re-thinking: In-depth Interactive Thinking with Retrieved Knowledge for Large Language Models

Anonymous ACL submission

Abstract

The hallucinations of large language models (LLMs) have the potential to be solved by Retrieval-Augmented Generation (RAG), which incorporates external knowledge during the generation process. Although effective, incorrectly retrieved knowledge uncontrollably carries rich noise, which damages RAG performance. In this paper, we propose a simple yet highly effective prompting strategy: *re-thinking*. Drawn inspiration from how humans selectively learn with external knowledge, *re-thinking* considers the retrieved knowledge cannot be treated equally, which means selectively retaining and removing knowledge. To gather insightful and comprehensive selection process, additionally, we develop a fine-grained and in-depth interaction mechanism, which equips knowledge with queries again, making them have richer, back-and-forth interactions, obtaining fine-grained correlation or slight differences. Experiments conducted on various reasoning benchmarks and LLMs demonstrate the effectiveness of the proposed *re-thinking* framework.

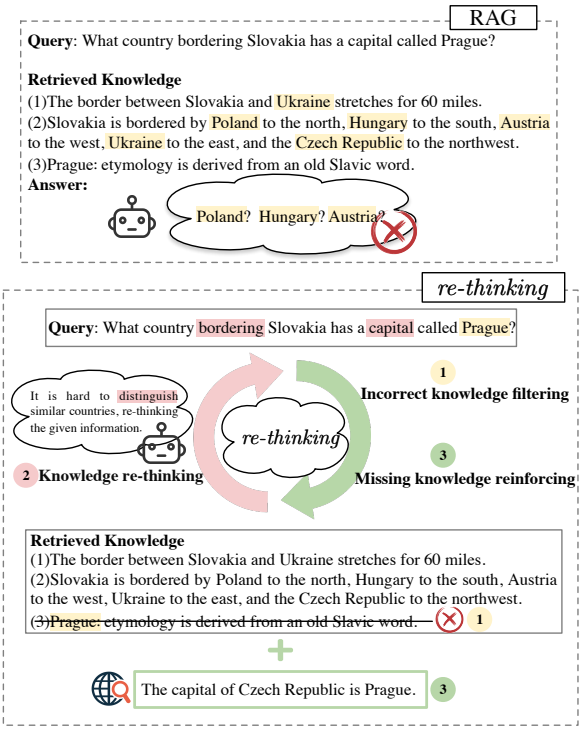

Figure 1: An example of adopting RAG and *re-thinking* on reasoning of LLMs respectively.

1 Introduction

Large language models (LLMs) have shown impressive performance in understanding and generating natural language (Touvron et al., 2023a; OpenAI, 2023). However, their training process heavily relies on large-scale static collections of text, overlooking the ever-changing nature of real-world data (Kandpal et al., 2023). As a result, LLMs still struggle with hallucinations (Zhang et al., 2023a), particularly when faced with queries that go beyond the scope of their training data. This issue significantly undermines the reliability of LLMs and hampers their practical application (Huang et al., 2023). One promising solution to address this challenge is Retrieval-Augmented Generation (RAG) (Lewis et al., 2020b). By incorporating external data during the generation process, RAG enhances the model's ability to produce dependable and accurate responses.

RAG offers a comprehensive and efficient approach to tackle hallucinations and knowledge delays in LLMs (Gao et al., 2023). While it can be beneficial in certain cases, there is also a risk of reduced performance for LLMs when retrieving incorrect knowledge with rich noise. Our analysis reveals that sometimes the retrieved knowledge is unrelated to the query at hand, and in other cases, it is only partially supported. For example, in Figure 1, although retrieved Knowledge 3 and the input query share the same word "Prague", they are not actually related. Besides, Knowledge 1 and 2 carry the country list bordering Slovakia but omit to indicate which country's capital is Prague, resulting in the retrieved knowledge only supporting par-

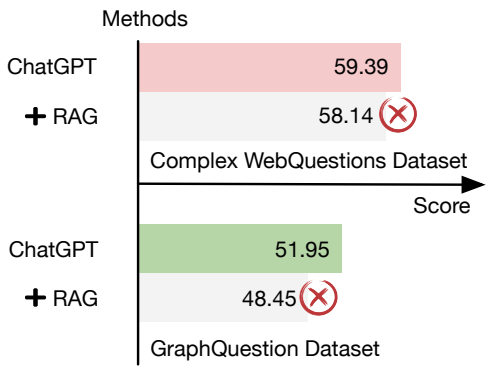

Figure 2: The EM score results of ChatGPT and (ChatGPT + RAG) on two-well known datasets. EM score measures the percentage of predicted responses that match the answer. We utilize the version of `gpt-3.5-turbo-0301` for ChatGPT. From the results, we find that RAG unexpectedly reduces the ChatGPT reasoning performance.

tial queries. Besides detailed case analysis, we further rigorously experiment on two well-known datasets (Talmor and Berant, 2018; Su et al., 2016) with RAG[1]. Figure 2 demonstrates that with the implementation of RAG, the score unexpectedly decreased by 1.25% and 3.5%. Based on the above analysis, we summarize the key factor briefly hinders RAG is the retrieval knowledge noise.

