ECAI 2024
U. Endriss et al. (Eds.)
© 2024 The Authors.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License 4.0 (CC BY-NC 4.0).
doi:10.3233/FAIA240748

Enhancing Retrieval and Managing Retrieval: A Four-Module Synergy for Improved Quality and Efficiency in RAG Systems

Yunxiao Shi[a], Xing Zi[a], Zijing Shi[a], Haimin Zhang[a], Qiang Wu[a] and Min Xu[a,*]

[a]University of Technology Sydney, Broadway, Sydney, 2007, NSW, Australia.

Abstract. Retrieval-augmented generation (RAG) techniques leverage the in-context learning capabilities of large language models (LLMs) to produce more accurate and relevant responses. Originating from the simple 'retrieve-then-read' approach, the RAG framework has evolved into a highly flexible and modular paradigm. A critical component, the Query Rewriter module, enhances knowledge retrieval by generating a search-friendly query. This method aligns input questions more closely with the knowledge base. Our research identifies opportunities to enhance the Query Rewriter module to **Query Rewriter+** by generating multiple queries to overcome the *Information Plateaus* associated with a single query and by rewriting questions to eliminate *Ambiguity*, thereby clarifying the underlying intent. We also find that current RAG systems exhibit issues with *Irrelevant Knowledge*; to overcome this, we propose the **Knowledge Filter**. These two modules are both based on the instruction-tuned Gemma-2B model, which together enhance response quality. The final identified issue is *Redundant Retrieval*; we introduce the **Memory Knowledge Reservoir** and the **Retriever Trigger** to solve this. The former supports the dynamic expansion of the RAG system's knowledge base in a parameter-free manner, while the latter optimizes the cost for accessing external knowledge, thereby improving resource utilization and response efficiency. These four RAG modules synergistically improve the response quality and efficiency of the RAG system. The effectiveness of these modules has been validated through experiments and ablation studies across six common QA datasets. The source code can be accessed at https://github.com/Ancientshi/ERM4.

1 Introduction

Large Language Models (LLMs) represent a significant leap in artificial intelligence, with breakthroughs in generalization and adaptability across diverse tasks [4, 6]. However, challenges such as hallucinations [32], temporal misalignments [27], context processing issues [1], and fine-tuning inefficiencies [8] have raised significant concerns about their reliability. In response, recent research has focused on enhancing LLMs' capabilities by integrating them with external knowledge sources through Retrieval-Augmented Generation (RAG) [2, 20, 13, 15]. This approach significantly improves LLMs' ability to answer questions more accurately and contextually.

The basic RAG system comprises a knowledge retrieval module and a read module, forming the retrieve-then-read pipeline [20, 15, 13]. However, this vanilla pipeline has low retrieval quality and produces unreliable answers. To transcend this, more advanced RAG modules have been developed and integrated into the basic pipeline. For example, the Query Rewriter module acts as a bridge between the input question and the retrieval module. Instead of directly using the original question as the query text, it generates a new query that better facilitates the retrieval of relevant information. This enhancement forms the Rewrite-Retrieve-Read pipeline [23, 22]. Furthermore, models like RETA-LLM [22] and RARR [10] integrate a post-reading and fact-checking component to further solidify the reliability of responses. Additional auxiliary modules such as the query router [21] and the resource ranker [1] [14] have also been proposed to be integrated into the RAG's framework to improve the practicality in complex application scenario. This integration of various modules into the RAG pipeline leading to the emergence of a modular RAG paradigm [11], transforming the RAG framework into a highly flexible system.

Despite significant advancements, several unresolved deficiencies persist in practical applications. In the Query Rewriter module, the reliance on generating a single query for retrieval leads to (1) *Information Plateau*, as this unidirectional search method limits the scope of retrievable information. Besides, the frequent misalignment between the input question and the underlying inquiry intent often exacerbated by (2) *ambiguous phrasing*, significantly impedes the LLM's accurately interpreting users' demand. Furthermore, while the Query Rewriter can facilitate the retrieval of relevant information, it cannot guarantee the accuracy of the retrieved information. The extensive retrieval process may also acquire (3) *irrelevant knowledge*, which detracts from the response quality by introducing noise into the context. At last, we have identified a phenomenon of (4) *redundant retrieval*, where users pose questions similar to previous inquiries, causing the RAG system to fetch the same external information repeatedly. This redundancy severely compromises the efficiency of the RAG system.

These deficiencies underscore the potential for enhancing the accuracy and efficiency of existing RAG framework, thereby guiding our investigative efforts to address these issues. We decompose the Query Rewriter module's functionality into two subtasks, resulting in the upgraded module **Query Rewriter+**. The first subtask involves crafting nuanced, multi-faceted queries for improving the comprehensive search of relevant information. The second subtask is rewriting the input text into a more intention-clear question. Given that both subtasks can be conceptualized as text-to-text task, we have de-

[*] Corresponding author with email: Min.Xu@uts.edu.au.

