TimeR^4: Time-aware Retrieval-Augmented Large Language Models for Temporal Knowledge Graph Question Answering

Xinying Qian[1] **Ying Zhang** [1*] **Yu Zhao**[1] **Baohang Zhou**[1]
Xuhui Sui[1] **Li Zhang**[1] **Kehui Song**[2]
[1] College of Computer Science, VCIP, TMCC, TBI Center, DISSec, Nankai University, China
[2] School of Software, Tiangong University, China
{qianxinying,zhaoyu,zhoubaohang,suixuhui,zhangli}@dbis.nankai.edu.cn,
yingzhang@nankai.edu.cn, songkehui@tiangong.edu.cn

Abstract

Temporal Knowledge Graph Question Answering (TKGQA) aims to answer temporal questions using knowledge in Temporal Knowledge Graphs (TKGs). Previous works employ pre-trained TKG embeddings or graph neural networks to incorporate the knowledge of TKGs. However, these methods fail to fully understand the complex semantic information of time constraints. In contrast, Large Language Models (LLMs) have shown exceptional performance in knowledge graph reasoning, unifying both semantic understanding and structural reasoning. To further enhance LLMs' temporal reasoning ability, this paper aims to integrate temporal knowledge from TKGs into LLMs through a Time-aware Retrieve-Rewrite-Retrieve-Rerank framework, which we named **TimeR^4**. Specifically, to reduce temporal hallucination in LLMs, we propose a *retrieve-rewrite* module to rewrite questions using background knowledge stored in the TKGs, thereby acquiring explicit time constraints. Then, we implement a *retrieve-rerank* module aimed at retrieving semantically and temporally relevant facts from the TKGs and reranking according to the temporal constraints. To achieve this, we fine-tune a retriever using the contrastive time-aware learning framework. Our approach achieves great improvements, with relative gains of 47.8% and 22.5% on two datasets, underscoring its effectiveness in boosting the temporal reasoning abilities of LLMs. Our code is available at https://github.com/qianxinying/TimeR4 .

1 Introduction

Knowledge graph question answering (KGQA) aims to provide answers based on knowledge from knowledge graphs (Sun et al., 2019). However, many real-world questions include temporal constraints, such as "Who is the president of the United States after Obama?" To address this, some knowledge graphs store time-aware facts as quadruples *(subject, predicate, object, timestamp)*, which are known as temporal knowledge graphs (TKGs). Temporal knowledge graph question answering (TKGQA) focuses on obtaining answers using the knowledge in TKGs (Saxena et al., 2021).

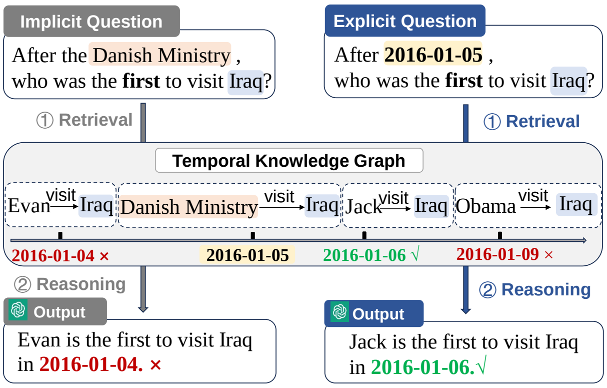


Figure 1: Examples of challenges in integrating temporal knowledge graphs with large language models.

Recent works (Saxena et al., 2021; Mavromatis et al., 2022) incorporate knowledge from TKGs by utilizing pre-trained TKG embeddings or graph neural networks (GNNs). However, these methods fail to fully understand the complex semantic information of time constraints in questions (Chen et al., 2023b). In contrast, Large language models (LLMs) have demonstrated exceptional performance in knowledge graph reasoning (Sun et al., 2024; Luo et al., 2024) and can unify semantic understanding and graph reasoning (Huang and Chang, 2023; Wei et al., 2023). To further enhance the temporal reasoning capabilities of LLMs, in this paper, we aim to integrate temporal knowledge into LLMs, thereby addressing complex and multi-granularity temporal questions. However, enhancing the temporal reasoning capabilities within LLMs remains several significant challenges:

(1) Hallucinated by implicit temporal ques-

[*] Corresponding author.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 6942–6952
November 12-16, 2024 ©2024 Association for Computational Linguistics

tions. In the TKGQA task, there are many implicit questions, such as *"After the Danish Ministry, who was the first to visit Iraq?"* Reasoning through such questions is very difficult because there are no explicit timestamps provided, requiring extra steps of inference. In Figure 1, LLMs tend to hallucinate when confronted with such questions, leading to incorrect reasoning. Conversely, when temporal events are replaced with specific timestamps, such as *"After 2016-01-05, who was the first to visit Iraq?"*, LLMs can more easily deduce the correct answer. Therefore, we believe that converting implicit questions into explicit ones is a crucial issue.

(2) Lack of Temporal Knowledge. To enhance the reasoning capability of LLMs, previous methods (Li et al., 2023) employ off-the-shelf retrieval tools such as BM25 to extract relevant facts from a knowledge graph as background knowledge. However, these retrieval methods focus solely on semantic matching, thus the retrieved knowledge neglects the time constraints of the question, rendering it ineffective for reasoning. For example, in Figure 1, quadruple *(Evan, visit, Iraq, 2016-01-04)* is irrelevant because the question requires retrieving facts after 2016-01-05. Therefore, constructing a retriever that concurrently pays attention to semantic similarity and temporal constraints is of great importance for the TKGQA task.

