Leveraging LLM and User Feedback to Improve Retrieval-Augmented Generation When Question and Answer Domains Shift

Anonymous ACL submission

Abstract

Retrieval-Augmented Generation (RAG) attempts to mitigate the issue of outdated knowledge and hallucinations in large language models (LLMs) by retrieving real-time and accurate information for LLMs. Nevertheless, we observe that the domain of user questions undergoes rapid changes over time, resulting in a significant decrease in RAG performance. Additionally, users have varying expectations for replies. However, current RAG approaches generally overlook the variations in preferred answer domains across different users. To this end, we proposes a method that utilizes both LLM and User Feedback (LUF) to improve RAG performance with shifts in question domains and answer domains. By leveraging feedback obtained from the classic RAG process, LUF can enhance its ability to adjust to variations in questions and user preferences through updates to the retriever and document database. Experiments on five datasets demonstrate that LUF significantly improves the accuracy of the retriever and the responses of the LLM. Compared to baselines, LUF provides more accurate responses aligned with different user preferences.

1 Introduction

Retrieval-Augmented Generation (RAG) (Lewis et al., 2021; Gao et al., 2024) tackles hallucinations resulting from the inability to access real-time information by employing an additional retriever to obtain the latest data. Existing RAG frameworks typically involve the following steps: first, an additional retriever is used to fetch relevant documents from a document database; then, an LLM is utilized to rerank and filter irrelevant documents; finally, the question is combined with the filtered documents and fed into an LLM to generate a response (Sachan et al., 2023).

Previous research (Yoran et al., 2024) has shown that LLMs exhibit a notable decrease in answer accuracy when irrelevant documents are retrieved. Thus, the output of the LLMs is highly dependent on the precision of the retriever and the quality of the document database. Several studies have attempted to improve the performance of RAG retrieval, including constructing better training data for retriever training (Yu et al., 2023), using data augmentation techniques such as query rewriting or expansion (Wang et al., 2023; Ma et al., 2023), and employing multiple retrievals or active retrieval methods (Borgeaud et al., 2021; Jiang et al., 2023). However, these studies failed to recognize RAG's real-time potential as their retrievers and document databases were frozen during testing.

Trained on	R@5 of Retriever/TF-IDF Similarity Tested on		
	NQ	TQA	SQuAD
NQ	**56.43/100**	51.91/42.76	55.68/64.64
TQA	60.55/42.76	**66.96/100**	61.67/51.83
SQuAD	58.39/64.64	50.05/51.83	**64.23/100**

Table 1: Accuracy of retriever trained and tested on different datasets and TF-IDF similarity of three datasets.

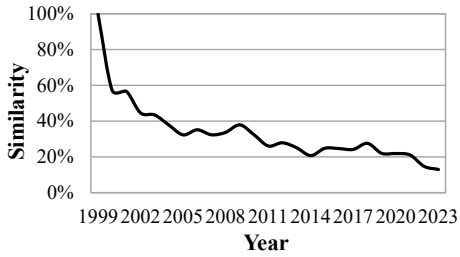


Figure 1: Similarity of most searched questions on Google across years.

However, our research shows that domain shifts in questions significantly reduce retriever performance, even with the same document database. By splitting the questions and answers from different datasets into word sets and calculating the TF-IDF similarity between them, Table 1 shows that the retriever's accuracy drops more when answering

questions less similar to those in the training set. In reality, as Fig. 1 shows, we analyzed the top 100 most searched keywords on Google each year and found that people's search interests change rapidly over time, indicating that RAG solutions may perform poorly. Moreover, current research utilizes the same database to answer all questions, providing indistinguishable responses to different users, who often have varying expectations for the domain of the answers. For instance, when asking about courses that would enhance one's skills, a computer science student would benefit from an answer of studying "Computer Systems", yet a biology student would find "General Biology" more appropriate.

Our insight is that the feedback from both the LLM and the user, as part of the existing RAG process, can assist with resolving the aforementioned decline in performance caused by domain shifts in questions and answers. On one hand, the results of LLM reranking can serve as feedback for improving the retriever, as LLMs, even without knowing the answer to the question, can identify which documents are beneficial for generation by semantic analysis. The documents retrieved by the retriever but filtered out by the LLM can be considered as hard negative samples. Utilizing these samples in training can rapidly enhance the retriever's accuracy in new question domains (Robinson et al., 2021). On the other hand, during interactions between the LLM and the user, the user naturally provides meaningful feedback, which can assist the LLM in understanding the user's preferences. By incorporating this information into the LLM's "memory", LLM can offer responses in future conversations that better align with the user's expectations, adapting to the shift in the answer domain.

With this in mind, we propose the LLM and User Feedback-based RAG (LUF), an RAG framework that autonomously adapts to domain shifts in questions and answers based on feedback. LUF initially follows the standard procedure of retrieving, reranking, and generating output. It then self-updates based on LLM and user feedback during the user's engagement. First, LUF uses LLM reranking results as pseudo-ground truth to update the retriever. This enhances the retriever's accuracy on unseen questions and helps it find more relevant documents for the LLM. Second, LUF prompts the LLM to identify if the user-LLM interaction yields unknown information. Such information is summarized and added to the document database to serve as evidence for future conversations. LUF does not need explicit annotations from the user. It exclusively relies on implicit feedback in the existing process to swiftly adapt to rapidly evolving questions and improve the accuracy of LLM outputs.

LUF can utilize any deep learning retriever, is compatible with all existing LLMs, and can be combined with other strategies to enhance RAG performance, such as active retrieval (Jiang et al., 2023) and query rewriting (Ma et al., 2023). Considering the differences between various LLMs and datasets, we used two LLMs to evaluate the influence of feedback on retriever and LLM across five datasets in two languages. Furthermore, we constructed three re-ask datasets to examine how LUF affects the responses generated by the LLM when similar questions are asked multiple times. LUF consistently achieved superior or competitive results compared to other LLM-enhanced baselines, indicating the effectiveness of our approach across all tasks. In summary, the contributions of this paper are:

- We propose LUF, an RAG framework that self-updates based on LLM and user feedback. It adapts the retriever and LLM output to handle question domain shifts resulting from cross-domain scenarios or public concern evolution, as well as answer domain shifts specified by user preferences.
- We evaluate the performance of RAG on three re-ask datasets. To our knowledge, we are the first to study LLM improvements from document database updates.
- We conducted evaluation experiments based on two language models across five public datasets to validate the effectiveness of our approach. Additionally, our experiments investigated how LLM and user feedback help RAG acquire real-time information and align with user preferences.