To eliminate knowledge noise, we consider the retrieved knowledge cannot be treated equally, which means selectively retaining and removing knowledge. Thereby, we propose a simple yet highly effective prompting strategy: *re-thinking*. LLM thinks and infers the retrieved knowledge again, identifying any contradictions or missing pieces of knowledge, later picking effective knowledge. For example, in Figure 1, with *re-thinking*, LLM discovers Knowledge 3 is not related to the query (Step 1).

However, the primary *re-thinking* mechanism in LLM is insufficient and superficial. LLM lacks the ability to thoroughly in-depth analyze the connection between a query and other coarse-related knowledge, especially when there is some minor noise, like presenting a list of similar countries that are hard to distinguish. This limitation stems from the fact that the current decoder-only causal language modeling architecture typically works in a linear manner. The retrieved knowledge cannot reflect and reinforce processed information, potentially overlooking the more intricate and interactive thinking process that humans employ to solve complex problems.

Drawing inspiration from how humans learn and solve problems, in the *re-thinking* mechanism, we furthermore develop knowledge re-thinking, which equips the knowledge with queries again, engaging them re-examining and in-depth reflecting on the relevance between the information and queries. Exemplified in Figure 1, with knowledge re-thinking (Step 2), LLM detects central but subtle missing information about "capital" in the query. By following this strategy, we can continuously reinforce and enhance our initial understanding of the relationship between the information and the queries. Overall, in the *re-thinking* mechanism, LLM firstly infers the correlation between the query and the retrieved knowledge to export clue information. With this clue information, LLM incorporates queries again, allowing the query and the retrieved knowledge to have richer, back-and-forth interactions, obtaining in-depth and fine-grained correlation information. With the fine-grained correlation information, LLM executes a re-search of knowledge and ultimately provides the responses.

To demonstrate the effectiveness of the proposed *re-thinking* mechanism, we conducted a comprehensive series of experiments, which proved that *re-thinking* can consistently improve reasoning accuracy on different datasets and LLMs. Meanwhile, the case studies for different scenarios specifically declare the capabilities of *re-thinking*.

The main contributions of this paper include:

- We focus on the RAG task and consider the noise existing in the retrieved knowledge. To eliminate knowledge noise, we propose a simple yet highly effective prompting strategy: *re-thinking*, which lets LLM identify any contradictions or missing pieces of knowledge, later picking effective knowledge.

- In the *re-thinking* mechanism, we furthermore develop knowledge re-thinking, which engaging the knowledge and queries re-examining and in-depth reflecting on the relevance, finding the minor knowledge noise.

- We evaluate *re-thinking* mechanism on various reasoning benchmarks and LLMs. The results demonstrate the effectiveness of *re-thinking*.

[1]We carefully select information retrieval tools that are open source and widely utilized by researchers (Pan et al., 2023): https://github.com/deedy5/duckduckgo_search

2 Related Work

2.1 Prompting and In-Context Learning

Large Language Models (LLMs) have become a focal point in Natural Language Processing (NLP) (Liu et al., 2023; Brown et al., 2020; Schick and Schütze, 2020). However, their training and upkeep are often prohibitively expensive. As a result, In-Context Learning (ICL) (Wei et al., 2022), which utilizes instruction prompts and adapts LLMs to various tasks without additional training (Dong et al., 2022), has gained popularity. Specifically, a specific prompting strategy that prefixes queries with zero-shot or few-shot example demonstrations, enables models to make analogous predictions.

2.2 Reasoning with LLM

The field of reasoning with LLMs has evolved significantly, leading to breakthroughs in several reasoning benchmarks. Key advancements in reasoning tasks have been achieved through typically decomposing complex queries into simpler, sequential steps, exemplified by Chain-of-Thought (CoT) (Zhang et al., 2023b; Kojima et al., 2022) prompting and its variations. For example, the typical variations such as Self-Consistency (Chen et al., 2023) use majority voting from multiple chains, and Least-to-most prompting (Zhou et al., 2022), which breaks down queries into simpler subqueries. More complex structures, such as CoRe (Zhu et al., 2022) and heuristic-based search methods, have also been explored to enhance search algorithms.

