Temporal Evidence Chain for Temporal Knowledge Graph Question Answering with Large Language Models

Shihao Liu[1,2], Xiaofei Zhou[1,2]*, Bo Wang[3], Geyuan Zhang[1,2]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3]University of International Business and Economics, Beijing, China
{liushihao, zhouxiaofei, zhanggeyuan}@iie.ac.cn
wangbo@uibe.edu.cn

Abstract

Temporal Knowledge Graph Question Answering (TKGQA) aims to answer temporal questions using knowledge from Temporal Knowledge Graphs (TKGs). Existing LLM-based TKGQA methods typically utilize RAG-based or agent-based paradigms, yet both struggle to construct reliable temporal evidence chains. RAG-based approaches primarily rely on semantic retrieval to fetch question-relevant contexts but overlook the structural dependencies within TKGs, leading to broken evidence chains, whereas iterative agents are prone to error propagation during multi-step reasoning. To address these limitations, we propose TECQA, a framework designed to construct temporal evidence chains for LLM reasoning. Firstly, TECQA employs structure-guided subgraph retrieval to capture structural dependencies and intermediate reasoning paths. Subsequently, it utilizes a k-nearest temporal neighbor pruning strategy to filter irrelevant noise while strictly preserving the continuous local history surrounding critical events. Finally, the retained temporal neighbors are serialized by temporal proximity to explicitly reconstruct a coherent temporal evidence chain. Extensive experiments on MultiTQ and CronQuestions demonstrate that TECQA achieves state-of-the-art performance, outperforming strong baselines by 45.3% particularly on complex queries. Code is available at https://github.com/SimonsLiu/TECQA.

1 Introduction

Temporal Knowledge Graph Question Answering (TKGQA) is pivotal for enabling intelligent systems to reason over evolving knowledge, with applications ranging from geopolitical analysis to financial monitoring (Chen et al., 2023; Su et al., 2025). Unlike static knowledge graphs, Temporal Knowledge Graphs (TKGs) incorporate timestamps into facts, requiring models to handle complex time-sensitive queries like *"Which country was the first to visit Thailand after John Baird?"*.

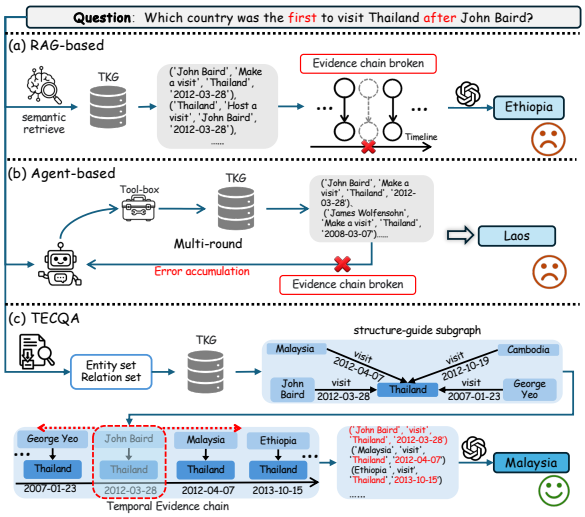


Figure 1: Comparison of (a) RAG-based methods, (b) agent-based methods, and (c) TECQA.

Early research primarily relied on semantic parsing (Jia et al., 2018; Neelam et al., 2021) or TKG embeddings (Saxena et al., 2021a; Chen et al., 2022), yet these methods are constrained by high annotation costs and limited semantic understanding, respectively. Recent advances integrate Large Language Models (LLMs) into TKGQA to exploit their strong reasoning ability. Existing LLM-based methods can be broadly grouped into two categories: (1) RAG-based paradigms, which encompass semantic retrieval (Qian et al., 2024) and decomposition-enhanced frameworks (Gong et al., 2025) that decompose complex queries into sub-questions; (2) agent-based paradigms (Chen et al., 2024; Hu et al., 2025), which employ LLMs as autonomous agents to interact with TKGs through iterative tool invocation.

However, although LLM-based methods excel on simple queries, they struggle with complex

*Corresponding author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 29515–29529

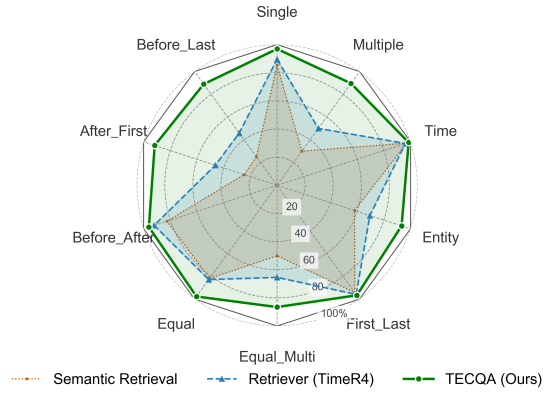


Figure 2: Answer Recall@20 across question categories. We compare TECQA against a standard semantic retriever[1] and retriever-TimeR[4] (fine-tuned on temporal facts). While semantic retrieval and TimeR[4] saturate at 68% and 78% respectively, TECQA achieves 94%.

queries requiring timeline-aware multi-step inference, primarily due to their failure to construct timeline-consistent evidence chains. As shown in Figure 1 (a), RAG-based methods retrieve and semantically match individual quadruples, thereby overlooking the explicit structural dependencies among them. Such a retrieval process often produces fragmented chains with missing crucial intermediate facts. These missing facts undermine the coherence and continuity of the evidence chain, leading to reduced answer recall. As shown in Figure 2, semantic retrieval plateaus at 78% recall even with temporal fine-tuning (Qian et al., 2024). This recall bottleneck is particularly pronounced for complex multiple queries, where limited recall directly constrains TKGQA performance. Conversely, as illustrated in Figure 1 (b), agent-based methods utilize an iterative exploration process vulnerable to error propagation, wherein a single erroneous tool invocation can cascade and prematurely terminate the search trajectory, leading to a failure to yield an answer.

To address these limitations, we propose **TECQA** (**T**emporal **E**vidence **C**hain for Temporal Knowledge Graph **Q**uestion **A**nswering), a framework designed to perform temporal evidence chain construction for LLM-based TKGQA. TECQA explicitly constructs timeline-consistent evidence chains that preserve relational connectivity to enable multi-step inference. Specifically, to address the fragmented chains caused by semantic retrieval, TECQA initiates with structure-guided retrieval. Expanding a subgraph via explicit entity-relation connectivity allows the framework to capture crucial intermediate hops missed by semantic retrieval, achieving 99.33% answer recall. Subsequently, to filter noise while preserving timeline coherence, the framework employs a k-nearest temporal neighbor pruning strategy that retains the continuous local history around critical anchors, effectively capturing the immediate event history required to resolve relative time constraints (e.g., before, after). As shown in Figure 2, such pruning maintains a high answer recall of 94.12% while condensing the context to an average of only 20 quadruples. Finally, TECQA serializes the facts based on temporal proximity to explicitly construct a compact, timeline-consistent evidence chain, serving as a plug-and-play module for accurate LLM inference.

Our contributions are summarized as follows:

- We analyze LLM-based TKGQA to reveal a critical limitation: existing methods suffer from fragmented or missing temporal contexts and fail to explicitly construct reasoning chains over event timelines.
- We propose TECQA, which integrates structure-guided retrieval, temporal anchor pruning, and proximity-based serialization to construct timeline-consistent evidence chains for accurate LLM reasoning.
- Extensive experiments on two widely-used benchmarks demonstrate that TECQA achieves state-of-the-art performance, significantly outperforming baselines on complex queries.

2 Related Work

2.1 Traditional TKGQA Methods

Semantic parsing-based methods convert natural language questions into executable logical forms via manually designed operators. Representative works such as TEQUILA (Jia et al., 2018), SYGMA (Neelam et al., 2021), and SF-TQA (Ding et al., 2023) explicitly model temporal constraints via symbolic representations. Despite their interpretability, these approaches require substantial manual engineering and are difficult to adapt to new schemas or domains.