2 Method

In this section, we provide a formal definition of the RAG and introduce the framework and technical details of our method.

2.1 Problem Definition

For the question-answering task, given a series of questions $\mathbb{Q} = \{q_i\}_{i=1}^{n}$, the generative lan-

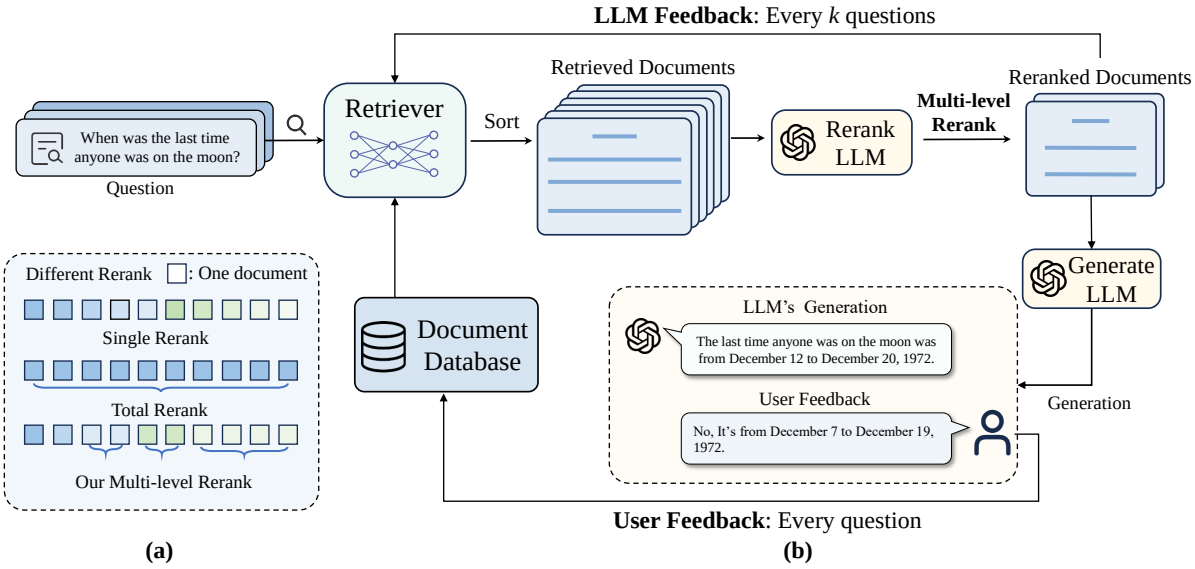


Figure 2: Framework of LUF, including the widely used RAG workflow and processors for LLM and user feedback.

guage model is expected to generate a series of responses $\mathbb{A} = \{a_i\}_{i=1}^{n}$, where each response contains the answer of the corresponding question. The RAG framework additionally includes a document database $\mathbb{D} = \{d_i\}_{i=1}^{m}$ and a retriever R. For a given user question q, RAG uses the retriever to fetch a set of potentially relevant documents $\mathbb{D}' = \{d'_i\}_{i=1}^{k}$ from the document repository. The simplest approach is to concatenate $\mathbb{D}'$ and q, and then send them to the LLM for generation, obtaining $\mathsf{Generate}([\mathbb{D}', q])$. However, due to the limited parameter size of the retriever, a significant portion of the retrieved documents may be irrelevant to the original question, and can interfere with the LLM's output. Therefore, a natural solution is to have the LLM perform a secondary identification of the retrieved documents, obtaining a filtered set of documents $\mathsf{Rerank}(\mathbb{D}')$, which is then concatenated with the query. The final output of the retrieval augmented LLM is:

$$a = \mathsf{Generate}([\mathsf{Rerank}(\mathbb{D}'), q])$$

2.2 Overview of the Proposed Method

The overall architecture of LUF is illustrated in Fig. 2. Given a user's question q, LUF first retrieves relevant documents from the database. Then, LLM performs multi-level reranking on the retrieved results to filter out irrelevant documents. Finally, the documents and the question are passed to the LLM for generation. To ensure the accuracy of the retriever on new questions, we treat the reranked results as pseudo ground truth to fine-tune the retriever. Additionally, we incorporate user feedback into the document database to better answer similar questions in the future.

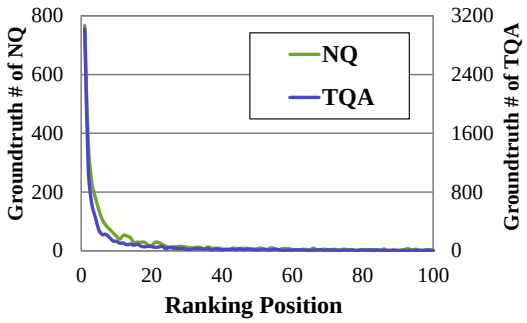


Figure 3: Retrieval results on Natural Questions and Trivia QA, as the rank position increases, the number of relevant documents decreases rapidly.

2.3 Updates Based on LLM Feedback

The accuracy of the retriever tends to decline when faced with unseen questions, whereas the LLM can still assess the relevance between the question and the document based on semantics, even without knowing the exact answer. LLM reranking exploits this characteristic by categorizing the retrieved documents into relevant ones D^+ and irrelevant ones D^-. The relevant documents filtered by the LLM not only provide the answer but are also very likely to contain relevant information that might help the LLM generate more diverse responses.

LUF advances by utilizing the reranked results

provided by LLM as pseudo ground truth to update the retriever during testing. To quickly improve the retriever's accuracy on new questions, we update the model every k questions. Specifically, for every k questions $\{q\}$ posed by users, we conduct offline training of the retriever based on the feedback from the LLM by the following loss function:

$$L(q_i, D_i^+, D_i^-) = -\log \frac{\sum_{d \in D_i^+} e^{\text{sim}(q_i,d)}}{\sum_{d \in D_i^+} e^{\text{sim}(q_i,d)} + \sum_{d \in D_i^-} e^{\text{sim}(q_i,d)}}, \quad (1)$$

where $\text{sim}(\cdot,\cdot)$ measures the similarity between the question and the document given by the retriever.

We emphasize that the ranking feedback used to update the retriever comes from the widely adopted LLM reranking process. LUF does not introduce any additional steps during retrieval and does not delay the time for users to receive a response.

2.4 Updates Based on User Feedback

Users naturally provide meaningful information during their interactions with the LLM, which can be utilized in future conversations. Compared to the documents in the database, the information provided by users is more aligned with their preferences. To leverage the memory capabilities that the document database provides to the LLM, LUF prompts the LLM after each conversation to summarize any new information unknown to the LLM and adds it as a new document to the database. In the experiment, we use LLMs to simulate user feedback, which will be detailed in Section 3.1.