To address the above challenges, we propose TimeR^4, a Time-aware Retrieve-Rewrite-Retrieve-Rerank framework for the TKGQA task. Specifically, to mitigate the issue of LLMs hallucinating when faced with implicit temporal questions, we employ the *retrieve-rewrite* strategy. We perform fact retrieval from the Facts Knowledge Store (FKS) to obtain relevant facts for implicit questions. Subsequently, we rewrite these questions by replacing temporal facts with specific timestamps, ensuring that all questions contain explicit temporal information. For constructing FKS, we fix the language model to obtain semantical fact embeddings. To simultaneously capture semantic similarity and temporal constraints, we employ the *retrieve-rerank* strategy. This involves conducting a time-aware retrieval from the Temporal Knowledge Store (TKS) and reranking the facts based on temporal constraints to refine the retrieval process. The TKS is constructed by fine-tuning a language model with a contrastive time-aware retrieval strategy, which develops an encoder capable of capturing both semantic similarity and temporal constraints by constructing three types of negatives for each question. Finally, we fine-tune open-source LLMs with the retrieved facts from TKS, leveraging the temporal knowledge in TKGs to enhance the model's temporal reasoning capabilities. Experiments on two datasets demonstrate that our strategy significantly enhances the temporal reasoning abilities of LLMs. Overall, our work makes the following contributions:

- We integrate LLMs with TKGs and propose TimeR^4, a Time-aware Retrieve-Rewrite-Retrieve-Rerank framework, which effectively overcomes the limitations in handling temporal knowledge in LLMs.

- We propose a contrastive time-aware retrieval strategy that simultaneously pays attention to semantic similarity and temporal constraints.

- The experimental results demonstrate that our approach achieves relative improvements of 47.8% and 22.5% respectively on two TKGQA datasets.

2 Related Work

2.1 KG-enhanced LLMs

Considering the excellent reasoning ability of large language models (LLMs) on NLP tasks (Bang et al., 2023), many recent works have applied LLMs on KGQA tasks. Based on how these methods integrate with knowledge graphs, they can be categorized into three primary types: retrieval-based reasoning, path-based reasoning, and agent-based reasoning. Retrieval-based methods (Baek et al., 2023b; He et al., 2024) focus on retrieving relevant subgraphs or triples from the knowledge graph that contain the information needed to answer a question. The LLMs are then used to process and reason over these retrieved information. However, they only consider semantic similarity and neglect temporal constraints. Path-based methods (Luo et al., 2023; Cheng et al., 2024) involve exploring paths within the knowledge graph to establish connections between the question and the potential answers. These methods typically utilize LLMs to traverse the graph and generate possible paths. However, they cannot be directly applied to TKGQA because they do not account for temporal dimension in path reasoning. Agent-based methods(Sun et al., 2023; Jiang et al., 2023) treat LLMs as an agent to search and prune on the KGs to find answers. However, they prove inefficient for complex reasoning tasks due to their reliance on multiple LLM-calls. Furthermore, the greedy decision-making process is susceptible to error propagation.

2.2 TKGQA Methods

The TKGQA task is more challenging than the KGQA task due to the added temporal dimension of reasoning. To incorporate temporal information, some methods pose a question as a TKG completion problem and utilize TKG embedding score functions to select entities or timestamps with the highest relevance as answers (Saxena et al., 2021). TempoQR (Mavromatis et al., 2022) augments the question embeddings with context, entity, and time-aware information by three designed modules. MultiQA (Chen et al., 2023b) adopts Transformer encoding layers to aggregate multi-granularity time information. Some works integrate temporal information by introducing RGCN (Relational Graph Convolutional Networks). EXAQT (Jia et al., 2021) utilizes the RGCN layer and augments it with dictionary matching. TwiRGCN (Sharma et al., 2022) adopts temporally weighted graph convolution followed by answer gating. LGQA (Liu et al., 2023) applies a multi-hop message passing graph neural network layer to combine the global and local information. However, such two methods perform poorly in complex reasoning tasks, especially those involving multi-granularity temporal questions. ARI (Chen et al., 2023a) integrates LLMs through a knowledge adaptability framework and abstract methodological guidance. However, it cannot be applied to smaller-scale LLMs.

3 Preliminaries

Temporal knowledge graph $\mathcal{G} = \{\mathcal{E}, \mathcal{P}, \mathcal{T}, \mathcal{F}\}$ is a directed graph where vertices are a set of entities $\mathcal{E}$. The edges are a set of predicates $\in \mathcal{P}$ with timestamps $\mathcal{T}$. The quadruple set $\mathcal{F} = \{(s, p, o, t)\} \subseteq \mathcal{E} \times \mathcal{P} \times \mathcal{E} \times \mathcal{T}$ represents the temporal facts, where s and o are subject and object, p is predicate between s and o at timestamp t.

TKGQA is a task to infer the correct answer to natural language question $q \in Q$ based on relevant quadruples $f = (s, p, o, t)$ in the TKG, where the answer can be either an entity name or timestamp.

4 Method

4.1 Overview

Figure 2 presents our proposed model, TimeR^4, a Retrieve-Rewrite-Retrieve-Rerank framework. The

Algorithm 1: The training procedure of TimeR^4

```
Input  : TKG $\mathcal{G}$, Questions $\mathcal{Q}$, Ground
         Truth $gt$, Language Model $LM$,
         raw LLM $M$
Output : Fine-tuned LLM $M'$
negatives
  $\leftarrow$ GenerateNegatives($\mathcal{G}$, $\mathcal{Q}$, $gt$)
$LM_t \leftarrow$ Optimize $LM$ as Equation 8
$FKS \leftarrow$ ConstructKS($\mathcal{G}$, $LM$)
$TKS \leftarrow$ ConstructKS($\mathcal{G}$, $LM_t$)
for $\{q\} \in loader(\mathcal{Q})$ do
    $f \leftarrow$ Retrieve($FKS$, $q$, $LM$);
    $q^* \leftarrow$ ReWrite($q, f$);
    $f' \leftarrow$ Retrieve($TKS$, $q^*$, $LM_t$);
    $f^+ \leftarrow$ ReRank($f'$, $q$);
    $M' \leftarrow$ Optimize $M$ as Equation 11
end
Function Retrieve($KS$, $q$, $LM$):
    $\mathbf{E_q} \leftarrow$ LM($q$);
    $\phi_s \leftarrow$ cos($\mathbf{E_q}$, $KS$);
    $f^+ \leftarrow$ TopN($\phi_s$);
    return $f^+$;
Function ConstructKS($\mathcal{G}$, $LM$):
    $KS \leftarrow$ LM($\mathcal{G}$);
    return $KS$;
```

