KBQA or LLM-QA: A Unified Benchmark for Question Answering

Anonymous ACL submission

Abstract

Recently, Large language models (LLMs) and retrieval-augmented generation (RAG) have demonstrated remarkable performance in question answering (QA) tasks. However, whether RAG can replace traditional supervised methods based on Knowledge Base (KB) remains to be further explored. The main difficulty is that in existing multi-source knowledge retrieval datasets, information from KBs and text is not equivalent and cannot be directly compared. To bridge this gap, we propose our Trace-then-Synthesize framework, synthesizing necessary knowledge from KBs into corpus. With this method, we have constructed a dataset with equivalent information in both KB and corpus. Compared to existing datasets, our dataset compensates for the weaknesses of the RAG dataset, such as its small number of questions and black-box reasoning process, while having a broader applicability than traditional complex QA datasets. Through extensive experiments, we have demonstrated the strengths and limitations of various existing QA methods and showcased the powerful capabilities of this dataset in QA tasks.

1 Introduction

With the recent advancements of large language models (LLMs), people leverage the knowledge of LLMs to answer various questions (Brown et al., 2020). Undoubtedly, LLMs excel in understanding questions (Drozdov et al., 2023) and generating fluent natural language, but they perform poorly on some long-tail questions (Kandpal et al., 2023) and, due to the hallucination - the tendency that LLMs confidently give incorrect answers, are unable to accurately answer questions (Zhang et al., 2023). Recently developed technologies such as RAG (Lewis et al., 2021; Gao et al., 2024; Edge et al., 2024) have given us hope again.

However, we cannot directly compare these methods that rely on different data sources because most of the previous influential datasets **contains only a single source**, either a knowledge base (KB) or a corpus. Some researchers try to build datasets from scratch that include both KB and the corpus, but knowledge from these two sources is not equivalent. For example, the questions of 2WikiMultiHopQA (Ho et al., 2020) are generated through linking Wikidata and Wikipedia. It cannot be guaranteed that questions can be fully answered through text alone, so this dataset can only be used to evaluate multi-source retrieval methods.

In this study, we create a large and unified dataset by our "trace-then-synthesis" method. Our dataset can be used for both training and evaluating models regardless of whether the data source is text (Asai et al., 2023; Wang et al., 2024b), knowledge base (Gu et al., 2023; Nie et al., 2022; Sun

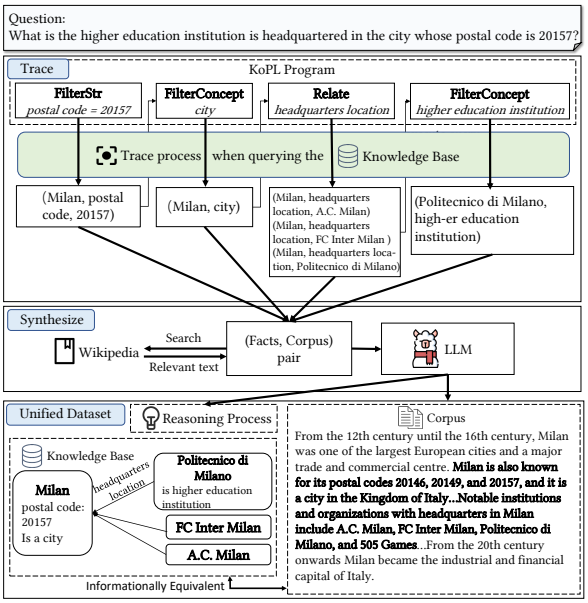

Figure 1: The pipeline of data construction for multiple QA methods. By tracing the execution process of queries, we extract all involved triples and use an LLM to synthesize them into the corresponding Wikipedia page, creating a corpus.

Dataset	Size	Knowledge Base	Corpus	Reasoning Path
WebQSP (Yih et al., 2016)	~5k	✓		
GrailQA (Gu et al., 2021)	~64k	✓		
MetaQA (Zhang et al., 2017)	~368k	✓		
HotpotQA (Yang et al., 2018)	~97k		✓	✓
Natural Questions (Kwiatkowski et al., 2019)	~315k		✓	
Musique (Trivedi et al., 2022)	~25k		✓	✓
2WikiMultihopQA (Ho et al., 2020)	~167k	✓	◯	✓
Ours	~10k	✓	✓	✓

Table 1: Existing question answering datasets. In 2WikiMultihopQA (Ho et al., 2020), although the text corresponding to the fact is provided, it cannot be guaranteed that questions can be fully answered through text alone.

et al., 2024), or both (Wang et al., 2024a), and it features a large number of questions, diversity, and interpretability.

Our main idea is to obtain all the facts required to answer questions from the KB and then supplement them into the existing corpus to make the information from the two data sources as equivalent as possible. This method involves three steps: 1) Use a modified query executor to retrieve facts and derive the reasoning process corresponding to the questions; 2) Collect all facts and convert them into natural language; 3) Use LLMs to naturally integrate the natural language descriptions of facts into the corresponding positions in the corpus. Additionally, there is an optional step, which is a stylized questions paraphrasing method.

Using this method, we have constructed a new unified dataset on the KQA Pro dataset, which includes an associated KB and corpus for evaluating various QA methods, paraphrased questions, and relevant reasoning processes for each question. These questions can be answered solely by the knowledge base or corpus. In addition to the information that must be used to answer the questions, there is some other irrelevant information in each data source, simulating "noise" in real-world.

We conduct extensive experiments to evaluate some of the current SOTA methods and models like Self-RAG (Asai et al., 2023) and REAR (Wang et al., 2024b) on this dataset. Llama3.1-8B model achieved an accuracy of 28.5% on our dataset which indicates that our data presents a certain level of difficulty. We also meticulously design a RAG pipeline that shows competitive performance to supervised semantic parsing methods and analyze the challenges that various methods may encounter, demonstrating the broad applicability of our dataset.