2.5 Multi-level Reranking

LLM reranking is a crucial component of LUF as it guides the retriever to adapt to the question domain shift. It is necessary to provide an additional prompt for LLM to outline the reranking task. As shown in Fig. 2, previous methods primarily adopt two forms of reranking: single reranking, where the LLM evaluates each retrieved document individually, and total reranking, which evaluates all documents together. Single reranking offers higher accuracy but incurs additional token computation costs, primarily due to the prompt. Conversely, total reranking yields the opposite results.

We observe that the results of retrieval typically follow a long-tail distribution, where higher-ranked positions have a much higher likelihood of providing relevant documents and thus have greater value compared to lower-ranked positions, as illustrated in Fig. 3. Therefore, we propose a multi-level reranking strategy, employing different reranking approaches for different positions, as shown in Fig. 2 (a). For higher-ranked positions, we use smaller reranking steps to enhance precision. As the process goes on, we gradually increase the step to reduce token consumption until at least one relevant document is found. It is a cost-effective solution that balances accuracy and computational overhead, allowing for accurate judgment without incurring excessive computational expenses.

	NQ	TQA	SQuAD
Kimi	37.34	60.16	18.93
GPT-4o	59.42	67.33	27.66

Table 2: Accuracy without retrieval of GPT-4o and Kimi.

3 Experiment

In this section, we first evaluated the retrieval performance of LUF on five QA datasets in two languages. Moreover, we evaluated the assistance of LUF to the LLMs' responses on re-ask datasets. Finally, we conducted experiments to demonstrate how feedback assists the LLM provide more accurate outcomes that fulfil user's requirements.

3.1 Implementation Details

Retriever. We used the pre-trained Contriever (Izacard et al., 2022a) as the retrieval model, which was trained on Wikipedia and CCNet and then tested on the test sets of all five datasets.

Base LLMs. We employed two language models: Kimi (Qin et al., 2024), primarily trained on Chinese corpora, and GPT-4o (OpenAI et al., 2024), primarily trained on English corpora. Table 2 shows the accuracy of both models' answers without any retrieval. On three English QA datasets, Kimi's accuracy was significantly lower than that of GPT-4o. We conducted our experiments based on these two models, which evidently have different knowledge bases, in order to demonstrate the generality of LUF.

Datasets. We used four English datasets to simulate domain shift scenarios and one Chinese dataset to simulate cross-language scenarios. The English datasets include Natural Questions (NQ) (Kwiatkowski et al., 2019), TriviaQA (TQA) (Joshi et al., 2017), SQuAD (Rajpurkar et al., 2016),

and Web Questions (WebQ) (Berant et al., 2013). All four datasets using articles extracted from Wikipedia as the initial document database as in previous work (Izacard et al., 2022a). The Chinese dataset is WebQA (Li et al., 2016), with the document database derived from all evidence in the WebQA.

Our method. To simulate domain shift in questions, the retriever used in LUF is pretrained only on Wikipedia and CCNET, without training on the training set, and is tested directly on the test set. The hyperparameter settings of LUF can be found in the Appendix A.

3.2 Baselines

We compare LUF with several methods that also utilize LLMs' feedback:

Retrieve and Rerank (R&R): The widely-used approach which involves retrieving documents based on the user-provided question and then using a LLM to rerank the results.

Query Rewrite: Query rewriting modifies or reformulates the initial user query to improve retrieval accuracy. We use the same LLM employed as a reranker to perform query rewriting.

Query2Doc: Query2Doc first transforms a user query into a pseudo-document, then concatenates the original query with the pseudo-document to serve as a new query. We use the same LLM serving as a reranker to generate pseudo-documents.

TENT: TENT (Test-Time Entropy Minimization) is a widely used test-time domain adaptation method, adjusting models during testing by minimizing prediction entropy on unlabeled test data.

R&R, Query Rewrite and Query2Doc use information from the LLM to enhance accuracy, while TENT is a data-driven method that aligns the target domain by training the retriever.

3.3 Main Results

3.3.1 Improvement of Retriever Accuracy

Table 3 shows the retrieval accuracy of different methods before and after LLM reranking. Across all baselines, the performance of GPT-4o surpassed that of Kimi, so we will only show the results achieved with GPT-4o. It is evident that LUF consistently achieved superior or comparable performance on all datasets, both before and after reranking. Notably, there was a significant improvement in R@5, which is particularly advantageous because it matches the standard procedure in RAG, where just a select number of very relevant papers are often presented as evidence to LLM.

In general, LUF exhibited more substantial enhancements on larger datasets due to the fact that smaller datasets, such as Web Questions, offer less feedback information from the LLM. In the case of Web Questions, the retriever was only updated once out of every 1000 questions. However, this update still led to an improvement.

LUF also demonstrated consistent improvements across different LLMs. Despite Kimi's ability to answer less than 20% of the questions correctly on SQuAD, the accuracy increase shown in Table 3 implies that when Kimi is used, it can still effectively assess the relevance between questions and documents and provide valuable feedback to the retriever through LUF. This suggests that LUF primarily use the LLM for analyzing semantic relevance, rather than relying on the LLM's knowledge for updates, hence showing LUF's generality. Compared to Kimi, GPT-4o has more knowledge and higher reranking accuracy, leading to greater improvements for the retriever.

For all methods, the reranked results show a significant improvement compared to the results without reranking, and the reranking accuracy of GPT-4o is higher than that of Kimi. Although the retriever's accuracy with GPT-4o is occasionally worse than with Kimi before reranking, the situations are generally reversed after reranking.

3.3.2 Improvement of LLMs' Response

To evaluate how LUF affects LLM outputs, we tested the accuracy of responses generated by LLMs on the NQ, TQA and SQuAD datasets. Specifically, we used GPT-4o to create two new questions for each original question in the datasets. The re-ask dataset was constructed with these semantically equivalent re-phrased questions. Initially, all methods were tested on the original datasets. LLMs then simulate user feedback to correct LLMs' false responses. Finally, we tested the methods with the re-ask datasets to simulate multiple users asking similar questions, mirroring the real-world scenario. The details of simulating user questions and feedback can be found in the appendix.