2.3 Retrieval-augmented LLM

Retrieval Augmentation Generation (RAG) (Lewis et al., 2020a) represents a pivotal innovation in the domain of LLMs, notably enhancing their generative capabilities and solving the hallucinations. Key enhancing factors of RAG's methodology involve an initial retrieval phase, where LLMs consult external databases to source relevant information before generating responses (Lazaridou et al., 2022). This approach anchors the outputs in factual data, thereby markedly elevating their precision and contextual relevance (Jiang et al., 2023), and effectively mitigates the production of factually inaccurate or "hallucinated" content (Jurafsky et al., 2020; Lee et al., 2018). Generally, most research focuses on designing different patterns for more accurate retrieval knowledge. However, the erroneous knowledge noise brought by retrieval results is inevitable, and handling knowledge noise after retrieving is ignored by researchers.

3 Method

3.1 Overall Framework

The reasoning task using an LLM $\mathcal{M}$ can be formally defined as $y = \mathcal{M}(x, \texttt{prompt})$, where x represents the input query, $\texttt{prompt}$ means the entering instructions and y corresponds to the predicted responses. Considering the responses often encounter hallucinations, due to the LLM memories outdated knowledge, an additional knowledge retrieval module is incorporated, which retrieval external latest knowledge K into the reasoning process. Therefore, the reasoning process can be re-formally defined as $y = \mathcal{M}(K, x, \texttt{prompt})$.

However, the noise existing in K hampers the reasoning performance. With these concerns, the knowledge noise will be substantially reduced based on our proposed *re-thinking* framework, which is a detailed analysis shown in Figure 3. (1) Incorrect knowledge filtering: we prompt the LLM to reason the relevance between the query and retrieved knowledge to obtain the coarse-grained clues, which aims to filter incorrect or unrelated knowledge pieces. (2) Knowledge re-thinking: we equip the input query again. Let the query and the retrieved knowledge to have richer, back-and-forth interactions, obtaining in-depth and fine-grained correlation information, which gains fine-grained clues that are beneficial to selecting knowledge. (3) Missing knowledge reinforcing and response prediction: based on correlation information, we prompt LLM to generate fine-grained queries to retrieve missing knowledge and finally to predicted responses. In the following subsection, we provide a detailed explanation.

3.2 Incorrect Knowledge Filtering

In this stage, the main goal is to select the retrieved knowledge and filter incorrect knowledge at coarse-grained level. We first utilize the retrieval APIs to recall the query-related knowledge. The retrieved knowledge is treated equally in the previous researches. However, as exemplified in Figure 3, the retrieved knowledge contains rich noise, which affects the reasoning performance. So the retrieved knowledge should be selectively distinguished. To achieve the selection mechanism, various additional models are designed to infer the correlation between the query and retrieved knowledge. How-