The contributions of this work are mainly summarized as follows:

- We propose a "trace-then-synthesize" data synthesis method. Based on this method, we constructed the first unified dataset that simultaneously includes knowledge base and corpus which builds a bridge between structured and unstructured data.
- We utilize this dataset to evaluate various QA models. We find that the current accuracy of RAG are already comparable to KB-based models, and even surpass supervised KB-based model's performance in certain scenarios. We also reveal the strengths and limitations of each approach in complex question answering tasks.

2 Related Work

Knowledge base question answering. KBQA methods can be classified into Information Retrieval-based (IR-based) and Semantic Parsing-based Methods (SP-based) (Lan et al., 2021). The main principle of IR-based methods (Wang et al., 2020; Sun et al., 2019, 2018; Yu et al., 2023) is to analyze the features and intentions in the question using natural language processing techniques, then retrieve matching triples in the knowledge graph to form a sub-graph related to the question. Then use this sub-graph to generate natural language answers. Some new work (Sun et al., 2024) also leverages the capabilities of large models to progressively explore and reason on knowledge graph. On the other hand, SP-based KBQA methods (Sun et al., 2020; Lan and Jiang, 2020; Bhutani et al., 2020; Kapanipathi et al.,

2021) focus on converting questions into logical expressions (such as SPARQL queries) that can operate on the knowledge base. Methods like UniKGQA(Jiang et al., 2023) are typical representatives based on semantic parsing. It proposes a unified model for multi-hop question answering tasks, consisting of a semantic matching module based on PLM for question-relation semantic matching and a matching information propagation module based on directed edges.

Retrieval-augmented models. After RAG was proposed, various works were presented in different directions, aiming to enhance the system's QA capabilities. Self-RAG (Asai et al., 2023) enhances the quality and factual accuracy of language models through retrieval and self-reflection, improving the performance of large language models across multiple tasks. FILCO (Wang et al., 2023) improves the quality of context provided to the generator in the generation model by identifying useful context and training a context filtering model, thereby addressing the generation output issues caused by the imperfection of retrieval systems. REAR (Wang et al., 2024b) significantly improves the efficiency of external knowledge utilization by accurately assessing the relevance of retrieved documents. ChatQA (Liu et al., 2024) proposes a two-stage instruction fine-tuning method that significantly improves the zero-shot conversational QA results of large language models, with ChatQA-70B's average score surpassing GPT-4.

Datasets for RAG. Question answering task has a long research history, accumulating a large number of excellent datasets, such as QALD-9-plus (Perevalov et al., 2022), MetaQA (Zhang et al., 2017), GrailQA (Gu et al., 2021). These datasets have played an important role in evaluating KBQA models. However, these datasets have some limitations when it comes to evaluating RAG systems. They lack assessments of the factual accuracy and refusal-to-answer capabilities of RAG systems. Therefore, some new datasets specifically designed for RAG have been proposed. CRUD-RAG (Lyu et al., 2024) categorizes the scope of RAG applications into four different types - Create, Read, Update, and Delete (CRUD), carefully evaluating all components of RAG systems. CRAG simulates APIs to mimic web and knowledge graph (KG) searches, evaluating the performance of RAG systems in multi-data source scenarios, fully representing the diversity and dynamics of real-world question-answering (QA) tasks. RGB (Chen et al., 2023) pays special attention to the performance of RAG in fundamental capabilities such as noise robustness, negative rejection, information integration, and counterfactual robustness. Fever (Thorne et al., 2018) is also commonly used in RAG tests to fact-check the text sources of RAG.

2.1 Preliminaries

Knowledge Base Knowledge Base (KB) is a collection of interlinked descriptions of entities. It can be defined as $G = (E, R, T)$ where E, R, and T represent the sets of entities, relations, and triples formed by them, respectively. A triple t contains a subject entity e, a predicate relationship r, and an object entity e', so it can be represented as $t = (e, r, e')$.

KoPL KoPL (Cao et al., 2022) is a programming language designed to represent the reasoning processes of complex problems. It has a tree structure where each node is a function like "Find" or "FilterConcept," and the parameters of each function are fixed values or the return values of child nodes. When a function is executed, it will filter data from the database that meets the definition. Running each function from the bottom up on the given knowledge base (KB) yields the answer.

2.2 Data Construction

Our primary objective is to collect relevant triples from knowledge base and seamlessly integrate them into a textual corpus. This integration is facilitated by the use of Large Language Models (LLMs), while ensuring the generation of high-quality data. Figure 1 illustrates the overall construction process.

Take KQA Pro as foundation When selecting the basic dataset, we compared various existing datasets. Table 1 compares some question-answering datasets. In terms of implementation difficulty, compared to SPARQL, KoPL used by KQA Pro (Cao et al., 2022) is simpler. In terms of problem difficulty, the evaluation by Tan (Tan et al., 2023) found that the difficulty of KQA Pro is quite high for LLMs, and KBQA methods have much higher accuracy than ChatGPT, indicating less problem leakage and greater difficulty, which can effectively benchmark these two categories

of methods. Taking all the above factors into consideration, we chose KQA Pro (Cao et al., 2022) as the base dataset. This does not mean the method cannot be extended to other datasets; it can be done by modifying the executor or using (Nie et al., 2022) to convert SPARQL to KoPL and then applying our method.

Trace Process In trace process, we need to find all relevant triples for each problem. This is achieved through a modification of the query executor. Specifically, we have the executor of the KoPL language record the function name, arguments, and the data retrieved during the KB query process while it is running.

After recording these information, it becomes feasible to translate these details into a natural language format through the use of predefined templates. For instance, we have a function called `QueryAttrQualifier` to query the qualifier value of the fact (Entity, Key, Value) in the knowledge graph. We manually wrote a template "QKEY of ENTITY is RES whose KEY is VALUE" for the function `res=QueryAttrQualifier(entities, key, value, qkey)`. That means when we record a query `QueryAttrQualifier("Bury My Heart at Wounded Knee", "publication date", "2007-05-20", "place of publication")` and its return value is `["United States of America"]`, we can convert it to `"The place of publication of Bury My Heart at Wounded Knee is the United States of America, whose publication date is 2007-05-20."`

Owing to the finite nature of the functions delineated by KoPL, it is possible to craft a specific template for each distinct function, thereby facilitating a seamless transformation process.

