RASTeR: Robust, Agentic, and Structured Temporal Reasoning

Anonymous ACL submission

Abstract

Temporal question answering (TQA) remains a persistent challenge for large language models (LLMs), particularly in retrieval-augmented generation (RAG) settings where retrieved content may be irrelevant, outdated, or temporally inconsistent. This is especially critical in applications like clinical event ordering, policy tracking, and real-time decision-making, which require reliable temporal reasoning even under noisy or misleading context. To address this challenge, we introduce RASTeR: **R**obust, **A**gentic, and **S**tructured, **Te**mporal **R**easoning, an agentic prompting framework that separates context evaluation from answer generation. RASTeR first assesses the relevance and temporal coherence of retrieved context, then constructs a structured temporal knowledge graph (TKG) to better facilitate reasoning. When inconsistencies are detected, RASTeR selectively corrects or discards context before generating an answer. Across multiple datasets and LLMs, RASTeR consistently improves robustness: defined here as the model's ability to generate correct predictions despite suboptimal context. We further validate our approach through a "needle-in-the-haystack" study, in which relevant context is buried among irrelevant distractors. Even with forty distractors, RASTeR achieves 75% accuracy, compared to the runner-up model, which reaches only 62%.[1]

1 Introduction

Large language models (LLMs) often answer factual questions directly from the knowledge stored in their parameters, as long as the necessary facts appear in their pre-training data (Petroni et al., 2019; Roberts et al., 2020). When a fact is missing, practitioners typically fall back on retrieval-augmented generation (RAG) (Lewis et al., 2020), which prepends retrieved passages to the prompt so the model can "read" before it

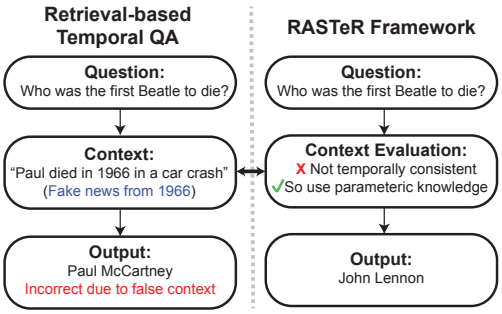

Figure 1: Example of temporal question answering failure due to irrelevant context. The retrieved statement is outdated, leading to an incorrect answer. RASTeR detects the inconsistency and defaults to parametric knowledge. We explore this and other context imperfections, including partially incorrect and fully irrelevant context.

"writes." Unfortunately, the retriever offers no guarantee of relevance (Yin et al., 2023). Irrelevant or adversarial snippets can mislead the generator and lower accuracy (Petroni et al., 2020). Recent work further underscores that today's QA benchmarks rarely stress a system's robustness[2] (Shaier et al., 2024). These issues become even more pronounced in temporal question answering (TQA), where answers depend on current facts and where stale, or simply wrong, documents are frequently retrieved (Wu et al., 2024).

TQA, therefore, poses a dual challenge: models must identify relevant entities (if any) and reason about their evolution over time. Robust temporal reasoning is critical for applications such as historical-event analysis (Lorenzini et al., 2022), time-sensitive retrieval (Wu et al., 2024), data-driven journalism, and real-time analytics, domains where a single date error can significantly alter the answer. Yet comprehensive benchmarks such as TimeBench and TRAM reveal that even GPT-4 lags behind human performance, despite access to gold context (Chu et al., 2023; Wang and Zhao,

[1] Code will be released upon acceptance.

[2] Some TQA work defines robustness as handling diverse temporal phenomena. Here, we define it as the ability to answer correctly despite suboptimal context

2024), and recent studies show that LLMs often hallucinate timelines or overlook explicit temporal cues (Beniwal et al., 2024). While prior work has evaluated models under clean context (Wallat et al., 2025; Luu et al., 2022; Tan et al., 2024), tested zero-shot generalization with synthetic data (Uddin et al., 2025), or explored robustness to irrelevant context in general QA (Yoran et al., 2024a; Cheng et al., 2024), these approaches do not directly address the temporal inconsistencies and ambiguities that arise in realistic retrieval.

