SCIENCE CHINA
Information Sciences

• Supplementary File •

Question answering over temporal knowledge graphs: a self-improving hierarchical retrieval-augmented approach

Ruishen Liu[1†], Shaorong Xie[1,2†], Xiangfeng Luo[1], Xinzhi Wang[1] & Hang Yu[1]

[1]*The School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China*
[2]*Engineering Research Center of Unmanned Intelligent Marine Equipment, Ministry of Education, Shanghai 200444, China*

Appendix A Related work

Appendix A.1 Traditional TKGQA methods

Traditional temporal knowledge graph question answering (TKGQA) has predominantly followed two paradigms: Semantic Parsing-based (SP-based) methods, which translate questions into formal queries, and TKG Embedding-based (TKGE-based) methods, which utilize representation learning.

1) Semantic parsing-based Methods: SP-based methods aim to convert a natural language question into a structured, executable logical form. These symbolic approaches offer high precision and inherent interpretability. The process typically includes parsing the question to identify semantic components, constructing an ungrounded logical form, grounding it with TKG entities and relations, and finally executing the query. Pioneering works in this area, such as TEQUILA [1], address complex temporal questions by decomposing them into simpler, answerable sub-questions. The final answer is then derived by applying rule-based temporal reasoning over the time intervals associated with the sub-answers. To create more robust and generalizable parsers, SYGMA [2] utilizes Abstract Meaning Representation (AMR) [3] to capture temporal semantics and map them to KB-agnostic λ-expressions. Based on this, a decomposition method of λ expressions is introduced to separately handle the primary event and the temporal constraints [4]. Other research has focused on designing specialized logical forms. For example, SF-TQA [5] designs a Semantic Framework of Temporal Constraints (SF-TCons) to systematically model the intricate relationship between events and temporal connectors. Prog-TQA [6] expands the set of temporal operators based on the Knowledge-oriented Programming Language (KoPL) [7], enabling more concise and powerful temporal queries. While powerful, the performance of SP-based methods is often constrained by the rigidity of parsing rules and templates, which can struggle with the linguistic diversity of natural language and the incompleteness of the TKG.

2) TKG embedding-based Methods: TKGE-based methods reframe TKGQA as a knowledge graph completion task, operating within a low-dimensional vector space. These approaches generally offer better robustness to linguistic variations and noise compared to SP-based methods. The core idea is to learn dense representations for the question and all TKG components, and then identify the answer via a scoring function. CronKGQA [8] first adapts this paradigm for the temporal domain. It leverages BERT to encode the question and a temporal KG embedding model to represent the TKG, subsequently ranking candidate answers based on their matching scores. Subsequent research has largely focused on two key challenges, enhancing the model temporal awareness and integrating structural graph information more effectively. To improve temporal sensitivity, researchers have developed various specialized modules. TSQA [9] further refins this by introducing a temporal order loss during the training of the KG embedding model, making it more sensitive to the sequence of events. To address questions with varying time granularities (e.g., year vs. day), MultiQA [10] introduces a multi-granularity temporal aggregation mechanism. To better leverage the graph structure, Graph Neural Networks (GNNs) have become a central component. TwiRGCN [11] designs a question-dependent attention mechanism over an R-GCN [12] to up-weight messages from relevant reasoning paths. LGQA [13] fuses global sentence-level semantics with local entity-level graph information using transformers. More complex multi-fact reasoning is addressed by JMFRN [14], which employs joint time-aware and entity-aware attention to aggregate evidence from multiple retrieved facts. Similarly, TMA [15] selects semantically similar facts and uses multiple token-level attentions to enhance the question representation.

Appendix A.2 LLM-based KGQA

Recently, large language models (LLMs) have shown great potential in graph reasoning and understanding complex semantics [16, 17]. This has led to a new paradigm where LLMs are integrated with knowledge graphs (KGs) for question answering, yielding significant advancements, particularly in the static KGQA domain. A variety of techniques have been explored to facilitate this synergy. A dominant approach is Retrieval-Augmented Generation (RAG) [18], where the quality of retrieved facts is crucial. AMQR [19] develops an adaptive multi-aspect retrieval mechanism to improve robustness against irrelevant facts, while GNN-Ret [20] and GraphRAG [21] have leveraged GNNs to enhance the retrieval of structured subgraphs. Once facts are retrieved, the focus shifts to the reasoning process. KG-COT [22] pioneers the use of Chain-of-Thought (CoT) prompting, guiding LLMs to perform interpretable, step-by-step reasoning.

* Corresponding author (email: wxz2017@shu.edu.cn, yuhang@shu.edu.cn)
† These authors contributed equally to this work.

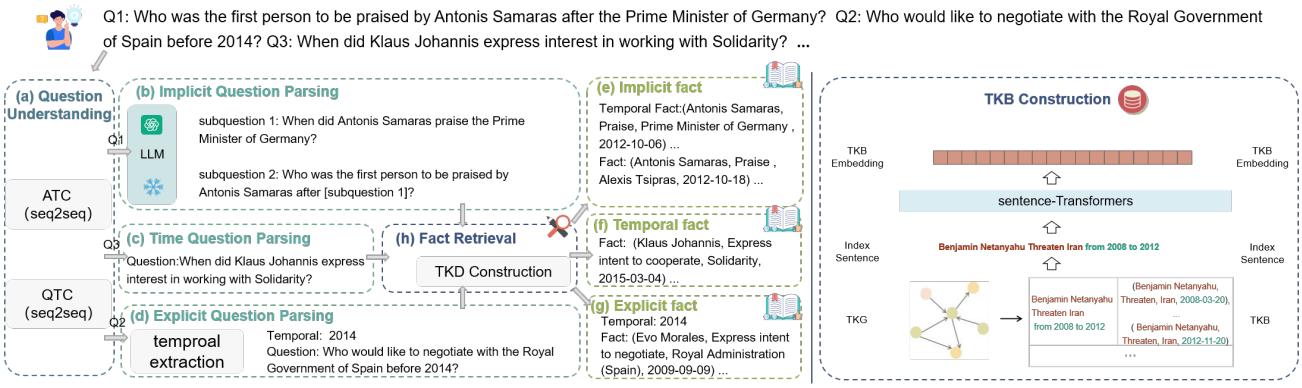

Figure C1 The process of question understanding and the fact retriever. The right part presents how to build the Temporal Knowledge Base(TKB) from the temporal knowledge graph.

