CoFE-RAG: A Comprehensive Full-chain Evaluation Framework for Retrieval-Augmented Generation with Enhanced Data Diversity

Anonymous ACL submission

Abstract

Retrieval-Augmented Generation (RAG) aims to enhance large language models (LLMs) to generate more accurate and reliable answers with the help of the retrieved context from external knowledge source. Despite the advancements, the evaluation of these systems remains a crucial research area due to the following issues: 1) Limited data diversity of knowledge sources and query types; 2) Difficult to locate problems in the RAG pipeline; 3) Unable to evaluate retrieval performance effectively. To tackle these challenges, we propose a **Co**mprehensive **F**ull-chain **E**valuation (CoFE-RAG) framework to facilitate thorough evaluation across the entire RAG pipeline, including chunking, retrieval, reranking, and generation. To effectively evaluate the first three phases, we introduce multi-granularity keywords, including coarse-grained and fine-grained keywords, to assess the retrieved context instead of relying on the annotation of golden chunks. Moreover, we release a holistic benchmark dataset tailored for diverse data scenarios covering a wide range of document formats and query types. We demonstrate the utility of the CoFE-RAG framework by conducting experiments to evaluate each stage of RAG systems.

1 Introduction

In recent years, Retrieval-Augmented Generation (RAG) has emerged as a powerful paradigm for improving the performance of large language models (LLMs). By integrating the retrieved context with queries, RAG systems can generate more accurate and reliable responses, thus mitigating the issue of hallucinations that often plagues standalone generative models (Izacard et al., 2023). With the development of this technology, comprehensively evaluating all stages of RAG systems becomes increasingly important as it offers guidelines for future improvement and enhances real-world applications.

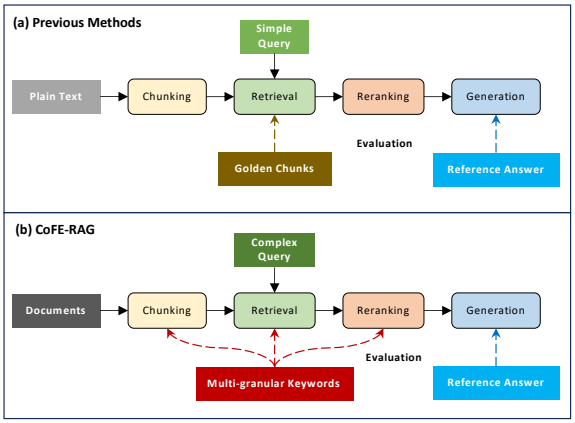

Figure 1: Overview of previous methods and the proposed CoFE-RAG for evaluating RAG systems.

Mainstream RAG evaluation methods can be broadly divided into reference-free and reference-required methods. Reference-free methods, such as AERS (Saad-Falcon et al., 2023) and RAGAS (ES et al., 2024), attempt to leverage LLMs to automatically evaluate context relevance, answer relevance, and faithfulness without benchmark datasets. Although these methods bypass the labor-intensive process of data labeling, they suffer from the absence of uniform evaluation standards and the potential risk of introducing subjective bias. On the other hand, reference-required methods, such as RECALL (Liu et al., 2023), RGB (Chen et al., 2024), and MultiHop-RAG (Tang and Yang, 2024), assess the output of the system against the ground truth reference.

Despite the promising capabilities of existing RAG evaluation methods, as illustrated in Fig. 1, they are still not effective due to the following issues: 1) The external knowledge base of existing evaluation methods basically derives from well-formed plain text crawled from HTML, which lacks data diversity and makes it difficult to incorporate complex documents such as PDF. Moreover, these methods mainly focus on simple queries, typically

factual queries, wherein the answers usually consist of specific entities. This narrows their applicability and hampers their ability to handle more complex analytical or tutorial queries. 2) Most previous methods predominantly evaluated the end-to-end results without performing step-by-step analysis. The RAG process can be divided into several stages: chunking, retrieval, reranking, and generation. By solely assessing the final generated outcomes, it becomes challenging to identify problems at specific stages within the RAG pipeline. Such approaches would result in poor interpretability and low optimization efficiency, hindering the ability to refine individual components effectively. 3) Previous methods evaluate the retrieval stage relying on the annotation of golden chunks with metrics such as Mean Reciprocal Rank and Hit Rate. Annotating all chunks is a tedious and labor-intensive process, and relabeling is required when the chunking strategy is modified.

To systematically address these challenges, we propose a Comprehensive Full-chain Evaluation (CoFE-RAG) framework to facilitate thorough evaluation across the entire RAG pipeline. We introduce multi-granularity keywords to effectively assess the chunking, retrieval, and reranking phases of RAG systems, which aims to address the dependency on golden chunk annotations for evaluation. The multi-granularity keywords encompass coarse-grained and fine-grained keywords. Specifically, coarse-grained keywords are the most representative and relevant words extracted from the query and context, serving as initial indicators for chunk relevance. Fine-grained keywords are formulated as a set of lists, where each list corresponds to an information point extracted from the context, providing detailed references for answering the query. CoFE-RAG employs coarse-grained keywords for the initial filtering of retrieved chunks and then uses fine-grained keywords to score the filtered results.