Table 4 shows the performance of all methods on both the original and re-ask datasets, with LUF delivering the most accurate responses across all datasets. On the original datasets, LUF's higher retrieval accuracy supplied LLMs with more pre-

Method	LLM	Natural Questions		Trivia QA		SQuaD		Web Questions		WebQA	
		R@5	R@20	R@5	R@20	R@5	R@20	R@5	R@20	R@5	R@20
Before Reranking											
R&R	-	47.80	67.80	59.40	67.80	43.19	62.33	36.66	55.86	12.73	20.93
Query Rewrite	GPT-4o	48.17	68.48	60.76	68.94	44.52	63.52	37.01	56.50	13.43	22.09
Query2Doc	GPT-4o	47.40	68.03	60.51	69.20	45.03	63.84	37.45	**56.94**	14.32	22.78
TENT	-	49.03	69.67	59.66	67.80	41.71	60.95	37.16	55.51	11.38	17.13
LUF	Kimi	53.13	73.57	62.16	70.13	**49.39**	**67.66**	**38.73**	56.84	16.90	26.92
LUF	GPT-4o	**53.57**	**74.02**	**62.42**	**70.17**	49.00	67.47	38.24	56.20	**17.56**	**28.04**
After Reranking											
R&R	Kimi	58.92	70.78	65.88	69.53	57.53	66.11	50.94	59.50	20.04	24.14
R&R	GPT-4o	59.86	71.00	67.05	69.84	58.28	66.26	51.67	59.65	20.57	24.37
Query Rewrite	GPT-4o	61.00	70.55	67.75	70.52	59.73	67.31	52.07	60.29	21.30	26.36
Query2Doc	GPT-4o	59.81	71.61	67.68	70.68	60.33	67.78	52.21	60.24	22.19	26.92
TENT	GPT-4o	58.67	68.78	65.92	68.96	59.89	67.83	51.62	59.10	16.87	19.58
LUF	Kimi	62.22	75.29	68.15	71.64	62.82	**70.98**	52.21	**60.33**	25.26	30.22
LUF	GPT-4o	**64.27**	**76.34**	**69.35**	**71.91**	**63.23**	70.91	**52.41**	59.99	**26.65**	**31.55**

Table 3: Retrieval results before and after reranking. The highest results are bolded.

cise evidence, resulting in better responses. On the re-ask datasets, feedback from users significantly improved LUF's retrieval accuracy, which in turn markedly enhanced the LLM's response accuracy. It is noteworthy that without using RAG, Kimi's response accuracy was significantly lower than that of GPT-4o across all datasets. However, with the help of evidence, the performance gap between Kimi and GPT-4o narrowed considerably, especially on the re-ask dataset. This demonstrates that LUF can effectively utilize user feedback to help LLMs learn unknown knowledge and provide desired answers. Furthur more, an interesting phenomenon was observed in the TQA dataset: GPT-4o's retrieval-augmented responses of baselines were less accurate than its direct answers. This corroborates the findings in previous research (Yoran et al., 2024) that irrelevant documents make LLMs' results less accurate.

Results presented in Table 4 assume that the users provide correct feedback, but in reality, users can provide erroneous feedback. We simulated incorrect feedback on 100 randomly sampled questions from Natural Questions. GPT-4o corrected the user's error on 50.88% of the questions answered correctly, preventing it from affecting future responses, indicates that LUF has a certain robustness to incorrect feedback. Examples of incorrect feedback and corresponding solutions will be further analyzed in Section 6 and Appendix A.2.1.

3.3.3 Multi-level Reranking

We conducted a comparative analysis of the tokens consumed by different reranking methods. For retrieval results from 500 randomly sampled questions, we applied the different methods and calculated the average token consumption (both input and output) to achieve a similar R@5. "Single" means each document is individually assessed by the LLM, while "Every 5" and "Every 20" refer to a reranking step of 5 or 20 documents, respectively.

Table 5 shows the result, the multi-level reranking consumes the fewest tokens while achieving similar R@5. The Single reranking strategy generally outperforms the "Every 5" and "Every 20" across most datasets, as it more effectively selects the most relevant documents from the top-ranked results.

3.4 Further Investigations

3.4.1 Impact of Reranking Accuracy

LUF is updated based on reranking results from LLMs, so relies on the accuracy of LLM feedback. We simulated the impact of LLMs' feedback with different accuracy on the retriever using the NQ and TQA datasets. In Fig. 4, the accuracy refers specifically to the LLM's ability to judge single relevant documents, as there is no significant differ-

Method	Natural Questions				TriviaQA				SQuAD			
	Original		Re-Ask		Original		Re-Ask		Original		Re-Ask	
	QA	R@5	QA	R@5	QA	R@5	QA	R@5	QA	R@5	QA	R@5
Kimi+	37.34	-	37.29	-	60.16	-	57.83	-	18.93	-	17.47	-
R&R	48.37	58.92	44.60	57.81	61.63	65.88	58.01	59.13	36.40	57.53	32.03	52.70
Query2Doc	49.34	61.61	45.15	59.17	61.81	66.37	58.37	60.24	35.84	58.19	31.50	54.76
Query Rewrite	48.39	58.86	44.65	57.34	61.76	66.23	58.54	60.39	36.13	59.35	32.72	56.16
TENT	47.98	56.93	44.29	57.53	61.65	64.76	58.21	58.39	36.82	58.79	32.20	55.46
LUF	**50.50**	**62.22**	**66.04**	**71.29**	**63.58**	**68.15**	**72.54**	**72.48**	**37.85**	**62.82**	**58.97**	**77.84**
GPT-4o +	59.42	-	57.42	-	67.33	-	65.35	-	27.66	-	24.75	-
R&R	63.38	59.86	59.42	58.84	64.27	67.05	68.85	59.86	34.95	58.28	35.67	53.02
Query2Doc	64.46	61.00	58.75	59.45	63.89	67.75	68.31	60.91	37.28	59.73	37.02	55.51
Query Rewrite	64.18	59.81	59.47	58.64	64.56	67.68	69.42	60.97	37.09	60.33	37.52	56.09
TENT	63.35	58.67	58.09	58.25	64.43	65.92	68.81	59.06	36.40	59.89	36.74	55.75
LUF	**67.31**	**64.27**	**79.20**	**72.77**	**68.32**	**69.35**	**75.25**	**72.85**	**40.04**	**63.23**	**54.60**	**79.81**

Table 4: The accuracy of the answers provided by the LLM, with "QA" representing the proportion of responses from the LLM that contain the correct answer. The highest results are bolded.