This CoT paradigm is further explored in works like ToG (Think-on-Graph) [23], which allows the LLM to perform an iterative beam search over the graph to actively explore reasoning paths. And IRCOT [24] conducts the interleaved retrieval with steps in a chain of thought.

Appendix B Preliminaries

Appendix B.1 Temporal knowledge graph

A temporal knowledge graph is denoted as $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T}, \mathcal{F})$, where $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{T}$ are a set of entities, relations and timestamps, respectively. The quadruple $\mathcal{F} = \{(s, r, o, t) | s, o \in \mathcal{E}, r \in \mathcal{R}, t \in \mathcal{T}\}$ represents that there is a relation r between the source entity s and target entity o at timestamp t. The inverse relations are added into the dataset, i.e., (o, r^{-inv}, s, t).

Appendix B.2 Temporal knowledge graph question answering

Given the temporal knowledge graph $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T}, \mathcal{F})$ and a question q, the task of TKGQA is to infer the correct answer based on the relevant quadruples $(s, r, o, t) \in \mathcal{F}$, where the answer can be an entity or a timestamp.

Appendix C Method

Appendix C.1 Question understanding

The semantics of complex questions is often hierarchical, demonstrating that the different components of the question contain different entities, relations and temporal constraints. Previous works map the whole question into a flat embedding, ignoring the hierarchical temporal semantics, which makes it difficult to distinguish entities and relations with different semantics, resulting in irrelevantly fetched facts. To address this issue, the question understanding module consists of two steps, i.e., Question Classification and Question Decomposition.

Question classification. As shown in Figure C1, the question is categorized into two aspects: 1) the type of question, which is a label from [Single, Complex]. 2) the answer type, which is a label from [time, entity]. We build two classifiers to perform prediction upon a pre-trained language model. The question type classifier (QTC) is as follows.

$$E_q = LM(q), \tag{C1}$$

$$h_q = W_q E_q, \tag{C2}$$

$LM(\cdot)$ is a language model, q denotes the questions, W_q is the learnable parameter, and h_q is the label of the question, i.e., [Single, Complex]. The answer type classifier (ATC) is similar to the question type classifier. We compute E_q with learnable parameter W_a.

Question decomposition. For the complex question, we need to focus on the distinct temporal information needs of the different components of the question. As shown in Figure C1(b), given the question *"Who was the first person to be praised by Antonis Samaras after the Prime Minister of Germany?"*, we need first to focus on *"when did Antonis Samaras praise the Prime Minister of Germany?"*, and then to focus on the question *"Who was the first person to be praised by Antonis Samaras"*. Based on this observation, we decompose the complex question into sub-questions. Since LLM has a strong ability to deal with complex questions, we use the closed-source LLM to break down the questions.

$$q' = LLM_c(Prompt(q)), \tag{C3}$$

where q is a question. q' is a set of sub-questions. LLM_c is a closed-source LLM, e.g., ChatGPT, Gemini, etc. $Prompt(\cdot)$ is the specific designed prompt.

The example of implicit temporal question

Instruction: Based on the relevant facts, please answer the given question. Please keep the answer as simple as possible and return all the possible answers as a list. You first think about the reasoning process and then provide the user with the answer. Respond in the following format: <think>...</think><answer>...</answer>
Question: Which country was the last to accuse the UN security council before the Military Personnel of Canada did?
Subqestion: [
Subquestion 1: When did the Military Personnel of Canada accuse the UN security council?,
Subquestion 2: Which country was the last to accuse the UN security council before [subquestion 1]?
]
Facts: [
[
"Military Personnel (Canada) Accuse UN Security Council in 2006-02-10.",
"UN Security Council Accuse-inv Military Personnel (Canada) in 2006-02-10.",
"Military Personnel (Canada) Accuse-inv Professor (France) in 2007-09-05.",
"Military Personnel (Canada) Accuse-inv Police (Canada) in 2010-04-30.",
"Military Personnel (Canada) Accuse-inv Police (Canada) in 2010-05-01."
],
[
"UN Security Council Accuse-inv Iran in 2006-12-23.",
"UN Security Council Accuse Iran in 2008-05-15.",
"UN Security Council Accuse-inv Eritrea in 2006-01-04."
]
]
Response:
<think>The question asks which country accused the UN security council last before the Military Personnel of Canada did. First, I need to find when the Military Personnel of Canada accused the UN security council, and then find which country was the last to accuse the UN security council before that date. From the facts, Military Personnel (Canada) accused UN Security Council in 2006-02-10. Before 2006-02-10, the last country to accuse the UN Security Council was Eritrea in 2006-01-04. So the answer is Eritrea.</think>
<answer>['Eritrea']</answer>

Figure C2 The example of complex temporal question answering.