detailed training procedure is illustrated in Algorithm 1. To enhance the performance of LLMs in handling complex problems, we first propose a *retrieve-rewrite* strategy. This strategy aims to retrieve implicit temporal knowledge within the questions from the Facts Knowledge Store (FKS) and reformulate the questions using this background knowledge to include explicit time constraints. For FKS, we utilize the language model to obtain embeddings of facts within TKGs. Next, we implement a *retrieve-rerank* module to retrieve both semantically and temporally relevant facts. This involves conducting a time-aware retrieval from the Temporal Knowledge Store (TKS) and reranking the facts based on temporal constraints. For TKS, we fine-tune a language model using contrastive learning to develop an encoder capable of simultaneously capturing semantic similarity and temporal constraints. Finally, we fine-tune the open-source LLMs, incorporating the retrieved facts from TKGs for enhanced LLMs' temporal reasoning.

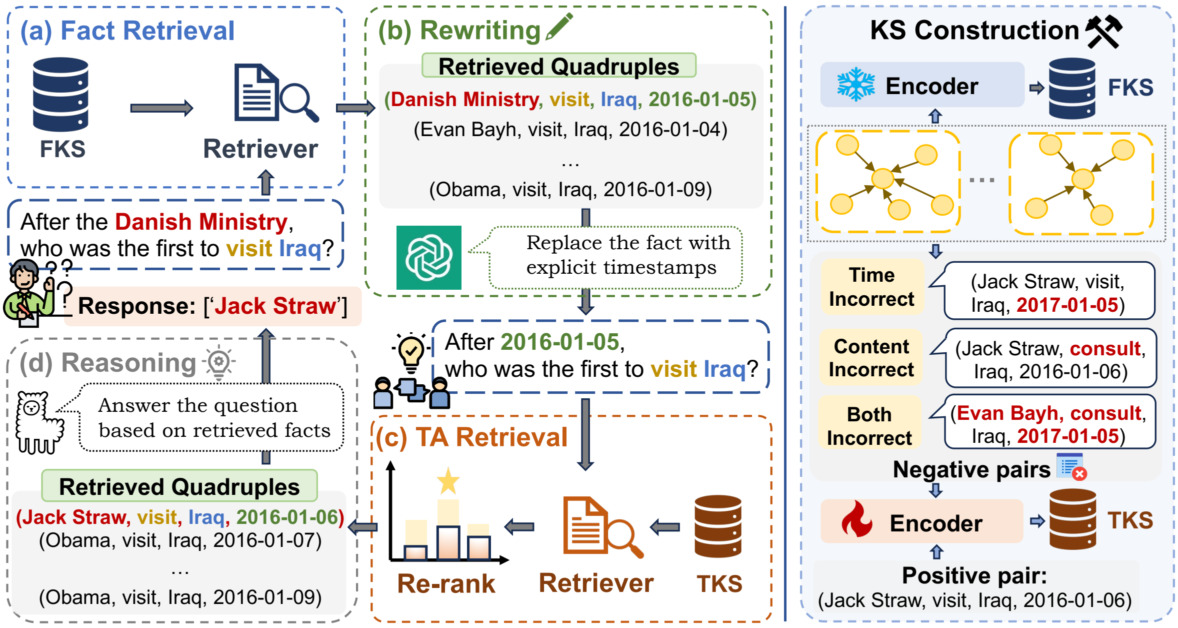

(a) Fact Retrieval
FKS
Retriever
After the Danish Ministry, who was the first to visit Iraq?
Response: ['Jack Straw']
(b) Rewriting
Retrieved Quadruples
(Danish Ministry, visit, Iraq, 2016-01-05)
(Evan Bayh, visit, Iraq, 2016-01-04)
...
(Obama, visit, Iraq, 2016-01-09)
Replace the fact with explicit timestamps
After 2016-01-05, who was the first to visit Iraq?
(c) TA Retrieval
Re-rank
Retriever
TKS
(d) Reasoning
Answer the question based on retrieved facts
Retrieved Quadruples
(Jack Straw, visit, Iraq, 2016-01-06)
(Obama, visit, Iraq, 2016-01-07)
...
(Obama, visit, Iraq, 2016-01-09)
KS Construction
Encoder
FKS
Time Incorrect
(Jack Straw, visit, Iraq, 2017-01-05)
Content Incorrect
(Jack Straw, consult, Iraq, 2016-01-06)
Both Incorrect
(Evan Bayh, consult, Iraq, 2017-01-05)
Negative pairs
Encoder
TKS
Positive pair:
(Jack Straw, visit, Iraq, 2016-01-06)

Figure 2: The architecture of TimeR[4] can be divided into four modules, fact retrieval, rewriting, time-aware retrieval, and reasoning. The right part shows how we construct the Knowledge Store (KS).

4.2 Fact Retrieval

Previous works (Chen et al., 2023b; Liu et al., 2023) employed entity-linking tools to identify entities and relations in question, subsequently utilizing these entities and relations for further retrieval. However, some TKGs, such as ICEWS (García-Durán et al., 2018a), lack entity-linking tools, resulting in poor performance with such methods (Chen et al., 2023b). Motivated by the recent study (Baek et al., 2023a), we adopt a direct fact retrieval strategy from TKGs without entity linking.