Method	Natural Questions			TriviaQA			SQuAD		
	R@5	Docs	Tokens	R@5	Docs	Tokens	R@5	Docs	Tokens
Single	55.75	39	3,257.6	57.64	38	3,313.7	55.09	38	3,174.1
Every 5	54.11	75	5,899.2	57.51	75	5,316.9	54.35	80	5,448.8
Every 20	55.44	80	4,920.1	57.95	80	5,004.3	55.02	80	4,851.9
Multi-level	55.78	40	**2,816.4**	57.9	40	**2,880.9**	55.08	40	**2,777**

Table 5: The number of tokens consumed by different reranking methods.

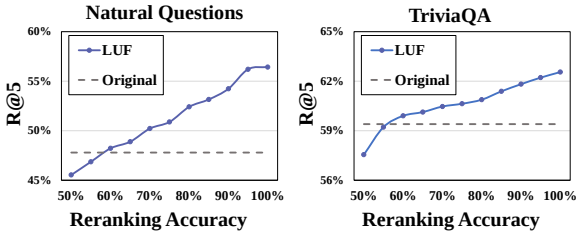


Figure 4: R@5 of LUF with different reranking accuracy.

ence in the accuracy of different LLMs in judging irrelevant documents.

Fig. 4 illustrates the impact of reranking accuracy on the retriever's performance. Even when the LLM correctly identifies only 60% of relevant samples, its feedback still improves the retriever's performance, demonstrating LUF is highly robust to feedback accuracy. This robustness comes from LLMs' high accuracy in filtering out irrelevant samples, the eliminated hard negative samples help the retriever improve. We tested three widely used LLMs: GPT-4o, Kimi, and DeepSeek-v2 (DeepSeek-AI, 2024), all of which achieved reranking accuracy above 70%, suggesting that LUF can be applied to them.

3.4.2 Case Study

In Fig. 5, we illustrate two examples where LUF enhances the performance of LLMs.

Feedback helps LLM correct erroneous knowledge. Fig. 5 (a) presents a scenario where GPT-4o initially provides an incorrect response to a question. After user correction, upon being asked a similar question later, GPT-4o successfully answers correctly by leveraging the newly added document in the document database.

Feedback helps LLM align with user preferences. Fig. 5 (b) demonstrates LUF's ability to help the LLM learn user's preferences: when the user asks, "What courses can I take to learn?" LUF retrieves information from the previous conversation indicating that the user is studying computer science and recommends a series of relevant

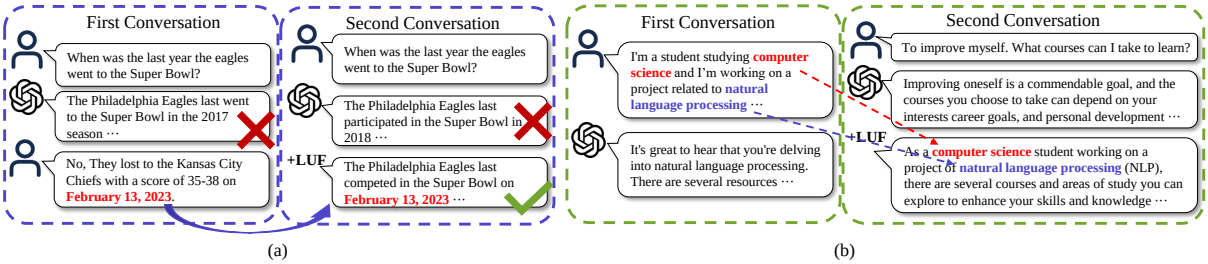


Figure 5: Examples of LUF's responses.

Method	Time per Question(s)	
	TriviaQA	SQuAD
R&R	3.6	4.6
Query Rewrite	5.4	6.5
Query2Doc	4.3	5.2
TENT	5.0	6.0
LUF	3.9	5.5

Table 6: Time consumption of different methods.

courses. In contrast, the standard RAG only provides courses from various fields, showing a weaker correlation with the user's preference.

3.4.3 Time Consumption

We compared the time consumption of different methods with two RTX 4090s. Each method answered 3000 questions and the average time required for each question is shown in Table 6. Compared to the simplest R&R, the additional time cost introduced by LUF is within an acceptable range. The extra time spent by LUF primarily comes from summarizing users' conversations and updating the retriever, which can be performed in parallel with retrieval in practice to reduce time cost.

4 Related Work

4.1 Retrieval-Augmented Generation

Research on RAG has advanced rapidly in recent years. Sparse retrieval techniques, such as BM25 (Sparck Jones, 1988), are simple and effective (Chen et al., 2024; Jiang et al., 2023; Ram et al., 2023). Dense retrieval methods like Dense Passage Retriever (DPR) (Karpukhin et al., 2020) demonstrate greater flexibility and adaptability (Izacard et al., 2022b; Shi et al., 2024; Sachan et al., 2024; Siriwardhana et al., 2023).

Recent studies have proposed various pre-retrieval and post-retrieval enhancement strategies. Pre-retrieval enhancement strategies, such as Query2doc (Wang et al., 2023), Hypothetical Document Embedding (HyDE) (Gao et al., 2023), and Query Rewriting (Ma et al., 2023) improve the relevance of retrieval results by reorganizing or expanding queries. Post-retrieval enhancement strategies, such as R2G (Sachan et al., 2023), filter irrelevant information by reranking retrieval results. These methods use static retrievers and overlook the role of feedback.

4.2 Feedback for Language Models

Feedback has been widely used in NLP and applied to many traditional tasks, such as question answering (Li et al., 2022; Harabagiu et al., 2001), text summarization (Nguyen et al., 2022; Liu et al., 2023), and machine translation (Saluja et al., 2012).

We note that two recent studies have also attempted to improve RAG using feedback too: RaFe (Wu et al., 2024) utilizes feedback to improve query rewriting, while InstructRAG (Wei et al., 2024) employs feedback to train LLMs. Both approaches only utilize feedback from LLMs, whereas LUF also considers user feedback. RaFe focuses more on enhancing query rewriter performance through feedback, while LUF's motivation is to adapt the entire RAG system to changes in questions and answers, so we adopt the most widely used RAG process without query rewriter. InstructRAG requires training the LLM, which is computationally costly and does not fit our use case.

5 Conclusion

To address the performance degradation of RAG and LLM on unseen questions, we propose an RAG framework called LUF based on user and LLM feedback. By updating the retriever and the document database, both the retriever and the LLM adapt to the shifts in question and answer domains, thereby improving their ability to provide responses that align with user preferences. Experiments conducted on five datasets demonstrate the effectiveness of our method.

6 Limitations

In this work, we primarily conducted evaluations on the Question-Answering task, the performance of LUF on other tasks such as commonsense reasoning and open-domain summarization remains unknown. Besides the LLMs mentioned in the paper, we also tested smaller LLMs like Qwen1.5-32b-chat. For these models with smaller parameters, their reranking accuracy was below the minimum threshold required to improve the retriever, and LUF did not provide any meaningful enhancement.