Despite this progress, a key gap remains. Existing benchmarks and methods tend to focus on either (**1**) scenarios in which the model has no prior knowledge of the event and must rely entirely on external context, (**2**) general robustness to distractors without temporal grounding, or (**3**) evaluation of questions that may have been seen during pretraining. However, real-world TQA systems must handle both: reasoning about known events under noisy, outdated, or conflicting context, and generalizing to novel or emerging events where memorized knowledge offers no support. Temporal ambiguity (In et al., 2025; Wang et al., 2025), misaligned retrievals (Fang et al., 2024; Yoran et al., 2024a), and hallucinations even under gold context (He et al., 2024; Wallat et al., 2024) highlight the need for methods that can diagnose and correct temporal inconsistencies. To our knowledge, no prior work systematically evaluates model robustness under both seen event settings and also evaluates unseen event settings with temporal context.

To address this gap, we propose RASTeR, an agent-based framework for temporal question answering that explicitly separates context evaluation from answer generation. RASTeR introduces modular agents that assess the temporal relevance and coherence of retrieved passages before transforming valid evidence into a structured Temporal Knowledge Graph (TKG). This structure enables precise, stepwise reasoning about time even under adversarial or outdated contexts. We systematically evaluate RASTeR across multiple models and four temporal QA datasets, including scenarios where events are known, unknown, or contextually distorted. Our results show that this agentic and structured decomposition not only improves robustness to noisy context, a key limitation in RAG pipelines, but also enables fine-grained reasoning over long, distractor-heavy passages. In doing so, RASTeR bridges the gap between robustness and temporal reasoning, offering a principled approach to TQA under realistic retrieval conditions. See Figure 1 for a high-level idea of our contribution.

Our contributions are as follows: (**1**) We introduce RASTeR, an agentic prompting pipeline that separates context evaluation from answer generation via temporal consistency agents and structured knowledge graph transformation. (**2**) We benchmark RASTeR across three LLMs and four temporal QA datasets, demonstrating consistent gains in both clean and noisy contexts. (**3**) We conduct granular robustness analyses, including adversarial retrieval settings (needle-in-the-haystack), altered temporal context, and relevance misclassification, to better understand the strengths and weaknesses of this approach.

2 Related Work

Temporal Question Answering. Temporal QA tasks involve understanding how events unfold over time, whether in text, video (Zhu et al., 2017), or structured data such as knowledge bases (Xiao et al., 2021; Jang et al., 2017; Saxena et al., 2021; Zhao and Rios, 2024; Tan et al., 2024). This includes applications like ordering clinical events (Sun et al., 2013; Zhao and Rios, 2024) or answering factoid questions such as "Who was president in 1998?" Several benchmarks have been proposed to evaluate temporal reasoning, including tests for time-sensitive fact verification and temporal reversal, where performance asymmetries between forward and backward questions reveal a reliance on memorized patterns rather than grounded temporal inference (Bajpai et al., 2024; Wallat et al., 2025).

Recent and historical work has exposed persistent limitations of LLMs in this setting. Models often struggle to reason over timelines, hallucinate events, or miss temporal cues entirely (Qiu et al., 2023; Beniwal et al., 2024). To probe these weaknesses, researchers have released new datasets (Gruber et al., 2024a; Jia et al., 2018; Velupillai et al., 2015; Wang et al., 2022; Gruber et al., 2024b) and diagnostic tasks (Llorens et al., 2015; Tan et al., 2023a; Gao et al., 2024a) that evaluate logical reasoning in time-sensitive settings. Zhu et al. (2025) highlight a related issue of temporal drift: LLMs tend to anchor their factual knowledge around 2015, resulting in degraded performance for domains like news or policy where timelines evolve. This drift presents a key challenge for retrieval-augmented QA, where

the context retrieved may be outdated, misleading, or temporally misaligned with the question.

Robustness in Retrieval-Augmented Generation. Improving the robustness of LLMs to imperfect context has been a focus of recent work on RAG. Broadly, these methods fall into three categories: filtering irrelevant context, adversarial training, and ambiguity-aware reasoning. For filtering, Yoran et al. (2024b) propose using NLI-based filters to exclude unsupported evidence before generation, and fine-tune models on mixed-quality data to improve resistance to distractors. He et al. (2024) introduce CoV-RAG, which incorporates a verification model and structured reasoning to select and integrate relevant information.