TKG embedding-based methods encode questions and candidate answers into continuous vec-

[1]The embedding model utilizes bge-base-en-v1.5. https://huggingface.co/BAAI/bge-base-en-v1.5

tor spaces to select answers via similarity ranking. CronKGQA (Saxena et al., 2021a) formulates the task as temporal KG completion, while TempoQR (Mavromatis et al., 2021a) and MultiQA (Chen et al., 2023) enhance representations using time-aware components and multi-granular information, respectively. Other works incorporate graph neural networks for structural encoding (Liu et al., 2023; Sharma et al., 2022). Although ensuring high execution rates, such methods typically lack the reasoning depth required for complex temporal queries (Gong et al., 2025).

2.2 LLM-based TKGQA Methods

RAG-based methods ground LLM outputs on external knowledge to mitigate hallucinations. TimeR^4 (Qian et al., 2024) employs a retrieve-rewrite-rerank pipeline to explicitly model temporal constraints, whereas RTQA (Gong et al., 2025) utilizes recursive thinking to decompose multi-constraint queries hierarchically. However, relying on semantic retrievers that overlook structural dependencies often yields incomplete evidence chains lacking intermediate facts, creating a recall bottleneck that strictly constrains performance on complex queries.

agent-based methods empower LLMs to navigate TKGs via tool invocation and planning. ARI (Chen et al., 2024) distills reasoning methodologies from historical trajectories, and TempAgent (Hu et al., 2025) extends the ReAct (Yao et al., 2023) framework with a temporal toolbox for multi-granularity filtering. Despite inherent flexibility, iterative exploration remains vulnerable to error propagation.

3 Preliminaries

Temporal knowledge graph. A 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 $\mathcal{P}$ with timestamps $\mathcal{T}$. The quadruple set $\mathcal{F} = \{(s, p, o, t) \mid s \in \mathcal{E},\ p \in \mathcal{P},\ o \in \mathcal{E},\ t \in \mathcal{T}\}$ represents the temporal facts, where s and o are subject and object, respectively, and p is the predicate between s and o at timestamp t.

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

Temporal Anchor. A temporal anchor is a critical time reference point derived from the question q that serves as the starting node of the evidence chain for answering the question. Anchors can be categorized into two types: (1) *Explicit Anchors*, which are specific timestamps (e.g., "2015", "April 2005") directly mentioned in the question text; and (2) *Implicit Anchors*, which are the timestamps associated with specific events mentioned in the question (e.g., the date when "John Baird visited Thailand") that act as hidden temporal constraints.

4 Method

4.1 Overview

Figure 3 provides an overview of the TECQA framework, which is designed to reconstruct a timeline-consistent evidence chain. The pipeline operates in three consecutive stages: (1) Structure-Guided Subgraph Construction (Section 4.2) retrieves a high-recall candidate set via explicit structural constraints, ensuring the inclusion of potential evidence paths; (2) Temporal Evidence Chain Construction (Section 4.3) identifies pivotal time anchors, filters noise via neighbor pruning, and explicitly reconstructs the timeline topology through proximity-based serialization; and (3) LLM Reasoning (Section 4.4) utilizes the constructed chain to generate the final answer. The detailed algorithmic procedure is presented in Algorithm 1 (see Appendix B).

4.2 Structure-Guided Subgraph Construction

4.2.1 Structural Constraint Parsing

To bridge the semantic gap between natural language queries and the rigid schema of the TKG, we employ a two-stage parsing strategy to map the question q into explicit structural constraints.

LLM-based Extraction. We initiate the process by employing an LLM to parse the query q and extract mentions of entities and relations. The model is explicitly prompted to identify the main entity (e_{main}), defined as the pivot node that directly connects to the potential answer. Given an instruction template $\mathcal{I}_{\text{parse}}$ (detailed in Appendix F.1), the extraction process is formalized as:

$$(\mathcal{E}_{\text{raw}}, \mathcal{R}_{\text{raw}}, e^{\text{raw}}_{\text{main}}) \leftarrow \text{LLM}(q, \mathcal{I}_{\text{parse}}) \quad (1)$$

where $\mathcal{E}_{\text{raw}}$ and $\mathcal{R}_{\text{raw}}$ are the sets of extracted entity and relation mentions, and $e^{\text{raw}}_{\text{main}} \in \mathcal{E}_{\text{raw}}$ is the identified main entity mention.

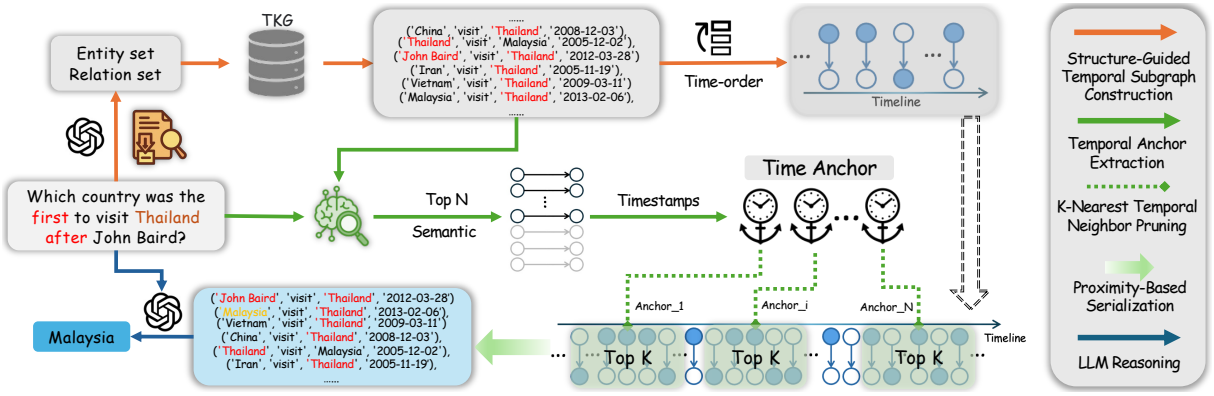


Figure 3: Overview of the TECQA framework. The process unfolds in three stages: Structure-Guided Subgraph Construction (in yellow), Temporal Evidence Chain Construction (in green), and LLM Reasoning (in blue).

Schema Grounding. Since raw mentions may not strictly match the TKG's vocabulary, we perform schema grounding to map them to the canonical forms in the graph. We define a grounding function $\text{Ground}(m, \mathcal{S})$ that maps a mention m to the most semantically similar element in a target schema set $\mathcal{S}$ based on the cosine similarity of their dense embeddings. We obtain the grounded constraints $\mathcal{E}_q$, $\mathcal{R}_q$, and the canonical main entity e_{main} as follows:

$$\begin{aligned} \mathcal{E}_q &\leftarrow \{\text{Ground}(e, \mathcal{E}_{\text{KG}}) \mid e \in \mathcal{E}_{\text{raw}}\} \\ \mathcal{R}_q &\leftarrow \{\text{Ground}(r, \mathcal{R}_{\text{KG}}) \mid r \in \mathcal{R}_{\text{raw}}\} \\ e_{\text{main}} &\leftarrow \text{Ground}(e^{\text{raw}}_{\text{main}}, \mathcal{E}_{\text{KG}}) \end{aligned} \tag{2}$$

where $\mathcal{E}_{\text{KG}}$ and $\mathcal{R}_{\text{KG}}$ denote the complete set of entities and relations in $\mathcal{G}$. This step ensures that subsequent retrieval targets the correct nodes and edges within the graph schema.

4.2.2 Subgraph Retrieval

We retrieve a candidate fact set $\mathcal{F}_{\text{cand}}$ from $\mathcal{G}$ by imposing the extracted structural constraints. To facilitate multi-hop reasoning, it is essential to capture not only the direct history of the main entity but also the intermediate interactions among auxiliary entities.