Compared to LLM feedback, user feedback is not as reliable as simulated in the experiments. Incorrect user feedback may add false information into the document database. For certain questions, LLM can identify the user's erroneous feedback through its own knowledge and refuse to update the document database, preventing the impact of incorrect feedback. However, if the user's incorrect answer is sufficiently misleading, it can still mislead the LLM's future responses to similar questions. Examples of LUF rejecting incorrect feedback and being misled by incorrect feedback can be found in Appendix A.2.1. By restricting newly added documents to be retrievable only by the corresponding user, we can prevent incorrect information from affecting other users, but this also limits the LUF's ability to utilize user feedback. In this paper, we focus more on the improvement brought by user feedback, and finding ways to more effectively identify and leverage the information provided by users is a promising direction for future research.

7 Ethics Statement

In this study, we utilized publicly available datasets that do not contain any personal or private information, ensuring full compliance with ethical guidelines. The prompts used and the outputs generated by the LLMs were selected to exclude any content that might be discriminatory, violent, or otherwise inappropriate. No personal data was collected throughout the experimental process, and the design and execution of the experiments pose no negative societal impact. Therefore, this research adheres to ethical standards, with no risks of privacy infringement or harmful social consequences.

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, and et al. 2021. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning*.

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2024. Benchmarking large language models in retrieval-augmented generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):17754–17762.

DeepSeek-AI. 2024. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model.

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)*, pages 1762–1777, Toronto, Canada. Association for Computational Linguistics.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey.

Sanda Harabagiu, Dan Moldovan, Marius Pasca, Rada Mihalcea, Mihai Surdeanu, Razvan Bunsecu, Roxana Girju, Vasile Rus, and Paul Morarescu. 2001. The role of lexico-semantic feedback in open-domain textual question-answering. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 282–289, Toulouse, France. Association for Computational Linguistics.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022a. Unsupervised dense information retrieval with contrastive learning.

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

Zhengbao Jiang, Frank 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*, pages 7969–7992, Singapore. Association for Computational Linguistics.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, and et al. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks.

Peng Li, Wei Li, Zhengyan He, Xuguang Wang, Ying Cao, Jie Zhou, and Wei Xu. 2016. Dataset and neural recurrent sequence labeling model for open-domain factoid question answering.

Zichao Li, Prakhar Sharma, Xing Han Lu, Jackie Cheung, and Siva Reddy. 2022. Using interactive feedback to improve the accuracy and explainability of question answering systems post-deployment. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 926–937, Dublin, Ireland. Association for Computational Linguistics.

Yixin Liu, Budhaditya Deb, Milagro Teruel, Aaron Halfaker, Dragomir Radev, and Ahmed Hassan Awadallah. 2023. On improving summarization factual consistency from natural language feedback. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15144–15161, Toronto, Canada. Association for Computational Linguistics.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315, Singapore. Association for Computational Linguistics.

Duy-Hung Nguyen, Nguyen Viet Dung Nghiem, Bao-Sinh Nguyen, Dung Tien Tien Le, Shahab Sabahi, Minh-Tien Nguyen, and Hung Le. 2022. Make the most of prior data: A solution for interactive text summarization with preference feedback. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1919–1930, Seattle, United States. Association for Computational Linguistics.

OpenAI, Josh Achiam, Steven Adler, and et al. 2024. Gpt-4 technical report.

Ruoyu Qin, Zheming Li, Weiran He, Mingxing Zhang, Yongwei Wu, Weimin Zheng, and Xinran Xu. 2024. Mooncake: A kvcache-centric disaggregated architecture for llm serving.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331.

Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. 2021. Contrastive learning with hard negative samples.

Devendra Singh Sachan, Mike Lewis, Mandar Joshi, Armen Aghajanyan, Wen tau Yih, Joelle Pineau, and Luke Zettlemoyer. 2023. Improving passage retrieval with zero-shot question generation.

Devendra Singh Sachan, Siva Reddy, William Hamilton, Chris Dyer, and Dani Yogatama. 2024. End-to-end training of multi-document reader and retriever for open-domain question answering. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA. Curran Associates Inc.

Avneesh Saluja, Ian Lane, and Ying Zhang. 2012. Machine translation with binary feedback: a large-margin approach. In *Proceedings of the 10th Conference of the Association for Machine Translation in the Americas: Research Papers*, San Diego, California, USA. Association for Machine Translation in the Americas.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2024. REPLUG: Retrieval-augmented black-box language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8371–8384, Mexico City, Mexico. Association for Computational Linguistics.

Shamane Siriwardhana, Rivindu Weerasekera, Elliott Wen, Tharindu Kaluarachchi, Rajib Rana, and Suranga Nanayakkara. 2023. Improving the domain adaptation of retrieval augmented generation (RAG) models for open domain question answering. *Transactions of the Association for Computational Linguistics*, 11:1–17.

Karen Sparck Jones. 1988. *A statistical interpretation of term specificity and its application in retrieval*, page 132–142. Taylor Graham Publishing, GBR.

Liang Wang, Nan Yang, and Furu Wei. 2023. Query2doc: Query expansion with large language models. In *Conference on Empirical Methods in Natural Language Processing*.

Zhepei Wei, Wei-Lin Chen, and Yu Meng. 2024. Instructrag: Instructing retrieval-augmented generation with explicit denoising.

Zhongkai Wu, Ziyu Wan, Jing Zhang, Jing Liao, and Dong Xu. 2024. Rafe: Generative radiance fields restoration.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024. Making retrieval-augmented language models robust to irrelevant context.

Zichun Yu, Chenyan Xiong, Shih Yuan Yu, and Zhiyuan Liu. 2023. Augmentation-adapted retriever improves generalization of language models as generic plug-in. In *Annual Meeting of the Association for Computational Linguistics*.

A Appendix

A.1 Prompt

Frequent LLM calls were required to generate responses in our experiment. In this section, we present the prompts used to accomplish different tasks.

Prompt A.1.1: Single Document Reranking

Please analyze whether this document can help answer this question, and provide your answer using single "yes" or "no" in the end.
Question:
$\langle Question \rangle$
Document:
$\langle Document \rangle$

Prompt A.1.2: Multiple Documents Reranking

Below is a question, along with $\langle Num\,of\,Docs \rangle$ documents that might be related to this question. Please judge if which document(s) are relevant to the question, and finally provide your answer using the document number + yes like "answer:1,yes" or "no".
Question:
$\langle Question \rangle$
Document 1:
$\langle First\,document \rangle$
Document 2:
$\langle Second\,ocument \rangle$
...