[1] https://txt.cohere.com/rerank/

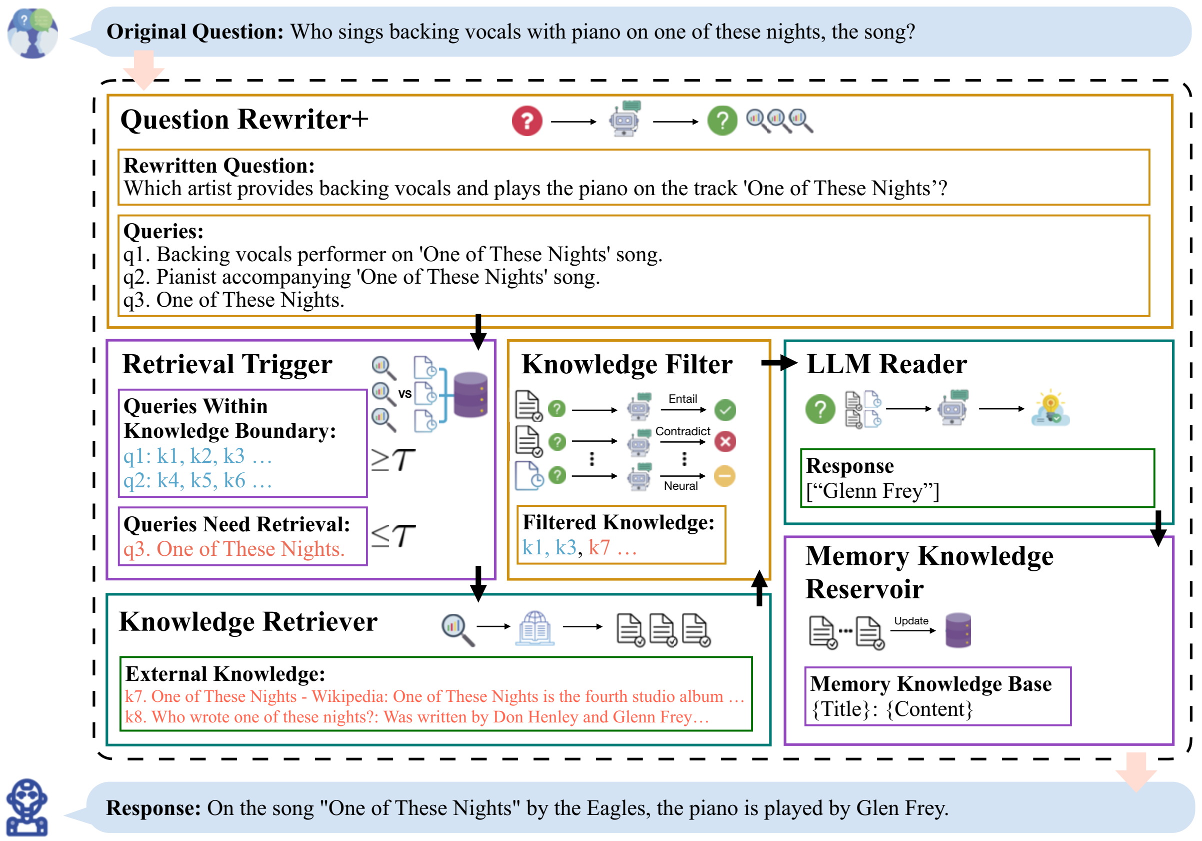

Original Question: Who sings backing vocals with piano on one of these nights, the song?
Question Rewriter+
Rewritten Question: Which artist provides backing vocals and plays the piano on the track 'One of These Nights'?
Queries: q1. Backing vocals performer on 'One of These Nights' song. q2. Pianist accompanying 'One of These Nights' song. q3. One of These Nights.
Retrieval Trigger
vs
Queries Within Knowledge Boundary: q1: k1, k2, k3 … q2: k4, k5, k6 …
≥T
Queries Need Retrieval: q3. One of These Nights.
≤T
Knowledge Filter
Entail
Contradict
Neural
Filtered Knowledge: k1, k3, k7 …
LLM Reader
Response ["Glenn Frey"]
Memory Knowledge Reservoir
Update
Memory Knowledge Base {Title}: {Content}
Knowledge Retriever
External Knowledge:
k7. One of These Nights - Wikipedia: One of These Nights is the fourth studio album …
k8. Who wrote one of these nights?: Was written by Don Henley and Glenn Frey…
Response: On the song "One of These Nights" by the Eagles, the piano is played by Glen Frey.

Figure 1. Flowchart depicting the integration of four modules into the basic Retrieve-then-Read (green) pipeline to enhance quality (orange) and efficiency (purple). Blue text represents cached knowledge retrieved from the Memory Knowledge Reservoir, while orange text indicates externally retrieved knowledge.

signed them to be executed concurrently for high efficiency. This is achieved through parameter-efficient fine-tuning of the Gemma-2B model, serving as the backbone of the Query Rewriter+. The model is trained on a meticulously constructed supervised dataset, enabling it to efficiently generate both appropriate queries and rewritten question from an original input text. To address the issue of retrieving irrelevant knowledge, we introduce the **Knowledge Filter** module, which performs the Natural Language Inference (NLI) task to sift through retrieved information and assess its relevance. This NLI model is also based on Gemma-2B and fine-tuned on a carefully designed dataset. The synergistic use of these two modules demonstrably enhances the accuracy of RAG responses across various datasets.

To address the issue of redundant retrieval, we introduce the **Memory Knowledge Reservoir**, a non-parametric module that enhances the RAG system's knowledge base through a caching mechanism. This module facilitates rapid information retrieval for recurring queries, thereby eliminating redundant external knowledge search. To avoid the situation that the retrieved cached information is insufficient, we design the **Retrieval Trigger**, which employs a simple and effective calibration-based approach, to determine whether to engage external knowledge retrieval.

These four modular advancements work synergistically to enhance the RAG framework, significantly improving the accuracy and efficiency of responses. The integration and functionality of our proposed modules within the RAG system are illustrated in Figure 1. In summary, our main contributions are as follows:

- We highlight the significance of clarifying input text and generating various queries, which informs the propose of Query Rewriter+. Furthermore, Knowledge Filter module is introduced to mitigate irrelevant knowledge issue. The synergistic operation of these two modules consistently enhances the response accuracy of RAG systems.
- We pinpoint the problem of redundant retrieval within the current RAG system. To address this efficiency concern, we introduce the Memory Knowledge Reservoir and the Retrieval Trigger module.
- Empirical evaluations and ablation studies across six QA datasets demonstrate the superior response accuracy and enhanced efficiency of our methods. Overall, our approach yields a 5%-10% increase in correctly hitting the target answers compared to direct inquiry. For historically similar questions, our method reduces the response time by 46% without compromising the response quality.

2 Motivation

In this section, we empirically investigate several preliminary studies to highlight the limitations of current RAG systems in Open-Domain QA tasks.

- **PS 1:** We investigate the maximum amount of relevant information that can be retrieved by converting input text into a single search-friendly query.
- **PS 2:** We examine whether using multiple queries that focus on different detailed semantic aspects can retrieve more relevant information than a single query.
- **PS 3:** We analyze how the proportion of irrelevant information changes as the volume of retrieved data increases.
- **PS 4:** We assess whether clarifying the input question is unnecessary for LLMs with strong semantic understanding capabilities.