Synthesize Corpus Some existing work on linearizing KBs primarily converts all triples in the KB into text form (Yu et al., 2023), resulting in text that is not natural enough to reflect the actual situation. We utilize the text from Wikipedia's English page dump dated March 1, 2022, as a foundational base. We referred to the approach in Wiki-40B (Guo et al., 2020) to trim the text, removing the "References" and "External Links" parts of the text. For each fact, we use a mapping table to find the corresponding Wikipedia pages for its head and tail entities, and associate the fact with the pages. Then for each Wikipedia page, we prompt the LLM once, instructing it to integrate the set of facts associated with that page into the content, and ensure the coherence of the context.

Reasoning Path Because we have detailed records of the intermediate information when executing query statements, we can directly construct the reasoning path without relying on crowdsourced annotations. For each question, we first execute its corresponding KoPL query. The KoPL language executor parses the query statement into a syntax tree and executes the functions on each node in a pre-order traversal. After execution, for each tree node, we place the facts traced when running the corresponding function onto the node, thus obtaining a tree-shaped reasoning path.

Paraphrase Questions. A notable limitation of the original questions within the KQA Pro dataset is their template-driven creation, which often results in a uniform format that may not accurately reflect natural language use. To address this, we employ ChatGLM4-9B (GLM et al., 2024) to paraphrase the questions and add an "extended" part in the dataset, aiming to diversify their structure while preserving original intent.

For each question, we generate six new variants, utilizing various instruction prompts to guide the process. Table 9 shows our paraphrasing strategy. This not only enriches the dataset with a wider range of question formulations but also ensures that the essence of the questions remains intact. The specific instructions utilized for this data collection process are provided in the Appendix A.2. Additionally, we opt to exclude questions that involve more than 50 number of facts, thereby streamlining the dataset and enhancing its usability.

2.3 Dataset Quality Assessment

We evaluated the quality of the synthesized corpus through three dimensions: correctness, consistency and fluency. We then removed questions that did not meet the above conditions to ensure all of the questions contain sufficient information to answer.

Correctness. We need to ensure every fact $F = \{F_1, F_2, \ldots, F_k\}$ obtained during the trace step is correctly synthesized into the corpus. First, we employ LLM-based automated verification. By comparing the synthesized text $S^{gen} = \{S_1^{gen}, S_2^{gen}, \ldots, S_n^{gen}\}$ with the original text $S^{origin} = \{S_1^{origin}, S_2^{origin}, \ldots, S_m^{origin}\}$,

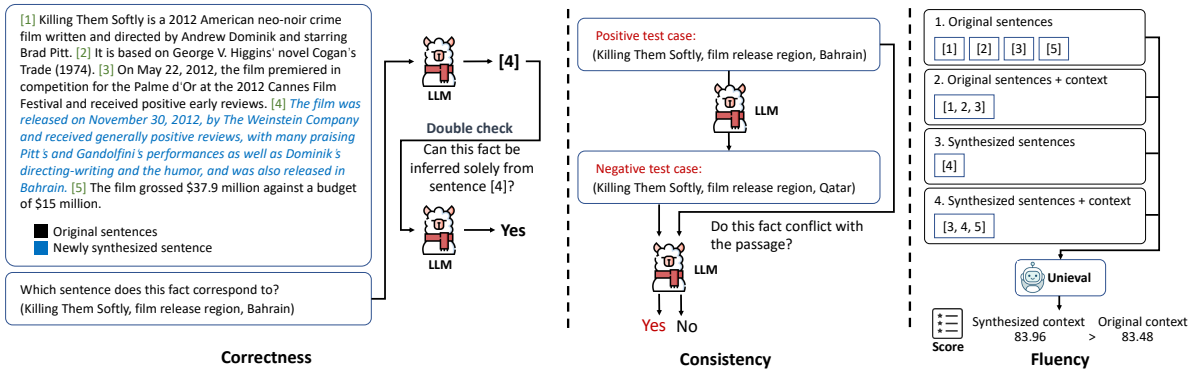

Figure 2: The pipeline of dataset quality assessment. We divide quality assessment into three dimensions: correctness, consistency, and fluency, all of which are evaluated using LLM automated assessment, with some results sampled for inspection.

we identify newly added text $S^{diff} = \{S_1^{diff}, S_2^{diff}, \ldots, S_{n-m}^{diff}\}$. For each fact F_i, we prompt the LLM to locate the corresponding text $S_{a_i}^{gen}$ in the new sentences $S^{gen} = \{S_1^{gen}, S_2^{gen}, \ldots, S_n^{gen}\}$, then re-prompt the LLM to verify whether the original fact F_i can be logically inferred from $S_{gen}^{a_i}$.

The results show that over 96% of facts are correctly synthesized. Additionally, we manually inspected 200 randomly sampled LLM-annotated text-fact pairs, confirming a correct annotation probability exceeding 98%.

Consistency. We are concerned that the facts in the KB may conflict with the information in the original corpus, so we need to check each fact. We use a combination of positive and negative test cases. For each fact F_i, we use various prompts to have LLM construct information F_i^* that conflicts with F_i. The specific prompts are detailed in Appendix A.4. Then, we prompt LLM to determine whether F_i and F_i^* conflict with the information in the synthesized text segment, respectively. We expect the first result to be "no" and the second to be "yes". The results show that 5241 out of 5764 (90.9%) the information has no conflicts.

Fluency. Compared to crowd workers, the advanced LLM can produce more fluent text. To test the contextual coherence of synthesized text, we used Unieval (Zhong et al., 2022)—a fine-tuned T5 model—to evaluate text fluency. We constructed the following four types of data: synthesized sentences, synthesized sentences + context, original sentences, and original sentences + context, sampling 1000 instances each to calculate their average fluency. For individual sentences, the fluency metric for synthesized text was , while for sentences from the original corpus it was . After concatenating with context, the fluency of the synthesized corpus actually improved.