We also release a holistic benchmark dataset specifically designed for diverse data scenarios and can be used to evaluate all stages of RAG systems. This dataset is equipped with a knowledge base encompassing a wide range of document forms. Each example is annotated with the query, multi-granularity keywords, and reference answer. We define four types of queries, including factual, analytical, comparative, and tutorial queries. In order to balance annotation efficiency and annotation quality, we use a combination of LLM automatic annotation and manual review to annotate data.

In our experimental evaluation, we conduct experiments with various models for each stage of the RAG system to assess their strengths and weaknesses. The experimental results demonstrate that existing retrieval models excel in handling factual queries but struggle significantly with analytical, comparative, and tutorial queries. Furthermore, existing LLMs also perform poorly in leveraging the retrieved context to produce more accurate and reliable responses. The main contributions of this paper can be summarized as follows:

1) We propose the CoFE-RAG framework. To the best of our knowledge, this is the first work to comprehensively evaluate all stages of RAG systems and utilize multi-granularity keywords to improve the evaluation of retrieval results.

2) This paper releases a benchmark dataset containing four types of queries, multi-granular keywords, and reference answers, along with a knowledge base covering various document formats to evaluate RAG systems in diverse data scenarios.

3) We conduct a series of experiments to benchmark existing methods at each stage of RAG systems, which facilitates an in-depth analysis of the performance of RAG systems.

2 Related Work

2.1 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) is a technology that combines information retrieval and text generation. It enables LLMs to incorporate retrieved context along with the query to generate more accurate and credible responses, thus reducing the generation of hallucinations (Izacard et al., 2023). Shi et al. (2023), Yu et al. (2023b), and Gao et al. (2023) have explored various methods to enhance the effectiveness of retrieval mechanisms. Yu et al. (2023a) and Tang et al. (2024) investigated the potential for LLMs to directly generate context, effectively bypassing the need for a separate retriever. Ding et al. (2024), Wang et al. (2023), and Jeong et al. (2024) used adaptive methods to dynamically determine whether retrieval is necessary to answer a query. Yoran et al. (2023), Li et al. (2023a), and Xu et al. (2024) aim to enhance the robustness of RAG models. Jiang et al. (2023), Asai et al. (2023), and Liu et al. (2024) focused on optimizing the overall RAG pipeline.

Type	Description
Factual	Seeking specific, clear facts or evidence *Where is the capital of the United States?*
Analytical	Seeking analysis for concepts, terms *Why is the earth warming?*
Comparative	Seeking comparisons in different dimensions *What are the differences between A and B?*
Tutorial	Seeking the steps to perform a task or process *What are the steps to install TensorFlow?*

Table 1: Definitions and examples of four types of queries.

2.2 Retrieval-Augmented Generation Evaluation

Evaluating the performance of RAG systems has garnered widespread attention, which enables a deeper understanding of the capabilities and limitations of RAG systems. Evaluation methods for RAG systems can be divided into two main categories: reference-free and reference-required methods. Reference-free methods, represented by AERS (Saad-Falcon et al., 2023) and RAGAS (ES et al., 2024), use LLMs to automatically evaluate context relevance, answer faithfulness, and answer relevance without relying on benchmark datasets. On the other hand, reference-required evaluations utilize ground truth references to assess the retrieval or generation process, remaining the predominant method for evaluating RAG systems. For instance, RGB (Chen et al., 2024) aims to evaluate noise robustness, negative rejection, information integration, and counterfactual robustness abilities of LLMs. RECALL (Liu et al., 2023) construct a benchmark to evaluate the ability of LLMs to discern the reliability of external knowledge. CRUD-RAG (Lyu et al., 2024) constructs a large-scale and more comprehensive benchmark to evaluate RAG applications in four distinct tasks: create, read, update, and delete. MultiHop-RAG (Tang and Yang, 2024) propose a comprehensive dataset for evaluating multi-hop queries using a knowledge base derived from news article. However, these methods fail to provide a comprehensive full-chain evaluation of RAG systems and suffer from limited data diversity.

3 Preliminaries

In this paper, we divide the whole process of RAG into four stages, including chunking, retrieval, reranking, and generation. **Chunking** involves dividing the entire knowledge base into chunks according to chunk size with overlap between adjacent chunks. **Retrieval** refers to converting both the query and chunks into numerical vectors using the embedding model and then selecting the top-K chunks as initial retrieved results based on the similarity between the query vector and the chunk vector. **Reranking** refers to using the reranking model to understand the query and chunk to further rank the initial retrieved chunks and select the top-k as the final results. **Generation** means leveraging LLMs to generate the response based on the query and final retrieved results.

4 The CoFE-RAG Framework

In this section, we demonstrate the proposed CoFE-RAG framework in detail, which aims to evaluate all phases of RAG systems containing chunking, retrieval, reranking, and generation. We introduce multi-granularity keywords to facilitate a robust evaluation of chunking, retrieval, and reranking performance.

4.1 Data Collection

4.1.1 Document Collection

We collect a variety of documents from open-source websites, encompassing multiple formats such as PDF, DOC, PPT, and XLSX. These documents cover various industries, including finance, technology, medical care, commerce, Internet, etc. Their content includes industry reports, manuals, statistics, etc., providing a rich source of information suitable for evaluating RAG systems. The majority of the documents were created in recent years, with a considerable portion dating from this year (2024). This time frame surpasses the knowledge cutoff range of many widely used LLMs, such as GPT-4 (OpenAI, 2023a).