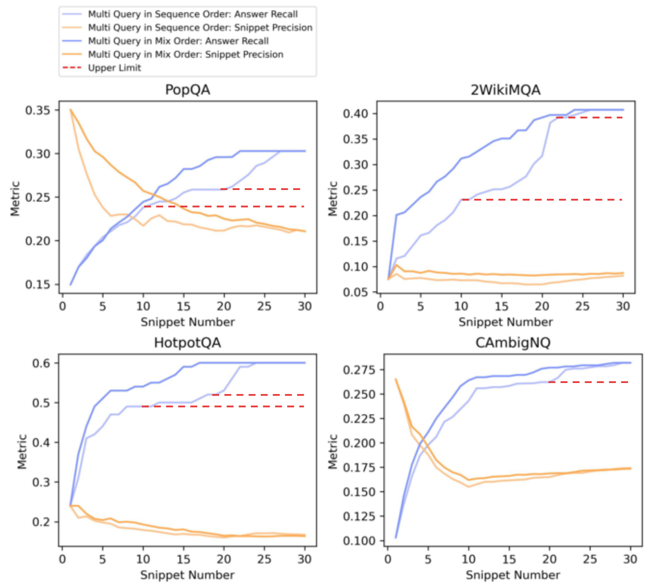


Figure 2. PS1 & PS2. Analysis of Information Plateaus in Retrieving Knowledge and Overcoming the Bottlenecks with New Queries.

Experimental Settings. The experiment is conducted using a randomly selected subset of 50 questions from each of the following datasets: PopQA [24], 2WikiMQA [30], HotpotQA [31], and CAmbigNQ [19]. We employ the Rewrite-Retrieve-Read RAG pipeline, and the rewriter follows the LLM-based method, specifically utilizes GPT-3.5-turbo-0613, more details are described in previous study [23]. The rewriter module is designed to generate one to three variable-length queries for each question, depending on the question's complexity. For retrieval, we use Bing Search V7 to identify the top 10 most relevant webpage snippets for each query. These snippets encapsulate the most query-related content from each webpage. For each question, a collection of retrieved snippets serves as the external knowledge to facilitate the LLM's in-context learning for generating response. We create two ways to present the snippets: *Sequential Order*, with snippets for each query following one another, and *Mix Order*, with top snippets evenly sampled from different queries. We increase the number of snippets to see how retrieval performance changes in two different snippet arrangements. The evaluation metrics are Answer Recall, which is the ratio of answer items found in the external knowledge to the total number of answer items, and Snippet Precision, which is the ratio of snippets containing any answer item to the total number of snippets used. These metrics provide a quantitative measure of retrieval performance. The experimental results are illustrated in Figure 2.

We conduct another experiment using 50 questions from the CAmbigNQ dataset. We obtain responses by directly inputting the original questions into the LLM, labeled as *org*. Another set of responses, labeled as *rewrt*, is generated by inputting rewritten questions into the LLM. The accuracy of these two sets of responses is quantified using the metrics: EM (Exact Match), Precision, Recall, and F1 Score. Additionally, we also use retrieval-enhanced method to generate answers again. These responses are labeled as *org_rag* for original questions and *rewrt_rag* for rewritten questions. A comparative analysis of the answer accuracy is conducted in the same manner. These results are illustrated in Figure 3.

PS1: The Information Plateaus of Single Query. From the analysis of Answer Recall in sequential order, it is observed that as the number of snippets increases, the Answer Recall metric improves, but they commonly plateauing before reaching 10 and 20 snippets (red dashed line). This phenomenon indicates there exist an upper limit to the retrievable useful information of a single query. This suggests there is a threshold beyond which additional information retrieved by the same query does not contribute too much to better retrieval quality.

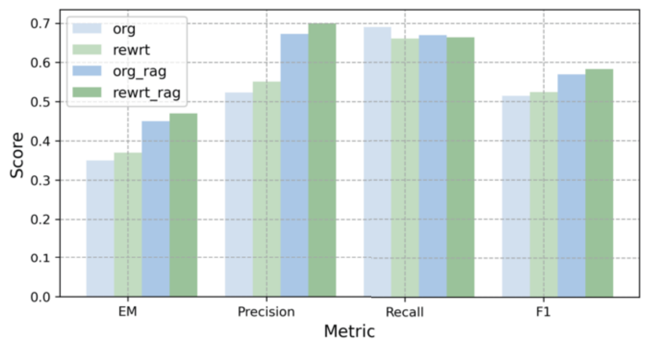


Figure 3. PS3. Evaluating the Impact of Rewriting Question for Q&A.

PS2: The Effect of Using Multiple Queries. The analysis of Answer Recall in Sequence Order indicates that introducing fresh snippets from new queries effectively mitigates plateauing in Answer Recall (light purple solid line). Additionally, the Answer Recall in Mixed Order consistently outperforms that in Sequence Order at the same snippet number (purple solid line), particularly when the number has not yet reached the maximum retrievable information limit (30 snippets). This underscores the significance of multiple queries in enhancing retrieval quality.

PS3: The Low Relevance of Retrieved Information. As shown in Figure 2, Snippet Precision notably decreases as the number of snippets increases, eventually stabilizing. This suggests a significant presence of retrieved external knowledge snippets that do not contain relevant answer information.

PS4: The Effect of Rewriting Questions. Figure 3 presents a bar graph comparing various metrics for original and rewritten questions in the CAmbigNQ dataset. Rewriting questions improves Exact Match (EM), Precision, and F1 Score—whether or not retrieval augmentation technique is used. However, the recall decreases with rewritten questions. This occurs because the CAmbigNQ dataset labels include all possible answers, and LLMs tend to provide all possible responses to vague questions. The Rewritten questions are more well-intended, prompting LLMs to generate specific answers.

Summary. Based on above results from a series of experiments on various datasets, our findings reveal that: (1) single query have an inherent upper limit of retrievable relevant information; (2) employing multiple queries that focus on different semantic aspects can surpass the information plateau, enhancing both the precision and recall of information retrieval; (3) The phenomenon of irrelevant knowledge is pervasive in RAG and becomes more pronounced with larger volumes of retrieved external information; and (4) rewriting ambiguous questions into intent-specific questions improves the precision of responses.