Let $\mathcal{E}_{\text{other}} = \mathcal{E}_q \setminus \{e_{\text{main}}\}$ denote the set of auxiliary entities. The candidate set is defined as the union of two subsets:

$$\mathcal{F}_{\text{cand}} = \mathcal{F}_{\text{main}} \cup \mathcal{F}_{\text{context}} \tag{3}$$

Here, $\mathcal{F}_{\text{main}}$ comprises facts anchored to the main entity, while $\mathcal{F}_{\text{context}}$ encompasses interactions strictly between auxiliary entities:

$$\begin{aligned} \mathcal{F}_{\text{main}} = \{(s,p,o,t) \in \mathcal{G} \mid \\ (s = e_{\text{main}} \vee o = e_{\text{main}}) \wedge p \in \mathcal{R}_q\} \end{aligned} \tag{4}$$

$$\begin{aligned} \mathcal{F}_{\text{context}} = \{(s,p,o,t) \in \mathcal{G} \mid \\ \{s,o\} \subseteq \mathcal{E}_{\text{other}} \wedge p \in \mathcal{R}_q\} \end{aligned} \tag{5}$$

We denote this comprehensive collection as the structure-guided temporal subgraph $\mathcal{F}_q$. At this stage, $\mathcal{F}_q$ serves as a high-recall candidate pool containing the global history of relevant entities.

4.3 Temporal Evidence Chain Construction

While the subgraph $\mathcal{F}_q$ ensures high recall, it inevitably includes numerous topically related but temporally irrelevant facts. To distill a precise evidence chain, we employ a three-step construction process: anchor extraction, neighbor pruning, and proximity-based serialization.

4.3.1 Temporal Anchor Extraction

We define the temporal anchor set $\mathcal{T}_{\text{anchor}}$ as the collection of critical reference timestamps derived from the question. Crucially, these anchors serve as the starting points for constructing the evidence chain required in temporal multi-hop reasoning. To handle both explicit and implicit temporal constraints, we extract anchors from two sources:

Explicit Anchors ($\mathcal{T}_{\text{exp}}$). We employ regular expressions to identify and extract explicit time mentions, such as specific years or dates, directly from the question q. These extracted values are subsequently normalized to the ISO 8601[2] format

[2] https://www.iso.org/iso-8601-date-and-time-format.html

(yyyy-mm-dd) to constitute the explicit anchor set $\mathcal{T}_{\text{exp}}$.

Implicit Anchors ($\mathcal{T}_{\text{imp}}$). For queries referencing events without explicit dates (e.g., "*after the election*"), we infer latent timestamps via semantic matching. Specifically, we compute the semantic similarity between the query q and the textual form of each fact $f \in \mathcal{F}_q$. The timestamps of the top-N most similar facts are selected as implicit anchors:

$$\mathcal{T}_{\text{imp}} = \Big\{ t_i \;\Big|\; (s, r, o, t_i) \in \text{Top-}N\big(\mathcal{F}_q, \text{sim}\big(q, \text{text}(f)\big)\big) \Big\}. \tag{6}$$

4.3.2 k-nearest temporal neighbor pruning

To capture the local history surrounding the critical time anchors, we focus on retaining the immediate context. For each anchor $t \in \mathcal{T}_{\text{anchor}}$, we employ a bidirectional search within the subgraph $\mathcal{F}_q$ to identify the K temporally closest facts. By simultaneously considering neighbors preceding and succeeding the anchor, this strategy ensures comprehensive coverage of diverse temporal relations, such as "*before*", "*after*", and "*during*".

Formally, the local evidence neighborhood $\mathcal{N}_K(t)$ is constructed as:

$$\mathcal{N}_K(t) = \text{Top-}K\big(\{f_i \in \mathcal{F}_q\}, |t_i - t|\big) \tag{7}$$

where $|t_i - t|$ denotes the absolute temporal distance between the timestamp t_i of fact f_i and the anchor t. By prioritizing facts based on strict temporal adjacency, this selection criterion naturally aligns with the local event density, ensuring that the context window captures the most immediate and relevant history around each anchor.

The final pruned subgraph $\tilde{\mathcal{F}}_q$ is obtained by unifying these local neighborhoods:

$$\tilde{\mathcal{F}}_q = \bigcup_{t \in \mathcal{T}_{\text{anchor}}} \mathcal{N}_K(t) \tag{8}$$

4.3.3 Proximity-Based Serialization

We serialize the subgraph $\tilde{\mathcal{F}}_q$ into the final evidence chain $\mathcal{S}_{\text{prox}}$ by prioritizing temporal proximity. For each fact $f_i = (s, r, o, t_i) \in \tilde{\mathcal{F}}_q$, we compute its proximity score $\delta(f_i)$ as the minimum absolute distance to the anchor set:

$$\delta(f_i) = \min_{t_a \in \mathcal{T}_{\text{anchor}}} |t_i - t_a| \tag{9}$$

We construct $\mathcal{S}_{\text{prox}}$ by sorting all facts in ascending order of their proximity scores. This arrangement ensures that events temporally closest to the critical anchors are processed first, effectively simulating an outward-propagating reasoning path that guides the LLM from known reference points into the surrounding historical context.

4.4 LLM Reasoning

Finally, we employ the LLM $\mathcal{M}$ to infer the answer a by reasoning over the proximity-ordered evidence $\mathcal{S}_{\text{prox}}$ under instruction $\mathcal{I}$ and query q:

$$a = \arg\max_{y} P_{\mathcal{M}}(y \mid \mathcal{I}, \mathcal{S}_{\text{prox}}, q) \tag{10}$$

This proximity-driven ordering imposes a temporal inductive bias, enabling the model to effectively trace sequential dependencies originating from the anchor. Prompt templates are provided in Appendix F.2.

5 Experiment

In this section, we seek to answer the following research questions: **Q1:** How does TECQA compare with existing methods? **Q2:** How do different components contribute to TECQA? **Q3:** How do hyperparameters affect TECQA? **Q4:** How does TECQA generalize across different LLMs? **Q5:** Does TECQA preserve evidence chain integrity? **Q6:** What do case studies and error analysis reveal about TECQA?

5.1 Experimental Setup

Datasets and Metrics. We evaluate TECQA on MultiTQ (Chen et al., 2023), comprising 500K pairs with multiple granularities, and CronQuestions (Saxena et al., 2021a), featuring 410K questions involving time intervals, utilizing Hits@1 as the primary metric. Detailed statistics are provided in Appendix A.

Baselines. Our comparison suite spans three paradigms: (1) Pre-trained Language Models (PLMs), including BERT (Devlin et al., 2019) and ALBERT (Lan et al., 2020); (2) Traditional TKGQA Methods, covering embedding-based and parsing-based approaches such as CronKGQA (Saxena et al., 2021b), TempoQR (Mavromatis et al., 2021b), MultiQA (Chen et al., 2023), LGQA (Liu et al., 2023), and QC-MHM (Xue et al., 2024); and (3) LLM-based Frameworks, including agent-based paradigms like ARI (Chen et al., 2024) and TempAgent (Hu et al., 2025), as well as RAG-based methods like

Table 1: Performance comparison of baselines and TECQA on Hits@1 across various question types and answer types on MultiTQ and CronQuestions testset. The best and second best results are marked in bold and underlined, respectively.

Model	MULTITQ					CRONQUESTIONS				
	Overall	Question Type		Answer Type		Overall	Question Type		Answer Type	
		Single	Multiple	Entity	Time		Simple	Complex	Entity	Time
BERT (Devlin et al., 2019)	0.083	0.092	0.061	0.101	0.040	0.243	0.249	0.239	0.277	0.179
ALBERT (Lan et al., 2020)	0.108	0.116	0.086	0.139	0.032	0.248	0.255	0.235	0.279	0.177
CronKGQA (Saxena et al., 2021b)	0.279	0.134	0.134	0.328	0.156	0.647	0.987	0.392	0.699	0.549
TempoQR (Mavromatis et al., 2021b)	–	–	–	–	–	0.918	0.990	0.864	0.926	0.903
MultiQA (Chen et al., 2023)	0.293	0.347	0.159	0.349	0.157	–	–	–	–	–
LGQA (Liu et al., 2023)	–	–	–	–	–	0.969	0.992	0.945	0.962	0.966
QC-MHM (Xue et al., 2024)	–	–	–	–	–	0.971	0.992	0.946	0.962	0.966
ARI (Chen et al., 2024)	0.380	0.680	0.210	0.394	0.344	0.707	0.860	0.570	0.660	0.800
GenTKGQA (Gao et al., 2024)	–	–	–	–	–	0.978	**0.999**	0.962	0.967	**0.990**
TimeR^4 (Qian et al., 2024)	0.728	0.887	0.335	0.639	**0.945**	0.856	0.963	0.764	0.788	0.971
TempAgent (Hu et al., 2025)	0.702	0.857	0.316	0.624	0.870	0.842	0.895	0.640	0.805	0.921
RTQA (Gong et al., 2025)	0.765	**0.902**	0.424	0.692	0.942	–	–	–	–	–
TECQA	**0.811**	0.889	**0.616**	**0.753**	0.939	**0.980**	0.995	**0.969**	**0.988**	0.965

TimeR^4 (Qian et al., 2024), RTQA (Gong et al., 2025).