4.1.2 Document Parsing and Splitting

In the initial phase, we parse the documents to extract content suitable for processing by language models. Documents in PDF, PPT, and DOC formats are parsed by the LlamaIndex tool (Liu, 2022), and the Pandas (pandas development team, 2020) library is used to table content from XLSX documents. Then we split the content of each document into multiple fragments for subsequent data construction. To address the potential absence of title information in intermediate fragments, we employ GPT-4 to extract key information from the first fragment of each document. Such key information is

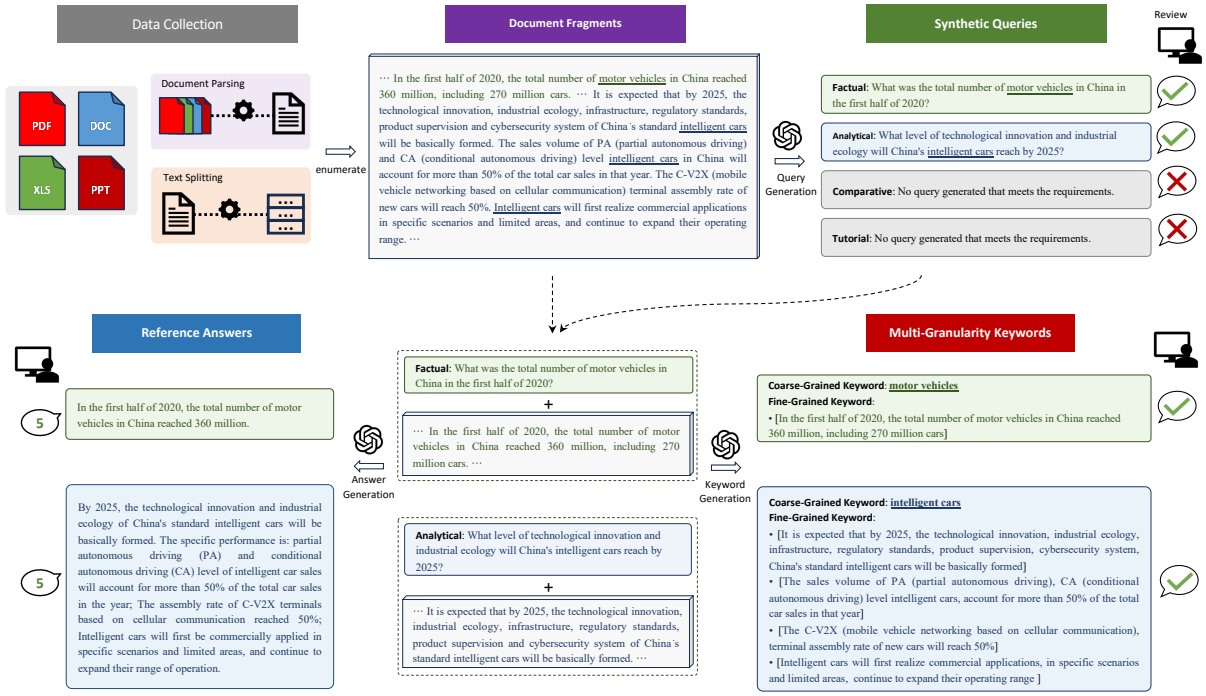

Figure 2: An example of the constructing process of query, multi-granularity keywords, and reference answers.

then used as the title and appended to the beginning of each fragment.

4.2 Data Construction

The data construction process includes query generation, multi-granularity keywords generation, and reference answer generation, which is illustrated in Fig. 2.

4.2.1 Query Generation

We define four distinct types of queries, including factual, analytical, comparative, and tutorial queries. Definitions for each query type are demonstrated in Table 1. We meticulously design prompts including task instruction, demonstration examples, and document fragment. For each document fragment, we employ GPT-4 to thoroughly comprehend the content and generate corresponding queries for all four types. It should be noted that if no applicable query can be generated for a specific query type that meets the requirements, the corresponding output will be *It cannot be generated*.

We establish three essential criteria that a high-quality query must satisfy: 1) The query must be clear, precise, and free from grammatical errors, avoiding the use of ambiguous pronouns such as he, it, this, etc; 2) The query must align with the definition of its respective query type; 3) The query should be inferable from the information presented in the corresponding document fragment. Then we employ well-trained annotators to assess the acceptability of each query. A query is deemed acceptable only if it fully complies with all the criteria.

4.2.2 Multi-granularity Keywords Generation

To address the issue of evaluating retrieval performance depending on golden chunks, we propose annotating multi-granularity keywords for each query instead. This approach eliminates the need for the labor-intensive process of re-labeling when the chunking strategy changes.

The multi-granularity keywords consist of coarse-grained and fine-grained keywords. Specifically, coarse-grained keywords are the most representative and relevant words extracted from the query and fragment, typically comprising one or two words that succinctly encapsulate the main topic. Fine-grained keywords are formulated as a set of lists, with each list corresponding to an information point extracted from the fragment. The elements of the list are specific spans of text taken directly from the original fragment, serving as reference points for answering the query.