3 Methodology

3.1 Question Rewriter+

The design of the Question Rewriter+ module encompasses two primary functions: (1) enhancing the original question semantically into a rewritten question, and (2) generating multiple search-friendly queries. Formally, the Question Rewriter+ is denoted as $G_\theta(\cdot)$, which takes an original question p as input:

$$G_\theta(p) \rightarrow (s, Q)$$

where s represents the rewritten question and $Q = \{q_1, q_2, \cdots, q_{|Q|}\}$ is the set of generated queries. A basic implementation of $G_\theta(\cdot)$ can adopt a prompt-based strategy, utilizing task descriptions, the original question, and exemplars to prompt black-box large language models. This approach capitalizes on the model's in-context learning capabilities, often yields effectiveness of question rewriting and query generation. Nevertheless, the effectiveness of this methodology is highly dependent on the meticulous construction of prompts tailored to specific domain datasets, which limits its general utility. Besides, the generated s and Q may be of low quality, failing to enhance RAG performance.

To address these limitations, we propose a more general and task-specific approach. This involves parameter-efficient LoRA [12] fine-tuning of the Gemma-2B model for $G_\theta(\cdot)$, utilizing a high-quality dataset that is semi-automatically constructed through LLMs' generation and human's quality validation. This dataset comprises instances (p, s, Q), each rigorously validated to ensure that responses derived from s are more accurate in hitting the labeled answer compared to those obtained by directly asking the LLM with p. Additionally, we manually verify the quality of generated queries Q to ensure the reliability. The prompt template for generating (s, Q) is as follows:

[Instruction]: Your task is to transform a potentially colloquial or jargon-heavy [Original Question] into a semantically enhanced Rewritten Question with a clear intention. Additionally, generating several search-friendly Queries that can help find relevant information for answering the question. You can consider the provided [Examples] and response following the [Format].
[Original Question]: *{User's original question is here.}*
[Examples]: *{The examples should be specially tailored for different datasets.}*
[Format]: *{The generated Rewritten Question is here}**{query1}**{query2}**{query3}...*

3.2 Knowledge Filter

The accuracy of responses generated by LLMs can be significantly compromised by noisy retrieved contexts [34]. To mitigate this, we introduce the Knowledge Filter module, designed to enhance response accuracy and robustness. This module utilizes LLMs to filter out irrelevant knowledge. Rather than directly querying an LLM to identify noise, we incorporate a Natural Language Inference (NLI) framework [3] for this purpose. Specifically, for a rewritten question s and retrieved knowledge k, the NLI task evaluates whether the knowledge (as the premise) contains reliable answers, or useful information aiding the response to the question (as the hypothesis). This results in a judgment j categorized as entailment, contradiction, or neutral. The operation of the Knowledge Filter can be mathematically represented as:

$$F_\theta(s, k) \rightarrow j \in \{\text{entailment}, \text{contradiction}, \text{neutral}\}$$

Knowledge is retained if the NLI result is classified as entailment. We can adjust the strength of the hypothesis based on the specific dataset. For single-hop questions, a stronger hypothesis can be set, requiring the knowledge to contain direct and explicit answer information. Conversely, for more complex multi-hop questions, we can set a weaker hypothesis, only requiring the knowledge to include information that possibly aids in answering the question. When valid knowledge is unavailable, a back-off strategy is invoked, where LLMs generate responses without the aid of external knowledge augmentation. The Knowledge Filter also employs the LoRA fine-tuning method [12] on the Gemma-2B model, offering enhanced applicability and adaptability compared to prompt-based approaches.

The NLI training dataset is constructed semi-automatically using the similar method we described in Section 3.1. We provide task instruction, rewritten question s, along with knowledge context k as prompt to GPT-4, which then generated a brief explanation e and a classification result j, resulting in the data instance $((s, k, (e, j))$. The prompt template is as follows:

[Instruction]: Your task is to solve the NLI problem: given the premise in [Knowledge] and the hypothesis that "The [Knowledge] contains reliable answers aiding the response to [Question]". You should classify the response as entailment, contradiction, or neutral.
[Question]: *{Question is here.}*
[Knowledge]: *{The judging knowledge is here.}*
[Format]: *{The explanation.}**{The NLI result.}*

Considering that LLMs are primarily designed for text regression rather than classification, using only j as a label for instruction tuning for Gemma-2B would prevent the LLM from accurately performing classification tasks in a generative manner. Therefore, we also incorporate the concise explanation e as part of the label, in addition to the NLI classification result j.

3.3 Memory Knowledge Reservoir

We present a Memory Knowledge Reservoir module designed to cache the retrieved knowledge. The knowledge is structured as title-content pairs, where the title serves as a brief summary and the content offers detailed context. The Memory Knowledge Reservoir updates by adding new title-content pairs and replacing older entries with newer ones for the same titles. Mathematically, the Memory Knowledge Reservoir can be represented as a set $K = \{k_1, k_2, \ldots, k_{|K|}\}$, where each k_i is a title-content pair.

3.4 Retrieval Trigger

This module assesses when to engage external knowledge retrieval. A calibration-based method is utilized, wherein the popularity serves as a metric to estimate a RAG system's proficiency with the related knowledge.

$K = \{k_1, k_2, \ldots, k_{|K|}\}$ is the set of knowledge in the Memory Knowledge Reservoir, and $q_i \in Q$ is a generated query. The cosine similarity between query q_i and a knowledge instance $k_j \in K$ is denoted by $S(q_i, title(k_j))$. The popularity of query q_i, denoted by $\text{Pop}(q_i)$, is defined as:

$$\text{Pop}(q_i) = |\{k_j \in K \mid S(q_i, title(k_j)) \geq \tau\}|$$

where τ is a similarity threshold, $|\cdot|$ indicates the cardinality of the set. The boundary conditions for a query being within or outside the knowledge of the RAG system are established using a popularity threshold θ. A query q_i is considered to be within the knowledge boundary if:

$$\text{Pop}(q_i) \geq \theta$$

Conversely, a query q_i is outside the knowledge boundary if:

$$\text{Pop}(q_i) < \theta$$

4 Experiments

4.1 Modular Setting

Fine-tuning Gemma-2B We follow the Alpaca's training method[2], employing the LoRa[12] method to instruction-tune the pre-trained Gemma-2B model[3] for the Question Rewriter+ and Knowledge Filter modules. We set the learning rate to 1e-4, batch size to 8, and epochs to 6. We set the rank of the LoRa low-rank matrix to 8, and the scaling factor, alpha to 16. Additionally, we utilize the 4-bit quantization method with NF4 quantization type [7]. The training and inference process are all conducted on a single Nvidia Quadro RTX 6000.