3 Experiment

Datasets. We construct a new dataset. It is built upon the foundation of KQA Pro, which consists of 117,970 diverse questions that involve varied reasoning skills. Most of the original questions and corresponding answers are retained. We add the natural language reasoning process corresponding to the questions and the corpus corresponding to the knowledge base. We also developed an extended version, where each original question is paraphrased into six different descriptions that are essentially the same but phrased differently. The construction method is detailed in Section 2.2.

Both versions keep the original split method in KQA Pro. For each question, it also provides up to 10 candidate answers by executing an abridged SPARQL, which randomly drops one clause from the complete SPARQL. We use these answer choices in multiple-choice settings.

Metrics. We report the accuracy in all datasets. We follow KoLA (Yu et al., 2024) to handle issues in answer comparison, including 1) Case insensitivity 2) Comparing all decimals at specific precision 3) Date comparison, converting dates like January 1, 1947 to 1947-01-01 format for comparison.

3.1 Baselines

For the evaluation of our data construction methodology and the subsequent question-answering performance, we have established a comprehensive baseline encompassing three distinct categories

of models. These categories are selected to represent the current state-of-the-art across different approaches to knowledge-based question answering (KBQA), reflecting the diversity and the wide range of applications.

• **KBQA** In the realm of KBQA models, we have chosen to benchmark against the latest state-of-the-art (SOTA) models, which have demonstrated exceptional performance on the KQA Pro dataset. We select RGCN and the current SOTA model BART+KOPL and GraphIR(Nie et al., 2022) on KQA Pro as representatives models. The KoPL programs generated by the semantic parsing model will be executed on the KG provided by KQA Pro.

• **RAG** We conduct our experiments on both naive and specially designed RAG methods. The Vanilla RAG framework is detailed in Section 3.1. The LLM we used in this framework includes ChatGLM4 (GLM et al., 2024), Llama 3.1 (Touvron et al., 2023), and Qwen 2 (Yang et al., 2024).

In addition to the naive approach, we evaluate models that have been specifically tailored to enhance the RAG framework. This includes REAR (Wang et al., 2024b), IR-CoT (Trivedi et al., 2023) and Self-RAG (Asai et al., 2023).

• **KG-LLM** KG-LLM combine knowledge graphs to address the limitations of LLMs in generating knowledge-based content. We selected FlexKBQA (Li et al., 2024) and symKGQA (Agarwal et al., 2024) as representatives of the KG-LLM section and excerpted data from their papers.

4 Results and Analysis

To further explore the model's performance under various conditions, we propose two special settings: multiple-choice questions and ground-truth triples.

• **Multiple choice**: We use the 10 alternative answers provided in KQA Pro as options and modify the prompt in the second step to output a single option without providing any other examples, i.e., zero-shot. In this setup, it reduces errors caused by subtle differences in output, which helps evaluate the model's true reasoning ability.

• **Ground-truth triples**: We provide the model with all triples retrieved during the tracing process. By providing all relevant triple information, this setup simulates a perfect retriever scenario, helping researchers understand the model's performance potential if retrieval is not a bottleneck. It aids in determining the model's performance ceiling under ideal conditions, providing researchers a target.

4.1 Overall results

Table 2 shows the experimental results on our datasets. Supervised semantic parsing method in KBQA class achieves extremely high accuracy on this dataset, with the BART model achieving over 90% accuracy. As a SOTA model, Graph IR achieved high scores of 97.2% and 94.2% respectively in the Comparison and Verify tasks, demonstrating its strong capabilities in handling complex queries and verifying information. It outperforms the best RAG model by 25%. In different categories of problems, there is no doubt that the two have the largest gap in count-type questions and the smallest gap in comparison-type questions. This may be because the retriever is prone to missing information when retrieving a large amount of data, and LLM also struggles with counting accurately (Arkoudas, 2023). The performance of LLM was not ideal, but the improvement after using RAG was about 30%, indicating that the main reason for the large model's incorrect answers on the dataset was insufficient information.

As the generator needs to infer and summarize the final answer based on relevant information, the reasoning capabilities of LLMs are crucial, directly affecting the final performance. Compared to the 8b model, the accuracy of the 70b model has increased by 10%, with significant improvements in count and multi-hop type problems, which are 25.2% and 15% respectively.

Finally, some specially designed algorithms seem to perform poorly, even below Vanilla RAG. We have observed similar situations in other works (Jin et al., 2024). This may be caused by two factors. First, fine-tuned models or specially designed methods may lose some ability to adhere to output formats, resulting in some outputs not meeting the requirements specified in the prompts. Second, the documents we retrieved were less confusing, and the order of the documents provided was sorted, resulting in these models not showing significant improvement. Nevertheless, we found that the CoT-based cross-retrieval method IR-CoT significantly outperforms the single-retrieval Self-RAG and REAR methods in multihop problems by a improvement of 7.7%.

Since we have the correct text corresponding to each question, we calculated the recall accuracy of the retriever. We are using the BGE model for

Category	Model	Overall	Multi-hop	Qualifier	Compari-son	Logical	Count	Verify
KBQA	RGCN	35.1	34.0	27.6	30.0	35.9	41.9	65.9
KBQA	BART+KoPL	90.6	89.5	84.8	95.5	89.3	86.7	93.3
KBQA	GraphQ IR	**91.7**	**90.4**	**84.9**	**97.2**	**92.6**	**89.4**	**94.2**
LLM	Glm4-9B	23.2	22.7	18.3	37.8	25.2	22.1	53.7
LLM	Mistral-7B	29.5	28.0	22.7	62.2	30.5	21.4	54.3
LLM	Llama3.1-8B	28.5	28.2	22.5	64.5	32.1	16.2	46.7
Vanilla RAG	Llama3-8b	56.4	53.9	57.2	89.1	53.2	30.6	70.4
Vanilla RAG	Llama3-70b	66.8	68.9	65.1	94.0	66.2	55.8	66.1
Vanilla RAG	Qwen2-72b	56.1	55.2	59.6	70.8	53.3	41.2	80.5
RAG	Self-RAG	39.7	37.3	32.7	79.3	41.7	29.6	64.3
RAG	REAR	38.9	37.3	37.5	64.4	37.2	18.6	50.7
RAG	IR-CoT	46.9	45.0	55.2	67.4	42.6	10.9	54.9
KG-LLM	FlexKBQA	46.8	-	-	-	-	-	
KG-LLM	symKGQA	51.1	44.3	32.5	49.2	37.3	37.0	54.3