For example in Fig. 2, for the analytical query *What level of technological innovation and industrial ecology will China's intelligent cars reach by 2025?*, we first extract the coarse-grained keywords

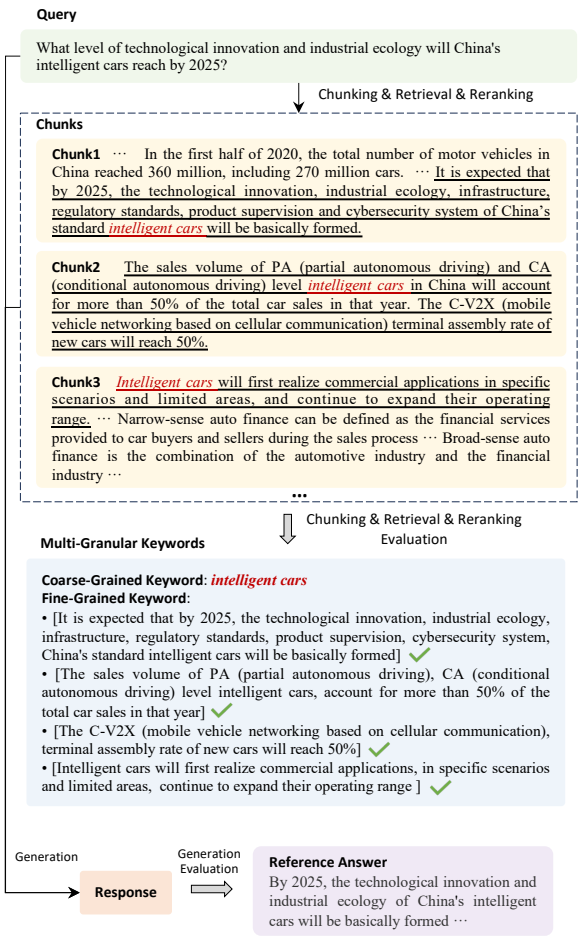

Figure 3: An example of the proposed CoFE-RAG framework. The red words denote coarse-grained keywords. The underlined words denote the corresponding content for fine-grained keywords.

intelligent cars. To adequately address this query, we identify four distinct information points from the document fragment, each corresponding to a separate list.

Like the query generation process, we utilize GPT-4 to generate coarse-grained keywords and fine-grained keywords with carefully designed prompt containing task instruction, demonstration examples, query, and document fragment. If no suitable coarse-grained or fine-grained keyword can be generated that meets the requirements, the resulting output list will be left blank.

To ensure quality, well-trained annotators are then employed to evaluate the acceptability of all coarse-grained keywords and calculate the acceptance rate for fine-grained keywords. We retain only those examples where all coarse-grained keywords are accepted and the acceptance rate for fine-grained keywords exceeds 80%. The acceptance rate means how many correct lists are recalled, with a list considered correct only when each of its elements is correct. This meticulous process ensures the reliability and quality of the multi-granularity keywords, facilitating a robust and nuanced evaluation of retrieval-augmented generation systems.

Query Type	Raw	Final	Accept Rate(%)
Factual	1786	1340	75.0
Analytical	1489	746	50.1
Comparative	903	498	55.1
Tutorial	513	242	47.2
Total	4691	2826	60.2

Table 2: The distribution of query types, where *Raw* and *Final* represent the number of queries before and after manual review.

4.2.3 Reference Answer Generation

We provide a reference answer for each query to serve as a benchmark for evaluating the generation performance of RAG systems. Similarly, we employ GPT-4 to generate reference answers with meticulously crafted prompt. To ensure the quality of these reference answers, we ask annotators to evaluate them based on five criteria: fluency, accuracy, relevance, readability, and practicality. Each answer is scored on a scale from 1 to 5 points. We then filter out samples with answer scores below 4 points to maintain a high standard of quality. This stringent filtering process ensures that only high-quality reference answers are retained for evaluations.

4.3 Data Statistics

After three generation steps, we obtain examples consisting of queries, multi-granularity keywords, and reference answers. The generated data went through rigorous human review to ensure high quality. The acceptance rates by human annotators were 92.2% for synthetic queries, 87.3% for synthetic multi-granularity keywords, and 74.8% for generated reference answers. The overall acceptance rate after manual review was 60.2%.

The distribution of query types is detailed in Table 2. Among all types of queries, factual queries account for the largest proportion. This is attributable to the higher generation rate and the larger proportion of factual queries meeting the filtering criteria. Conversely, tutorial queries have the smallest proportion, largely due to the original documents containing limited tutorial information, which in turn results in fewer queries of this type.

4.4 Evaluation Metrics

We utilize a series of evaluation metrics to assess all stages of RAG systems. The detailed process of the proposed CoFE-RAG framework is illustrated in Fig. 3.

4.4.1 Chunking & Retrieval & Reranking Evaluation

The proposed CoFE-RAG aims to evaluate the chunking, retrieval, and reranking quality based on multi-granularity keywords rather than golden chunks. For the top-K retrieval chunks, we regard coarse-grained keywords as a loose constraint and filter out the results that do not contain any coarse-grained keywords. This step ensures that only contextually relevant chunks are considered for further evaluation. After filtering, we concatenate the remaining chunks and use two metrics to evaluate the results, including Recall and Accuracy.

Specifically, Recall evaluates how many fine-grained keyword lists are correctly recalled from all the annotated fine-grained keyword lists of the whole dataset.

$$\text{Recall} = \frac{\sum_{i=1}^{N} |C_i|}{\sum_{i=1}^{N} |A_i|} \tag{1}$$

where N is the number of examples, C_i is the set of correctly recalled lists of the i-th example, A_i is the set of all annotated lists of the i-th example. Accuracy reflects the ratio of completely correct retrieved results among all examples. A result is considered completely correct when all fine-grained keyword lists of an example are correctly recalled.

$$\text{Accuracy} = \frac{\sum_{i=1}^{N} \mathbb{I}(S_i = 1)}{N} \tag{2}$$

where S_i denotes the recall rate of fine-grained keyword lists for the i-th example.