Knowledge Retriever We utilize the Bing Search Engine v7 as the information retrieval method. For each query q, we select the top-n items from the search results, and each item is regarded as a knowledge instance. We utilize the snippet of a search item as the content of a knowledge instance. The hyperparameter n is predetermined at 10.

LLM Reader We primarily used GPT-3.5-turbo-0613 as the black-box LLM model for generating answers. The prompt structure includes task instruction, question, external knowledge, examples, and response format. The external knowledge section comprises up to 30 knowledge instances arranged in mixed order, as discussed in Section 2.

4.2 Task Setting

We evaluate the efficacy of our proposed methodologies under open-domain QA task. This evaluation leverages three distinct open-domain QA datasets that do not require logical reasoning. These include: (i) The Natural Questions (NQ) dataset [18], which is a real-world queries compiled from search engines. (ii) PopQA [24], a dataset with a long-tail distribution, emphasizing less popular topics within Wikidata. (iii) AmbigNQ [25], an enhanced version of NQ that transforms ambiguous questions into a set of discrete, yet closely related queries. Additionally, we incorporate two benchmark datasets that require logical reasoning: (iv) 2WIKIMQA [30] and (v) HotPotQA [31]. Due to the costs associated with API calls for LLMs and Bing Search, and following common practices[16, 34, 23, 33], we test on a stratified sample of 300 questions from each dataset rather than the entire test dataset.

We assess performance using the F1 Score and Hit Rate metrics. Due to the discrepancy between the verbose outputs of LLMs and the concise format of the dataset answers, we chose not to utilize the Exact Match (EM) metric. Instead, we considered a response as correct if the text hit any item of the labeled answers.

4.3 Baselines

Rewrite-Retrieve-Read [23] represents the current state-of-the-art improvement on the basic Retrieve-then-Read RAG pipeline. Our approach enhances the existing RAG pipeline by augmenting the Query Rewriter to Query Rewriter+ and introducing a new Knowledge Filter module. To highlight the effectiveness of our method, we implemented the following configurations:

(i) Direct: Ask the LLM directly with the original question.

(ii) Rewriter-Retriever-Reader: Prior to retrieval, a Query Rewriter module is employed to generate a query that fetches external knowledge. The external knowledge, along with the original question, is used to prompt the response generation.

[2] https://github.com/tloen/alpaca-lora
[3] https://huggingface.co/google/gemma-2b

Table 1. Open-Domain Question Answering Performance.

Dataset	Method	F1	Hit Rate
CAmbigNQ	Direct	37.38	55.67
	Rewriter-Retriever-Reader	39.65	57.67
	Rewriter+-Retriever-Reader	41.58	59.00
	Rewriter+-Retriever-Filter-Reader	**43.39**	**64.33**
NQ	Direct	41.50	42.00
	Rewriter-Retriever-Reader	48.70	46.00
	Rewriter+-Retriever-Reader	51.43	50.33
	Rewriter+-Retriever-Filter-Reader	**52.68**	**51.33**
PopQA	Direct	35.24	42.33
	Rewriter-Retriever-Reader	37.84	44.00
	Rewriter+-Retriever-Reader	38.51	46.67
	Rewriter+-Retriever-Filter-Reader	**41.77**	**51.33**
AmbigNQ	Direct	45.21	46.67
	Rewriter-Retriever-Reader	49.85	50.00
	Rewriter+-Retriever-Reader	51.84	52.33
	Rewriter+-Retriever-Filter-Reader	**53.47**	**55.67**
HotPot	Direct	46.33	41.33
	Rewriter-Retriever-Reader	48.24	44.00
	Rewriter+-Retriever-Reader	53.67	45.33
	Rewriter+-Retriever-Filter-Reader	**57.59**	**50.00**
2WikiMQA	Direct	41.85	42.33
	Rewriter-Retriever-Reader	43.24	45.00
	Rewriter+-Retriever-Reader	45.71	47.67
	Rewriter+-Retriever-Filter-Reader	**46.83**	**49.33**

(iii) Rewriter+-Retriever-Reader: Prior to retrieval, the Enhanced Query Rewriter module is utilized, generating multiple queries to acquire external knowledge and clarify the original question. Responses are generated using both the rewritten question and all retrieved external knowledge.

(iv) Rewriter+-Retriever-Filter-Reader: Applied before retrieval, the Enhanced Query Rewriter module generates multiple queries and clarifies the original question. A Knowledge Filter is used to discard external knowledge unrelated to the rewritten question. The final response is then generated using the filtered external knowledge and the rewritten question.

Comparing the Rewriter+-Retriever-Reader setup with the Rewriter-Retriever-Reader setup validates the superiority of the proposed Question Rewriter+ module. Additionally, comparing the Rewriter+-Retriever-Filter-Reader setup with the Rewriter+-Retriever-Reader setup demonstrates the effectiveness of the 4-step RAG pipeline incorporating the Knowledge Filter.

4.4 Results

Experimental results are reported in Table 1. The scores indicate that the Query Rewriter+ module outperforms the Query Rewriter module across all datasets, substantiating that multiple queries and clarified question are more effective for a RAG system correctly response to user's questions than single query and unrefined questions. Specifically, adding the Knowledge Filter module to the traditional 3-step RAG pipeline significantly improves performance. This indicates that merely adding external knowledge to the RAG system can be detrimental, especially for multi-hop questions. The Knowledge Filter module effectively eliminates noise and irrelevant content, enhancing the accuracy and robustness of the RAG system's responses.