Adversarial training methods expose models to noisy or counterfactual inputs to encourage robustness. For instance, Fang et al. (2024) train models on irrelevant and contradictory passages to improve reliability under real-world retrieval errors. However, these approaches typically focus on general QA and do not account for temporal-specific failure modes. Ambiguity-aware pipelines offer a complementary strategy. In et al. (2025) retrieve diverse evidence to accommodate questions with multiple valid answers. Wang et al. (2025) propose a multi-agent architecture where separate models handle different retrieved passages, and a judge model resolves conflicts. Other work uses search-based methods (Hu et al., 2025b), eligibility assessment (Kim et al., 2024), or similarity-based example selection (Park et al., 2024) to guide reasoning under ambiguity. Finally, GraphRAG (Han et al., 2025) combines RAG with graph-structured knowledge, showing that graph-based retrieval can improve reasoning. This motivates us to explore how transforming retrieved temporal context into graph form to can support more robust reasoning.

Structured Knowledge and Reasoning. Structured representations such TKGs enable reasoning over time. Most prior research assumes access to structured datasets and focuses on TKG question answering (TKGQA), which typically involves either interpolation (inferring missing facts within a timeline) or extrapolation (predicting events beyond observed data) (Chen et al., 2024). A central challenge in TKGQA is identifying the most salient nodes. Zhang et al. (2024) use reinforcement learning to sample reasoning chains, while Gao et al. (2024b) first filter relevant relations and then restrict them temporally.

Others focus on improving question formulation. Hu et al. (2025a) show that LLMs perform better on explicit temporal queries and propose a two-stage retrieval-and-rewriting pipeline to make implicit questions more solvable. Xia et al. (2022) also advocate for a two-step strategy that first retrieves direct evidence and then expands it using related entities to capture second-order temporal relationships. These methods assume relatively clean data and often ignore the noisy, conflicting nature of real-world context.

In contrast, our work examines how structured temporal representations impact model robustness when the context is messy, misaligned, or adversarial. Rather than using TKGs solely for interpolation or extrapolation, we dynamically construct TKGs from retrieved text and assess their utility under imperfect retrieval conditions. The closest to our approach is the Chain-of-Timeline framework (Wu and Hooi, 2025), which constructs structured TKGs based on a question and its associated context. However, their work evaluates only on golden context and a single dataset. We extend it by developing a system that handles a variety of context and generalizes across models and datasets.

3 Method

TQA presents unique challenges that standard RAG pipelines are not designed to handle. Retrieved context may be outdated, partially relevant, or temporally inconsistent, yet current systems often assume that any retrieved passage can be treated as reliable input. Our approach addresses this gap by explicitly separating context evaluation from answer generation. We first assess whether the context is relevant and temporally coherent with respect to the question. When the context passes these checks, we convert it into a structured TKG to support precise, time-aware reasoning. If the context is found to be unreliable or inconsistent, we either attempt to correct it or disregard it and rely on the model's parametric knowledge. This modular, agent-based design enables robust performance across a wide range of temporal QA scenarios and offers greater interpretability through structured intermediate representations.

We formalize the robust TQA task as follows. Let Q denote a temporal question, and C denote the context provided to answer it. The LLM is modeled as a function $\mathcal{M}_\theta$ that produces an answer $A = \mathcal{M}_\theta(Q, C)$, where $Q = q_1, \ldots, q_n$,