4.4.2 Generation Evaluation

We utilize various metrics to evaluate the quality of generated response, including BLEU (Papineni et al., 2002), Rouge-L (Lin, 2004), Faithfulness, Relevance, and Correctness.

Specifically, BLEU measures the similarity between the generated response and the reference answer by calculating the n-gram exact match between them. Rouge-L measures the similarity between the generated response and the reference answer by the Longest Common Subsequence (LCS), focusing on order and coverage. Faithfulness, Relevance, and Correctness are calculated by the built-in evaluator of LlamaIndex, which uses GPT-4 to automatically evaluate via in-context learning. Faithfulness evaluates whether a generated response is faithful to the retrieved context. Relevance evaluates the relevancy of retrieved context and generated response to a query. Correctness evaluates the correctness of the system. This evaluator can output a score between 1 and 5 based on the query, generated response, and reference answer, where 1 is the worst and 5 is the best, as well as the reason for the score. Score represents the average correctness score of all examples. Pass is defined as the ratio of examples whose score is greater than or equal to 4.

5 Experiments

The proposed dataset can be used as a benchmark for evaluating RAG systems in more diverse data scenarios. In this section, we conduct experiments to demonstrate the effect of retrieval, reranking, generation, and chunking, respectively.

5.1 Effect of Retrieval

We first split all documents into chunks with a size of 512 tokens, with an overlap of 100 tokens between two adjacent chunks. We use the top 30 chunks as initial retrieved results to evaluate retrieval performance. We choose a variety of embedding models, include text-embedding-ada-002 and text-embedding-3-large by OpenAI (OpenAI, 2023b), stella-large-zh-v2 (infgrad, 2023), m3e-large (Wang Yuxin, 2023), piccolo-large-zh-v2 (Huang et al., 2024), gte-large-zh (Li et al., 2023b), bge-base-zh-v1.5, and bge-large-zh-v1.5 (Xiao et al., 2023).

The experimental results for different embedding models are shown in Table 3. We observed that the bge-large model outperforms others in terms of Recall and Accuracy across all types of queries and overall performance. This indicates that the model has a strong ability to capture the semantic relationship between queries and their context. Among all embedding models, factual queries generally perform better than analytical, comparative, and tutorial queries. This may be because the relevant context for factual queries is usually contained within a single chunk, making it easier to retrieve. In contrast, other types of queries are more complex, with relevant context potentially spread across multiple chunks, making retrieval more challeng-

Embedding	Factual		Analytical		Comparative		Tutorial		Overall	
	Recall	Accuracy	Recall	Accuracy	Recall	Accuracy	Recall	Accuracy	Recall	Accuracy
text-embedding-ada-002	0.6288	0.5833	0.6027	0.5691	0.6067	0.5594	0.5772	0.4938	0.6080	0.5669
text-embedding-3-large	0.6763	0.6385	0.6603	0.6067	0.6471	0.6056	0.6131	0.5477	0.6565	0.6157
stella-large	0.7525	0.6968	0.7091	0.6443	0.6700	0.6298	0.7006	0.6224	0.7142	0.6638
m3e-large	0.6915	0.6303	0.6496	0.5732	0.6096	0.5493	0.6608	0.5726	0.6566	0.5952
piccolo-large	0.7442	0.6893	0.6827	0.6255	0.6630	0.6237	**0.7070**	0.6100	0.7011	0.6532
gte-large	0.6898	0.6378	0.6537	0.5933	0.6348	0.5875	0.6752	0.5892	0.6641	0.6122
bge-base	0.7470	0.6871	0.7108	0.6443	0.6717	0.6258	0.6855	0.6141	0.7114	0.6578
bge-large	**0.7612**	**0.7028**	**0.7124**	**0.6591**	**0.6735**	**0.6378**	0.7030	**0.6224**	**0.7190**	**0.6720**

Table 3: Retrieval performance of baselines on the dataset.

Reranking	Factual		Analytical		Comparative		Tutorial		Overall	
	Recall	Accuracy	Recall	Accuracy	Recall	Accuracy	Recall	Accuracy	Recall	Accuracy
jina-reranker-v2-base	0.7175	0.6699	0.6559	0.5987	0.6096	0.5714	0.6330	0.5560	0.6633	0.6231
bce-reranker-base	0.7251	0.6721	**0.6678**	0.6040	0.6102	0.5775	0.6457	0.5613	**0.6719**	0.6270
bge-reranker-base	0.7220	0.6714	0.6537	0.5919	**0.6120**	0.5782	0.6417	0.5602	0.6654	0.6238
bge-reranker-large	**0.7262**	**0.6759**	0.6625	**0.6067**	0.6114	**0.5795**	**0.6529**	**0.5685**	0.6714	**0.6306**

Table 4: Reranking performance of baselines on the dataset.