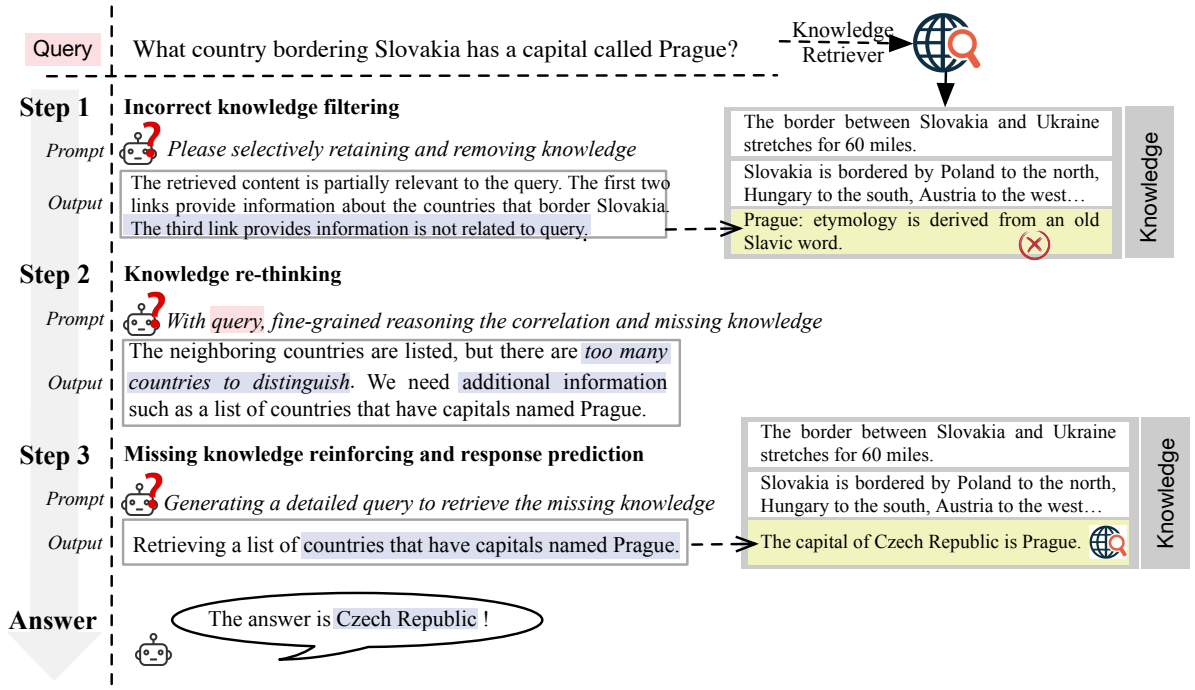

Figure 3: The *re-thinking* framework, accompanied by a case presentation. With the given query, *re-thinking* mainly consists of three parts: 1) Incorrect knowledge filtering; 2) Knowledge re-thinking; and 3) Missing knowledge reinforcing and response prediction.

ever, the design of the models relies heavily on prior features, and adding additional models will increase the burden on LLM. To overcome this limitation, we prompt LLM to automatically infer the relevance due to its powerful semantic analysis and understanding ability. Specifically, we enter the query x and knowledge K with the prompt, which contains instructions $\texttt{prompt}_1$ to determine the relevance between the query and retrieved knowledge. The above process is defined as follows:

$$O_{relevance} = \mathcal{M}(x, K, \texttt{prompt}_1). \quad (1)$$

LLMs will infer the correlation and output the coarse-grained inference result $O_{relevance}$.

3.3 Knowledge Re-thinking

The coarse-grained selection clues are perfunctory and superficial, based on the fact that the current decoder-only casual language modeling architecture typically works in a linear manner, which may miss the richer, back-and-forth interactions. With the coarse-grained correlation results, in this stage, we re-think the retrieved knowledge, which reflects the processed information and further infers the correlation between the query and retrieved knowledge at a fine-grained level to better identify the knowledge noise. Specifically, we input the query x again, then enter instructions $\texttt{prompt}_2$ to reason the correlation and missing knowledge, which is defined as:

$$O_{thinking} = \mathcal{M}(O_{relevance}, x, K, \texttt{prompt}_2). \quad (2)$$

LLM will output fine-grained correlation analysis, as well as the missing knowledge clues: $O_{thinking}$

3.4 Missing Knowledge Reinforcing and Response Prediction

Considering that the retrained knowledge may support partial query, at this stage, we further recall the missing knowledge and finally predict the responses. We perform fine-grained decomposition of the query, identify incomplete fragments, and generate a retrieved sub-query for the incomplete fragments. Specifically, we enter prompt instructions $\texttt{prompt}_3$ that aim at letting the LLMs give a most critical sub-query O_{query} that needs to be retrieved, which is defined as:

$$O_{query} = \mathcal{M}(O_{thinking}, \texttt{prompt}_3). \quad (3)$$

Based on the generated sub-query, we perform a search again and obtain the missing knowledge K^+. Finally, based on all the aforementioned inference results, LLMs output the response $O_{response}$ of the input query.

4 Experiments

In this section, we conduct a series of experiments. We first provide an overall description of the datasets, baselines and experimental details. Then we perform experiments on a series of reasoning benchmarks and backend LLMs to prove the effectiveness of *re-thinking*.

4.1 Datasets

We experiment on eight datasets and assign them into three categories: (1) **Multi-hop question answering**, including ComplexWebQuestions (CWQ) (Talmor and Berant, 2018), GraphQuestions (GraphQ) (Su et al., 2016), KQAPro (Cao et al., 2020) and GrailQA (Gu et al., 2021); (2) **Commonsense reasoning**, including StrategyQA (Geva et al., 2021) and ARC-c (Clark et al., 2018); (3) **Arithmetic reasoning**, including AQUA-RAT (Ling et al., 2017), MultiArith (Roy and Roth, 2015). Arithmetic reasoning relies on the understanding of mathematical formulas and measurement unit conversation, such as converting $1m/s$ to $3.6km/h$. Considering the limited memory knowledge of LLM parameters, arithmetic reasoning also relies on external mathematical knowledge.

4.2 Backend LLM Methods

We utilize `gpt-3.5-turbo-0301` and `gpt-3.5-turbo-0631` version of OpenAI GPT (Ouyang et al., 2022) as the backend LLM. Meanwhile, we also conduct experiments on Gemini-Pro (Team et al., 2023) version of Google Bard (Touvron et al., 2023b).

4.3 Compared Methods

From another perspective, we compare with existing popular reasoning frameworks to aspects demonstrate the effectiveness of *re-thinking*. The compared methods are listed as follows

(1) ITER-RETGEN (Shao et al., 2023a): A retrieval-augmented method that synergizes retrieval and generation in an iterative manner. Unlike this strategy, our proposed framework further focuses on the retrieved knowledge and reduces the slight knowledge noise. In ITER-RETGEN, we concatenate the generated responses and queries to conduct knowledge retrieval. For fair comparison, we only consider the two iterations, which is represented as ITER-RETGEN 2.