5 Ablation Studies

In this section, we analyze the individual and combined effects of question rewriting and knowledge filtering. The results presented in Table 3 indicate that the question rewriting process consistently improves answer accuracy across the setups of direct generation,

Table 2. The Impact of the Similarity Threshold τ of the Retriever Trigger module in Response Efficiency and Quality.

τ	Time Cost (s)	External Knowledge	Memory Knowledge	Irrelevant Knowledge	Hit Rate (%)
0.2	5.68	0.00	30.30	11.62	48.5
0.4	5.89	0.00	19.58	3.52	50.5
0.6	3.97	4.39	11.57	1.53	53.0
0.8	7.17	14.33	1.72	0.86	55.0
1.0	7.45	15.00	0.00	0.79	53.5

Table 3. Investigation of the Individual and Combined Effects of Question Rewriting and Knowledge Filtering on CAMbigNQ and PopQA

Dataset	Setting		F1	Hit Rate
	Question	Knowledge		
CAmbigNQ	Original	\	37.38	55.67
	Rewritten	\	38.45	57.67
	Original	All	38.24	54.00
	Rewritten	All	41.58	58.00
	Original	Filtered	39.62	59.67
	Rewritten	Filtered	43.39	64.33
PopQA	Original	\	35.24	42.33
	Rewritten	\	36.73	42.67
	Original	All	38.14	44.33
	Rewritten	All	38.51	46.67
	Original	Filtered	39.79	47.33
	Rewritten	Filtered	41.77	51.33

retrieval-augmented generation using all knowledge, and retrieval-augmented generation using filtered knowledge.

The results also shows that using all (unfiltered) external knowledge for retrieval-augmented generation can sometimes lead to marginal improvements or even decreased performance. For instance, on the CAmbigNQ dataset, when LLMs are asked with rewritten questions, introducing all external knowledge only raise the hit rate from 57.67% to 58%. Besides, when LLM are queried with the original questions, introducing all external knowledge makes the hit rate decreased from 55.67% to 54.00%. On the other hand, we observe that filtering knowledge can significantly boost the response accuracy of the RAG system, whether asking with the original question or rewritten question.

Assessing the synergistic effect of two modules, we find that while each module individually improves response accuracy, the effect is sometimes modest. However, their combined yields a significant enhancement. For instance, on the CAmbigNQ dataset, the individual application of each module resulted in a maximum of 2% more correctly answered questions, whereas their combined application led to a 7% increase in correctly answered questions. A similar phenomenon can also been observed on the PopQA dataset.

6 Efficiency Improvement Investigation

In this section, we explore how efficiently our proposed method reduces redundant retrieval when answering recurring questions with historically similar semantics. We also examine the hyperparameter τ to balance efficiency and response accuracy. The experimental procedure is as follows:

Initially, we randomly selected 100 questions from the AmbigNQ dataset to generate responses using our proposed method. Unlike previous sections, we set the parameter n in the Knowledge Retriever module to 5. Instead of utilizing webpage snippets as the content of knowledge instances, we visited the searched URLs and read the entire webpage text, filtering out irrelevant information using the BM25 algorithm. After the response finished, the webpage content was then cached in the Memory Knowledge Reservoir. Subsequently, we selected an additional 200 questions from AmbigNQ that are semantically similar to the previously solved questions. These questions were answered with the support of the Memory Knowledge Reservoir and the Retrieval Trigger module, with the popularity threshold θ set as 3.

We design several metrics to evaluate the resource cost of per-question, include the average time spent in the RAG pipeline (Time Cost), the average number of external knowledge instances (External Knowledge), the average number of memory knowledge instances (Memory Knowledge), the average number of knowledge instances filtered out (Irrelevant Knowledge), and the performance metric Hit Rate. These metrics are recorded during the question-answering process.

The analysis of the trade-off between response quality and efficiency for answering historically similar questions across different τ settings is presented in Table 2. A significant finding is that the Time Cost metric reaches minimum when setting the similarity threshold $\tau = 0.6$. This is accompanied by the External Knowledge metric being very small, approximately 4.39, which is roughly equivalent to one query search. This suggests that this configuration predominantly leverages memory knowledge rather than external sources for generating responses, thereby enhancing response efficiency. Remarkably, at the $\tau = 0.6$ setup, the quality of the responses is not heavily affected and remains very close to that achieved by relying entirely on external knowledge at $\tau = 1.0$. This suggests that deploying the Memory Knowledge module can achieve a significant reduction in response time—by approximately 46%—without substantially compromising the quality of the answers. Furthermore, adjusting the threshold to 0.8 enhances the response quality beyond that at $\tau = 1.0$, underscoring that leveraging highly relevant historical experience can generate responses with superior quality.

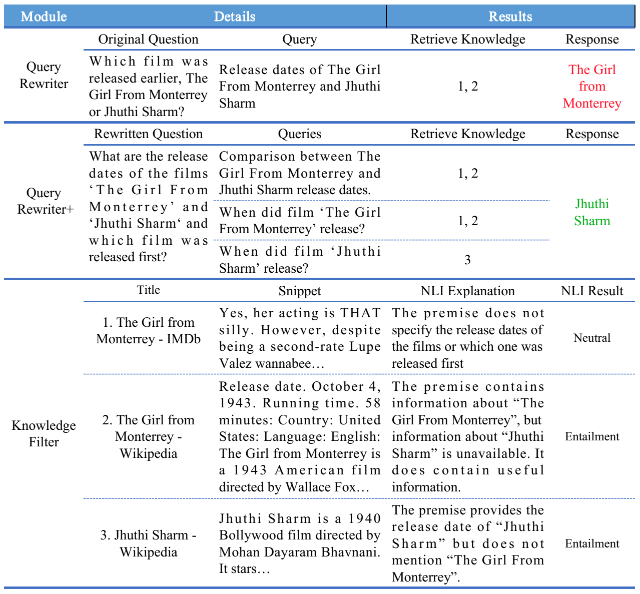
7 Case Study

To intuitively demonstrate how the Query Rewriter+ Module enhances the original question and generates multiple queries, as compared to traditional Query Rewriter Modules, we present a question examples from 2WikiMQA dataset in Figure 4. It can be observed that the Query Rewriter+ Module semantically enhances the original question and, unlike the Query Rewriter which generates only one query, it produces three distinct queries, each focusing on different semantic aspects. The Query Rewriter+ module can retrieve external knowledge sources 1, 2, and 3, whereas the Query Rewriter module only retrieves sources 1 and 2, showcasing its advantage in improving knowledge recall. The Knowledge Filter module subsequently ensures the precision of the external knowledge by filtering out irrelevant knowledge instances (neutral, contradict) and retaining only those that provide valuable information for answering the question (entailment).