Implementation Details. We utilize Gemini-2.5-flash (Comanici et al., 2025) for structural constraint parsing, bge-base-en-v1.5 (Xiao et al., 2023) for grounding, and Qwen3-8B (Yang et al., 2025) for reasoning (temperature fixed at 0). For dataset-specific processing, we normalize MultiTQ timestamps to ISO 8601 format; for CronQuestions, we compute the representative timestamp of intervals using the midpoint formula $t_m = (t_s + t_e)/2$. Based on sensitivity analysis, we configure the neighbor count $K = 40$ for both datasets, with implicit anchors $N = 2$ for MultiTQ and $N = 5$ for CronQuestions.

5.2 Main Results (Q1)

Table 1 demonstrates that TECQA achieves state-of-the-art results across both benchmarks, securing Overall Hits@1 scores of 0.811 on MultiTQ and 0.980 on CronQuestions. This consistent superiority validates the effectiveness of our framework over existing paradigms: unlike RAG-based methods which are limited by semantic recall bottlenecks, and agent-based methods which suffer from error propagation during iterative execution, TECQA explicitly constructs timeline-consistent evidence chains that enable accurate LLM reasoning.

Crucially, TECQA demonstrates superior effectiveness in handling complex queries within the *Multiple* category of MultiTQ, which involves nested constraints such as "after_first". Achieving a Hits@1 of 0.616, TECQA significantly outperforms strong baselines. Such a substantial margin underscores the necessity of an explicitly connected evidence chain over logical decomposition. By reconstructing the temporal topology, our framework enables precise in-context comparisons along a continuous, proximity-ordered timeline, effectively resolving multi-hop constraints where fragmented retrieval or iterative planning fail.

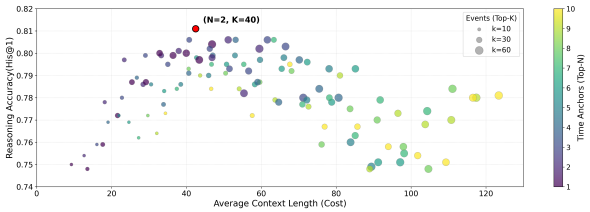
Table 2: Ablation study on MultiTQ and CronQuestions.

Model	MultiTQ			CronQuestions		
	Overall	Multiple	Single	Overall	Complex	Simple
TECQA	**0.811**	**0.616**	**0.889**	**0.980**	**0.969**	**0.995**
w/o SG	0.657	0.172	0.835	0.760	0.701	0.810
w/o KNTN	0.726	0.291	0.882	0.938	0.900	0.983
w/o PS	0.742	0.425	0.858	0.961	0.939	0.987

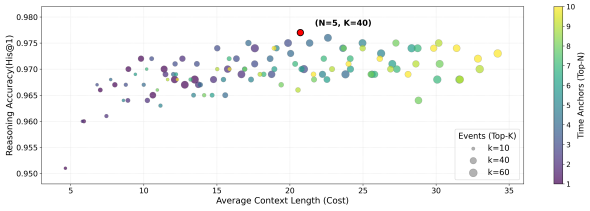
5.3 Ablation Studies (Q2)

To verify the contribution of each component, we conduct comprehensive ablation studies on both MultiTQ and CronQuestions. We evaluate TECQA against three distinct variants: w/o SG (without Structure-Guided Retrieval), w/o KNTN (without k-nearest temporal neighbor pruning), and w/o PS (without Proximity-Based Serialization). Detailed configurations are provided in Appendix C.

Impact of Structure-Guided Retrieval. Replacing structure-guided retrieval with semantic retrieval causes a substantial performance degradation across both datasets, most notably on *Multiple* questions in MultiTQ. This decline confirms that semantic retrieval, lacking explicit entity-relation



(a) MultiTQ



(b) CronQuestions

Figure 4: Hyperparameter sensitivity analysis on MultiTQ and CronQuestions. The red marker highlights the optimal configuration.

anchoring, fails to assemble the complete evidence chains required for complex constraints, instead identifying only simple, single-hop associations.

Impact of KNTN Pruning. Replacing temporal K-NN pruning with semantic selection on the structure-guided subgraph leads to a significant performance decline on *Multiple* queries. While the initial subgraph ensures a high-recall candidate set, semantic pruning remains suboptimal for TKGQA as it prioritizes topically similar but temporally discontinuous facts. This selection bias hinders the construction of a continuous evidence chain along the timeline, failing to capture the temporally adjacent events required for resolving fine-grained sequential dependencies.

Impact of Proximity-Based Serialization. Replacing proximity-based serialization with a randomized fact order leads to a substantial accuracy drop, particularly on *Multiple* queries in MultiTQ where performance falls to 0.425. This result confirms that presenting evidence in a proximity-ordered sequence is critical for accurate response generation.

5.4 Hyperparameter Sensitivity and Efficiency Analysis (Q3)

To identify the optimal configuration for TECQA, we conduct a comprehensive grid search on the validation set targeting two key hyperparameters: the number of implicit anchors N and the temporal neighbor count K, varying N from 1 to 10 and K from 5 to 60. Figure 4 illustrates the trade-off between reasoning accuracy and computational cost, measured by average context length.

Optimal Configurations. Distinct optimal settings emerge driven by intrinsic dataset characteristics. MultiTQ achieves optimal accuracy with $N = 2$ and $K = 40$, whereas CronQuestions requires a higher anchor density ($N = 5$) with the same neighbor count. This divergence stems from the interval-based structure of CronQuestions necessitating broader anchor coverage to capture event spans, in contrast to the denser point-based facts of MultiTQ where fewer anchors suffice.

Impact of Context Length. The analysis reveals a clear trade-off between accuracy and context size. Increasing K initially boosts accuracy by retrieving richer contextual details but eventually leads to diminishing returns or performance degradation. This trend aligns with the "lost-in-the-middle" phenomenon, where excessive context length introduces irrelevant information that dilutes the reasoning focus of the LLM.

Table 3: Cross-model evaluation decoupling retrieval quality from reasoning capability.

Reasoner	Retrieval	Overall	Question Type		Answer Type	
			Single	Mult.	Ent.	Time
Llama2-7B	TimeR^4	0.391	**0.442**	0.266	0.370	**0.442**
Generalist	**TECQA**	**0.408**	0.415	**0.388**	**0.408**	0.407
Llama2-7B	TimeR^4	0.728	**0.887**	0.335	0.639	**0.945**
Fine-tuned	**TECQA**	**0.731**	0.884	**0.353**	**0.680**	0.843
Qwen3-8B	TimeR^4	0.533	0.589	0.396	0.496	0.627
Generalist	**TECQA**	**0.778**	**0.847**	**0.589**	**0.711**	**0.926**

5.5 Impact of Model Capability and Scalability (Q4)

To assess the interplay between model capabilities and our framework, we conduct a decoupled analysis on the more challenging MultiTQ dataset.

Interplay between Retrieval and Reasoning. Table 3 reveals a crucial dependency between retrieval strategy and model capability. TimeR^4 exhibits a significant performance drop when switching from its fine-tuned version to the generalist Qwen3-8B (Yang et al., 2025), suggesting its reliance on parametric knowledge to mitigate retrieval noise. In contrast, TECQA achieves a dramatic 27.8% lead over TimeR^4 with Qwen3-8B, yet shows only marginal gains on the weaker Llama2-7B (Touvron et al., 2023) base model. This disparity confirms that TECQA's structured evidence

Table 4: Performance comparison of TECQA across different LLM backbones on MultiTQ, sorted by Overall Score.