(2) Plan-and-Solve (PS) (Wang et al., 2023b): A plan-and-solve prompting strategy, which consists of two components. First, devising a plan to divide the entire task into smaller subtasks. Second, carrying out the subtasks according to the plan. We retrieve external knowledge before executing this prompting strategy.

(3) Self-Criticism (Kim et al., 2023): A simple reasoning architecture that prompts LLMs to find problems in their output and improves the output based on what they find. We retrieve external knowledge before executing this prompting strategy.

4.4 Experimental Details

We directly use the pre-trained dense retriever[2]. and use the browser web page information as the retrieval corpus for all datasets. Considering balancing model performance and computational cost, we retrieve the top-3 paragraphs for each query. For multi-hop question answering tasks, generated answers are evaluated with the standard exact match metric (EM score): a generated answer is considered correct if it matches any answer of the answer list after normalization. For other tasks, we utilize accuracy as an evaluation metric. For all evaluation experiments, we use the complete validation sets of the CWQ, AQUA-RAT and ARC-c datasets, the complete test sets of the MultiArith datasets, as well as the complete train sets and test sets of the GraphQ datasets. Due to reasoning time constraints, we refer to the previous related work (Shao et al., 2023b; Hao et al., 2023; Feng et al., 2023; Wang et al., 2023a) and select the first 1000 samples in the validation sets of KQAPro and GrailQA, the first 500 samples in the test set of StrategyQA dataset. In Section 4.7, for efficiency, we select the first 500 validation samples of CWQ dataset for evaluation. To avoid the impact of randomness introduced by the demonstrations in a few-shot setting, we assess our method in a zero-shot setting.

4.5 Main Results

We experiment with the proposed framework and baselines on eight datasets, which are assigned into three categories: multi-hop question answering, commonsense reasoning and arithmetic reasoning. The results are shown in Table 1 and Table 2. Table 1 presents the evaluation results for multi-hop question answering. From the experimental results, it can be seen that our *re-thinking* framework has achieved consistent performance improvement on different benchmark LLMs and datasets.

[2]We use the same information retrieval tools, which are detailed stated in the introduction section.

LLMs	Methods	CWQ	GraphQ	KQApro	GrailQA
gpt-3.5-turbo-0301	Vanilla	59.39	51.95	45.70	39.40
	Vanilla + RAG	58.14	48.45	46.40	41.20
	Vanilla + *re-thinking*	**62.83**	**56.11**	**52.20**	**46.20**
		↑3.44	↑4.16	↑5.80	↑5.00
Gemini-Pro	Vanilla	55.66	55.82	43.40	40.90
	Vanilla + RAG	50.92	46.57	48.00	41.90
	Vanilla + *re-thinking*	**70.73**	**60.76**	**53.50**	**49.80**
		↑15.07	↑4.94	↑5.50	↑7.90

Table 1: Evaluation results with EM score on multi-hop question answering datasets.

LLMs	Methods	AQuA	MultiArith	StrategyQA	ARC-c
gpt-3.5-turbo-0301	Vanilla	30.70	85.00	61.20	81.93
	Vanilla + RAG	37.79	83.88	57.00	82.27
	Vanilla + *re-thinking*	**54.72**	**89.44**	**66.80**	**84.28**
		↑16.93	↑4.44	↑5.60	↑2.01
Gemini-Pro	Vanilla	34.25	71.11	64.20	89.29
	Vanilla + RAG	36.61	65.55	58.60	78.59
	Vanilla + *re-thinking*	**37.79**	**90.00**	**67.80**	**94.64**
		↑1.18	↑18.89	↑3.60	↑5.35

Table 2: Evaluation results with accuracy score on arithmetic reasoning and commonsense reasoning datasets.

Specifically, GPT-based LLM, which equipped *re-thinking*, has improved by 3.44%, 4.16%, 5.80%, and 5.00% on the CWQ, GraphQ, KQApro and GrailQA datasets, respectively. Similarly, the performance of Gemini-Pro has been improved with the help of *re-thinking*, raising 15.07%, 4.94%, 5.50%, and 7.90% respectively.

In addition, we discover that some Vanilla baselines' performance drops after only utilizing RAG. For example, compared to GPT-based LLM, using RAG reduces 1.25% on the CWQ dataset. Through analysis, we find that the retrieved knowledge contains rich noise, which is shown in Figure 1. The above phenomenon leads to a direct decrease in RAG performance. Unlike RAG, *re-thinking* infers and thinks the retrieved knowledge again, in-depth identifying any contradictions or missing pieces of knowledge, later picking effective knowledge. Therefore, *re-thinking* significantly reduces the interference of knowledge noise.