Table 2: The performance scores of KBQA models, LLMs, RAG models, and Knowledge Graph-enhanced Large Language Models (KG-LLM) across various task types

indexing; the proportion of the correct text appearing within the top 64 passages is 65%, with an average ranking of 6.7. Adding more irrelevant passages does not significantly affect the LLM's performance. More chunks will slightly improve the overall system's accuracy, but less than 1%.

4.2 Results on special settings

We designed several other scenarios to explore the performance limits of RAG, as shown in the table 3. The models in the table are divided into two main categories: KBQA models and RAG models based on Llama3.1, with different configurations for the RAG models, such as multiple-choice, providing real triples, and combinations of both.

For small size models (less than 8B), a good retriever is crucial. Compared to the retriever we built, a perfect retriever could increase accuracy by 20%. If the candidate answers are given in a close-set format, accuracy could also be increased by 20%. When both are combined, accuracy can be increased by 35%, reaching a level comparable to supervised KBQA models.

The most impressive finding is that we discovered that with the 70b model, the accuracy under perfect settings can reach 95%, which significantly exceeds the level of the SOTA KBQA models, indicating the great potential of the RAG system.

4.3 Results on extended part

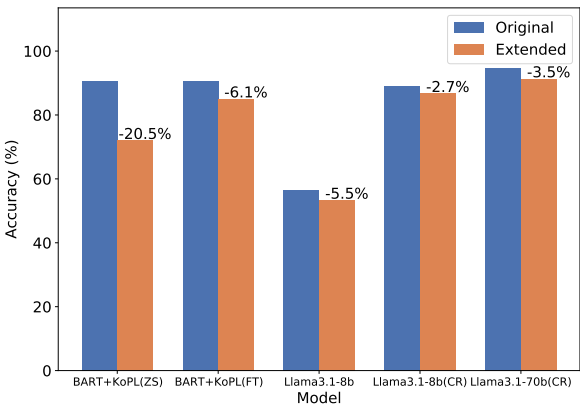

Figure 3: Results on extended part. The model's performance decline on unseen questions was compared. **ZS** represents the BART model trained on the original questions, and **FT** represents the BART model further trained on paraphrased questions. **CR** stands for RAG with ground-truth triples under the multiple-choice setting.

In this section, we conduct experiments using the extended portion of the constructed dataset, as described in section 4.1, where each question is paraphrased into six semantically equivalent questions to explore the generalization capabilities of different models. We use the BART model trained on the original questions (ZS) and the BART model further trained on paraphrased questions (FT) as the Baseline for the KBQA model.

Category	Model	Overall	Multi-hop	Qualifier	Compari-son	Logical	Count	Verify
KBQA	GraphQ IR	91.7	90.4	84.9	97.2	**92.6**	**89.4**	**94.2**
LLM	Llama3.1-8B	28.5	28.2	22.5	64.5	32.1	16.2	46.7
RAG	Llama3.1-8b	54.4	53.1	60.3	71.8	51.6	28.7	74.9
RAG	Llama3.1-70b	66.8	68.9	65.1	94.0	66.2	55.8	66.1
RAG under the multiple-choice setting								
RAG	Llama3.1-8b	75.9	71.2	81.8	74.1	71.8	41.4	76.1
RAG	Llama3.1-70b	80.7	75.7	85.0	85.4	78.5	43.1	82.4
RAG with ground-truth triples								
RAG	Llama3.1-8b	76.3	78.5	82.8	97.3	70.3	48.8	75.8
RAG	Llama3.1-70b	83.5	86.6	88.7	98.3	79.8	65.4	91.5
RAG with ground-truth triples under the multiple-choice setting								
RAG	Llama3.1-8b	89.1	88.5	92.7	87.5	82.8	58.9	88.0
RAG	Llama3.1-70b	**94.6**	**94.0**	**95.8**	**98.0**	92.1	73.7	93.8

Table 3: Overall Model Performance On Special Settings.

In Figure 3, we found that KBQA model trained on the original dataset experienced an 18% performance drop on the extended portion, and after fine-tuning, it dropped by 5% compared to the original setting. While the impact of diverse problem formulations on RAG is less significant, with an average decrease of 3%, which indicates that LLM-based model has better generalization performance.

4.4 Error analysis on counting problems

LLM-based methods exhibit relatively low accuracy in count tasks, which drags down the overall accuracy. In this section, we discuss the reasons for their failure. Thanks to our dataset having corresponding KB and corpus, by prompting LLM to output all eligible entities and collecting ground-truth entities from KB queries, we can calculate the overlap of entities between these two sets and categorize them into three types based on the situation.

Wrong format error(1.7% of the total). The model sometimes does not output in the specified format, instead outputting that it cannot find relevant information.

Intersected(26.5% of the total). Indicates that the model output and the ground-truth entities have an intersection. This situation occurs due to insufficient retrieval of information, leading to missed answers, or the model not carefully checking each entity to determine if it meets the question requirements, resulting in the output of additional entities.

Disjoint(39.1% of the total). Indicates that the model output does not overlap with the ground-truth entities. This situation arises because the model fails to understand the question or does not correctly decompose the question into sub-problems, resulting in completely irrelevant information being output.