Model	Overall	Question Type		Answer Type	
		Single	Multiple	Entity	Time
Non-Reasoning Models					
Qwen3-4B-Instruct	0.571	0.708	0.235	0.578	0.554
Llama-3-8B-Instruct	0.616	0.732	0.299	0.548	0.766
Qwen3-8B (no thinking mode)	0.627	0.772	0.231	0.597	0.692
DeepSeek-v3-0324	0.774	0.902	0.425	0.705	0.926
Reasoning Models					
Qwen3-4B-Thinking-2507	0.806	0.875	0.637	0.755	0.933
Qwen3-8B (thinking mode)	0.811	0.889	0.616	0.753	0.939
Gemini-2.5-Pro	0.836	0.893	0.682	0.786	0.948
DeepSeek-R1-0528	0.838	0.918	0.619	0.792	0.939
GPT-5-thinking	0.842	0.901	0.682	0.786	0.968
Gemini-3-Pro-Preview	**0.854**	**0.915**	**0.689**	**0.803**	**0.968**

chain functions as a sophisticated reasoning context requiring a capable reasoner to traverse the proximity-ordered timeline, a capability distinct from simple knowledge recall.

Effectiveness of Reasoning-Enhanced Models. We extend our evaluation to diverse LLM backbones in Table 4 to assess scalability. Results indicate that reasoning models consistently outperform non-reasoning counterparts, with smaller reasoning models (e.g., Qwen3-4B-Thinking) even surpassing massive generalist models (e.g., DeepSeek-V3). This pattern confirms that TECQA provides a sophisticated reasoning context requiring explicit deduction capabilities to fully exploit the timeline-consistent topology. Consequently, TECQA empowers smaller reasoning models to achieve state-of-the-art performance while scaling synergistically with frontier models like Gemini-3-Pro-Preview.

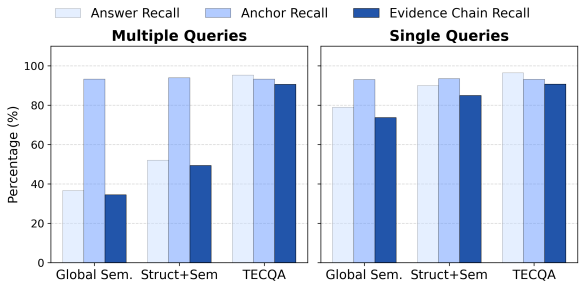


Figure 5: Comparison of Evidence Chain Integrity.

5.6 Evidence Chain Integrity Analysis (Q5)

To verify whether retrieved contexts form valid reasoning paths, we benchmark TECQA against "Global Semantic Retrieval" and "Structure-Guided + Semantic Pruning". Figure 5 illustrates performance on complex and single queries using three metrics: Answer Recall, Anchor Recall, and Evidence Chain Recall. Detailed metric definitions are provided in Appendix D.

The left panel shows that semantic-based baselines suffer a sharp drop in Evidence Chain Recall on complex queries despite maintaining high Anchor Recall. This disparity confirms that semantic similarity can locate the starting anchor but often fails to retrieve the temporally associated answer, resulting in fragmented evidence chains. In contrast, TECQA utilizes temporal K-NN pruning to bridge the anchor and answer, ensuring the context completeness required for temporal comparison. Such an advantage diminishes on single queries where semantic retrieval suffices for identifying simple one-hop relations.

5.7 Case Study and Error Analysis (Q6)

To illustrate how TECQA succeeds under both implicit and explicit temporal constraints, we present two representative successful cases in the main text, while detailed failure examples are provided in Appendix E. In the implicit-constraint case (Table 5), TECQA identifies the latent anchor date of January 5, 2006 via semantic matching over the structure-guided subgraph. Proximity-based serialization then places the target answer *Jack Straw* immediately around the anchor, allowing the LLM to resolve the *after* constraint against nearby distractors. In the explicit-constraint case (Table 6), TECQA normalizes "April 2005" into a concrete anchor and preserves a compact cluster of locally relevant interactions within the target time window, enabling the LLM to recover a complete answer set.

We further analyze the MultiTQ test set containing 54,584 queries. Only 159 queries yield an empty subgraph, accounting for just 0.29% of the test set, and 86 of these failures are directly caused by incorrect main-entity extraction. In such cases, the failure occurs before temporal evidence chain construction begins, leaving the LLM with an empty context and increasing the risk of hallucination. A second failure mode arises on *first/last* queries, where the entity and relation may be extracted correctly but conservative anchor exploration and neighbor retention truncate the temporal span and exclude the true earliest or latest event. Appendix E presents representative examples of both failure modes.

Table 5: Successful case 1: Implicit temporal constraint resolved by anchor localization and proximity-based serialization.

Question: *After the Danish Ministry of Defence and Security, who was the first to visit Iraq?*	
Step 1	**Structure-Guided Subgraph Construction**
Goal	Retrieve a high-recall candidate pool via structural constraints.
Process	• **Extraction:** Entities: `Iraq`, `Danish Ministry`; Relation: `Make a visit`. • **Retrieval:** 1,065 facts retrieved. Context is chronologically unordered and noisy. **Representative facts:** Adil Abdul-Mahdi → Iraq (2015-05-15), China → Iraq (2010-02-18), *etc.*
Step 2	**Temporal Evidence Chain Construction**
Goal	Extract anchors, prune noise, and serialize facts into a timeline-consistent chain.
Process	**1. Temporal Anchor Extraction (Implicit):** ↪ Semantic matching identifies event timestamp: **Anchor = `2006-01-05`**. **2. K-Nearest Pruning & Proximity Serialization:** compute $\delta = \lvert t - \texttt{2006-01-05} \rvert$ and sort by proximity. **[Anchor, $\delta = 0$]** Danish Ministry → Iraq (2006-01-05) **[Answer, $\delta = 1$]** Jack Straw → Iraq (2006-01-06) $\delta = 1$ Iraq → Iran (2006-01-04) $\delta = 2$ Evan Bayh → Iraq (2006-01-07) $\delta = 2$ Peter Pace → Iraq (2006-01-03) *... (context strictly ordered by proximity) ...*
Step 3	**LLM Reasoning**
Goal	Generate the answer using the constructed evidence chain.
Process	**Input Prompt:** *...Based on the timeline above, who visited Iraq first after 2006-01-05?* **Output: Jack Straw**

Table 6: Successful case 2: Explicit temporal constraint resolved by timestamp normalization and local temporal pruning.

Question: *Who signed an agreement with China in April 2005?*	
Step 1	**Structure-Guided Subgraph Construction**
Goal	Retrieve a high-recall candidate subgraph consistent with the extracted entity and relation.
Process	• **Extraction:** Entity: `China`; Relation: `Sign agreement`. • **Retrieval:** 2,311 facts retrieved. Context spans 2005-2015 and is heavily redundant. **Representative facts:** China ↔ Japan (2005-09-20), Ethiopia ↔ China (2015-11-13), *etc.*
Step 2	**Temporal Evidence Chain Construction**
Goal	Normalize the explicit time expression and serialize the relevant local evidence chain.
Process	**1. Temporal Anchor Extraction (Explicit):** ↪ Regex extraction and normalization: "April 2005" → **Anchor = `2005-04-01`**. **2. K-Nearest Pruning & Proximity Serialization:** compute $\delta = \lvert t - \texttt{2005-04-01} \rvert$ and retain the local window. **[Ref Anchor]** Target date: 2005-04-01 **[Answer, $\delta = 6$]** China ↔ Colombia (2005-04-07) **[Answer, $\delta = 7$]** South Korea ↔ China (2005-04-08) **[Answer, $\delta = 7$]** Japan ↔ China (2005-04-08) **[Answer, $\delta = 10$]** Iran ↔ China (2005-04-11) **[Answer, $\delta = 14$]** Kuomintang ↔ China (2005-04-15) **[Answer, $\delta = 20$]** France ↔ China (2005-04-21) *... (pruned evidence chain preserves local cluster) ...*
Step 3	**LLM Reasoning**
Goal	Generate the answer set from the serialized evidence chain.
Process	**Input Prompt:** *...Who signed an agreement with China around 2005-04-01?* **Output: Colombia, South Korea, Japan, Iran, Kuomintang, France**

6 Conclusion

In this paper, we propose TECQA, a framework designed to perform temporal evidence chain construction for LLM-based TKGQA. Specifically, TECQA utilizes structure-guided retrieval to capture dependencies missed by semantic retrieval to ensure high recall, while employing k-nearest temporal neighbor pruning to filter noise and preserve local history. By serializing the retrieved facts into a timeline-consistent evidence chain, TECQA provides a coherent context that mitigates both the fragmentation of semantic retrievers and the instability of iterative agents. Extensive experiments on MultiTQ and CronQuestions demonstrate that TECQA achieves state-of-the-art performance, significantly outperforming strong baselines on complex queries. Additional analyses confirm that TECQA preserves high integrity of evidence chains and scales synergistically with stronger LLMs.