Appendix C.2 Temporal fact retrieval

After question understanding, we need to retrieve the facts that are related to the questions. A significant challenge in TKGs is data redundancy. The same core fact (e.g., *(Benjamin Netanyahu, threatens, Iran)*) can appear multiple times with different timestamps, leading to retrieving more similar information. To handle this efficiently, we preprocess the TKG into a structured Temporal Knowledge Base (TKB). In the TKB, all temporal instances of a core fact are consolidated into a single group. Moreover, since textual representations expose explicit temporal values, enabling the LLM to perform direct logical comparisons, which are crucial for temporal reasoning, we textualize the facts to align with the semantic space of LLMs. For example, the fact *(Benjamin Netanyahu, Threaten, Iran, 2012-03-07),. . . , (Benjamin Netanyahu, Threaten, Iran, 2012-11-20)* is organized in a group, i.e., *"Benjamin Netanyahu threaten Iran in 2012 year"*. We encode the TKB through the fine-tuned temporal-fact retriever. The details of the fine-tuned temporal-fact retriever are shown in the Appendix C.5.

$$g_{s,r,o} = \{(s', r', o', t') \in \mathcal{G} \mid s' = s, r' = r, o' = o\}, \tag{C4}$$

$$Group(\mathcal{G}) = \{g_{s,r,o} \mid \forall (s, r, o) \in I \wedge g_{s,r,o} \neq \emptyset\}, \tag{C5}$$

$$TKB = \{\mathbf{E}_f \mid \mathbf{E}_f = LM_t(g_{s,r,o}), g_{s,r,o} \in Group(\mathcal{G})\}, \tag{C6}$$

where I is the index set of all unique (s, r, o) tuples. $LM_t(\cdot)$ is a temporal-fact retriever. $\mathcal{G}$ is the temporal knowledge graph.

After question understanding and TKB construction, given the question, we first parse the question based on the question type and answer type. If the question contains the explicit time, we extract it as the temporal signal. If the question type is complex, we define which subquestion is the temporal signal, and then retrieve each fact based on the subquestions(Figure C1). Cosine similarity is used to identify the top-k most relevant fact groups:

$$E_{q'} = LM_t(q'), \tag{C7}$$

$$f' = \underset{E_f \in \text{TKB}}{\text{Topk}} \cos\left(E_{q'}, E_f\right), \tag{C8}$$

where f' is the relevant facts, LM_t is a temporal-fact retriever, and q' denotes the processed questions.

Appendix C.3 Data augmentation

Recent work has shown that LLMs reasoning abilities are significantly enhanced when they generate explicit chain-of-thought (CoT) [25]. Moreover, reasoning-based LLMs, like DeepSeek-R1 [26] and O1 [27], have shown a significant paradigm for complex reasoning tasks with reasoning process generation. Therefore, in this paper, we also enhance the LLM temporal reasoning ability through guiding the model to generate the reasoning process.

Based on the processed questions and the retrieved facts, we leverage closed-source LLMs, such as ChatGPT, Gemini, etc, to perform data augmentation. The data augmentation step includes data distillation and data filtering. During the distillation phase, we utilize the closed-source LLM to generate the reasoning process and the answer. The input of LLM is the processed questions and the relevant facts. The settings are as follows:

- The temperature is set to 0.5.
- For reasoning data, the specific prompts (Figure C2) are designed to ensure the reasoning process and answer format.

Table D1 The details of the MultiTQ dataset.

Question Type		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	57,979	54,584

After data distillation, we perform an answer-check to filter data. The ground truth is used to check whether the generated answer is aligned with the ground truth. If the generated answer matches the ground truth, the generated data remains. Otherwise, it is discarded.

Appendix C.4 LLM fine-tuning

After data augmentation, we train the model through supervised fine-tuning (SFT). During SFT, each training instance d includes six componentes, i.e., $d = (q, q', f, p, y^*, y)$. q *and* q' denote the questions and the processed questions, respectively. f is the relevant facts. p is the reasoning process formatted as $< think > ... < /think >$ and y^* denotes the generated answers contained in $< answer > ... < /answer >$. y is the ground truth. During training, q, q' *and* f are the input of the model. p and y^* is the output of the model. This phase ensures the model learns the temporal reasoning patterns and generates a well-formed reasoning process and answer.

Appendix C.5 Self-improvement strategies

Retriever self-improvement. A high-precision retriever is paramount for TKGQA, yet training one is hampered by the lack of direct supervision. Existing datasets, which only contain questions and answers, do not label the intermediate temporal facts. To deal with this issue, we propose a retriever self-improvement strategy that leverages weak supervision from the final answer.

In the initial stage, we leverage the closed-source LLM to generate the reasoning process and the answer. Then the answer-check is used to filter data. If the generated answer is aligned with the ground truth, it means there are correct facts contained in the fetched facts. Therefore, the facts used in its reasoning chain are regarded as a positive question-fact pair. Then, we randomly corrupt the relations, entities, and time of the correct facts to generate the corresponding negative facts. There are three types of negative facts: inverse fact, incorrect fact, and incorrect time. Specifically, the incorrect time negatives are critical for TKGQA, which forces the retriever to distinguish between the same event occurring at different timestamps.

$$\phi_{TKB} = cos(E_{q'}, E_f), \tag{C9}$$

$$\mathcal{L} = \sum_i \left[w_p Y \cdot \exp\left(\phi_{TKB}\right) + w_n (1 - Y) \cdot \exp\left(1 - \phi_{TKB}\right) \right], \tag{C10}$$

where $E_{q'}$ and E_f are the embeddings of the questions and facts, respectively. ϕ_{TKB} is the cosine similarity between the question embedding and the fact embedding. Y is 1 for a positive pair and 0 for a negative pair.

