSE Bert (Gao et al.,

#Knowledge#: The nine-mile byway starts south of Morehead, Kentucky and can be accessed by U.S. Highway 60. Morehead is a home rule-class city located along US 60 (the historic Midland Trail) and Interstate 64 in Rowan County, Kentucky, in the United States. #Question#: What U.S Highway gives access to Zilpo Road, and is also known as Midland Trail?
#Right Answer#: U.S. Highway 60
#Hallucinated Answer#: U.S. Highway 70

Table 9: A sample question from the HaluEval Dataset.

2021). The evaluation metric is Balanced Accuracy (Balanced-Acc).

For the HaluEval Dataset, the retrieval corpus and setup were similar to those in other works (Karpukhin et al., 2020; Gao et al., 2023a), employing a 2018 snapshot of Wikipedia and a state-of-the-art BERT-based retriever, All-mpnet-base-v2[1]. The evaluation metric is Accuracy (Acc).

In this scenario, due to the lack of a unified retrieval paradigm or specifically constructed retrieval corpus for such datasets, the contribution of documents to answering questions was inherently limited. We cap the number of retrieved documents at 10. Since the number of documents is small, τ is set to 1 here to help the LLM summarize the documents more effectively.

B.3 Detailed Introduction of Baselines

The baselines for FELM include: 1) prompts enhanced with Chain-of-Thought (CoT) reasoning (Kojima et al., 2022), 2) prompts augmented with hyperlinks to reference documents, and 3) prompts supplemented by human-annotated reference documents (Chen et al., 2023).

The baselines for WikiBio GPT-3 comprise: 1) HalluDetector(Wang et al., 2023), which leverages external knowledge sources along with a dedicated classification model and a Naive Bayes classifier to identify hallucinations, and 2) Focus(Zhang et al., 2023), which employs a multi-stage decision-making framework combining both pre-retrieval and task-specific classifiers.

[1] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

C Prompts Used in Our Experiments

C.1 Hallucination Detection Datasets

C.1.1 FELM & HaluEval

Prompt of Compression

##Instruction##:
You are an AI assistant specializing in information extraction. Your task is to analyze a given statement and a set of related documents, and extract only the directly relevant information.

##Extraction Guidelines##:
- Identify key points, evidence, or details that **directly support, refute, or elaborate** on the statement.
- Ensure that the extracted content is **concise, objective, verifiable, and directly traceable** to the original documents.
- **Do not make inferences or draw conclusions** beyond what is explicitly stated.
- If the documents contain **no relevant information**, respond with **No content to extract.**

##Example Output Format##:
`{few-shots}`

##Statement##:
`{query}`

##Documents##:
`{docs}`

##Extracted Information##:

Eval Prompt of HaluEval

##Instruction##:
I want you to act as an answer judge. Given a question, two answers, and related knowledge, your objective is to select the best and correct answer without hallucination and non-factual information.
You should try your best to select the best and correct answer. If the two answers are the same, you can choose one randomly. If both answers are incorrect, choose the better one. You MUST select an answer from the two provided answers.
Think step by step. Give your reasoning first and then output your choice. Output in the following format:

”#Reasoning#: Your Reasoning
#Choice#: ”X””.
”X” should only be either ”Answer 1” or ”Answer 2”, rather than specific answer content.

##Knowledge##:
{knowledge}

##Question##:
{question}

##Answer 1##:
{answer 1}

##Answer 2##:
{answer 2}

C.1.2 WikiBio GPT-3

Prompt of Compression

##Instruction##:
You have been provided with a statement about {a person} and a collection of related documents. Your task is to extract relevant information from these documents that directly supports, refutes, or elaborates on the given statement.
Focus on identifying key points, evidence, or details that are clearly connected to the statement. Ensure the extracted content is concise, directly relevant, and maintains the context of the original documents.
The extracted content must be objective, verifiable, and directly traceable to the original documents. Avoid making inferences or drawing conclusions based on the extracted content.
If you find that the documents contain no relevant information, please output ”No content to extract”. Below is an example.

{One shot}

##Person##:
{person}

##Statement##:
{query}

##Documents##:
{docs}

##Extracted Information##:

Prompt of Evaluation

##Instruction##:
Assess whether the given statement about {a person} contains factual errors or not with the help of the reference docs.
If you believe given statement contains factual errors, your answer should be ”Nonfactual”; if there is no factual error in this statement, your answer should be ”Factual”. This means that the answer is ”Nonfactual” only if there are some factual errors in the given statement. When there is no factual judgment in the given statement or the given statement has no clear meaning, your answer should be ”Factual”. At the same time, please consider all aspects of the given statement thoroughly during the evaluation and avoid focusing excessively on any single factual aspect. Any factual errors should be considered.
Reference docs can be classified into three types: documents that support the response segment as ”Nonfactual”, documents that support the response segment as ”Factual”, and documents that provide supplementary or explanatory information for the response segment. Please consider these documents comprehensively when answering.
Think it step by step. Give your ”Reasoning” first and then output the ”Answer”.

##Statement##:
{statement}

##Reference docs##:
{passage}

##Output##:

C.2 Knowledge-QA Datasets

The prompts used for compression and generation in KQA tasks are shown below. These prompts differ from those used in previous datasets because we aim to elicit more informative chunks by having the model respond to the question first. This approach encourages the model to provide supporting evidence, which we then use to extract and compress relevant information. In contrast, directly prompting the model to summarize often leads it to provide answers directly without grounding them in the source content. If there is no strong formatting requirement, the quality of the LLM's re-

sponses remains stable; however, if strict formatting requirements are imposed, the response quality drops sharply, causing a significant decline in performance. Accordingly, during the final generation stage, we also have the model consider these outputted answers and their corresponding evidence. The model integrates all the evidence to select the most appropriate answer.