When performing rerank for a single document and multiple documents using LLM, to ensure that the LLM's response adheres to a fixed format suitable for code analysis, we use prompts as shown in Prompt A.1.1 and A.1.2.

Prompt A.1.3: User Feedback Stimulation

Here is a conversation between a user and the LLM, where the LLM incorrectly answered the user's question. The correct answer to the user's question is $\langle Answer \rangle$. Please simulate how the user would correct the LLM's response, providing only the user's reply.
User:$\langle Question \rangle$
LLM:$\langle Response \rangle$

Prompt A.1.4: User Feedback Summarization

The following is a conversation between a user and an LLM. Did the user provide any information that the LLM does not know or has misunderstood? If so, please summarize the information given by the user.
User:$\langle Question \rangle$
LLM:$\langle Response \rangle$
User:$\langle Simulated\,user\,feedback \rangle$

In the experiment in Table 4, we used LLMs to simulate and summarize user feedback. Prompt A.1.3 asks the LLM to simulate user feedback, specifically correcting the errors in the responses provided by the LLM. Prompt A.1.4 then instructs the LLM to summarize the user feedback and create a new document to be added to the database.

Prompt A.1.5: Query Rewrite

Provide a better search query for retriever to search the given question.
Original question: What 2000 movie does the song "All Star" appear in?
New question: 2000 movie "All Star" song
Original question:$\langle Question \rangle$
New question:

Query rewrite uses the original prompt from previous work (Ma et al., 2023) where the LLM was used as a rewriter. as shown in Prompt A.1.5. Query2doc follows the original prompt in previous work (Wang et al., 2023).

A.2 Additional Experiment Results

A.2.1 Error Feedback

We simulated scenarios where users provide incorrect feedback on 100 randomly sampled questions

Method	Natural Questions		TriviaQA		SQuAD	
	QA	R@5	QA	R@5	QA	R@5
R&R	44.18	57.01	57.21	58.45	31.79	52.26
+Multi-level Reranking	44.60	57.81	58.01	59.13	32.03	52.70
+LLM Feedback	47.23	61.69	60.67	60.81	34.32	57.79
+User Feedback	**66.04**	**71.29**	**72.54**	**72.48**	**58.97**	**77.84**

Table 7: Ablation study results.

Method	Time to Retrieve 3,000 Questions (s)					
	TriviaQA			SQuAD		
	Retrieve	LLM	Train	Retrieve	LLM	Train
R&R	4,096.4	6,696.8	-	4,077.1	9,706.9	-
Query Rewrite	8,157.2	8,110.0	-	8,134.8	11,321.4	-
Query2Doc	4,092.9	8,931.2	-	4,069.6	11,635.4	-
TENT	8,162.4	6,709.2	66.0	8,138.4	9,704.8	66.0
LUF	4,091.4	7,639.5	65.8	4,069.5	12,261.8	65.6

Table 8: The detailed time required to retrieve 3,000 questions of different methods.

Method	Web Questions			WebQA		
	R@5	Docs	Tokens	R@5	Docs	Tokens
Single	49.16	35	2,853.3	18.68	35	2,069.6
Every 5	47.64	70	5,129.8	18.29	75	2,406.5
Every 20	49.02	80	4,933.5	18.72	60	1,623.9
Multi-level	49.26	40	2,791.6	18.72	40	1,183.0

Table 9: The number of tokens consumed by different reranking methods on Web Questions and WebQA.

from the Natural Questions dataset. For questions that GPT-4o correctly answered, after receiving incorrect feedback, the LLM was still able to provide correct answers for 50.88% of them. Fig. 6 (a) shows examples where the LLM identified the user's incorrect feedback and refused to update the article database, preventing it from affecting future conversations. However, when the incorrect feedback was more misleading, the LLM was misled and gave incorrect answers in future interactions, as shown in Fig. 6 (b). Our future work will focus on improving the identification of user feedback to make LUF more robust.

A.2.2 Ablation Study

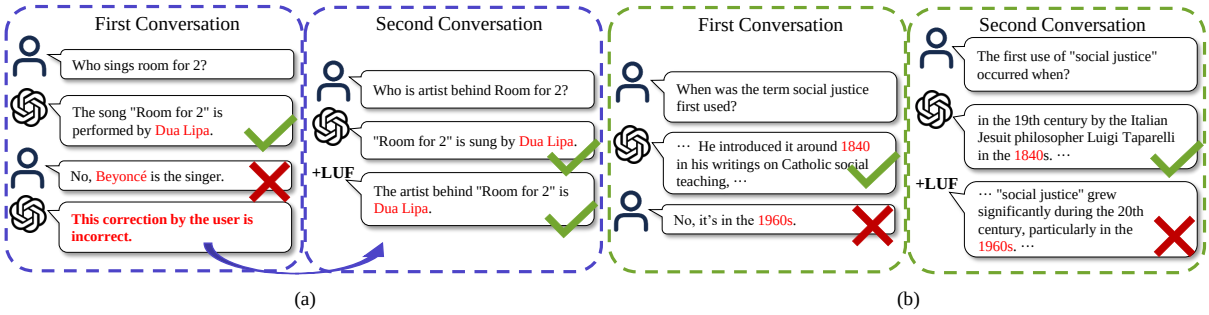
To understand the specific roles of different components in LUF, we conducted a series of ablation experiments. We tested the improvements brought by different strategies within LUF across three reask datasets, as shown in Table 7.

After applying different strategies, both retrieval and response accuracy improved. Replacing the single reranking with multi-level reranking that consumes a similar number of tokens, there has been a certain improvement retrieval accuracy. LLM Feedback mainly helps the retriever adapt to the shift in the domain of questions, while user feedback significantly improves the results obtained by the retriever and LLM by adding new information to the database.

A.2.3 Time Consumption

Table 8 shows the detailed time required by different methods to answer 3,000 questions, with all results tested on an dual RTX 4090 and Intel Xeon Gold 6430 machine. "Retrieve" refers to the time spent on dense retrieval, "LLM" indicates the time taken for LLM generation, and "Train" represents the time required for model training. For Query rewrite, we adopted the approach from (Wu et al., 2024), where the results of two rewritten queries are fused during retrieval.

Compared to the simplest R&R process, the additional time introduced by LUF is within an acceptable range. Moreover, aside from reranking, LUF does not require any preprocessing before retrieval, meaning the time from when the user asks a question to receiving an answer is the same as in the R&R. The additional time is attributed to summarizing user feedback with the LLM and training the retriever (as the "LLM" and "Train" in the Table 6), both of which can be performed in parallel with retrieval, thus not adding extra time.