After temporal-fact retriever training, the fine-tuned temporal-fact retriever is used to fetch the facts. Afterward, the facts and the processed questions are processed by the closed-source LLM to generate the answer and reasoning step. The iterative process persists until either the maximum number of rounds is reached or the accuracy of the reasoning does not increase.

LLM self-improvement. Recently, reinforcement learning (RL) has shown great potential in improving the LLM reasoning ability. Compared with SFT, the RL can dynamically adjust the LLM reasoning process, thereby achieving superior performance in complex reasoning tasks [26]. Therefore, we further enhance the model performance through reinforcement learning. In the initial stage, the SFT model can generate the reasoning path and a final answer for each question in the training set. Then, each generated answer is validated against the ground truth. If an answer is correct, the corresponding data is added to the correct dataset. Afterwards, the model is further trained on the correct dataset by using reinforcement learning. We regard the GRPO [28] as reinforcement learning.

This entire process, i.e., generation, validation, filtering, and refinement, is repeated iteratively. The training loop continues until a predefined stopping criterion is met, such as reaching a maximum number of rounds or the accuracy of the reasoning does not increase. This approach enables the model to refine its own reasoning policy by learning from its successful problem-solving attempts.

Appendix D Experiment

Appendix D.1 Dataset

We evaluate our model on two benchmarks, MultiTQ [29] and TimeQuestions [30], avoiding the CronQuestions dataset due to known issues with spurious relations [31]. MultiTQ is a large-scale, complex TKGQA dataset derived from ICEWS05-15, containing over 500k [32] question pairs. The dataset can be divided into simple and complex questions. Furthermore, it can be categorized into six types, such as "Equal", "Before/After", "Before Last", etc. TimeQuestions is a dataset from Wikidata containing 16k complex questions across four categories, i.e., Explicit, Implicit, Temporal and Ordinal. More details are shown in Table D1 and D2.

Table D2 The details of the TimeQuestions dataset.

Question Type	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	6970	3,236	3,237

Table D3 The main results of SHRA on MultiTQ (Hit@1). The best results are in bold. And the second best results are in underlined.

		Question Type		Answer Type	
Model	Overall	Multiple	Single	Entity	Time
ALBERT	0.108	0.086	0.116	0.139	0.032
EmbedKGQA	0.206	0.134	0.235	0.290	0.001
CronKGQA	0.279	0.134	0.337	0.328	0.156
MultiQA	0.293	0.159	0.347	0.349	0.157
MTQADC	0.342	0.185	0.406	0.430	0.128
ARI	0.380	0.210	0.680	0.394	0.344
ChatGPT	0.078	0.052	0.094	0.087	0.058
LLamAv3	0.064	0.043	0.073	0.078	0.012
TimeR[4]	0.728	0.335	**0.887**	**0.639**	0.945
SHRA	**0.741**	**0.395**	0.832	0.617	**0.966**

Table D4 The comparative results of SHRA on TimeQuestions (Hit@1). The best results are in bold. And the second best results are in underlined.

Model	Overall	Explicit	Implicit	Temporal	Ordinal
GRAFT-NET	0.452	0.445	0.428	0.515	0.322
PullNet	0.105	0.022	0.081	0.234	0.029
Uniqorn	0.331	0.318	0.316	0.392	0.202
TempoQR	0.459	0.503	0.442	0.485	0.367
CronKGQA	0.393	0.388	0.380	0.347	0.457
LGQA	0.529	0.532	0.506	0.605	0.402
JMFRN	0.628	0.662	0.530	0.646	0.553
ChatGPT	0.459	0.433	0.511	0.465	0.481
LLaMAv3	0.241	0.282	0.256	0.194	0.281
GenTKGQA	0.584	0.596	0.611	0.563	0.578
TimeR[4]	0.781	0.823	0.730	0.830	0.649
SHRA	**0.804**	**0.831**	**0.736**	**0.871**	**0.666**

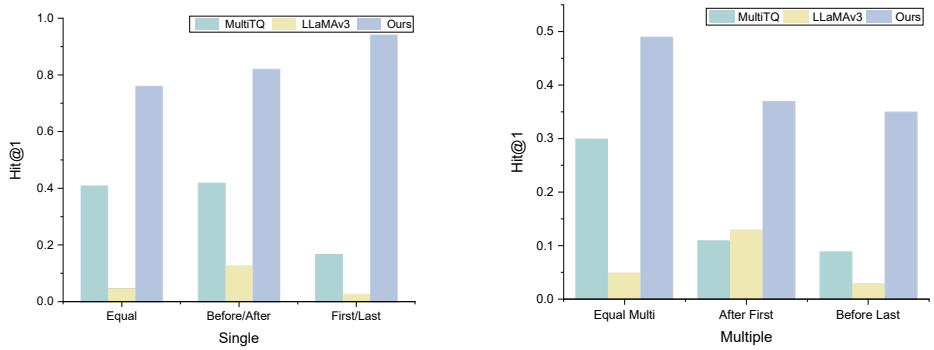

Figure D1 The results of SHRA on MultiTQ across different question types.

Appendix D.2 Baseline and evaluation metrics

For MultiTQ, we use three types of baselines: (1) LM-based models, including ALBERT. (2) Embedding-based models, including EmbedKGQA, CronKGQA, MultiQA, and MTQADC. (3) LLM-based models, including ARI, ChatGPT, LLaMAv3 and TimeR[4]. For TimeQuestions, three types baselines are utilized: (1) KG embedding-based models, including GRAFT-NET, PullNet and Uniqorn. (2) TKG embedding-based models, including CronKGQA, TempoQR, LGQA, JMFRN. (3) LLM-based models, including ChatGPT, LLaMAv3, TimeR[4] and GenTKGQA. We evaluate all models using the Hits@k metric, reporting Hits@1 and Hits@10.