Limitations

Despite the strong performance of TECQA, several limitations warrant discussion. First, the framework depends on upstream structural parsing accuracy. The effectiveness of structure-guided retrieval hinges on the LLM's ability to precisely extract entities and map them to the TKG schema. Performance may degrade on ambiguous queries or out-of-domain entities where schema grounding fails, introducing noise into the initial candidate pool. Second, the k-nearest temporal neighbor pruning strategy operates on a temporal locality assumption. While effective for event-centric TKGQA, the method may underperform in scenarios requiring long-horizon reasoning, where the target answer is temporally distant from the anchor and lacks a dense chain of connecting events within the selected set of K neighbors. Lastly, our current evaluation is confined to structured TKGQA. Extending temporal evidence chain construction to unstructured text-based temporal reasoning or real-time streaming data remains a challenging direction for future research.

Acknowledgment

This work was supported by National Natural Science Foundation of China (No.62176252).

References

Ziyang Chen, Dongfang Li, Xiang Zhao, Baotian Hu, and Min Zhang. 2024. Temporal knowledge question answering via abstract reasoning induction. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4872–4889, Bangkok, Thailand. Association for Computational Linguistics.

Ziyang Chen, Jinzhi Liao, and Xiang Zhao. 2023. 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, Toronto, Canada. Association for Computational Linguistics.

Ziyang Chen, Xiang Zhao, Jinzhi Liao, Xinyi Li, and Evangelos Kanoulas. 2022. Temporal knowledge graph question answering via subgraph reasoning. *Knowledge-Based Systems*, 251:109134.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-Jiang Jiang, and 3416 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *Preprint*, arXiv:2507.06261.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025a. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 181 others. 2025b. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. Association for Computational Linguistics.

Wentao Ding, Hao Chen, Huayu Li, and Yuzhong Qu. 2023. Semantic Framework based Query Generation for Temporal Question Answering over Knowledge Graphs. *Preprint*, arxiv:2210.04490.

Yifu Gao, Linbo Qiao, Zhigang Kan, Zhihua Wen, Yongquan He, and Dongsheng Li. 2024. Two-stage Generative Question Answering on Temporal Knowledge Graph Using Large Language Models. *Preprint*, arxiv:2402.16568.

Alberto García-Durán, Sebastijan Dumancic, and Mathias Niepert. 2018. Learning sequence encoders for temporal knowledge graph completion. In *EMNLP*, pages 4816–4821. Association for Computational Linguistics.

Zhaoyan Gong, Juan Li, Zhiqiang Liu, Lei Liang, Huajun Chen, and Wen Zhang. 2025. RTQA : Recursive thinking for complex temporal knowledge graph question answering with large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 9864–9881, Suzhou, China. Association for Computational Linguistics.

Google DeepMind. 2025. Gemini 3 pro. Accessed: 2026-01-05.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Qianyi Hu, Xinhui Tu, Cong Guo, and Shunping Zhang. 2025. Time-aware ReAct agent for temporal knowledge graph question answering. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 6013–6024, Albuquerque, New Mexico. Association for Computational Linguistics.

Zhen Jia, Abdalghani Abujabal, Rishiraj Saha Roy, Jannik Strötgen, and Gerhard Weikum. 2018. TEQUILA: Temporal Question Answering over Knowledge Bases. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1807–1810, Torino Italy. ACM.

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. In *ICLR*. OpenReview.net.

Yonghao Liu, Di Liang, Mengyu Li, 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, IJCAI-23*, pages 5141–5149. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Costas Mavromatis, Prasanna Lakkur Subramanyam, Vassilis N. Ioannidis, Soji Adeshina, Phillip R. Howard, Tetiana Grinberg, Nagib Hakim, and George Karypis. 2021a. TempoQR: Temporal Question Reasoning over Knowledge Graphs. *Preprint*, arxiv:2112.05785.

Costas Mavromatis, Prasanna Lakkur Subramanyam, Vassilis N. Ioannidis, Soji Adeshina, Phillip R. Howard, Tetiana Grinberg, Nagib Hakim, and George Karypis. 2021b. Tempoqr: Temporal question reasoning over knowledge graphs. *Preprint*, arXiv:2112.05785.

Sumit Neelam, Udit Sharma, Hima Karanam, Shajith Ikbal, Pavan Kapanipathi, Ibrahim Abdelaziz, Nandana Mihindukulasooriya, Young-Suk Lee, Santosh Srivastava, Cezar Pendus, Saswati Dana, Dinesh Garg, Achille Fokoue, G. P. Shrivatsa Bhargav, Dinesh Khandelwal, Srinivas Ravishankar, Sairam Gurajada, Maria Chang, Rosario Uceda-Sosa, and 5 others. 2021. SYGMA: System for Generalizable Modular Question Answering OverKnowledge Bases. *Preprint*, arxiv:2109.13430.

OpenAI. 2025. Introducing GPT-5. Accessed: 2026-01-05.

Xinying Qian, Ying Zhang, Yu Zhao, Baohang Zhou, Xuhui Sui, and Xiaojie Yuan. 2025. Plan of knowledge: Retrieval-augmented large language models for temporal knowledge graph question answering. *Preprint*, arXiv:2511.04072.

Xinying Qian, Ying Zhang, Yu Zhao, Baohang Zhou, Xuhui Sui, Li Zhang, and Kehui Song. 2024. TimeR^4 : Time-aware retrieval-augmented large language models for temporal knowledge graph question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6942–6952, Miami, Florida, USA. Association for Computational Linguistics.

Apoorv Saxena, Soumen Chakrabarti, and Partha Talukdar. 2021a. Question Answering Over Temporal Knowledge Graphs. *Preprint*, arxiv:2106.01515.

Apoorv Saxena, Soumen Chakrabarti, and Partha Talukdar. 2021b. Question answering over temporal knowledge graphs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6663–6676, 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. *Preprint*, arxiv:2210.06281.

Miao Su, Zixuan Li, Zhuo Chen, Long Bai, Xiaolong Jin, and Jiafeng Guo. 2025. Temporal knowledge graph question answering: A survey. *Preprint*, arXiv:2406.14191.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

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

Chao Xue, Di Liang, Pengfei Wang, and Jing Zhang. 2024. Question calibration and multi-hop modeling for temporal question answering. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. *Preprint*, arXiv:2210.03629.

A Experimental Setup Details

In this section, we provide comprehensive details regarding the datasets, baseline comparisons, and implementation specifics used in our experiments.

A.1 Datasets

MULTITQ MultiTQ (Chen et al., 2023) is derived from the ICEWS05-15 dataset (García-Durán et al., 2018) and stands as the largest TKGQA dataset to date, containing 500K unique question-answer pairs. In this dataset, all facts are standardized as quadruples (s, r, o, t). A key advantage of MultiTQ is its rich semantic information, featuring a higher average number of relation types per entity compared to other TKGs. Furthermore, while ICEWS originally provides time information at a day granularity, MultiTQ generates questions with multiple temporal granularities (including years, months, and days), making it a significant resource for testing multi-granularity temporal reasoning. The distribution of its question categories is shown in Table 7.

CRONQUESTIONS CronQuestions (Saxena et al., 2021a) is another widely used benchmark utilizing a subset of Wikidata annotated with

Table 7: Statistics of question categories in MultiTQ.