(a) (b)

Figure 6: Examples of error feedback.

Dataset	Test Set Size		Number of Documents
	Kimi	GPT-4o	
Natural Questions	3,602	3,610	21,015,324
TriviaQA	11,297	11,313	21,015,324
SQuAD	10,554	10,570	21,015,324
Web Questions	2,029	2,032	21,015,324
WebQA	3,024	3,024	3,024

Table 10: The number of test questions and documents of different dataset.

Dataset	Update Interval	Epochs	Learning Rate
Natural Questions	2,000	4	5e-06
TriviaQA	4,000	4	1e-05
SQuAD	4,000	4	1e-05
Web Questions	1,000	1	5e-06
WebQA	2,000	3	5e-06

Table 11: Training details on different datasets.

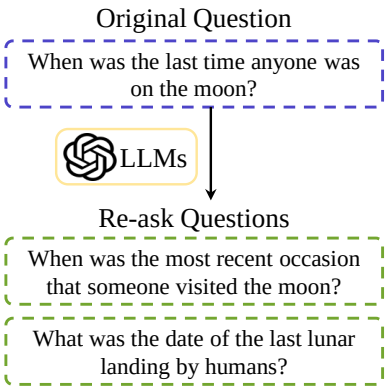


Figure 7: Examples of constructed re-ask dataset.

A.2.4 Token Consumption

Table 9 shows the token consumption of different reranking strategies on Web Questions and WebQA. Compared to other methods, Multi-level reranking used the fewest tokens on the both English and Chinese datasets.

A.2.5 Additional Examples

Fig. 8 presents two examples of responses provided by LUF.

Fig. 8 (a) illustrates how user feedback can assist the LLM in correcting outdated information. For recent events, the LLM's knowledge may not be promptly updated; however, LUF can prevent giving outdated and incorrect answers by leveraging user feedback.

In Fig. 8 (b), the user's question has multiple correct answers, but the answer directly provided by the LLM was not the one the user desired. Through the first conversation, LUF learned the user's preferences and then provided the answer that met the user's expectations.

B Implementation Details

B.0.1 Dataset

Table 10 shows the number of questions used for testing and the number of documents in the database across different datasets. For the four English datasets, we used the splited Wikipedia from previous studies (Karpukhin et al., 2020) as the document database, while for the Chinese dataset WebQA, we used the evidence provided within the dataset. Since Kimi was unable to answer a few questions, the number of questions tested by Kimi is smaller than that tested by GPT-4o.

B.0.2 Training

Table 11 presents the training details of different datasets. Updating the retriever with more than 2,000 questions each time results in more significant improvements.

B.0.3 Construction of Re-ask Datasets

Fig. 7 shows an example of how we constructed the re-ask datasets. We used an LLM to rephrase the original question, generating two new questions with the same meaning but different structures, simulating the scenario where different users ask similar questions.

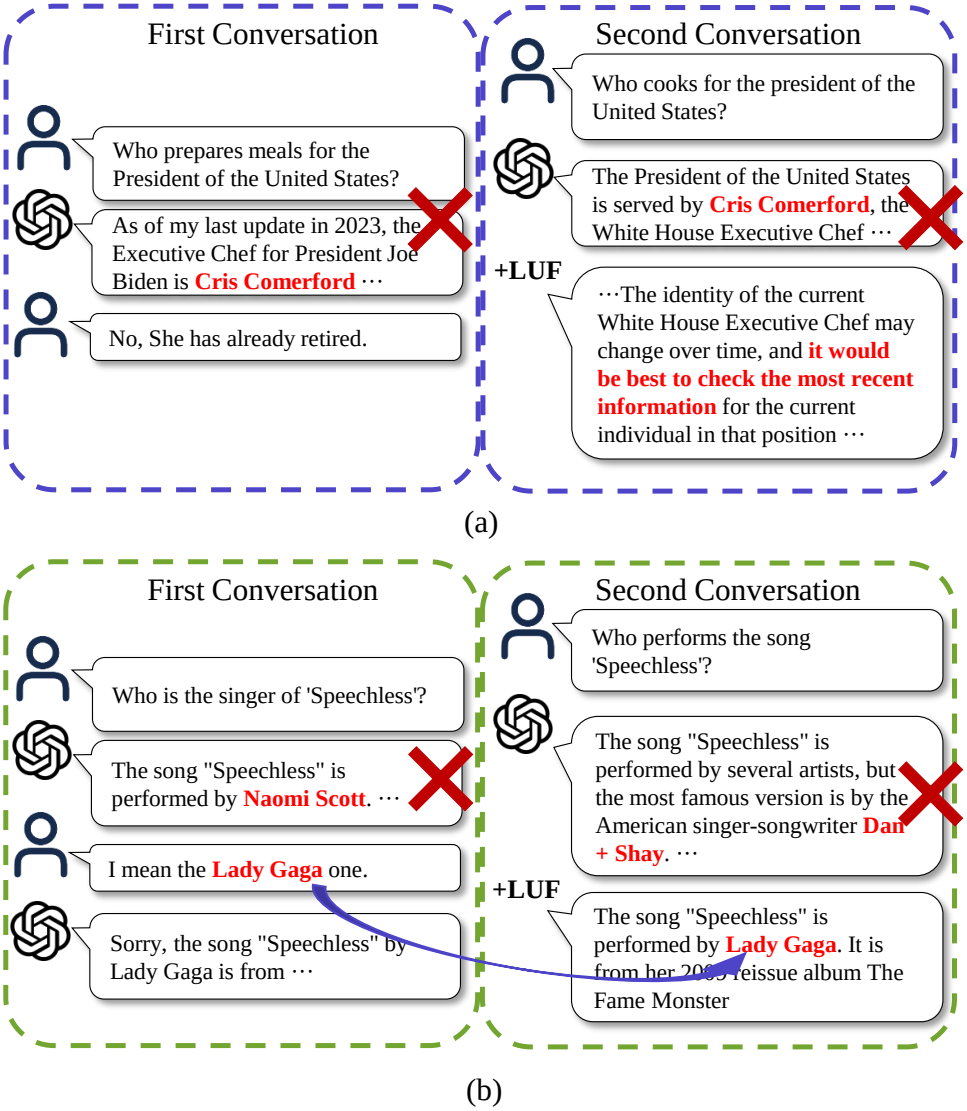


Figure 8: Examples of other LUF reponses.