Appendix D.3 Implementation details

LLaMAv3-8B and Gemini (Gemini-2.0-flash) are utilized as the open-source LLM and closed-source LLM, respectively. Since the MultiTQ is too large, we sample the MultiTQ to conduct experiments. The temporal-fact retriever is implemented using the sentence-transformers library [33]. SHRA is trained on 4 NVIDIA A100(40GB) GPUs. For the GRPO refinement stage, we set the number of generated candidate outputs to 4 [1)].

Appendix D.4 Main results

Table D3 and Table D4 report the main results of SHRA in comparison with other approaches on MultiTQ and TimeQuestions. The proposed model outperforms other baselines.

Performance on MultiTQ. As shown in Table D3, SHRA achieves an overall Hits@1 score of 0.741, surpassing other baselines. Traditional LM-based methods like ALBERT perform poorly, lacking access to the necessary factual knowledge. Even though Embedding-based models (e.g., CronKGQA, MTQADC) find the relevant facts relying on the temporal questions embedding, their limited semantic understanding restricts their performance on complex questions. Among the LLM-based approaches, SHRA demonstrates a significant advantage, with the improvement of 1.3% in Hit@1. This highlights the effectiveness of our self-improvement and retrieval architecture. Notably, SHRA achieves a remarkable Hits@1 score of 0.966 on questions requiring a time answer, indicating its exceptional temporal reasoning capability.

Performance on TimeQuestions. SHRA also achieves the best performance across all question types, achieving an overall Hits@1 score of 0.804. Compared with the KG-based methods, our model provides the relevant facts, improving the model performance. SHRA outperforms the traditional TKGQA baseline (JMFRN) by a substantial margin of 17.6%, demonstrating the superiority of LLMs in comprehending complex questions. Furthermore, SHRA also outperforms the LLM-based methods, TimeR[4], demonstrating the benefits of our proposed method.

Analysis of performance by question type. Furthermore, we conduct a fine-grained analysis of its performance on different question subtypes from MultiTQ. As illustrated in Figure D1, for both single and multiple questions, SHRA obtains the best results. Since the MultiTQ questions contain implicit temporal constraints, requiring multiple facts, the result of "single" questions is better than the multiple questions. Within the "Single" category, SHRA excels at "First/Last" and "Before/After" questions. Performance on "Equal" questions is comparatively lower. This can be attributed to the nature of these questions, which often involve year-level granularity. Such broad temporal scopes can lead to the retrieval of numerous similar and potentially ambiguous facts, posing a greater challenge for the model. Despite this, the superior performance of SHRA across all types underscores its robustness in handling diverse and complex temporal questions.

Appendix D.5 Retrieval effect

The number of retrieved facts. We investigate how the quantity of retrieved facts affects the model reasoning performance. Figure D2 plots the model Hits@1 and Hits@10 on MultiTQ. And we vary the number of retrieved facts (k) from 0 to 20. The results show a sharp performance increase as k goes from 0 to 2, confirming that providing even a small number of relevant facts is critical for this task. When the number exceeds 10, the model performance tends to stabilize, reaching a maximum at 15. If the number is larger than 15, the model performance degrades slightly. Eventhough retrieving more facts increases the likelihood of including the ground truth, it also introduces irrelevant information that can hinder the model ability to reason effectively.

1) The source code and data are released on request.

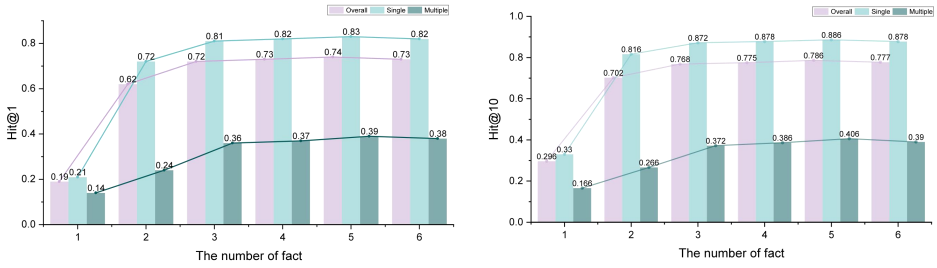

Figure D2 The effect of different fact numbers on MultiTQ.

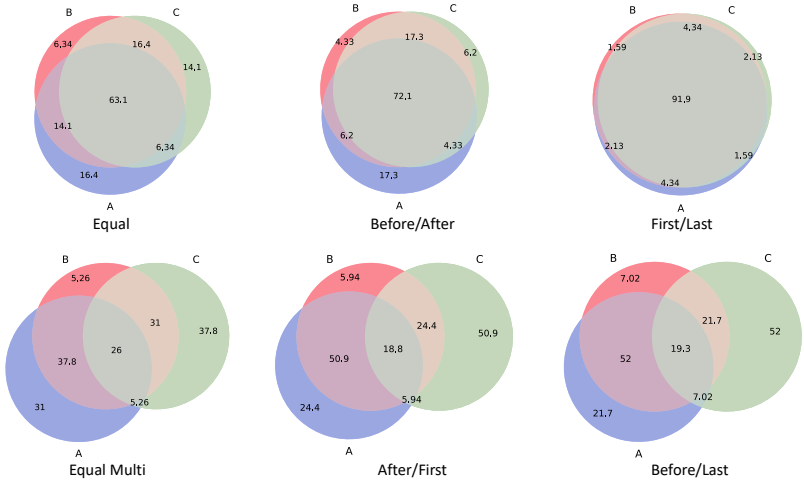

Figure D3 The answers coverage overlap of different retrievers across various question types on MultiTQ. A is the non-fine-tune retriever. B denotes the temporal-fact retriever. And C is the ground truth.