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**	**58,979**	**54,584**

temporal information. Unlike MultiTQ, facts in CronQuestions are often represented as time intervals (e.g., ⟨Barack Obama, held position, President of USA, 2008, 2016⟩). Entities with both "start time" and "end time" annotations are transformed into an event format (e.g., ⟨WWII, significant event, occurred, 1939, 1945⟩). The dataset comprises a temporal knowledge graph with 125k entities and 328k facts, along with 410k natural language questions that require rigorous temporal reasoning. The statistical breakdown is presented in Table 8.

Table 8: Statistics of question categories in CronQuestions.

Category	Train	Dev	Test
Simple Questions			
Simple Entity	90,651	7,745	7,812
Simple Time	61,471	5,197	5,046
Complex Questions			
Before/After	23,869	1,982	2,151
First/Last	118,556	11,198	11,159
Time Join	55,453	3,878	3,832
By Answer Type			
Entity Answer	225,672	19,362	19,524
Time Answer	124,328	10,638	10,476
Total	**350,000**	**30,000**	**30,000**

A.2 Baselines

We evaluate TECQA against a diverse set of strong baselines categorized into three groups. The experimental results for these baselines are directly reported from their original papers:

- **Pre-trained Language Models (PLMs)**: We evaluate standard PLMs including BERT (Devlin et al., 2019) and ALBERT (Lan et al., 2020) to assess the performance of direct fine-tuning on TKGQA tasks.
- **Traditional TKGQA Methods**: This category includes CronKGQA (Saxena et al., 2021b), TempoQR (Mavromatis et al., 2021b), MultiQA (Chen et al., 2023), LGQA (Liu et al., 2023), and QC-MHM (Xue et al., 2024). These methods primarily rely on TKG embeddings or semantic parsing techniques.
- **LLM-based Frameworks**: We compare against state-of-the-art LLM-based approaches, including the agent-based frameworks ARI (Chen et al., 2024) and TempAgent (Hu et al., 2025), as well as retrieval-augmented generation methods like TimeR^4 (Qian et al., 2024), RTQA (Gong et al., 2025), and PoK (Qian et al., 2025).

A.3 Implementation Details

Model Configuration. To further evaluate framework scalability (as presented in Table 4), we extend our evaluation to a diverse suite of LLM backbones. This includes non-reasoning models: Qwen3-4B-Instruct (Yang et al., 2025), Llama-3-8B-Instruct (Grattafiori et al., 2024), and DeepSeek-V3 (DeepSeek-AI et al., 2025b); as well as reasoning-enhanced models: Qwen3-4B-Thinking (Yang et al., 2025), Gemini-2.5-Pro (Comanici et al., 2025), DeepSeek-R1 (DeepSeek-AI et al., 2025a), GPT-5-thinking (OpenAI, 2025), and Gemini-3-Pro-Preview (Google DeepMind, 2025).

B Algorithm 1

In this section, we provide the detailed pseudocode for the inference procedure of **TECQA**, as discussed in Section 4. Algorithm 1 formally outlines the three-stage framework: (1) *Structure-Guided Subgraph Construction*, which retrieves a high-recall candidate set; (2) *Temporal Evidence Chain Construction*, which filters noise and organizes the context into a timeline-consistent topology; and (3) *LLM Reasoning*, which utilizes the constructed chain to derive the final answer.

C Ablation Study Details

We define the three variants used in the ablation study as follows:

- **w/o SG** (without Structure-Guided): Replaces the structure-guided subgraph construction with a standard dense retrieval method based on semantic similarity (using bge-base-en-v1.5).

Algorithm 1: The Inference Procedure of TECQA

Input: TKG $\mathcal{G}$, Question q, LLM $\mathcal{M}$, Instruction $\mathcal{I}$, Hyperparameters N, K

Output: Predicted Answer a

// Phase 1: Structure-Guided Subgraph Construction

$(\mathcal{E}_{\text{raw}}, \mathcal{R}_{\text{raw}}, e^{\text{raw}}_{\text{main}}) \leftarrow \mathcal{M}(q, \mathcal{I}_{\text{parse}})$;

$\mathcal{E}_q, \mathcal{R}_q, e_{\text{main}} \leftarrow \text{Ground}(\mathcal{E}_{\text{raw}}, \mathcal{R}_{\text{raw}}, e^{\text{raw}}_{\text{main}}, \mathcal{G})$;

$\mathcal{F}_{\text{cand}} \leftarrow \text{Retrieve}(\mathcal{G}, \mathcal{E}_q, \mathcal{R}_q, e_{\text{main}})$;

$\mathcal{F}_q \leftarrow \mathcal{F}_{\text{cand}}$; // Unordered high-recall pool

// Phase 2: Temporal Evidence Chain Construction

$\mathcal{T}_{\text{exp}} \leftarrow \text{ExtractRegex}(q)$;

$\mathcal{T}_{\text{imp}} \leftarrow \{t_i \mid f_i \in \text{Top-}N(\mathcal{F}_q, \text{sim}(q, \text{text}(f)))\}$;

$\mathcal{T}_{\text{anchor}} \leftarrow \mathcal{T}_{\text{exp}} \cup \mathcal{T}_{\text{imp}}$;

$\tilde{\mathcal{F}}_q \leftarrow \emptyset$;

for $t \in \mathcal{T}_{\text{anchor}}$ **do**

  $\mathcal{N}_K(t) \leftarrow \arg\text{min-K}_{f_i \in \mathcal{F}_q}(|t_i - t|)$;

  $\tilde{\mathcal{F}}_q \leftarrow \tilde{\mathcal{F}}_q \cup \mathcal{N}_K(t)$;

$\mathcal{S}_{\text{prox}} \leftarrow \text{SortByProximity}(\tilde{\mathcal{F}}_q, \mathcal{T}_{\text{anchor}})$;

// Phase 3: LLM Reasoning

$a \leftarrow \mathcal{M}(\mathcal{I}, \mathcal{S}_{\text{prox}}, q)$;

return a;

- **w/o KNTN** (without k-nearest temporal neighbor pruning): Replaces the k-nearest temporal neighbor pruning strategy with a semantic Top-K selection, retaining facts solely based on their semantic similarity to the question.
- **w/o PS** (without Proximity-Based Serialization): Retains the same set of pruned facts but randomizes their order in the final prompt, removing the distance-based sorting to disrupt the explicit temporal topology.

D Evidence Chain Integrity Analysis Details

Baselines for Comparison. To isolate the impact of our temporal pruning strategy, we benchmark against two retrieval paradigms:

- **Global Semantic Retrieval:** A standard RAG baseline that retrieves the top-k facts directly from the entire TKG based solely on semantic similarity with the query, disregarding structural or temporal constraints.
- **Structure-Guided + Semantic Pruning:** An ablation variant that constructs the candidate subgraph using our structure-guided method but selects the final top-k facts via semantic similarity, replacing our K-NN temporal pruning.

Evaluation Metrics. Figure 5 reports the retrieval coverage of critical reasoning elements:

- **Answer Recall:** The proportion of questions where the retrieved context contains the ground truth answer entity or timestamp.
- **Anchor Recall:** The proportion of questions where the retrieved context contains the critical temporal reference point (Anchor) required to anchor the reasoning timeline.
- **Evidence Chain Recall:** The proportion of questions where the retrieved context simultaneously contains *both* the Anchor and the Answer. Given that complex queries in MultiTQ (e.g., *after_first*, *before_last*) primarily involve 2-step temporal comparisons, the co-occurrence of the reference anchor and the target answer constitutes the complete evidence chain required to resolve the temporal constraint.

E Case Study Details

This section provides two representative failure cases that complement the two successful cases shown in the main text.

E.1 Failure Case 1: Retrieval Collapse from Incorrect Main-Entity Extraction

Table 9 shows a representative error where the parser identifies the wrong pivot entity, causing the retrieved subgraph to become empty before evidence chain construction begins.

This case shows that main-entity identification is a hard prerequisite for TECQA. Once retrieval is rooted in an incorrect pivot node, the downstream stages cannot recover: high-recall retrieval, anchor matching, and temporal serialization all fail simultaneously, yielding a catastrophic rather than gradual error.

It also explains why this failure mode is particularly damaging for LLM-based reasoning. With an empty context, the model must rely on parametric memory instead of grounded evidence, substantially increasing hallucination risk. This example therefore highlights parser robustness—especially

Table 9: Failure case 1: Incorrect main-entity extraction causes complete retrieval collapse.