We first convert each quadruple into natural language sequences. Then we embed all quadruples $T(s,p,o,t) \in \mathcal{G}$ in TKGs onto a dense embedding space by using a pre-trained language model as in Equation 1 and memorize the knowledge representations in a Fact Knowledge Store (FKS). We also embed the given questions q as in Equation 2. d is the dimension of the output vector.

$$\text{FKS} = \{\mathbf{E}_f | \mathbf{E}_f = LM(S(s,p,o,t)), (s,p,o,t) \in \mathcal{G}\} \quad (1)$$

$$\mathbf{E}_q = LM(q) \in \mathbb{R}^d \quad (2)$$

To retrieve the k-nearest semantic quadruples $\mathbf{f}_1^+$ according to the representation distance for the given question, we calculate the similarity between $\mathbf{E}_q$ and $\mathbf{E}_f$. We use the FAISS library (Johnson et al., 2021) for indexing and similarity calculation.

$$\phi_{FKS}(\mathbf{E}_q, \mathbf{E}_t) = \cos(\mathbf{E}_q, \mathbf{E}_f) = \mathbf{E}_q \cdot \mathbf{E}_f \quad (3)$$

$$\mathbf{f} = \arg\max \phi_{FKS}(\mathbf{E}_q, \mathbf{E}_f) \quad (4)$$

4.3 Rewrite

Complex questions often contain implicit temporal information, posing a challenge to the TKGQA task. To address the hallucination issues of LLMs with implicit questions, we plan to rewrite the questions to ensure that all questions have explicit timestamps. Specifically, we retrieve the necessary background facts through the FKS and then input them along with the question into the LLM for inference and rewriting as in Equation 5. We apply the in-context learning strategy (Dong et al., 2023), which encodes structural knowledge into demonstrations to guide the LLM. The specific prompt is shown in Appendix A.

$$q^* = LLM(Prompt(q, f)), q \in Q \quad (5)$$

After being rewritten by the LLMs, questions containing implicit temporal facts are modified to include explicit timestamps based on the retrieved background facts. Additionally, for certain questions involving common knowledge not present in TKGs, such as *"Who was the first president of the US after World War II?"*, the LLMs can utilize their inherent knowledge to rewrite them. Knowing that World War II ended in 1945, the question can successfully be transformed into *"Who was the first president of the US after 1945?"*.

4.4 Time-aware Retrieval

We further propose the Time-aware Retrieval module to retrieve facts that satisfy both semantic similarity and time constraints simultaneously from the temporal knowledge store (TKS). To construct the TKS, we fine-tuned the language model to model the time-aware facts and store them in the knowledge base as in Equation 6.

$$\text{TKS} = \{\mathbf{E}_t | \mathbf{E}_t = LM_t(S(s,p,o,t)), (s,p,o,t) \in \mathcal{G}\} \tag{6}$$

In order to enhance the time-awareness of the language model, we employ the contrastive time-aware retrieval strategy. We randomly corrupt the time, relations, and entities of the positive pair separately and generate three types of negative pairs: time incorrect, content incorrect, and both incorrect, as shown in Figure 2. Contrastive loss is calculated based on the cosine similarity between the question representation $\mathbf{E}_q$ and the quadruples representation $\mathbf{E}_t \in$ TKS as Equation 7.

$$\phi_{TKS} = \texttt{cos}(\mathbf{E}_{q^*}, \mathbf{E}_t) \tag{7}$$

We aim to minimize the distance between positive pairs and maximize the distance between negative pairs. The contrastive label Y=1 signifies that the context corresponds to a positive pair of the question, whereas Y=0 indicates that the context represents the negative pair. Subsequently, following Son and Oh (2023), the contrastive loss is computed in Equation 8, with w_p and w_n as the weights for positive and negative samples.

$$\mathcal{L} = \sum_i [w_p Y \cdot \exp(\phi_{TKS}) + w_n(1-Y) \cdot \exp(1-\phi_{TKS})] \tag{8}$$

4.5 Rerank

To further refine the retrieval process, we design a time-filtering function to filter out irrelevant facts and focus more on time-related ones. After the rewriting module, each question q^* contains a specific timestamp t_q, except for the questions that require answering the timestamp. In that case, to introduce the influence of time intervals, we design a time-filtering function for questions containing time constraints. For each quadruple (s,p,o,t) in TKG $\mathcal{G}$, we calculate the time difference between t_q and t, filter out quadruples that fall outside the range, and normalize the time differences within the range as the results of time filtering function. Equation 9 represents the time-filtering function for "before" type questions.

$$\phi_t(t_q,t) = \begin{cases} 1 - \frac{|t_q - t|}{max(t_q - t)}, & \text{if } (t_q - t) > 0 \\ -100, & \text{otherwise} \end{cases} \tag{9}$$

Specifically, we add the score obtained from the time filtering function $\phi_t(t_q,t)$ to the score of the Time-aware Retirver $\phi_{TKS}(\mathbf{E}_q, \mathbf{E}_t)$, thereby reranking the scores of the retrieved facts. μ is the weight of two scores.

$$\phi(q,t) = \mu \cdot \phi_{TKS}(\mathbf{E}_{q^*}, \mathbf{E}_t) + (1-\mu) \cdot \phi_t(t_q,t) \tag{10}$$

4.6 Reasoning

After obtaining the retrieved quadruples we formulate the reasoning part as an LLM optimization problem, aiming to maximize the probability of inferring the answer a to question q from the knowledge graph $\mathcal{G}$ by using the retrieved quadruples f^+ as history facts as in Equation 11. To guide the LLM in generating final answers, we design a simple instruction prompt in Appendix A.

$$\mathcal{L} = \max_{\Phi} \sum_{(q^*,a)\in\hat{\mathcal{Q}}} \sum_{t=1}^{|a|} \log\left(P_\Phi\left(a_t \mid (q^*, f^+), a_{<t}\right)\right) \tag{11}$$

5 Experiments

5.1 Experimental Settings

Datasets. Considering that the CronQuestions (Saxena et al., 2021) dataset has been reported to contain spurious correlations that different models can exploit to achieve high accuracy (Sharma et al., 2022), we base our experiments on two recent more challenging datasets, i.e., MULTITQ (Chen et al., 2023b) and TimeQuestions (Jia et al., 2021). **MULTITQ** is the largest known TKGQA dataset constructed from the ICEWS05-15 (García-Durán et al., 2018b), which has 500K unique question-answer pairs. Besides, MULTITQ features multiple temporal granularities, including years, months, and days, with questions spanning over 3600 days. **TimeQuestions** is another challenging dataset, which has 16K temporal questions and is divided into four categories. However, TimeQuestions only includes a time granularity of years and is much smaller in size. The statistical information is presented in Table 1 and Table 2 separately.