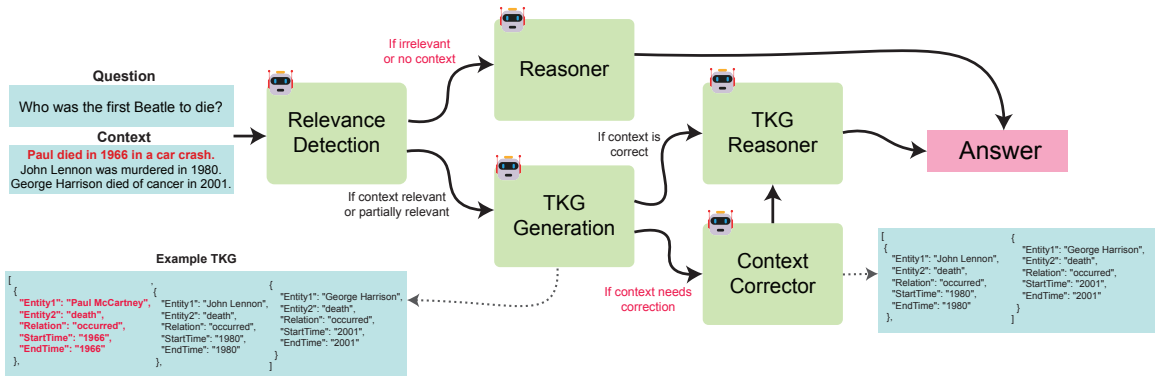

Figure 2: Overview of the RASTeR framework. Given a question and retrieved context, the system first determines whether the context is relevant and temporally coherent. If necessary, it corrects temporal inconsistencies before generating a structured TKG. The final answer is produced either by reasoning over the TKG or, in cases of irrelevant or missing context, via a fallback zero-shot reasoner.

$C = c_1, \ldots, c_m$, and $A = a_1, \ldots, a_k$ are token sequences. We evaluate model performance across several context settings: relevant (C_r), irrelevant (C_i), altered (C_a), and no context (C_0).

The RASTeR pipeline is structured into distinct modules, each handled by a dedicated agent: context evaluation, TKG construction, context correction, and answer generation (via reasoner agents). We show a high-level overview of our method in Figure 2. First, a question and context are evaluated to test if the context is relevant to the question. If not, or if there is no context, a reasoner answers the question directly using the models internal parametric knowledge if it is answerable. Otherwise, a TKG is generated using the context. If the relevance detector determined that the context was only partially relevant, then the context corrector is called to fix the TKG. Finally, the TKG reasoner is called to reason over the TKG to answer the question. We describe each part below.

Context Evaluation. Before reasoning over the retrieved context, we must establish whether it is temporally aligned and semantically relevant to the question. To achieve this, we introduce a Relevance Reasoning Chain (RRC) that decomposes context evaluation into discrete steps. Given a question Q and context C, the model identifies the question's entities Q_e, checks for their presence (e_{pres}) in the context, and generates a Correction Reasoning Chain (CRC) $D = (d_1, d_2, d_3, d_4)$ assessing d_1: chronological coherence of dates, d_2: alignment of context dates with the question, d_3: realism of the overall time span, and d_4: agreement with parametric knowledge. These outputs inform a final decision C_{nc} on whether the context requires correction by the "Context Correction" agent. The exact prompting format for this step is shown in Appendix A.5.1 Figures 5

Temporal Knowledge Graph Construction. When the context is deemed usable, we convert it into a TKG that supports symbolic reasoning over events and temporal intervals. TKGs can be formally defined as a sequence of quadruples $(e_1, r_i, e_2, t_i)_{i=1}^{N}$, where each tuple represents a fact consisting of a subject entity e_i, a relation r_i, an object entity e_2, and an associated timestamp t_i.. We begin by splitting the context into sentences and chunking it with overlap (batch size = 12, overlap = 6). For each chunk c_i, the model conditions on the previous TKG state TKG_{i-1} to expand the graph: $TKG_i = f_{\text{TKG}}(c_i, TKG_{i-1})$. The final graph is the union of all iterations. As an example, if there are three sentences, s_1, s_2, ands_3, and we use a batch size of 2 and an overlap of 1, then a TKG_1 will be generated using s_1 and s_2. TKG_2 will be generated by s_2 and s_3. Both TKG_1 and TKG_2 will be combined to form the final TKG. Intuitively, generating a TKG by passing the entire context as input causes the model to hallucinate nodes and edges, and worse, miss important information. By generating it in an iterative and overlapping fashion, information is seen multiple times and in small contexts to generate the final graph better. An example of the prompting for this procedure is shown in Appendix A.5.1 Figure 6.

Context Correction. If C_{nc} is true, we trigger a context correction mechanism. For each TKG node, we replace the temporal fields (`starttime` and `endtime`) with placeholders and prompt the model to infer plausible time spans. The model then generates a natural language sentence that articu-

lates the relation. Formally, each corrected node is $(e_1, rel, e_2, starttime'_i, endtime'_i, sentence_i)$. This results in a corrected graph TKG' with both symbolic and textual representations. Appendix A.5.1 Figure 7 shows the full correction prompt.