The above errors mainly stem from insufficiently accurate recall information and the model's failure to verify whether each entity meets the requirements. The key to improving count capabilities lies in enhancing the large model's ability to retrieve information that meets the requirements.

5 Conclusion

This study primarily aims to build a unified dataset to support various methods whether single-source or multi-source. To construct this dataset, we propose a "trace-then-synthesis" framework to supplement the corpus with knowledge base and generate natural language reasoning processes while expanding the original question formats to enhance diversity. We have built a unified dataset using this framework. In our benchmark, we compare the performance of various methods in the same dataset, analyzing their respective strengths and weaknesses. Our findings reveal the significant potential of RAG in QA tasks, outperforming supervised KBQA algorithms in certain environments and many other interesting things. We hope this dataset can support different tasks and contribute to the further development of RAG and KG-LLM.

6 Limitations

Our dataset is primarily based on the KoPL program, but due to the expressive capabilities of the KoPL program, the types of questions included in this dataset are still limited. And although we have synthesized some new problems, we have not generated fundamentally different new problems. Because the number of problems in the current KB is already sufficient, we simply hope to integrate all the content. In future work, we hope to see more complex and user-friendly questions that are closer to how users describe them.

Another limitation is in benchmark section, although RAG can achieve performance comparable to supervised models, the resource requirements for running the LLM model are several times that of the supervised model, as RAG needs to store the corpus and the space required for the vector database. On the other hand, since we built the corpus from the KB, in practical scenarios, constructing the KB from corpus may result in further information loss (Nayak and Timmapathini, 2023), leading to a further decline in KBQA's performance.

In future work, there is a focus on optimizing the retriever. By improving the performance of the retriever, the performance of the LLM-based system can be effectively enhanced.

References

Prerna Agarwal, Nishant Kumar, and Srikanta Bedathur. 2024. SymKGQA: Few-shot knowledge graph question answering via symbolic program generation and execution. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10119–10140, Bangkok, Thailand. Association for Computational Linguistics.

Konstantine Arkoudas. 2023. Gpt-4 can't reason. *Preprint*, arXiv:2308.03762.

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

Nikita Bhutani, Xinyi Zheng, Kun Qian, Yunyao Li, and H. Jagadish. 2020. Answering complex questions by combining information from curated and extracted knowledge bases. In *Proceedings of the First Workshop on Natural Language Interfaces*, pages 1–10, Online. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Shulin Cao, Jiaxin Shi, Liangming Pan, Lunyiu Nie, Yutong Xiang, Lei Hou, Juanzi Li, Bin He, and Hanwang Zhang. 2022. KQA pro: A dataset with explicit compositional programs for complex question answering over knowledge base. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6101–6119, Dublin, Ireland. Association for Computational Linguistics.

Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. 2023. Benchmarking large language models in retrieval-augmented generation. *Preprint*, arXiv:2309.01431.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library.

Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. 2023. Compositional semantic parsing with large language models. In *ICLR*.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *Preprint*, arXiv:2404.16130.

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. *Preprint*, arXiv:2312.10997.

Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Juanzi Li, Lei Zhao, Lindong Wu, Lucen Zhong, Mingdao Liu, Minlie Huang, Peng Zhang, Qinkai Zheng, Rui Lu, Shuaiqi Duan, Shudan Zhang, Shulin Cao, Shuxun Yang, Weng Lam Tam, Wenyi Zhao, Xiao Liu, Xiao Xia, Xiaohan Zhang, Xiaotao Gu, Xin Lv, Xinghan Liu, Xinyi Liu, Xinyue Yang, Xixuan Song, Xunkai Zhang, Yifan An, Yifan Xu, Yilin Niu, Yuantao Yang, Yueyan Li, Yushi Bai, Yuxiao Dong, Zehan Qi, Zhaoyu Wang, Zhen Yang, Zhengxiao Du, Zhenyu Hou, and Zihan

Wang. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *Preprint*, arXiv:2406.12793.

Yu Gu, Xiang Deng, and Yu Su. 2023. Don't generate, discriminate: A proposal for grounding language models to real-world environments. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4928–4949, Toronto, Canada. Association for Computational Linguistics.

Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond i.i.d.: Three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference 2021*, WWW '21. ACM.

Mandy Guo, Zihang Dai, Denny Vrandečić, and Rami Al-Rfou. 2020. Wiki-40B: Multilingual language model dataset. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2440–2452, Marseille, France. European Language Resources Association.

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

Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Unikgqa: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph. *Preprint*, arXiv:2212.00959.

Jiajie Jin, Yutao Zhu, Xinyu Yang, Chenghao Zhang, and Zhicheng Dou. 2024. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. *Preprint*, arXiv:2405.13576.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge. *Preprint*, arXiv:2211.08411.

Pavan Kapanipathi, Ibrahim Abdelaziz, Srinivas Ravishankar, Salim Roukos, Alexander Gray, Ramon Astudillo, Maria Chang, Cristina Cornelio, Saswati Dana, Achille Fokoue, Dinesh Garg, Alfio Gliozzo, Sairam Gurajada, Hima Karanam, Naweed Khan, Dinesh Khandelwal, Young-Suk Lee, Yunyao Li, Francois Luus, Ndivhuwo Makondo, Nandana Mihindukulasooriya, Tahira Naseem, Sumit Neelam, Lucian Popa, Revanth Reddy, Ryan Riegel, Gaetano Rossiello, Udit Sharma, G P Shrivatsa Bhargav, and Mo Yu. 2021. Leveraging abstract meaning representation for knowledge base question answering. *Preprint*, arXiv:2012.01707.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*.

Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. A survey on complex knowledge base question answering: Methods, challenges and solutions. *Preprint*, arXiv:2105.11644.

Yunshi Lan and Jing Jiang. 2020. Query graph generation for answering multi-hop complex questions from knowledge bases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 969–974, Online. Association for Computational Linguistics.

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. *Preprint*, arXiv:2005.11401.