Table 2 exhibits the evaluation results for arithmetic reasoning and commonsense reasoning. For LLMs in commonsense reasoning scenarios, retrieving some additional knowledge can enhance their cognition of commonsense queries, thereby improving their reasoning ability. For arithmetic reasoning, although this scenario measures the LLMs' mathematical calculation ability, the additional retrieval knowledge, such as "To solve for distance use the formula for distance $d = st$ or distance = speed x time", can inspire LLMs to solve mathematical queries. Therefore, from the experimental results, we find that our *re-thinking* framework can consistently improve the reasoning ability of arithmetic reasoning and commonsense reasoning scenarios. Specifically, GPT-based LLM, which equipped *re-thinking*, has improved by 16.93% and 4.44% on the AQUA-RAT and MultiArith datasets, respectively. In addition, we find that some methods only equipped with RAG had reduced reasoning ability. For example, GPT-based LLM, which utilizes RAG, reduces 1.12% and 4.2% on the MultiArith and StrategyQA datasets. This phenomenon indicates that if too much noise exists in the retrieved knowledge, it will actually hamper LLM and affect its reasoning ability.

4.6 Ablation Study

To measure the contribution of the main components in *re-thinking*, we conduct ablation experi-

Methods	CWQ	GraphQ	KQApro	GrailQA
Gemini-Pro + *re-thinking*	**70.73**	**60.76**	**53.50**	**49.80**
– w/o incorrect knowledge filtering	67.54	57.26	53.10	48.80
– w/o knowledge re-thinking	60.81	54.08	51.40	46.90
– w/o missing knowledge reinforcing	68.57	59.38	51.20	48.90
Gemini-Pro	55.66	55.82	43.40	40.90
ChatGPT + *re-thinking*	**62.83**	**56.11**	**52.20**	**46.20**
– w/o incorrect knowledge filtering	61.35	52.87	51.20	44.50
– w/o knowledge re-thinking	58.60	50.80	48.90	43.20
– w/o missing knowledge reinforcing	62.42	54.48	51.50	45.10
ChatGPT	59.39	51.95	45.70	39.40

Table 3: Ablation experimental results of our *re-thinking* mechanism.

LLMs	Methods	CWQ
`turbo-0301`	Self-Criticism (Kim et al., 2023)	64.2
	ITER-RETGEN 2 (Shao et al., 2023a)	65.0
	PS (Wang et al., 2023b)	69.8
	re-thinking	**71.9**
`turbo-0613`	Self-Criticism	62.6
	ITER-RETGEN 2	55.4
	PS	63.4
	re-thinking	**65.6**

Table 4: Evaluation results of comparing *re-thinking* with other well-known frameworks on different backend LLMs.

ments on different LLMs and datasets. Specifically, we remove **incorrect knowledge filtering**, **knowledge re-thinking**, and **missing knowledge reinforcing** modules from *re-thinking* framework respectively. The experimental results are presented in Table 3. From the results, we can draw some conclusions:

(1) After removing the incorrect knowledge filtering module, the performance of *re-thinking* is consistently decreased. For example, Gemini-Pro based *re-thinking* drops 3.19% on the CWQ dataset. This indicates that making a coarse-grained filter of the retrieved knowledge is beneficial, which exports coarse-grained clues for subsequent fine-grained reasoning between the query and knowledge to further discover slight noise of retrieved knowledge.

(2) Without knowledge re-thinking module, the performance of *re-thinking* is expectable decreased. which indicates that only coarse-grained knowledge filter and noise selection are not complete and the results may be perfunctory and superficial. With the knowledge *re-thinking* module, we allow the query and the retrieved knowledge to have richer, back-and-forth interactions, obtaining in-depth and fine-grained correlation information, which can detect the slight knowledge noise

(3) Knowledge re-thinking module may discover missing knowledge, and using the Missing Knowledge Reinforcing module can retrieve and recall missing knowledge, improving the LLM's reasoning ability.

4.7 Comparing with other Reasoning Frameworks

We compare the proposed model *re-thinking* with existing well-known reasoning frameworks based on two different backend LLMs: `gpt-3.5-turbo-0301` and `gpt-3.5-turbo-0631`. The evaluation results are shown in Table 4. From the results, we find that *re-thinking* is consistently higher than other compared frameworks. PS focuses on task planning for a given query, while Self-Criticism pays more attention to the criticism of predicted responses. Although these strategies can enhance reasoning ability, they ignore the concerns about retrieved knowledge. ITER-RETGEN concatenates the generated responses and queries to perform knowledge retrieval through multiple iterations. However, this strategy neglects fine-grained analysis of the retrieved knowledge at each iteration, which cannot effectively analyze the noise and missing information in the retrieved knowledge, resulting in unsatisfactory iteration prediction results.