LLM	BLEU	Rouge-L	Faithfulness	Relevance	Correctness	
					Pass	Score
Qwen2-0.5B	0.1650	0.3126	0.7367	0.7824	0.3443	2.7093
Qwen2-1.5B	0.1437	0.3022	0.7385	0.7785	0.3439	2.9338
Qwen2-7B	0.2649	0.4925	0.8372	0.9253	0.6348	3.7699
Llama2-7B	0.2323	0.3345	0.8461	0.7611	0.3808	3.1175
ChatGLM3-6B	0.2662	0.4100	0.8659	0.8255	0.5180	3.3942
Claude-2.1	0.2141	0.4060	0.8742	0.9018	0.5612	3.3349
Claude-3-Opus	0.2623	0.5209	0.8846	**0.9565**	0.6684	3.8613
GPT-3.5-Turbo	0.2934	0.4215	**0.9222**	0.9176	0.5690	3.5290
GPT-4o	**0.4565**	**0.5519**	0.8977	0.9441	**0.7389**	**4.0777**

Table 5: Generation performance of baselines on the dataset.

ing. Additionally, existing retrieval models generally suffer from poor performance, highlighting the ongoing challenge of searching relevant chunks that accurately match the query.

5.2 Effect of Reranking

We rerank the initial retrieved results and select the top 4 chunks to assess the reranking performance. To evaluate the reranking methods, we use the chunks retrieved by bge-large-zh-v1.5 and conduct experiments with various reranking models, including jina-reranker-v2-base-multilingual (Günther et al., 2023), bce-reranker-base (NetEase Youdao, 2023), bge-reranker-base, and bge-reranker-large (Xiao et al., 2023).

The experimental results with different reranking models are reported in Table 4. We can observe that bge-reranker-large stands out with the best performance. Additionally, using the reranked top 4 results proves less effective compared to utilizing all retrieved results. This indicates that the current reranking methods are still not performing well and may miss some relevant chunks. After the retrieval and reranking phases, the performance of factual queries still outperforms the other three queries, which further demonstrates our analysis.

5.3 Effect of Generation

The generation stage has a great impact on the RAG system, as different LLMs vary in their ability to integrate queries and retrieved chunks to generate responses. We feed the query and top 4 chunks reranked by bge-reranker-large into various LLMs for evaluation. Our experiments encompass a diverse array of LLMs, including GPT-4o, GPT-3.5-Turbo (OpenAI, 2023a), Claude-2.1, Claude-3-Opus (Anthropic, 2023), Qwen2 (qwe, 2024), Llama2 (Touvron et al., 2023), and ChatGLM3 (Du et al., 2022).

The generation performance with different

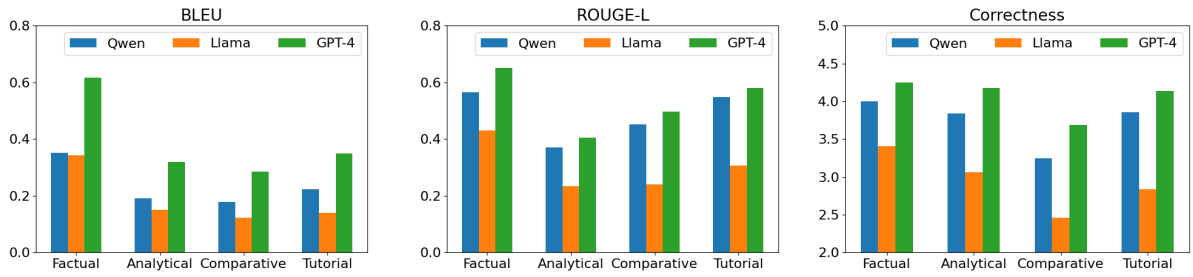

Figure 4: BLEU, Rouge-L, and Correctness score over different query types.

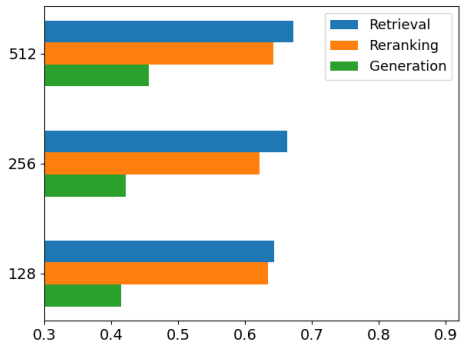

Figure 5: Experimental results with different chunk size. The retrieval and reranking phases are evaluated by Accuracy, while the generation stage is assessed by BLEU.

LLMs is reported in Table 5. We observed that GPT-4 achieved the best results across various LLMs, significantly outperforming other models. Models with larger parameters, such as GPT-4 and Claude-3 generally perform better than models with smaller parameters, such as Qwen-7B, Llama-7B. This may be because models with larger parameters have stronger reasoning and generalization capabilities, reduce the risk of hallucinations, and can handle more complex tasks. Qwen2-7B performs the best among Qwen2-7B, Llama2-7B, and ChatGLM-6B, demonstrating its ability to generate accurate and reliable answers in the RAG system.

To provide a more detailed comparison, we present the BLEU, Rouge-L, and Correctness scores for Qwen2-7B, Llama2-7B, and GPT-4 across different query types in Figure 4. We can observe that the performance on factual queries generally outperforms the other query types. This observation highlights the complexity and challenging nature of analytical, comparative, and tutorial queries, suggesting that further efforts are required to enhance performance on these more intricate query types.