Baseline. We select three types of baselines for comparison on MULTITQ: (1) Pre-trained LMs, including BERT (Devlin et al., 2019) and ALBERT (Lan et al., 2020). (2) Embedding-based methods, including EmbedKGQA (Saxena et al.,

Category		Train	Dev	Test
Single	Equal	135,890	18,983	17,311
	Before/After	75,340	11,655	11,073
	First/Last	72,252	11,097	10,480
Multiple	Equal Multi	16,893	3,213	3,207
	After First	43,305	6,499	6,266
	Before Last	43,107	6,532	6,247
Total		386,787	587,979	54,584

Table 1: Statistics of MULTITQ dataset.

Category	Train	Dev	Test
Explicit	2,724	1,302	1,311
Implicit	651	291	292
Temporal	2,657	1,073	1,067
Ordinal	938	570	567
Total	6,970	3,236	3,237

Table 2: Statistics of TimeQuestions dataset.

2020), CronKGQA (Saxena et al., 2021), MultiQA (Chen et al., 2023b). (3) LLM-based methods, including ARI (Chen et al., 2023a), LLaMA2 (Touvron et al., 2023), ChatGPT. For TimeQuestions, we also use three types of baselines: (1) KGQA method, including PullNet (Sun et al., 2019), Uniqorn (Pramanik et al., 2023), and GRAFT-Net (Sun et al., 2018). (2) TKGQA methods, including CronKGQA, TempoQR (Mavromatis et al., 2022), EXAQT (Jia et al., 2021), LGQA (Liu et al., 2023), and TwiRGCN (Sharma et al., 2022). (3) LLM-based methods, including LLaMA2 and ChatGPT. We only input the given questions into ChatGPT and LLaMA2 without any explanation.

Implementations Details. We use LLaMA2-Chat-7B (Touvron et al., 2023) as the LLM backbone. We fine-tune the LLaMA2 for 2 epochs on 2 NVIDIA A6000 GPUs. We only use 20% of the training data for MULTITQ datasets for training because it is very large. For Fact Retriever, We utilize off-the-shell SentenceBert (Reimers and Gurevych, 2019) as the base encoder. For Time-aware Retriever, we fine-tune the SentenceBert for 10 epochs. For re-writing, we use the OpenAI-API [1] (gpt-3.5-turbo-0125[2]). We set the μ as 0.4. We fine-tune each question using in-batch negatives and three hard negatives.

5.2 Main Results

We present the experimental results in comparison with other methods of TimeR^4 on the MultiTQ and TimeQuestions datasets in Table 3 and Table 4, where the highest results are highlighted in bold font and the second highest results are marked underlined. TimeR^4 achieves the best performance in all experimental settings, indicating its superiority on the TKGQA task.

For the MultiTQ dataset, TimeR^4 achieves state-of-the-art performance across all question types. Specifically, We find that PLMs (BERT. ALBERT) and LLMs (LLaMA2, ChatGPT) exhibit the lowest performance on the TKGQA task. This might be due to the lack of necessary temporal knowledge, thus leading to errors in reasoning. Compared to traditional methods, TimeR^4 shows a significant improvement in hits@1 performance, achieving a 59.8% enhancement. This highlights the capability of LLMs in reasoning on complex temporal questions, particularly those involving multi-granularity timestamps and complex reasoning. Furthermore, compared with the recent ChatGPT-based method ARI, TimeR^4 demonstrates a 47.8% improvement, showcasing the effectiveness of our proposed temporal reasoning framework.

As for TimeQuestions, TimeR^4 also achieves the best results across all question types. The results of KGQA methods are the poorest for they lack the ability to retrieve temporal facts or reason in temporal facts. Compared to traditional methods, TimeR^4 still achieves a 22.5% relative improvement on Hits@1. The results demonstrate the capability of TimeR^4 for precisely answering temporal questions. Compared with other LLMs, TimeR^4 also performs the best. The results of ChatGPT and LLaMA2 are much higher than on MULTITQ datasets. This might be because Timequestions are built on the Wikidata (Vrandečić and Krötzsch, 2014) knowledge graph, and most LLMs are pre-trained on the Wikidata knowledge graph corpus. Therefore, they store some relevant information, enabling them to answer such questions.

5.3 Ablation Study

The ablation results are shown in Table 5. The results for TimeQuestions do not decline significantly, primarily because the MULTITQ dataset is more challenging. MULTITQ is larger and covers a wider time range, making it harder to retrieve relevant facts. Conversely, in the TimeQuestions dataset, relevant facts can often be retrieved by the fact retrieval module, which is why our strategy is less prominent on this dataset.

Effect of fact retrieval module. We first replace fact retriever with entity-linking tools. Since no

[1] https://platform.openai.com/docs/api-reference

[2] https://platform.openai.com/docs/models/gpt-3-5-turbo

Model	Overall	Question Type		Answer Type	
		Single	Multiple	Entity	Time
BERT	8.3	9.2	6.1	10.1	4
ALBERT	10.8	11.6	8.6	13.9	3.2
EmbedKGQA	20.6	23.5	13.4	29	0.1
CronKGQA	27.9	13.4	13.4	32.8	15.6
MultiQA	29.3	34.7	15.9	34.9	15.7
ARI	38.0	68.0	21.0	39.0	34.0
LLaMA2	18.5	22.0	10.1	23.9	5.5
ChatGPT	10.2	14.7	7.7	13.7	2
TimeR^4	**72.8**	**88.7**	**33.5**	**63.9**	**94.5**

Table 3: Performance comparison of different models (in percentage) on MUULTITQ.