Prompt of Summarization

##Instruction##:
Please refer to the following text and answer the following question, **providing supporting evidence**.

##Question##:
{question}

##Reference text##:
{docs}

##Answer##:

Prompt of Response

##Task##:
Analyze the following set of candidate answers to a question and select the single most consistent/plausible answer based on majority consensus and logical coherence.

##Instructions##:
1. Carefully compare all candidate answers.
2. Identify the core factual claims or entities in each answer.
3. Group semantically equivalent answers (e.g., "1990", "the year 1990", "nineteen ninety").
4. Select the answer that: - Appears most frequently in the candidate set - Has strong internal consistency (no self-contradictions)
5. If multiple answers have equal validity, prefer the most specific and concise one.

##Format Requirements##:
Reasoning: Concise justification for selection.
Selected_Answer:...

Below is an example.
Candidate Answers: ["Paris", "The capital is Paris", "France", "paris", "It's Paris in France"]
Question: What is the capital of France?
Expected Response:
Reasoning: 4/5 answers directly state 'Paris'. While 'France' is incorrect alone, the most frequent and unambiguous consensus is 'Paris' Selected_Answer: Paris

##Candidate Answers##:
{answers}

##Question##:
{question}

D Additional Experimental Results

D.1 Experiments on Open-Source Models

Additional experiments are conducted using Qwen-3-8B in think mode on the TwoWiki dataset under a noise rate of 0%, constrained by available computational resources. These experiments, summarized in Table 10, utilized only this 8B model. The results reveal a notable performance gap compared to closed-source LLMs, attributable to the limited summarization and evidence-filtering capabilities of smaller models.

Top-k	RALM	Ours (Qwen-3-8B)
5	66.96	60.33
10	72.39	67.71
20	73.90	75.64
30	78.44	71.01
50	80.76	69.88
70	80.30	72.17
100	81.56	71.18

Table 10: Performance comparison on TwoWiki dataset (noise rate 0%) using Qwen-3-8B in think mode.

We anticipate improved outcomes with larger open-source models and intend to incorporate corresponding experiments in future iterations, subject to resource availability.

D.2 Additional Experimental Results on Noise Resistence

Tables 11 summarizes performance under varying noise levels with Top-$k = 20$.

Dataset	Method	Noise Rates (%) at Top-k=20						
		0	20	40	60	80	100	Avg
		gpt-3.5-turbo-1106						
MusiQue	Vanilla RALM	74.75	77.82	78.07	74.92	74.42	74.30	75.71
	Chunk Compression	74.15	75.38	77.70	78.01	71.89	76.08	75.54
	Long Agent	**84.21**	83.41	**79.02**	76.12	78.91	75.78	79.58
	Ours	82.55	**85.50**	78.28	**83.58**	**82.53**	**79.88**	**82.05**
WebQ	Vanilla RALM	90.07	89.62	90.12	90.14	**90.06**	86.36	89.40
	Chunk Compression	90.77	89.68	90.03	90.79	89.68	87.64	89.77
	Long Agent	90.49	**91.91**	90.54	89.46	88.81	**87.91**	89.85
	Ours	**90.79**	91.87	**90.75**	**91.00**	89.23	87.87	**90.25**
2Wiki	Vanilla RALM	**77.51**	71.48	71.84	68.40	67.57	66.01	70.47
	Chunk Compression	72.41	71.52	71.06	68.13	69.75	67.28	70.03
	Long Agent	76.06	**77.05**	74.20	71.07	69.35	66.99	72.45
	Ours	76.20	76.66	**76.75**	**72.43**	**72.92**	**68.99**	**73.99**
		gpt-4o-mini-2024-07-18						
MusiQue	Vanilla RALM	77.78	73.39	76.25	68.08	65.42	70.32	71.87
	Chunk Compression	75.67	75.33	76.82	75.29	67.41	68.26	73.13
	RAPTOR	72.07	78.46	75.95	71.15	76.64	70.78	74.18
	Long Agent	80.43	76.67	72.50	77.69	73.93	**78.05**	76.55
	Ours	**81.71**	**80.44**	**81.10**	**78.98**	**77.50**	74.91	**79.11**
WebQ	Vanilla RALM	85.07	89.89	90.82	88.70	88.27	85.20	87.99
	Chunk Compression	90.77	90.49	90.08	**90.53**	**89.40**	**86.98**	89.71
	Long Agent	**91.94**	**91.49**	**90.86**	90.13	88.60	86.79	**89.80**
	Ours	91.89	90.36	90.76	89.43	88.40	86.90	89.62
2Wiki	Vanilla RALM	**73.84**	73.03	71.43	69.03	**67.53**	**60.88**	**69.29**
	Chunk Compression	69.24	68.63	67.84	68.45	66.12	59.14	66.51
	Long Agent	71.33	**73.32**	70.52	64.27	62.69	57.29	66.57
	Ours	72.86	71.92	**72.58**	**69.60**	66.44	**60.88**	69.05

Table 11: Comparison of F1 scores under different noise levels at Top-k=20 on MusiQue, WebQ, and 2Wiki datasets for multiple retrieval methods.