Zhenyu Li, Sunqi Fan, Yu Gu, Xiuxing Li, Zhichao Duan, Bowen Dong, Ning Liu, and Jianyong Wang. 2024. Flexkbqa: A flexible llm-powered framework for few-shot knowledge base question answering. *Preprint*, arXiv:2308.12060.

Zihan Liu, Wei Ping, Rajarshi Roy, Peng Xu, Chankyu Lee, Mohammad Shoeybi, and Bryan Catanzaro. 2024. Chatqa: Surpassing gpt-4 on conversational qa and rag. *Preprint*, arXiv:2401.10225.

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

Anmol Nayak and Hari Prasad Timmapathini. 2023. Llm2kb: Constructing knowledge bases using instruction tuned context aware large language models. *Preprint*, arXiv:2308.13207.

Lunyiu Nie, Shulin Cao, Jiaxin Shi, Jiuding Sun, Qi Tian, Lei Hou, Juanzi Li, and Jidong Zhai. 2022. Graphq ir: Unifying the semantic parsing of graph query languages with one intermediate representation. *Preprint*, arXiv:2205.12078.

A. Perevalov, D. Diefenbach, R. Usbeck, and A. Both. 2022. Qald-9-plus: A multilingual dataset for question answering over dbpedia and wikidata translated by native speakers. In *2022 IEEE 16th International Conference on Semantic Computing (ICSC)*, pages 229–234, Los Alamitos, CA, USA. IEEE Computer Society.

Haitian Sun, Tania Bedrax-Weiss, and William Cohen. 2019. PullNet: Open domain question answering with iterative retrieval on knowledge bases and text.

In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2380–2390, Hong Kong, China. Association for Computational Linguistics.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4231–4242, Brussels, Belgium. Association for Computational Linguistics.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M. Ni, Heung-Yeung Shum, and Jian Guo. 2024. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. *Preprint*, arXiv:2307.07697.

Yawei Sun, Lingling Zhang, Gong Cheng, and Yuzhong Qu. 2020. Sparqa: Skeleton-based semantic parsing for complex questions over knowledge bases. *Preprint*, arXiv:2003.13956.

Yiming Tan, Dehai Min, Yu Li, Wenbo Li, Nan Hu, Yongrui Chen, and Guilin Qi. 2023. Can chatgpt replace traditional kbqa models? an in-depth analysis of the question answering performance of the gpt llm family. *Preprint*, arXiv:2303.07992.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *Preprint*, arXiv:1803.05355.

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

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Preprint*, arXiv:2108.00573.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *Preprint*, arXiv:2212.10509.

Hongru Wang, Wenyu Huang, Yang Deng, Rui Wang, Zezhong Wang, Yufei Wang, Fei Mi, Jeff Z. Pan, and Kam-Fai Wong. 2024a. Unims-rag: A unified multi-source retrieval-augmented generation for personalized dialogue systems. *Preprint*, arXiv:2401.13256.

Xu Wang, Shuai Zhao, Jiale Han, Bo Cheng, Hao Yang, Jianchang Ao, and Zhenzi Li. 2020. Modelling long-distance node relations for KBQA with global dynamic graph. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2572–2582, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yuhao Wang, Ruiyang Ren, Junyi Li, Wayne Xin Zhao, Jing Liu, and Ji-Rong Wen. 2024b. Rear: A relevance-aware retrieval-augmented framework for open-domain question answering. *Preprint*, arXiv:2402.17497.

Zhiruo Wang, Jun Araki, Zhengbao Jiang, Md Rizwan Parvez, and Graham Neubig. 2023. Learning to filter context for retrieval-augmented generation. *Preprint*, arXiv:2311.08377.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding. *Preprint*, arXiv:2309.07597.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *Preprint*, arXiv:1809.09600.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206, Berlin, Germany. Association for Computational Linguistics.

Donghan Yu, Sheng Zhang, Patrick Ng, Henghui Zhu, Alexander Hanbo Li, Jun Wang, Yiqun Hu, William Wang, Zhiguo Wang, and Bing Xiang. 2023. Decaf: Joint decoding of answers and logical forms for question answering over knowledge bases. *Preprint*, arXiv:2210.00063.

Jifan Yu, Xiaozhi Wang, Shangqing Tu, Shulin Cao, Daniel Zhang-Li, Xin Lv, Hao Peng, Zijun Yao, Xiaohan Zhang, Hanming Li, Chunyang Li, Zheyuan Zhang, Yushi Bai, Yantao Liu, Amy Xin, Nianyi Lin, Kaifeng Yun, Linlu Gong, Jianhui Chen, Zhili Wu, Yunjia Qi, Weikai Li, Yong Guan, Kaisheng Zeng, Ji Qi, Hailong Jin, Jinxin Liu, Yu Gu, Yuan Yao, Ning Ding, Lei Hou, Zhiyuan Liu, Bin Xu, Jie Tang, and Juanzi Li. 2024. Kola: Carefully benchmarking world knowledge of large language models. *Preprint*, arXiv:2306.09296.

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. 2023. Siren's song in the ai ocean: A survey on hallucination in large language models. *Preprint*, arXiv:2309.01219.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J. Smola, and Le Song. 2017. Variational reasoning for question answering with knowledge graph. *Preprint*, arXiv:1709.04071.

Ming Zhong, Yang Liu, Da Yin, Yuning Mao, Yizhu Jiao, Pengfei Liu, Chenguang Zhu, Heng Ji, and Jiawei Han. 2022. Towards a unified multi-dimensional evaluator for text generation. *Preprint*, arXiv:2210.07197.

A Appendix

A.1 Implementation details

RAG components. We implement a Vanilla RAG framework. We first use a splitter to split Wikipedia into chunks. We let the splitter cut each page into chunks of 256 words, with an overlap of 128 words between chunks, and we try to keep entire sentences within a single chunk. We are using bge-en-1.5 (Xiao et al., 2023) model to create text embedding vectors, which has achieved the best performance on the text embedding benchmark. The encoder will invoke the model to convert chunks into 768-dimensional vectors. All vectors are indexed using FAISS (Douze et al., 2024) and are used for retrieval. By default, the L2-norm is used as the similarity metric to compare embeddings. When retrieving, we recall the top 100 vectors each time. Then we use BM25 to filter the top-72 passages and sort them using bge-reranker, then take the top 64 vectors. We search the library for each question and record the retrieved passages offline to ensure that the passages obtained by each RAG model in the same set of experiments are the same.