Model	Overall	Question Type		Answer Type	
		Explicit	Implicit	Temporal	Ordinal
PullNet	10.5	2.2	8.1	23.4	2.9
Uniqorn	33.1	31.8	31.6	39.2	20.2
GRAFT-Net	45.2	44.5	42.8	51.5	32.2
CronKGQA	46.2	46.6	44.5	51.1	36.9
TempoQR	41.6	46.5	3.6	40	34.9
EXAQT	57.2	56.8	51.2	64.2	42
TwiRGCN	60.5	60.2	58.6	64.1	51.8
LGQA	52.9	53.2	50.6	60.5	40.2
LLaMA2	27.1	26.8	32.5	27.9	23.4
ChatGPT	45.9	43.3	51.1	46.5	48.1
GenTKGQA	58.4	59.6	61.1	56.3	57.8
TimeR^4	**78.1**	**82.3**	**73.0**	**83.0**	**64.9**

Table 4: Performance comparison of different models (in percentage) on TimeQuestions.

linking tool is available for the ICEWS, we adopted the approach outlined in Chen et al. (2023b), employing a NER tool to identify entities. It can be observed that after replacing the fact retriever module, the overall performances decrease by 13.2% on hits@1, which indicates that the fact retriever module achieves accurate fact recognition.

Effect of rewriting strategy. To verify the role of the rewriting strategy, we then conduct an experiment where we removed the rewriting strategy and solely relied on the original questions and retrieved facts. The results showed a significant decrease, indicating that the rewriting strategy effectively aids LLMs in mitigating the hallucination of implicit temporal questions.

Effect of time-aware retrieval module. We conducted an experiment where we replaced the time-aware retriever with the fact retriever. The results show a significant decline, indicating that our proposed time-aware retriever method is inherently time-aware and performs better than the non-time-aware fact retriever.

Model	Hit@1	
	MULTITQ	TimeQuestions
TimeR^4	**72.78**	**78.1**
w/o fact retrieval	41.04	54.3
w/o rewrite	61.12	77.2
w/o temporal retrieval	70.34	77.3
w/o rerank	63.04	77.9

Table 5: Results of the ablation study. "w/o" means removing the module.

Effect of reranking strategy. Removing the rerank strategy resulted in a significant decrease in model performance, indicating that filtering out irrelevant time information is indeed crucial.

5.4 Further Discussion

To further analyze the superior performance of TimeR^4 compared to other models, we conduct a series of experiments to gain further insights.

5.4.1 Comparison with LLMs

We compare TimeR^4 with other LLMs in both datasets in Table 6. LLMs *w/ TimeR*^{4} indicates that we input the facts retrieved by our strategy and rewritten questions into the LLMs. LLMs *w/ fine-tuned* indicates that we fine-tuned the LLMs with only questions.

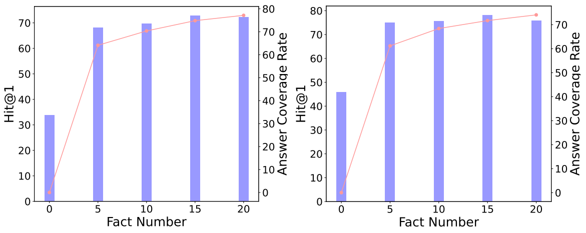
First, the results on two LLMs show that with the enhancement of the facts retrieved from TKGs and our retrieve-rewrite-retrieve-rerank strategy, LLMs *w/ TimeR*^{4} have significantly better performance. This suggests that the LLMs possess some degree of simple temporal reasoning capability. However, for more precise and effective reasoning in complex temporal questions, TimeR^4 effectively overcomes the limitations in handling and interpreting time-sensitive knowledge.

To further explore the effectiveness of our method, we also compared the results of LLama with fine-tuned and without fine-tuned. It can be found that the results of LLaMA2 with fine-tuning almost doubled, indicating that fine-tuning can effectively regulate the output of LLMs and help LLMs learn patterns of temporal reasoning, thereby enhancing the results.

It is worth noting that in multiple questions, ChatGPT performs better than TimeR^4. This is mainly because ChatGPT tends to provide all possible answers. For example, in Table 7, for questions with only one entity as the answer, ChatGPT often returns a list of relevant results, which can result in higher performance.

Model	MULTITQ					Timequestions				
	Overall	Question Type		Answer Type		Overall	Question Type		Answer Type	
		Single	Multiple	Entity	Time		Explicit	Implicit	Temporal	Ordinal
TimeR^4	**72.8**	**88.7**	33.5	**63.9**	**94.5**	**78.1**	**82.3**	**73.0**	**83.0**	**64.9**
LLaMA2	18.5	22.0	10.1	23.9	5.5	28.9	26.8	41.9	33.7	33.8
LLaMA2 *w/ finetuned*	33.9	38.4	22.7	45.0	7.8	45.8	44.4	46.0	51.9	37.8
LLaMA2 *w/ TimeR^4*	39.1	44.2	26.6	37.0	44.2	59.3	57.4	51.5	73.4	41.0
ChatGPT	10.2	14.7	7.7	13.7	2	45.9	43.3	51.1	46.5	42.1
ChatGPT *w/ TimeR^4*	41.4	58.5	**41.2**	56.1	57.1	64.8	66.0	52.9	77.6	45.5

Table 6: Effects of integrating the TimeR^4 framework with different LLMs for reasoning.



(a) Results on MULTITQ. (b) Results on TimeQuestions.

Figure 3: The Hits@1 results of different fact numbers.