Question: *After China, with whom did the Indian Prime Minister first want to engage in diplomatic cooperation?*	
Step 1	**Structure-Guided Subgraph Construction**
Goal	Identify the main entity and retrieve a high-recall candidate subgraph.
Process	• **Extraction:** The LLM incorrectly identifies `Prime Minister Chaudhry` as the main entity. • **Retrieval:** Because the extracted pivot node is incorrect, the KG returns 0 relevant facts and the subgraph becomes empty. **Outcome:** `Subgraph = {}`
Step 2	**Temporal Evidence Chain Construction**
Goal	Match temporal anchors and serialize a valid local evidence chain.
Process	With an empty subgraph, no anchor can be matched and no temporal neighbors can be retained. The chain-construction stage therefore collapses before serialization.
Step 3	**LLM Reasoning**
Goal	Infer the answer from the constructed chain.
Process	Given an **[Empty Context]**, the LLM is forced to rely on parametric knowledge rather than retrieved evidence. **Output:** Incorrect answer (hallucination); **Ground Truth: Bhupinder Singh Hooda**

under ambiguous political titles or role mentions—as a key factor in reducing empty-subgraph failures.

E.2 Failure Case 2: Temporal Truncation on a Last Query

Table 10 presents a failure where the entity and relation are extracted correctly, but conservative temporal coverage excludes the true last event from the retained chain.

Table 10: Failure case 2: Limited temporal coverage truncates the evidence chain for a last query.

Question: *In which month did the African Union last ask Ethiopia?*	
Step 1	**Structure-Guided Subgraph Construction**
Goal	Retrieve a candidate pool that covers the full temporal span required by the query.
Process	• **Extraction:** The entity and relation are identified correctly. • **Retrieval:** Owing to limited anchor and neighbor budgets, the retrieved subgraph covers only 2005-10 to 2007-06. The true last event in 2008-12 is therefore excluded. **Outcome:** The retrieved context is locally relevant but temporally truncated.
Step 2	**Temporal Evidence Chain Construction**
Goal	Serialize the subgraph into a chain sufficient for resolving the ordinal constraint *last*.
Process	The system correctly serializes the retained facts, but the latest interaction with Ethiopia in the truncated chain occurs at 2006-12-28 rather than at the true endpoint in 2008-12.
Step 3	**LLM Reasoning**
Goal	Select the latest valid event from the evidence chain.
Process	The LLM reasons correctly over the incomplete chain and predicts the final month supported by the truncated evidence. **Output: 2006-12**; **Ground Truth: 2008-12**

Unlike Failure Case 1, this error does not stem from parsing. The retrieved subgraph is locally relevant, but local temporal pruning induces a coverage–precision trade-off: while emphasizing nearby evidence improves context compactness, it may exclude distant events that determine ordinal constraints such as *last*.

As a result, the LLM reasons correctly over the retained chain but still produces a globally incorrect answer because the true endpoint event is missing. The failure therefore reflects incomplete temporal coverage rather than faulty reasoning. This suggests that *first*/*last* queries may require more adaptive neighbor budgets or broader anchor expansion to reduce long-range truncation.

F Prompts

F.1 Prompts for entity set and relation set extraction

To transform natural language questions into structured graph constraints, we employ a sequential prompting pipeline. This process consists of three steps: (1) **Entity Extraction**, which identifies core mentions (people, organizations, locations) while filtering generic terms; (2) **Relation Extraction**, which maps the question's primary action to a standardized relation in the TKG schema using context-aware mapping rules; and (3) **Main Entity Identification**, which pinpoints the pivot entity directly connected to the target answer based on grammatical structure (e.g., selecting the object when the subject is interrogative).

The first prompt extracts the entity set from the question while suppressing generic mentions. The second prompt selects the most appropriate relation from the candidate schema based on the question semantics and the extracted entities. The third prompt determines the main entity that should act as the pivot node for subsequent retrieval.

F.2 Prompts for question answer reasoning

This section presents the structured prompt template used for the final one-shot reasoning stage (see Section 4.4). The `{facts}` field is populated with the proximity-based serialized subgraph $\tilde{\mathcal{F}}_q$, where events temporally closest to the anchors are positioned at the beginning to simulate the reasoning path. The system instruction strictly enforces a closed-book reasoning setting, requiring the model to derive answers solely from the provided evidence to prevent hallucination. Furthermore, it defines explicit execution rules for processing temporal constraints (e.g., *before*, *after*), resolving ordinal dependencies (e.g., *first*, *last*), and formatting the final output based on the target answer type.

The fourth prompt is used only after the evidence

chain has been constructed. It guides the LLM to answer the question from the serialized facts, apply temporal constraints explicitly, and return the final answer in a normalized list format.

F.3 Prompt Templates

Prompt 1: Entity set extraction

```
Extract core topic entities (people, places, orgs) from the question as a Python list. Use double quotes if a string contains an apostrophe (e.g., "Xi'an's"), otherwise use single quotes. Exclude generic terms with "country". Output ONLY the list (e.g., ['A', "B's"]) without explanation. Return [] if empty.

Examples:
Question: After the Danish Ministry of Defence and Security, who was the first to visit Iraq?
response: ['Iraq', 'the Danish Ministry of Defence and Security']

Question: When did the al-Shabaab insurgency use unconventional violence against Muslims in the United Kingdom?
response: ['al-Shabaab insurgency', 'Muslims in the United Kingdom']

Question: To whom did John Dramani Mahama make an appeal after 2010?
response: ['John Dramani Mahama']

Question: {question}
response:
```

Prompt 2: Relation set extraction

```
Select exactly ONE relation from the Relation Set that best matches the core action in the Question, using the Entities List for context. Output ONLY the exact relation name.
Mapping Rules: wish/plan/aim → "Express intent..."; ask/urge/call on → "Appeal for..."; negotiate/talks → "Engage in negotiation..." or "Express intent to meet..."; visit/travel → "Make/Host a visit"; attack/force → "Use ... force"; criticize → "Make a statement..."
Examples:
Q: After the Danish Ministry..., who was the first to visit Iraq?
E: ["the Danish Ministry...", "Iraq"]
A: Make a visit

Q: Before China, with whom did Spain wish to cooperate economically?
E: ["Spain", "China"]
A: Express intent to cooperate economically

Q: Who did Greece appeal to for humanitarian aid in 2015?
E: ["Greece"]
A: Appeal for humanitarian aid

Q: {question}
E: {entities_list}
Relation Set: {relation_set}
A:
```

Prompt 3: Main entity identification

```
Task: Identify the single Main Entity from the list. Output ONLY the exact name.
Selection Logic: 1. If the grammatical subject is a specific entity (e.g., "Kitti..."), select the Subject. 2. If the subject is interrogative (e.g., "Who/What"), select the focus Object.
Examples:
Q: After the Danish Ministry..., who was the first to visit Iraq?
L: ["Iraq", "Defense / Security Ministry (Denmark)"]
A: Iraq (Subject is "who" → pick Object)

Q: When did Kitti Wasinondh last negotiate with Thailand?
L: ["Kitti Wasinondh", "Thailand"]
A: Kitti Wasinondh (Specific Subject → pick Subject)

Q: With whom did Catherine Ashton last wish to meet before Cambodia?
L: ["Catherine Ashton", "Cambodia"]
A: Catherine Ashton (Specific Subject → pick Subject)

Q: {question}
L: {entities_list}
A:
```

Prompt 4: Question answering

```
Task: Answer Q using ONLY provided Facts. Output strictly a Python list string (e.g., ['A', '2024']) with NO extra text. Return [] if empty.
Execution Rules:
1. Data: Facts are [sub, rel, obj, date]. Ignore invalid dates if temporal reasoning is needed.
2. Type: If 'time', format as YYYY, YYYY-MM, or YYYY-MM-DD to match Q granularity. If 'entity', extract sub or obj based on Q direction (e.g., Who acted? → Sub).
3. Logic: Filter facts by time constraints (before/after). For "first" → sort earliest; "last" → sort latest.
4. Final: Deduplicate answers, preserve temporal order, apply Top K.
Input:
Q: {question}
Type: {answer_type} | Top K: {topk}
Facts: {facts}
Response:
```