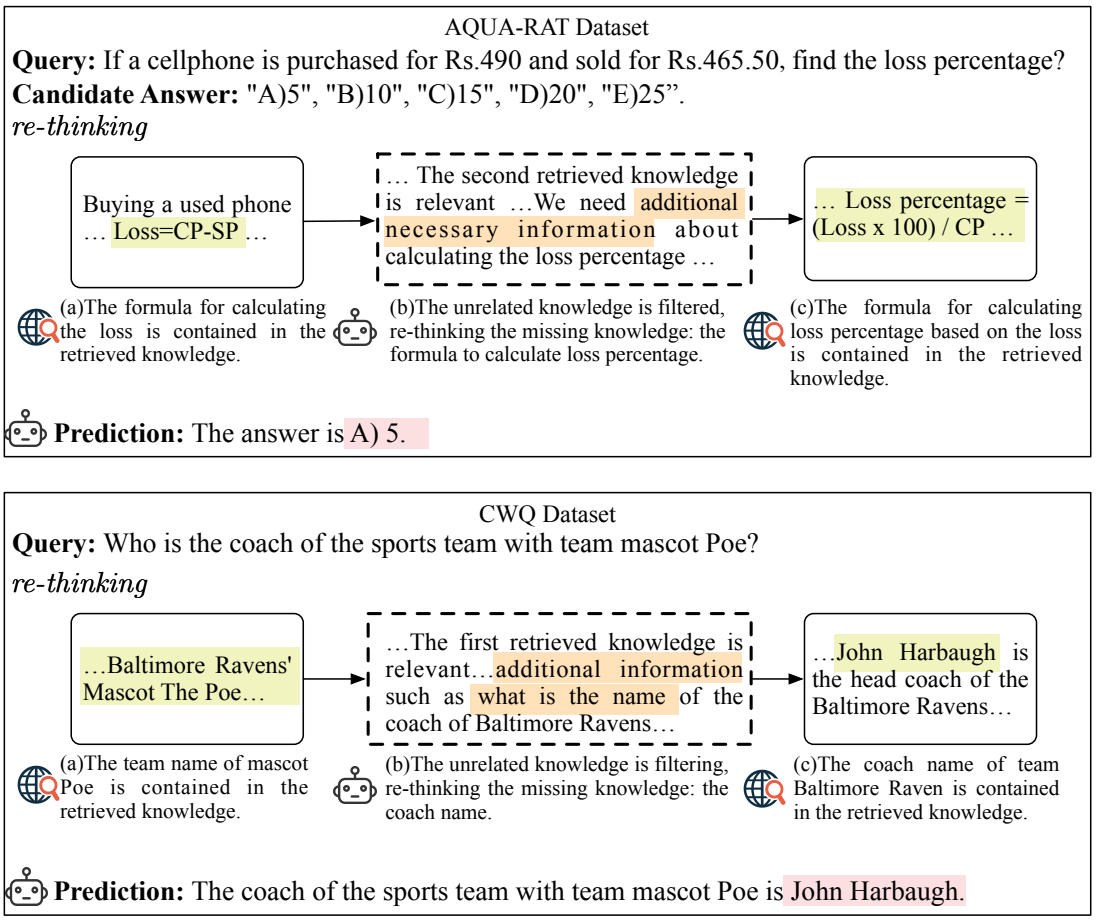

Figure 4: We display two examples of *re-thinking* on the AQUA-RAT and CWQ datasets. The (b) recognizes the knowledge noise recalled by the (a), and then infers the missing knowledge. The (c) retrieves the missing knowledge and ultimately predicts the response. Some unimportant information is not shown for brevity.

Unlike aforementioned frameworks, *re-thinking* allows questions and the retrieved knowledge to have richer, back-and-forth interactions, obtaining in-depth and fine-grained correlation or slight differences, further gathering the noise and missing knowledge.

4.8 Case Study

To present *re-thinking* more intuitively, we have presented two examples in Figure 4. The first example relies on reasoning for numerical calculations, while the second example relies on reasoning for multi-hop questions. For the first example, the retrieved knowledge with previous methods includes the calculation formula of Loss, but it is still insufficient to answer the question, as shown in (a). *Re-thinking* can measure the relevance between the query and retrieved knowledge, eliminates noise information, and further infers missing information: "*how to calculate the loss percentage with the loss*". With the retrieved formula information "*Loss percentage = (Loss x 100) / CP*", LLM can infer the correct answer. The above analysis also exists in the second example. With *re-thinking*, LLM can think deeply, find the first retrieved knowledge is relevant. LLM re-thinks the missing knowledge "*The coach name of Baltimore Ravens*", and then searches for richer and more important knowledge "*John Harbaugh is the head coach of the Ravens*", and ultimately obtain the correct answer.