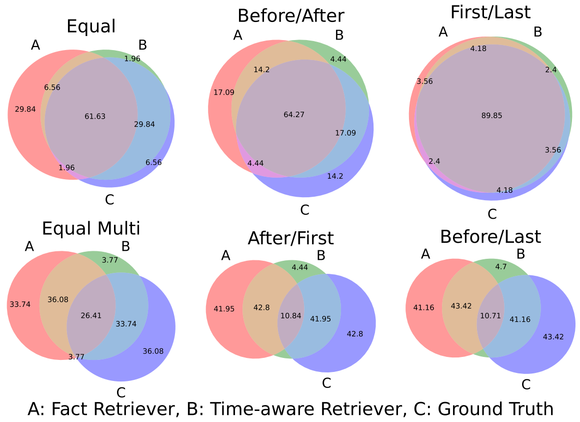


Figure 4: Venn diagrams for the answers coverage overlap of time-aware retrieval, fact retrieval, and ground truth for six question types in MULTITQ dataset.

5.4.2 Number of Retrieved Facts

To explore the impact of varying numbers of retrieved facts on the results, we present the performance changes and the corresponding answer coverage on two datasets by adjusting the number of retrieved facts n in Figure 3. It is evident that the model achieves its peak performance with 15 relevant facts, the same number included in our strategy. Interestingly, there is a slight performance dip at $n = 20$, despite the larger answer coverage obtained with 20 retrieved facts. This suggests that an excessive amount of facts may hinder performance by introducing noise, thereby making it more difficult for the LLMs to distinguish relevant information from irrelevant ones. On the other hand, fewer facts do not provide sufficient temporal knowledge, also leading to poor performance. Hence, $n = 15$ represents the optimal choice.

5.4.3 Effectiveness of the Retrieval

To verify the effectiveness of our time-aware retrieval module, we investigate the overlap of answer coverage between the results of the Time-aware Retriever, the Fact Retriever, and ground truth in different question types in Figure 4. Across various question types, our Time-Aware Retriever answers most questions correctly that the Fact Retriever does, while also handling a significantly larger set of complex questions (second row). This indicates that after fine-tuning, the Time-Aware Retriever can retrieve more temporal information, while the Fact Retriever struggles to capture temporal information in the facts, leading to low answer coverage in the retrieved results.

5.5 Case Study

We provide examples for each question type in the MULTITQ dataset of ChatGPT *w/ TimeR^4*, LLaMA2 *w/ TimeR^4*, and TimeR^4 in Table 7 with the same input.

It can be observed in the results that without fine-tuned, LLMs tend to randomly generate a list of irrelevant answers, but in comparison, ChatGPT performs better than LLaMA2. LLaMA2 is more prone to generating irrelevant responses without fine-tuning. For instance, when asked *"After Okada Katsuya, who wishes to visit Cambodia first?"*, LLaMA2 directly outputs the entity mentioned in the question: '*Okada Katsuya*'. TimeR^4, on the other hand, achieves significantly higher accuracy and only generates the relevant answers.

Furthermore, after fine-tuning, the output of TimeR^4 adheres more closely to standard conventions. For questions like *"in which month...,"* where the ground truth answer is *['2012-05']*, LLMs of-

Type	Question	Response		
		ChatGPT *w/ TimeR*^{4}	**LLaMA2 *w/ TimeR*^{4}**	**TimeR^4**
Equal	In 2005-01, who used light weapons to attack Thailand?	-Insurgent (Thailand) **-Citizen (Thailand)**	1. Insurgent (Thailand) 2. **Citizen (Thailand)** 3. Armed Opposition (Thailand)	[**’Citizen (Thailand)’**, **’Armed Gang (Thailand)’**]
Before/ After	Who investigated China after 22 July 2015?	1.Police (South Korea) 2. Mainland Affairs Council 3. Police (South Africa)	1. Japan 2. South Korea 3. France 4. China	**[’Xi Jinping’]**
First/ Last	In which month did Benny Gantz first visit China?	- **May** - August	**May**, August, or July	**[’2012-05’]**
Equal Multi	Who was the first to praise Iraq in 2015?	Iran	1. Iran 2. Iraq 3. el-Tayeb	**[’Foreign Affairs (France)’]**
After /First	After Okada Katsuya, who wish to visit Cambodia first?	- South Korea - Thailand - **Foreign Affairs (South Korea)**	1. Okada Katsuya 2. John Faulkner 3. Anupong Paochinda	**[’Foreign Affairs(South Korea)’]**
Before /Last	Who did Zimbabwe’s Foreign Minister praise last before Kuwait?	- South Sudan - **Iran**	Guy Scott Mark Simmonds Tony Blair Faith Pansy Tlakula	**[’Iran’]**

Table 7: Comparison of responses to six different question types between our TimeR^4 and ChatGPT *w/ TimeR*^{4}, LLaMA2 *w/ TimeR*^{4}. The correct answers are highlighted in **bold font**.

ten generate *"May"*, which, although correct, is considered incorrect during evaluation. Exploring methods to construct a more reasonable evaluation or to ensure LLMs’ outputs conform to standard conventions will be our future research. These results demonstrate the effectiveness of TimeR^4 in conducting temporal reasoning.

6 Conclusion

In this work, we address two key challenges that LLMs face when handling temporal questions and introduce a time-aware retrieve-rewrite-retrieve-rerank framework named TimeR^4.

To mitigate the issue of the temporal hallucination of LLMs, we utilize a *retrieve-rewrite* strategy to fetch relevant facts in FKS and integrate specific timestamps into the questions. Afterward, in order to retrieve facts that satisfy both time constraints and semantic similarity, we implement a *retrieve-rerank* strategy. We perform time-aware retrieval from the TKS and rerank them based on temporal constraints. Finally, we fine-tune open-source LLMs, leveraging the knowledge in TKGs to enhance the temporal reasoning capabilities in LLMs. Experiments on two datasets demonstrate that our framework significantly enhances the temporal reasoning abilities of LLMs.

Limitations

Although our approach achieves significant improvements, with relative gains of 47.8% and 22.5% on two TKGQA datasets, the performance of complex temporal fact retrieval can still be further enhanced. How to retrieve more precise and effective temporal information is a question worthy of exploration.

Additionally, controlling the answer format during the generation process of LLMs without fine-tuning is difficult, as discussed in Section 5.5. Thus, standardizing the answer formats of LLMs or developing a more reasonable evaluation method is another important future task.