5.4 Effect of Chunking

To demonstrate the effect of chunking, we conduct experiments with chunk sizes of 128, 256, and 512 tokens, respectively. The corresponding overlap sizes are set to 25, 50, and 100 tokens, and the final number of chunks after reranking is set to 16, 8, and 4, respectively. For these experiments, we employed the bge-large-zh-v1.5 model for retrieval, the bge-reranker-large model for reranking, and GPT-4o for generation. The performance with different chunk sizes is illustrated in Fig. 5. We can observe that using a size of 512 can achieve better retrieval, reranking, and generation performance. This indicates that larger chunks are more effective at preserving the original information from the document, thereby benefiting the ability of the system to address complex queries.

6 Conclusion

In this paper, we present the CoFE-RAG framework to facilitate thorough evaluation across the entire RAG pipeline. We introduce multi-granularity keywords to assess the retrieved context instead of relying on the annotation of golden chunks, which can effectively evaluate chunking, retrieval, and reranking performance particularly when the chunking strategy changes. Moreover, we release a holistic benchmark dataset tailored for diverse data scenarios covering a wide range of document formats and query types. The experimental results indicate that while there have been significant advancements, current methods still have substantial room for improvement, particularly in handling complex query types and diverse knowledge sources.

Limitations

In the evaluation experiments, we implemented a series of experiments with different models at each stage of the RAG systems to demonstrate the effectiveness of the CoFE-RAG framework. Given that current representative RAG methods mainly focus on augmenting or modifying specific stages to improve model performance for particular query types, we opted against conducting direct experiments on these existing methods.

In addition, to prove the quality of the evaluation dataset, we employ well-trained annotators to evaluate the acceptability of the generated data. In many cases, the manual review process can be removed due to the high level of acceptability observed. In the future, we will explore more reliable methods to achieve efficient automatic evaluation of RAG systems.

References

2024. Qwen2 technical report.

Anthropic. 2023. Claude 2. Large language model.

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

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2024. Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17754–17762.

Hanxing Ding, Liang Pang, Zihao Wei, Huawei Shen, and Xueqi Cheng. 2024. Retrieve only when it needs: Adaptive retrieval augmentation for hallucination mitigation in large language models. *CoRR*, abs/2402.10612.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335.

Shahul ES, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. Ragas: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2024 - System Demonstrations, St. Julians, Malta, March 17-22, 2024*, pages 150–158. Association for Computational Linguistics.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 1762–1777. Association for Computational Linguistics.

Michael Günther, Louis Milliken, Jonathan Geuter, Georgios Mastrapas, Bo Wang, Han Xiao, and AI Jina. 2023. Jina embeddings: A novel set of high-performance sentence embedding models. In *The 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS)*, page 8.

Junqin Huang, Zhongjie Hu, Zihao Jing, Mengya Gao, and Yichao Wu. 2024. Piccolo2: General text embedding with multi-task hybrid loss training. *CoRR*, abs/2405.06932.

infgrad. 2023. stella-large-zh. `https://huggingface.co/infgrad/stella-large-zh`.

Gautier Izacard, Patrick S. H. Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *J. Mach. Learn. Res.*, 24:251:1–251:43.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C. Park. 2024. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. *CoRR*, abs/2403.14403.

Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 7969–7992. Association for Computational Linguistics.

Daliang Li, Ankit Singh Rawat, Manzil Zaheer, Xin Wang, Michal Lukasik, Andreas Veit, Felix X. Yu, and Sanjiv Kumar. 2023a. Large language models with controllable working memory. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 1774–1793. Association for Computational Linguistics.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023b. Towards general text embeddings with multi-stage contrastive learning. *CoRR*, abs/2308.03281.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Jerry Liu. 2022. LlamaIndex.

Yanming Liu, Xinyue Peng, Xuhong Zhang, Weihao Liu, Jianwei Yin, Jiannan Cao, and Tianyu Du. 2024. RA-ISF: learning to answer and understand from retrieval augmentation via iterative self-feedback. *CoRR*, abs/2403.06840.

Yi Liu, Lianzhe Huang, Shicheng Li, Sishuo Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023. RECALL: A benchmark for llms robustness against external counterfactual knowledge. *CoRR*, abs/2311.08147.

Yuanjie Lyu, Zhiyu Li, Simin Niu, Feiyu Xiong, Bo Tang, Wenjin Wang, Hao Wu, Huanyong Liu, Tong Xu, and Enhong Chen. 2024. CRUD-RAG: A comprehensive chinese benchmark for retrieval-augmented generation of large language models. *CoRR*, abs/2401.17043.

Inc. NetEase Youdao. 2023. Bcembedding: Bilingual and crosslingual embedding for rag. https://github.com/netease-youdao/BCEmbedding.

OpenAI. 2023a. GPT-4 technical report. *CoRR*, abs/2303.08774.

OpenAI. 2023b. text-embedding-ada-002.

The pandas development team. 2020. pandas-dev/pandas: Pandas.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. 2023. ARES: an automated evaluation framework for retrieval-augmented generation systems. *CoRR*, abs/2311.09476.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2023. REPLUG: retrieval-augmented black-box language models. *CoRR*, abs/2301.12652.

Qiaoyu Tang, Jiawei Chen, Bowen Yu, Yaojie Lu, Cheng Fu, Haiyang Yu, Hongyu Lin, Fei Huang, Ben He, Xianpei Han, Le Sun, and Yongbin Li. 2024. Self-retrieval: Building an information retrieval system with one large language model. *CoRR*, abs/2403.00801.

Yixuan Tang and Yi Yang. 2024. Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries. *CoRR*, abs/2401.15391.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Yile Wang, Peng Li, Maosong Sun, and Yang Liu. 2023. Self-knowledge guided retrieval augmentation for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 10303–10315. Association for Computational Linguistics.