The effectiveness of the temporal-fact retriever. A core component of our method is the self-improvement strategy for fine-tuning the temporal-fact retriever. To verify the effectiveness of the proposed strategy, we report the ground truth coverage of the temporal-fact retriever across different question types. As shown in Figure D3, the results demonstrate the superiority of our approach. Across all question types, our temporal-fact retriever (B) achieves a higher overlap with the ground-truth facts (C) than the baseline retriever (A). This enhanced retrieval accuracy is crucial, as it supplies the downstream LLM with higher-quality facts, directly enabling more accurate reasoning. This analysis validates that the self-improvement strategy, which leverages weak supervision from question-answer pairs, automatically generates effective training data for the retriever.

Appendix D.6 Ablation study

Table D5 The ablation study on MultiTQ. w/o removes the corresponding component. QU, DA, RSI, LSI denote Question Understanding, Data Augmentation, Retriever Self-Improvement and LLM Self-Improvement, respectively.

	Hits@1					Hits@10				
		Question Type		Answer Type			Question Type		Answer Type	
Model	Overall	Multiple	Single	Entity	Time	Overall	Multiple	Single	Entity	Time
w/o QU	0.694	0.283	0.802	0.469	0.694	0.744	0.292	0.863	0.511	0.744
w/o DA	0.729	0.398	0.815	0.651	0.872	0.764	0.419	0.853	0.690	0.901
w/o RSI	0.626	0.206	0.737	0.455	0.936	0.663	0.217	0.780	0.512	0.936
w/o LSI	0.730	0.370	0.825	0.601	0.966	0.767	0.379	0.869	0.657	0.966
SHRA	0.741	0.395	0.832	0.617	0.966	0.786	0.406	0.886	0.686	0.966

To evaluate the contribution of each component within the SHRA, we conduct an ablation study on the MultiTQ dataset. The study demonstrates that each component has a positive effect on the model performance.

The most critical component is Retriever Self-Improvement. Its removal causes the most substantial performance drop, with the overall Hits@1 falling from 0.741 to 0.626. This decrease underscores the importance of accurate and relevant fact retrieval.

Model	Overall	Multiple	Single
LLaMAv3	0.064	0.043	0.073
LLaMAv3 w.tkg	0.384	0.140	0.482
SHRA w/o.tkg	0.198	0.145	0.212
SHRA	0.741	0.395	0.832

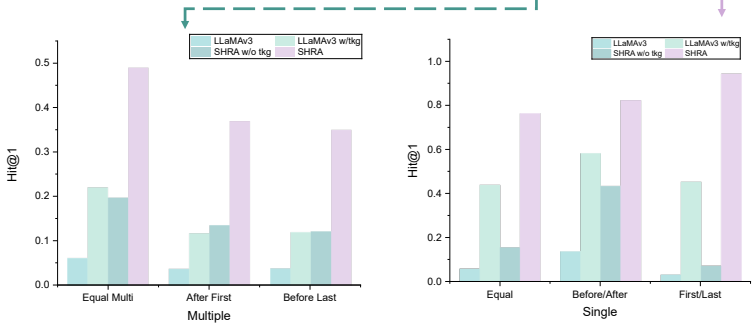

Figure D4 The LLM effect on MultiTQ. w.tkg means the model uses the retrieved facts. w/o.tkg denotes the model answers the questions without facts.

The Question Understanding module is the second important component. Removing it reduces the overall Hits@1 to 0.694. Notably, its contribution is most pronounced for complex "Multiple" questions, where the score drops from 0.395 to 0.283. This result validates that it is essential to decompose the hierarchical semantics of the complex question.

Finally, the Data Augmentation and LLM Self-Improvement modules both provide significant performance enhancements. Their removal lowers the Hits@1 to 0.729 and 0.730, respectively. These components are responsible for refining the model reasoning ability by generating and learning from reasoning paths. Their positive impact highlights the value of guiding the model to generate correct thought processes. In summary, the ablation study demonstrates that each component plays a distinct and vital role in SHRA.

Appendix D.7 Compared with LLMs

To understand the interplay between the base LLM capabilities, the retrieval module, and our fine-tuning framework, we conduct a detailed comparative analysis on the MultiTQ dataset (Figure D4). We evaluated four distinct model configurations: (1) LLaMAv3, the base LLM in a zero-shot setting, without retrieved facts. (2) LLaMAv3 w.tkg, the base LLM augmented with retrieved facts. (3) SHRA w/o. tkg, the fine-tuned SHRA model, but without retrieved facts. (4) SHRA, the proposed model with all components.

The effectiveness of retrieval. Comparing the base LLaMAv3 with its retrieval-augmented counterpart (LLaMAv3 w.tkg), the overall Hits@1 jumps from 0.064 to 0.384. The improvement is especially pronounced for "Single" answer questions, with a remarkable 40.9% gain (from 0.073 to 0.482). This demonstrates that the parametric knowledge of a base LLM is insufficient for the TKGQA task and that external facts are essential for effective reasoning.

The effectiveness of fine-tuning. We compare the retrieval augmented baseline (LLaMAv3 w.tkg) with our fine-tuned model that lacks retrieval (SHRA w/o. tkg). Notably, our fine-tuned model performs comparably to the retrieval-augmented baseline on complex "Multiple" questions, i.e., 0.145 vs. 0.140 Hits@1. This denotes that our training framework enables the model to internalize not just factual knowledge, but also the complex reasoning patterns required to solve complex questions, partially compensating for the absence of retrieved facts.