Hyperparameters. The parameters in the experiment are set as follows: temperature=0.01 to ensure stable output. The vector embedding dimension is 768, the TopK retrieved documents is set to 100. The default chunk size is set to 256 with overlaps of 128.

A.2 Prompt for paraphrasing questions

During the process of paraphrasing questions, we will prompt LLM multiple times, and the prompts for this part are shown in Table 4. Table 9 demonstrates the 6 paraphrasing prompts we have proposed. The right column shows example sentences under different methods

A.3 Prompt for synthesizing

During the process of synthesizing the corpus, we concatenate the text segments and the facts that need to be synthesized into the segments and send them to LLM. The detailed prompts are shown in the Table 5.

A.4 Prompt for dataset quality assessment

In the first two steps of quality assessment, we used LLM, with the prompt in Table 6.

A.5 Prompt for RAG

In the RAG pipeline, the first step is to let LLMs freely infer step by step based on the context and the second step is to let LLMs summarize the answers according to the format. The prompt we use are in Table 7.

A.6 Prompt for special settings

For the scenario of multiple-choice questions, we use the prompt in Table 8 in the second step.

Paraphrase Please rewrite this question: {question} **Synonym substitution for verbs** Please rewrite the following question by replacing the verb with a synonym, ensuring that the information contained in the original sentence remains unchanged: {question}
Alter sentence structure Please rewrite the following question by changing the sentence structure, ensuring that the information contained in the original sentence remains unchanged: {question}
Expand the sentence Please rewrite the following question using extensions, ensuring that the information contained in the original sentence remains unchanged: {question}
Change to passive voice Please rewrite the following question in a passive manner, ensuring that the information contained in the original sentence remains unchanged: {question}
More colloquial Please make the following question more colloquial and ensure that the information contained in the original sentence remains unchanged: {question}

Table 4: Prompt for paraphrasing question.

I will give you a paragraph and some sentences, please help me add the information from these sentences to the text. Require information to remain unchanged, language to be fluent, and conform to the original logic. paragraph:{paragraph} sentences:{triples}

Table 5: Prompt for synthesizing

Correctness {sentence} Please select a sentence from the following that can induce the above information and output its identifier. These sentences are talking about {title}. The number at the beginning of each sentence within [] is the identifier; do not output numbers outside this range. If there is none, output None. Please do not output any other explanation. choices:{list of facts}
Consistency Please help me change the given information to be incorrect. You can modify numbers, change to negative forms, or alter objects, etc. Only output the modified information without any other explanation. information:{fact}
I will give you a piece of information and paragraphs. Please help me determine if there are any facts in the paragraph that conflict with the information. Output "conflict" or "not conflict" without any other explanation. information:{fact} paragraph:{paragraph}

Table 6: Prompt for dataset quality assessment

Step 1

context: {context}
<end of context>
Now you are an assistant to answer questions. Please infer the answer based on your knowledge and the information provided to you in English.
Please try to answer using the original words in the information I gave you, especially when asking about the relationship between the two.
Q: {question}
Let's think step by step.

Step 2

Context:
{Output_from_1st_step}
Please answer the following question.
If the question is a true or false question, please output 'yes' for true or 'no' for false directly.
If it is a question about counting quantities and there is no relevant information, output 0.
If it is a comparison question, please output the answer directly.
Please try to answer using the original words in the information I gave you, especially when asking about the relationship between the two
Otherwise output the answer in the format of the following example:
Q: When was Neville A. Stanton's employer founded?
A: The employer of Neville A. Stanton is University of Southampton. The University of Southampton was founded in 1862. So the answer is: 1862.
Q: What weekly publication in the Connecticut city with the most Zagat rated restaurants is issued by university of Americaǎ212Lite: How Imperial Academia Dismantled Our Culture's author?
A: The author of America Lite: How Imperial Academia Dismantled Our Culture is David Gelernter. David Gelernter was educated at the Yale University. The city in Connecticut that has the highest number of Zagat rated restaurants is New Haven. The weekly publication in New Haven that is issued by Yale University is Yale Herald. So the answer is: Yale Herald.
Q: What is the headquarters for the organization who sets the standards for ISO 21500?
A: The standards for ISO 21500 were set by International Organization for Standardization. The International Organization for Standardization has headquarters in Geneva. So the answer is: Geneva.
Q: How many children does LeBron James have?
A: LeBron James have kids LeBrony James Jr., Bryce Maximus James, Zhuri James. So LeBron James have 3 kids. So the answer is: 3.
Q: Does LeBron James and Yao Ming come from the same country?
A: LeBron James is from the U.S.. Yao Ming is from China. So the answer is: no.
Q: {question}
A:

Table 7: The prompt design for RAG.

This is some known information: {context} The following are single choice questions (with answers). You only need to output one character representing your option, without any additional output. Question: {question} {choices} Answer:

Table 8: The prompt design for multiple-choice setting.

Strategy	Example
Original question	Which cost more, World Trade Center or An Ideal Husband?
Paraphrase	What has a higher cost: the World Trade Center or "An Ideal Husband"?,
Synonym substitution for verbs	Which was more expensive, the World Trade Center or An Ideal Husband?
Alter sentence structure	Between the World Trade Center and An Ideal Husband, which incurred greater expenses?
Expand the sentence	What is the higher price between the World Trade Center and the production of "An Ideal Husband"?
Change to passive voice	The cost of which is greater, the World Trade Center or An Ideal Husband?
More colloquial	So, which one is pricier, the World Trade Center or "An Ideal Husband"?

Table 9: The prompt design for paraphrasing.