8 Related Work

8.1 Retrieval Augmented Generation

Retrieval Augmented Generation (RAG) [20] leverages a retriever that provides substantial external information to enhance the output



Module	Details		Results	
	Original Question	Query	Retrieve Knowledge	Response
Query Rewriter	Which film was released earlier, The Girl From Monterrey or Jhuthi Sharm?	Release dates of The Girl From Monterrey and Jhuthi Sharm	1, 2	The Girl from Monterrey
	Rewritten Question	Queries	Retrieve Knowledge	Response
Query Rewriter+	What are the release dates of the films 'The Girl From Monterrey' and 'Jhuthi Sharm' and which film was released first?	Comparison between The Girl From Monterrey and Jhuthi Sharm release dates.	1, 2	Jhuthi Sharm
		When did film 'The Girl From Monterrey' release?	1, 2	
		When did film 'Jhuthi Sharm' release?	3	
	Title	Snippet	NLI Explanation	NLI Result
Knowledge Filter	1. The Girl from Monterrey - IMDb	Yes, her acting is THAT silly. However, despite being a second-rate Lupe Valez wannabee…	The premise does not specify the release dates of the films or which one was released first	Neutral
	2. The Girl from Monterrey - Wikipedia	Release date. October 4, 1943. Running time. 58 minutes: Country: United States: Language: English: The Girl from Monterrey is a 1943 American film directed by Wallace Fox…	The premise contains information about "The Girl From Monterrey", but information about "Jhuthi Sharm" is unavailable. It does contain useful information.	Entailment
	3. Jhuthi Sharm - Wikipedia	Jhuthi Sharm is a 1940 Bollywood film directed by Mohan Dayaram Bhavnani. It stars…	The premise provides the release date of "Jhuthi Sharm" but does not mention "The Girl From Monterrey".	Entailment

Figure 4. Examples for answering question "Which film was released earlier, The Girl From Monterrey or Jhuthi Sharm?" in the 2WIKIMQA dataset.

of Large Language Models (LLMs). This strategy utilizes knowledge in a parameter-free manner and circumvents the high training costs associated with LLMs' parameterized knowledge. Furthermore, it alleviates hallucination issues, significantly enhancing the factual accuracy and relevance of the generated content. The concept of RAG is rooted in the DrQA framework [5], which marked the initial phase of integrating retrieval mechanisms with Language Models through heuristic retrievers like TF-IDF for sourcing evidence. Subsequently, RAG evolved with the introduction of Dense Passage Retrieval [15] and REALM [26]. These methods utilize pre-trained transformers and are characterized by the joint optimization of retrieval and generation process. Recent advancements have extended RAG's capabilities by integrating Large Language Models (LLMs), with developments such as REPLUG [29] and IC-RALM [26] demonstrating the potent generalization abilities of LLMs in zero-shot or few-shot scenarios. These models can follow complex instructions, understand retrieved information, and utilize limited demonstrations for generating high-quality responses.

8.2 *Modular RAG*

The core of RAG framework consists of retriever and reader modules. This retrieve-read pipeline has been enhanced, leading to the Modular RAG paradigm with various integrated modules. This section describes related modules in our work.

Rewriter: The introduction of a Question Rewriter module [23] led to the development of a Rewrite-Retrieve-Read RAG pipeline. This module generates a query that bridges the gap between the input text and the knowledge base, facilitating the retrieval of relevant knowledge and enhancing response accuracy. Our empirical studies indicate that while a single query retrieves limited the recall of useful information, and multiple queries significantly enhance the retrieval.

Clarification: Represented by [17], this module generates clarification questions to ascertain user intent, thus refining vague questions to uncover the underlying inquiry intent. We have integrated the functionalities of the Rewriter and Clarification modules into a single unit, Query Rewriter+, employing a fine-tuned Gemma-2B model to perform both tasks generatively for efficiency.

Post-Retrieval Process: After information retrieval, presenting all data to a LLMs may exceed the context window limit, and the noise context may also affect the LLM's performance in answering questions. The Re-Ranking module relocates information based on relevance. In our study, we consider this post-retrieval process primarily as a classification de-noising task, rather than focusing on ranking.

Memory: Modules in this category leverage historically similar question-answer records to enhance current problem-solving capabilities, reflecting an evolutionary learning process within the agent [36]. Drawing on this concept, we employ a parameter-free caching mechanism to expand the knowledge boundaries of RAG system for question-answering, effectively improving response efficiency.

Retrieve Trigger: Understanding the parameterized knowledge boundaries of LLMs is crucial for optimizing the cost for external knowledge retrieval. Calibration-based judgment methods have proven both efficient and practical. Such methods heavily rely on threshold tuning [35], therefore we explore appropriate thresholds that balance accuracy and efficiency.

Additional modules include Knowledge Retriever, LLM Reader, Fact Checking, Revising [22, 10], and iterative RAG pipeline [28] with further details available in [11, 9].

9 Conclusion

In this paper, we present a four-module strategy to enhance RAG systems. The Query Rewriter+ module generates clearer questions for better intent understanding by LLMs and produces multiple, semantically distinct queries to find more relevant information. The Knowledge Filter refines retrieved information by eliminating irrelevant and noisy context, enhancing the precision and robustness of LLM-generated responses. The Memory Knowledge Reservoir and the Retrieval Trigger module optimize the use of historical data and dynamically manage external information retrieval needs, increasing system efficiency. Collectively, these advancements improve the accuracy and efficiency of the RAG system.

Acknowledgements

This work was sponsored by the `Australian Research Council under the Linkage Projects Grant LP210100129`, and by the program of `China Scholarships Council (No. 202308200014).`

References

[1] C. An, S. Gong, M. Zhong, X. Zhao, M. Li, J. Zhang, L. Kong, and X. Qiu. L-eval: Instituting standardized evaluation for long context language models, 2023.