The effectiveness of SHRA. The SHRA model outperforms other variants. This demonstrates that the fine-tuned model is not only better at reasoning but is also more adept at utilizing the provided facts, leading to the state-of-the-art overall Hits@1 score of 0.741.

Appendix D.8 Case study

To further illustrate the reasoning process of the SHRA, we present a series of case studies from the MultiTQ dataset in Fig D5. These examples compare the proposed model against the retrieval-augmented baseline (LLaMAv3 w.tkg). From Figure D5, four advantages of SHRA are highlighted: (1) **Improving the reasoning performance**. Eventhough we provide the relevant facts, the model without fine-tuning, like LLaMAv3, fails to capture the complex question answering patterns, generating irrelevant responses. For example, the LLaMAv3 generates the entity contained in the question(Fig D5 (e)). Conversely, SHRA has a higher accuracy in understanding and reasoning about complex questions. (2) **Enhanced answer completeness.** Even when the baseline provides a partially correct response, it often suffers from incompleteness. For example, in Figure D5(a), the question requires two entities *'Japan'* and *'France'*, but LLaMAv3 w.tkg identifies only one. Likewise, in the "Equal Multiple" question (d), the baseline answer is incomplete. SHRA, having learned from reasoning paths during its self-improvement phase, identifies all valid entities and generates a complete answer set. (3) **Sensitive to the temporal**. Compared with LLaMAV3, the SHRA is more sensitive to the temporal predicate, like *before* and *after*. As shown in Fig D5 (c), the generated entity *"France"* and *"Sudan"* are beyond the target temporal i.e. *2012-01-30*. The

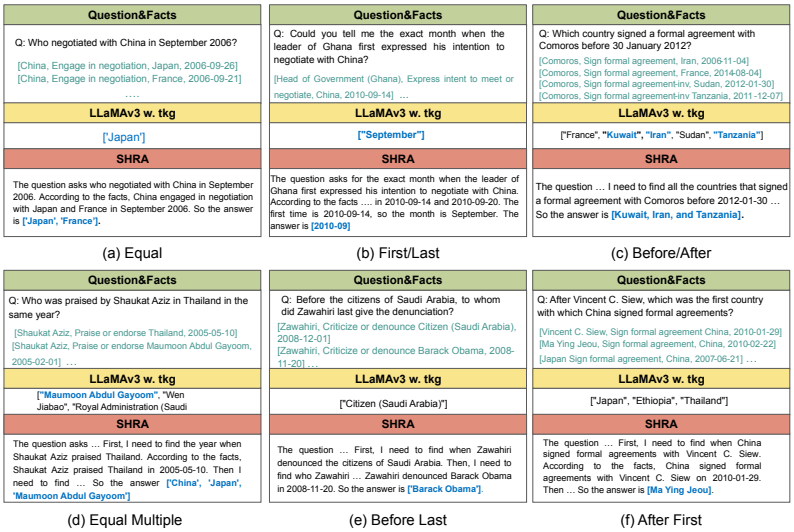

Figure D5 The case study of the SHRA and LLaMAv3 for six question types on MultiTQ. w.tkg means the model uses the retrieved facts. Green and Blue denote the retrieved facts and the correct answers, respectively.

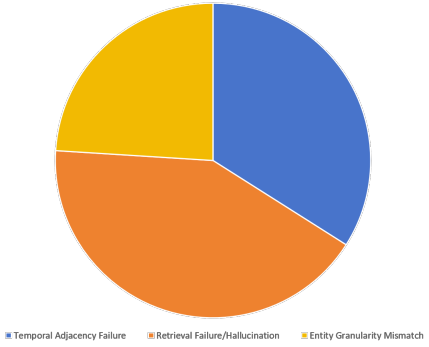

Figure D6 Error analysis.

SHRA can distinguish the difference between *"before"* and *"after"*, improving the understanding of complex questions. (4) **More standardized generation**. Given the question *"exact month..."*(Figure D5 (b)), the ground truth is *"2010-09"*. Although LLaMAv3 generates *"September"*, which is semantically correct, it does not match the required format and would be marked as incorrect during the evaluation. However, SHRA generates the precise *"2010-09"* format, demonstrating that it has learned not only what to answer but also how to present it. Meanwhile, *"2010-09"* is more specific than *"September"*.

Overall, the proposed model improves reasoning by enhancing the LLM ability to understand complex questions. As well, the self-improvement strategies further boost the accuracy of retrieval and temporal reasoning.

Appendix D.9 Error analysis

We also conduct error analysis to point out the challenges(Figure D6). 1) Temporal Adjacency Failure. While the model correctly identifies the reference time anchor, it struggles to identify the immediate successor or predecessor in a dense list of facts. It often retrieves an event that satisfies the "after" constraint but is not the first one chronologically. For example, for the question "After Tony Blair, which country was the first to express an interest in cooperating with Iraq?", the model correctly identifies Tony Blair's action in May 2006. However, it selects China (2008) as the answer, overlooking intermediate events in late 2006 found in the fact list. 2) Retrieval Failure/Hallucination. When the retrieval module fails to find the necessary facts, the reasoning module occasionally exhibits wrong answers. Instead of outputting "unknown" or an empty set, it attempts to infer a plausible answer from irrelevant facts or timestamps present in the context. For example, the question asks about a specific threat from a "Somali criminal". The retrieved facts contain no information regarding Somali criminals. The model explicitly notes in its reasoning trace that "there is no information about Somali criminals," yet it still hallucinates a numeric answer "2006" based on the dates of unrelated consultations in the context. 3) Entity Granularity Mismatch. The model occasionally confuses the granularity of the target entity, responding with a country name when a specific person or organization is required. For example, the question asks "With whom did Catherine Ashton last wish to meet...?", which implies a person. The model correctly traces the timeline but answers "China" rather than the specific individual involved in the corresponding fact, e.g., "Wen Jiabao".