7 Acknowledgements

This research is supported by the National Natural Science Foundation of China (No. 62302243, 62272250), the Natural Science Foundation of Tianjin, China (No. 22JCQNJC01580, 22JCJQJC00150, 23JCYBJC01230), and the Fundamental Research Funds for the Central Universities, Nankai University (63241442).

References

Jinheon Baek, Alham Fikri Aji, Jens Lehmann, and Sung Ju Hwang. 2023a. Direct fact retrieval from knowledge graphs without entity linking. *Preprint*, arXiv:2305.12416.

Jinheon Baek, Alham Fikri Aji, and Amir Saffari. 2023b. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. *arXiv preprint arXiv:2306.04136*.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. *arXiv preprint arXiv:2302.04023*.

Ziyang Chen, Dongfang Li, Xiang Zhao, Baotian Hu, and Min Zhang. 2023a. Temporal knowledge question answering via abstract reasoning induction. *Preprint*, arXiv:2311.09149.

Ziyang Chen, Jinzhi Liao, and Xiang Zhao. 2023b. Multi-granularity temporal question answering over knowledge graphs. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11378–11392.

Sitao Cheng, Ziyuan Zhuang, Yong Xu, Fangkai Yang, Chaoyun Zhang, Xiaoting Qin, Xiang Huang, Ling Chen, Qingwei Lin, Dongmei Zhang, et al. 2024. Call me when necessary: Llms can efficiently and faithfully reason over structured environments. *arXiv preprint arXiv:2403.08593*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. 2023. A survey on in-context learning. *Preprint*, arXiv:2301.00234.

Alberto García-Durán, Sebastijan Dumančić, and Mathias Niepert. 2018a. Learning sequence encoders for temporal knowledge graph completion. *arXiv preprint arXiv:1809.03202*.

Alberto García-Durán, Sebastijan Dumančić, and Mathias Niepert. 2018b. Learning sequence encoders for temporal knowledge graph completion. *arXiv preprint arXiv:1809.03202*.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *Preprint*, arXiv:2402.07630.

Jie Huang and Kevin Chen-Chuan Chang. 2023. Towards reasoning in large language models: A survey. *Preprint*, arXiv:2212.10403.

Zhen Jia, Soumajit Pramanik, Rishiraj Saha Roy, and Gerhard Weikum. 2021. Complex temporal question answering on knowledge graphs. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pages 792–802.

Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Structgpt: A general framework for large language model to reason over structured data. *arXiv preprint arXiv:2305.09645*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. *Preprint*, arXiv:1909.11942.

Tianle Li, Xueguang Ma, Alex Zhuang, Yu Gu, Yu Su, and Wenhu Chen. 2023. Few-shot in-context learning for knowledge base question answering. *Preprint*, arXiv:2305.01750.

Yonghao Liu, Mengyu Li Di Liang, Fausto Giunchiglia, Ximing Li, Sirui Wang, Wei Wu, Lan Huang, Xiaoyue Feng, and Renchu Guan. 2023. Local and global: temporal question answering via information fusion. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 5141–5149.

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. 2023. Reasoning on graphs: Faithful and interpretable large language model reasoning. *arXiv preprint arXiv:2310.01061*.

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. 2024. Reasoning on graphs: Faithful and interpretable large language model reasoning. *Preprint*, arXiv:2310.01061.

Costas Mavromatis, Prasanna Lakkur Subramanyam, Vassilis N Ioannidis, Adesoji Adeshina, Phillip R Howard, Tetiana Grinberg, Nagib Hakim, and George Karypis. 2022. Tempoqr: temporal question reasoning over knowledge graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 5825–5833.

Soumajit Pramanik, Jesujoba Alabi, Rishiraj Saha Roy, and Gerhard Weikum. 2023. Uniqorn: Unified question answering over rdf knowledge graphs and natural language text. *Preprint*, arXiv:2108.08614.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084.

Apoorv Saxena, Soumen Chakrabarti, and Partha Talukdar. 2021. Question answering over temporal knowledge graphs. *arXiv preprint arXiv:2106.01515*.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4498–4507, Online. Association for Computational Linguistics.

Aditya Sharma, Apoorv Saxena, Chitrank Gupta, Seyed Mehran Kazemi, Partha Talukdar, and Soumen Chakrabarti. 2022. Twirgcn: Temporally weighted graph convolution for question answering over temporal knowledge graphs. *arXiv preprint arXiv:2210.06281*.

Jungbin Son and Alice Oh. 2023. Time-aware representation learning for time-sensitive question answering. *Preprint*, arXiv:2310.12585.

Haitian Sun, Tania Bedrax-Weiss, and William W Cohen. 2019. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. *arXiv preprint arXiv:1904.09537*.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W. Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. *Preprint*, arXiv:1809.00782.

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.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Heung-Yeung Shum, and Jian Guo. 2023. Think-on-graph: Deep and responsible reasoning of large language model with knowledge graph. *arXiv preprint arXiv:2307.07697*.

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

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

A Prompt Template

The prompts for rewriting can be found in Figure 5. The template used for instruction tuning and reasoning is shown in Figure 6.

Rewriting Prompt Template

Replace the temporal fact in questions with explicit timestamps from the provided facts or your knowledge without any explanation. If you are not sure about the answer, return the original questions.

For instance, from the fact:
"[Juan Carlos I, Praise or endorse, Vietnam, 2006-02-22]",
We can modify the question:
"After Vietnam, who was the first to praise Juan Carlos I?"
to "After 2006-02-22, who was the first to praise Juan Carlos I?"

Here is your turn:
Facts: `<fact>`
Question: `<question>`

Figure 5: Rewriting prompt template.

Reasoning Prompt Template

Based on the historical facts, please answer the given question. Please keep the answer as simple as possible and return all the possible answers as a list.
Historical facts: `<fact>`
Question: `<question>`

Figure 6: Reasoning prompt template.