[2] S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. B. Van Den Driessche, J.-B. Lespiau, B. Damoc, A. Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR, 2022.

[3] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning. A large annotated corpus for learning natural language inference. In L. Màrquez, C. Callison-Burch, and J. Su, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal, Sept. 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1075. URL https://aclanthology.org/D15-1075.

[4] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners, 2020.

[5] D. Chen, A. Fisch, J. Weston, and A. Bordes. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*, 2017.

[6] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel. Palm: Scaling language modeling with pathways, 2022.

[7] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.

[8] M. Du, F. He, N. Zou, D. Tao, and X. Hu. Shortcut learning of large language models in natural language understanding, 2023.

[9] Z. Feng, W. Ma, W. Yu, L. Huang, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, and T. liu. Trends in integration of knowledge and large language models: A survey and taxonomy of methods, benchmarks, and applications, 2023.

[10] L. Gao, Z. Dai, P. Pasupat, A. Chen, A. T. Chaganty, Y. Fan, V. Zhao, N. Lao, H. Lee, D.-C. Juan, and K. Guu. RARR: Researching and revising what language models say, using language models. In A. Rogers, J. Boyd-Graber, and N. Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16477–16508, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.910. URL https://aclanthology.org/2023.acl-long.910.

[11] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, Q. Guo, M. Wang, and H. Wang. Retrieval-augmented generation for large language models: A survey, 2024.

[12] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models, 2021.

[13] G. Izacard, P. Lewis, M. Lomeli, L. Hosseini, F. Petroni, T. Schick, J. Dwivedi-Yu, A. Joulin, S. Riedel, and E. Grave. Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*, 2022.

[14] H. Jiang, Q. Wu, C.-Y. Lin, Y. Yang, and L. Qiu. Llmlingua: Compressing prompts for accelerated inference of large language models, 2023.

[15] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W. tau Yih. Dense passage retrieval for open-domain question answering, 2020.

[16] O. Khattab, K. Santhanam, X. L. Li, D. Hall, P. Liang, C. Potts, and M. Zaharia. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive nlp. *arXiv preprint arXiv:2212.14024*, 2022.

[17] G. Kim, S. Kim, B. Jeon, J. Park, and J. Kang. Tree of clarifications: Answering ambiguous questions with retrieval-augmented large language models. In H. Bouamor, J. Pino, and K. Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 996–1009, Singapore, Dec. 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.63. URL https://aclanthology.org/2023.emnlp-main.63.

[18] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, K. Toutanova, L. Jones, M. Kelcey, M.-W. Chang, A. M. Dai, J. Uszkoreit, Q. Le, and S. Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL https://aclanthology.org/Q19-1026.

[19] D. Lee, S. Kim, M. Lee, H. Lee, J. Park, S.-W. Lee, and K. Jung. Asking clarification questions to handle ambiguity in open-domain QA. In H. Bouamor, J. Pino, and K. Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11526–11544, Singapore, Dec. 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.772. URL https://aclanthology.org/2023.findings-emnlp.772.

[20] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[21] X. Li, E. Nie, and S. Liang. From classification to generation: Insights into crosslingual retrieval augmented icl, 2023.

[22] J. Liu, J. Jin, Z. Wang, J. Cheng, Z. Dou, and J.-R. Wen. Reta-llm: A retrieval-augmented large language model toolkit. *arXiv preprint arXiv:2306.05212*, 2023.

[23] X. Ma, Y. Gong, P. He, H. Zhao, and N. Duan. Query rewriting for retrieval-augmented large language models, 2023.

[24] A. Mallen, A. Asai, V. Zhong, R. Das, D. Khashabi, and H. Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822, 2023.

[25] S. Min, J. Michael, H. Hajishirzi, and L. Zettlemoyer. AmbigQA: Answering ambiguous open-domain questions. In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5783–5797, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.466. URL https://aclanthology.org/2020.emnlp-main.466.

[26] O. Ram, Y. Levine, I. Dalmedigos, D. Muhlgay, A. Shashua, K. Leyton-Brown, and Y. Shoham. In-context retrieval-augmented language models, 2023.

[27] P. Röttger and J. Pierrehumbert. Temporal adaptation of BERT and performance on downstream document classification: Insights from social media. In M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2400–2412, Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.206. URL https://aclanthology.org/2021.findings-emnlp.206.

[28] Z. Shao, Y. Gong, Y. Shen, M. Huang, N. Duan, and W. Chen. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In H. Bouamor, J. Pino, and K. Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9248–9274, Singapore, Dec. 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.620. URL https://aclanthology.org/2023.findings-emnlp.620.

[29] W. Shi, S. Min, M. Yasunaga, M. Seo, R. James, M. Lewis, L. Zettlemoyer, and W. tau Yih. Replug: Retrieval-augmented black-box language models, 2023.

[30] J. Welbl, P. Stenetorp, and S. Riedel. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302, 2018.

[31] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. Cohen, R. Salakhutdinov, and C. D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1259. URL https://aclanthology.org/D18-1259.

[32] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao. React: Synergizing reasoning and acting in language models, 2023.

[33] O. Yoran, T. Wolfson, B. Bogin, U. Katz, D. Deutch, and J. Berant. Answering questions by meta-reasoning over multiple chains of thought. In H. Bouamor, J. Pino, and K. Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5942–5966, Singapore, Dec. 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.364. URL https://aclanthology.org/2023.emnlp-main.364.

[34] O. Yoran, T. Wolfson, O. Ram, and J. Berant. Making retrieval-augmented language models robust to irrelevant context, 2023.

[35] Z. Zhang, M. Fang, and L. Chen. Retrievalqa: Assessing adaptive retrieval-augmented generation for short-form open-domain question answering, 2024. URL https://arxiv.org/abs/2402.16457.

[36] A. Zhao, D. Huang, Q. Xu, M. Lin, Y.-J. Liu, and G. Huang. Expel: Llm agents are experiential learners. In *Proceedings of THE 37th Association for the Advancement of Artificial Intelligence Conference.*, 2023.