References

1 Jia Z, Abujabal A, Saha Roy R, et al. Tempquestions: A benchmark for temporal question answering. In: Proceedings of Companion Proceedings of the The Web Conference 2018, 2018. 1057–1062
2 Neelam S, Sharma U, Karanam H, et al. Sygma: System for generalizable modular question answering overknowledge bases, 2021
3 Mansouri B. Survey of abstract meaning representation: Then, now, future, 2025
4 Kannen N, Sharma U, Neelam S, et al. Best of both worlds: Towards improving temporal knowledge base question answering via targeted fact extraction. In: Proceedings of Conference on Empirical Methods in Natural Language Processing, 2023
5 Ding W, Chen H, Li H, et al. Semantic framework based query generation for temporal question answering over knowledge graphs, 2023
6 Chen Z, Zhang Z, Li Z, et al. Self-improvement programming for temporal knowledge graph question answering. ArXiv, 2024, abs/2404.01720
7 Cao S, Shi J, Pan L, et al. Kqa pro: A dataset with explicit compositional programs for complex question answering over knowledge base, 2022
8 Saxena A, Chakrabarti S, Talukdar P P. Question answering over temporal knowledge graphs. In: Proceedings of Annual Meeting of the Association for Computational Linguistics, 2021
9 Shang C, Wang G, Qi P, et al. Improving time sensitivity for question answering over temporal knowledge graphs. ArXiv, 2022, abs/2203.00255
10 Chen Z, Liao J, Zhao X. Multi-granularity temporal question answering over knowledge graphs. In: Proceedings of Annual Meeting of the Association for Computational Linguistics, 2023
11 Sharma A, Saxena A, Gupta C, et al. Twirgcn: Temporally weighted graph convolution for question answering over temporal knowledge graphs. In: Proceedings of Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, 2023. 2049–2060
12 Schlichtkrull M, Kipf T N, Bloem P, et al. Modeling relational data with graph convolutional networks, 2017
13 Liu Y, Liang D, Li M, et al. Local and global: Temporal question answering via information fusion. In: Proceedings of Elkind E, editor, Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23, 2023. 5141–5149. Main Track
14 Huang R, Wei W, Qu X, et al. Joint multi-facts reasoning network for complex temporal question answering over knowledge graph. ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2024, pages 10331–10335
15 Liu Y, Liang D, Fang F, et al. Time-aware multiway adaptive fusion network for temporal knowledge graph question answering. In: Proceedings of ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2023. 1-5
16 He X, Tian Y, Sun Y, et al. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. ArXiv, 2024, abs/2402.07630
17 Gutierrez B J, Shu Y, Gu Y, et al. Hipporag: Neurobiologically inspired long-term memory for large language models. ArXiv, 2024, abs/2405.14831
18 Gao Y, Xiong Y, Gao X, et al. Retrieval-augmented generation for large language models: A survey, 2024
19 Xu D, Li X, Zhang Z, et al. Harnessing large language models for knowledge graph question answering via adaptive multi-aspect retrieval-augmentation. In: Proceedings of AAAI Conference on Artificial Intelligence, 2025
20 Li Z, Guo Q, Shao J, et al. Graph neural network enhanced retrieval for question answering of large language models. In: Proceedings of Chiruzzo L, Ritter A, Wang L, editors, Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), 2025. 6612–6633
21 Zhang Q, Chen S, Bei Y, et al. A survey of graph retrieval-augmented generation for customized large language models, 2025
22 Zhao R, Zhao F, Wang L, et al. Kg-cot: Chain-of-thought prompting of large language models over knowledge graphs for knowledge-aware question answering. In: Proceedings of International Joint Conference on Artificial Intelligence, 2024
23 Sun J, Xu C, Tang L, et al. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In: Proceedings of The Twelfth International Conference on Learning Representations, 2024
24 Trivedi H, Balasubramanian N, Khot T, et al. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions, 2023
25 Wei J, Wang X, Schuurmans D, et al. Chain-of-thought prompting elicits reasoning in large language models. In: Proceedings of Proceedings of the 36th International Conference on Neural Information Processing Systems, 2022
26 DeepSeek-AI, Guo D, Yang D, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025
27 OpenAI, :, Jaech A, et al. Openai o1 system card, 2024
28 Shao Z, Wang P, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024
29 Chen Z, Li D, Zhao X, et al. Temporal knowledge question answering via abstract reasoning induction. In: Proceedings of Annual Meeting of the Association for Computational Linguistics, 2023
30 Jia Z, Pramanik S, Saha Roy R, et al. Complex temporal question answering on knowledge graphs. In: Proceedings of Proceedings of the 30th ACM International Conference on Information & Knowledge Management, 2021. 792–802
31 Sharma A, Saxena A, Gupta C, et al. TwiRGCN: Temporally weighted graph convolution for question answering over temporal knowledge graphs. In: Proceedings of Vlachos A, Augenstein I, editors, Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, 2023. 2049–2060
32 García-Durán A, Dumančić S, Niepert M. Learning sequence encoders for temporal knowledge graph completion. In: Proceedings of Riloff E, Chiang D, Hockenmaier J, et al., editors, Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018. 4816–4821
33 Reimers N, Gurevych I. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In: Proceedings of Inui K, Jiang J, Ng V, et al., editors, Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019. 3982–3992