5 Conclusion

In this paper, we focus on the RAG and propose a simple yet highly effective prompting strategy: *re-thinking*, which aims to denoise the retrieved knowledge and further enhance the RAG ability. Specifically, we develop a fine-grained interaction mechanism, which allows queries and the retrieved knowledge to have richer, back-and-forth interactions, obtaining in-depth and fine-grained correlations. Experiments prove the validity of *re-thinking*. We hope *re-thinking* mechanism can be a strong baseline for future research on RAG.

Ethical Consideration

We propose a simple yet highly effective prompting strategy: *re-thinking*, which aims to fine-grain analysis of the noise existing in the retrieved knowledge. All experiments utilized publicly accessible datasets that are widely employed in the field of reasoning research. We confirm that we contribute to society without any harm.

Limitations

re-thinking has been evaluated on three categories of datasets: multi-hop question answering, commonsense reasoning and arithmetic reasoning, with no experiments on more diverse types of datasets. At the same time, we only apply *re-thinking* on two backend LLMs. In the future, we will further apply *re-thinking* on more black-box models, such as GPT-4, or locally deployable models, such as Llama2, proving the universality and adaptability of *re-thinking*.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Shulin Cao, Jiaxin Shi, Liangming Pan, Lunyiu Nie, Yutong Xiang, Lei Hou, Juanzi Li, Bin He, and Hanwang Zhang. 2020. Kqa pro: A dataset with explicit compositional programs for complex question answering over knowledge base. *arXiv preprint arXiv:2007.03875*.

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. 2023. Universal self-consistency for large language model generation. *arXiv preprint arXiv:2311.17311*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*.

Zhangyin Feng, Xiaocheng Feng, Dezhi Zhao, Maojin Yang, and Bing Qin. 2023. Retrieval-generation synergy augmented large language models. *arXiv preprint arXiv:2310.05149*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *CoRR*, abs/2312.10997.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? A question answering benchmark with implicit reasoning strategies. *Trans. Assoc. Comput. Linguistics*, 9:346–361.

Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond iid: three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference 2021*, pages 3477–3488.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *CoRR*, abs/2311.05232.

Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983*.

Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault. 2020. Proceedings of the 58th annual meeting of the association for computational linguistics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 15696–15707. PMLR.

Geunwoo Kim, Pierre Baldi, and Stephen McAleer. 2023. Language models can solve computer tasks. *CoRR*, abs/2303.17491.

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

Angeliki Lazaridou, Elena Gribovskaya, Wojciech Stokowiec, and Nikolai Grigorev. 2022. Internet-augmented language models through few-shot prompting for open-domain question answering. *arXiv preprint arXiv:2203.05115*.

Katherine Lee, Orhan Firat, Ashish Agarwal, Clara Fannjiang, and David Sussillo. 2018. Hallucinations in neural machine translation.

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

Patrick S. H. 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. 2020b. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 158–167. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

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

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Haojie Pan, Zepeng Zhai, Hao Yuan, Yaojia Lv, Ruiji Fu, Ming Liu, Zhongyuan Wang, and Bing Qin. 2023. Kwaiagents: Generalized information-seeking agent system with large language models. *CoRR*, abs/2312.04889.

Subhro Roy and Dan Roth. 2015. Solving general arithmetic word problems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1743–1752. The Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2020. It's not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023a. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 9248–9274. Association for Computational Linguistics.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023b. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. *arXiv preprint arXiv:2305.15294*.

Yu Su, Huan Sun, Brian Sadler, Mudhakar Srivatsa, Izzeddin Gür, Zenghui Yan, and Xifeng Yan. 2016. On generating characteristic-rich question sets for qa evaluation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 562–572.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models, 2023. *URL https://arxiv.org/abs/2307.09288*.

Jinyuan Wang, Junlong Li, and Hai Zhao. 2023a. Self-prompted chain-of-thought on large language models for open-domain multi-hop reasoning. *arXiv preprint arXiv:2310.13552*.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023b. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 2609–2634. Association for Computational Linguistics.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. 2023a. Siren's song in the AI ocean: A survey on hallucination in large language models. *CoRR*, abs/2309.01219.

Zhuosheng Zhang, Yao Yao, Aston Zhang, Xiangru Tang, Xinbei Ma, Zhiwei He, Yiming Wang, Mark Gerstein, Rui Wang, Gongshen Liu, et al. 2023b. Igniting language intelligence: The hitchhiker's guide from chain-of-thought reasoning to language agents. *arXiv preprint arXiv:2311.11797*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

Xinyu Zhu, Junjie Wang, Lin Zhang, Yuxiang Zhang, Ruyi Gan, Jiaxing Zhang, and Yujiu Yang. 2022. Solving math word problem via cooperative reasoning induced language models. *arXiv preprint arXiv:2210.16257*.