He sicheng Wang Yuxin, Sun Qingxuan. 2023. M3e: Moka massive mixed embedding model.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighof. 2023. C-pack: Packaged resources to advance general chinese embedding. *CoRR*, abs/2309.07597.

Shicheng Xu, Liang Pang, Mo Yu, Fandong Meng, Huawei Shen, Xueqi Cheng, and Jie Zhou. 2024. Unsupervised information refinement training of large language models for retrieval-augmented generation. *CoRR*, abs/2402.18150.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2023. Making retrieval-augmented language models robust to irrelevant context. *CoRR*, abs/2310.01558.

Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. 2023a. Generate rather than retrieve: Large language models are strong context generators. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Zichun Yu, Chenyan Xiong, Shi Yu, and Zhiyuan Liu. 2023b. Augmentation-adapted retriever improves generalization of language models as generic plug-in. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 2421–2436. Association for Computational Linguistics.

A Experimental Results on English Queries

Query Type	English		Chinese	
	Count	Ratio	Count	Ratio
Factual	364	36.3%	1340	47.4%
Analytical	260	25.9%	746	26.4%
Comparative	226	22.5%	498	17.6%
Tutorial	153	15.3%	242	8.6%
Total	1003	-	2826	-

Table 6: The distribution of query types on English and Chinese queries.

The proposed dataset contains queries in both Chinese and English languages. The distributions of the query types are shown in Table 6. In the

	Factual		Analytical		Comparative		Tutorial		Overall	
	Recall	Accuracy	Recall	Accuracy	Recall	Accuracy	Recall	Accuracy	Recall	Accuracy
Retrieval	0.7648	0.7308	0.6661	0.4077	0.6348	0.4425	0.6519	0.5098	0.6765	0.5484
Reranking	0.7402	0.7198	0.5931	0.3731	0.5703	0.4159	0.5711	0.4771	0.6129	0.5244

Table 7: Retrieval and reranking performance of baselines on the English queries.

	BLEU	Rouge-L	Faithfulness	Relevance	Correctness	
					Pass	Score
Generation	0.5016	0.5666	0.9332	0.9671	0.7358	4.0304

Table 8: Generation performance of baselines on the English queries.

Format	Avg. Tokens	Avg. Pages	Count
PDF	88495.9	115.4	485
PPT	5662.6	25.9	269
DOC	7894.3	20.2	433
XLSX	3565.2	3.2	227
Total	-	-	1414

Table 9: Distributions of documents in different formats.

main body of the paper, we mainly conduct experiments and analysis on Chinese queries. In the appendix, we present benchmark experimental results on English queries with the same document base.

In the implementations, we employ bge-large-en-v1.5 as the embedding model, bge-reranker-large as the reranking model, and GPT-4o as LLMs for generation. We use a chunk size of 512 tokens with an overlap of 100 tokens. We first retrieve the top 30 chunks using the embedding model. Then we rank these chunks using the reranking model and select the top 4 chunks for generation. The experimental results are demonstrated in Table 7 and Table 8.

B Distributions of the Documents

The distributions of documents across different formats are shown in Table 9.

C An Example of the Dataset

We leverage coarse-grained keywords and fine-grained keywords to evaluate the chunking, retrieval, and reranking stages, while using reference answer to assess the generated response. Here we present an example in json format:

{
"**query type**": "Analytical",
"**query**": "What are the main responsibilities of a Program Support Assistant (Office Automation) in the Research and Development Service?",
"**coarse-grained keywords**": [
"Program Support Assistant"
],
"**fine-grained keywords**": [
["prepare proposal forms", "monitors each research proposal", "reviews each proposal for conformance with policies, regulations, format, completeness and accuracy"],
["Establishes agendas", "prepares or coordinates preparation of monthly packages", "schedules meetings", "takes notes of the proceedings", "converts the notes into formal minutes", "prepares these minutes for the chairperson's review and signature", "distributes them"],
["Monitors each project", "notifies the investigators of the need for reports", "incorporates these reports into the project package", "initiates and maintains a file of all abstracts and publications"],
["Manages all project data activities utilizing the PROMISE system"],
["Acts on requests for information", "answers questions related to procedure", "providing copies of application forms and instructions", "reviewing with them the major requirements"],
["Establishes and maintains a personal calendar", "notation of action dates for initial submission of material or follow-up"],
["Performs other duties as assigned"]
],
"**reference answer**":
"The main responsibilities of a Program Support Assistant (Office Automation) in the Research and Development Service include:
1. Assisting investigators with proposal preparation and monitoring research proposals from submission to completion, ensuring conformance with

policies, regulations, and accuracy.
2. Establishing agendas and scheduling meetings for the Research and Development Committee, Human Studies Subcommittee, and Animal Studies Subcommittee.
3. Preparing and coordinating monthly packages for committee members, taking meeting notes, converting them into formal minutes, and distributing them.
4. Monitoring project reports (initial, progress, final), notifying investigators of required reports, maintaining files of abstracts and publications, and preparing them for distribution.
5. Managing project data activities using the PROMISE system.
6. Responding to information requests from potential grant applicants, providing application forms and instructions, and reviewing major requirements with them.
7. Maintaining a personal calendar for various grant deadlines and action dates.
8. Performing other assigned duties."
}