Answer Generation. When a TKG is available, we filter relevant nodes, extract justifications, and synthesize an answer A as part of a larger output $(A, sn, r) = f_{\text{tkg\_answerer}}(Q, TKG)$, where sn denotes supporting nodes and r is the reasoning trace. We do this using an LLM agent without any rule-based methods. The full answer generation prompt is shown in Appendix A.5.1 Figure 8. In the absence of usable context (without TKG), the model falls back to zero-shot reasoning using parametric knowledge: $(Q^1, r, A) = f_{\text{zs\_answerer}}(Q)$, where Q^1 is a restated version of the question and r is the internal reasoning trace. The prompt for this setup is included in Appendix A.5.1 Figure 9.

4 Experiments

In this section, we describe the datasets, metrics, baseliens, and our overall results. We also include a detailed error analysis and ablation of the various components in our agent-based framework.

Datasets Each subset of our collected datasets benchmarks a distinct aspect of temporal reasoning, thus testing different dimensions of temporal question answering. We describe each dataset below.

MenatQA (MQA). In MQA (Wei et al., 2023), the *counterfactual* subset explores imaginative temporal reasoning. The *scope* subset evaluates a model's ability to handle questions with variable time spans, while the *scope_expand* subset challenges models to reason over extended temporal intervals that go beyond the typical bounds of the context. The *order* subset targets reasoning over shuffled event sequences.

TimeSensitiveQA (TSQA). TSQA (Chen et al., 2021) evaluates temporal reasoning over time-evolving passages, with a focus on alignment between temporal expressions in the question and timeline boundaries in the context. The dataset is split into two levels: *easy* and *hard*. In the *easy* subset, the time specifier in the question exactly matches a boundary event (e.g., the start or end of a time interval) that is explicitly mentioned in the passage, allowing models to answer via surface-level matching. In the *hard* subset, the time specifier falls within the middle of a temporal span, requiring models to infer implicit time alignment and reason beyond explicit timestamps.

TempReason (TR). TR (Tan et al., 2023b) focuses on factual temporal reasoning across two levels. The *l2* subset asks for specific facts grounded in time (e.g., "Who coached the team in 2010?"), while the *l3* subset requires reasoning over event sequences (e.g., "Who coached the team before Ted Lasso?"), combining time understanding with knowledge of event order.

UnSeenTimeQA (UTQA). UTQA (Uddin et al., 2025) is a dataset of logistics-style word problems designed to test temporal reasoning without relying on prior knowledge. Because the problems are synthetic and domain-specific, models cannot answer them without using the provided context. This reduces concerns about training data contamination. We focus on the *hardSerial* and *hardParallel* subsets. *HardSerial* assumes events occur in sequence but only provides durations, requiring models to simulate a timeline mentally. *HardParallel* allows events to overlap and introduces distractors that resemble irrelevant but plausible context.

Metrics. We report Exact Match (EM), Contains Accuracy (Acc), and word-level F1 to evaluate model performance. EM measures whether the predicted answer exactly matches any reference answer (e.g "Barack Obama" is not "Obama"). Acc is more lenient and considers a prediction correct if it is a subset of, or contains, any reference answer (e.g "Barack Obama" contains "Obama"). Finally, F1 captures the overlap between the predicted and reference answers at the word level by computing the harmonic mean of precision and recall. Formal definitions of these metrics are provided in Appendix A.1. To conserve space, our main tables only show Acc, but full results for EM and F1 are available in Appendix A.3 (Tables 8 and 9).

Baselines. We compare three baseline prompting strategies against our proposed method. (**1**) generic Few-Shot prompting, (**2**) a simple reasoning prompt, and (**3**) a TKG prompt with no agentic steps. For each baseline we include four few-shot examples, one each for relevant-, irrelevant-, slightly altered, and no-context.

Few-Shot. In the Few-Shot approach, we provide the question and context and ask for an answer.

Model	Prompt Type	MQA	TR	TSQA	Avg
gemma-3-12b-it	Few-Shot	**.332**	.257	.176	.293
	Reasoning	.222	.275	**.190**	.225
	TKG	.302	.254	.164	.271
	RASTeR	.327	**.290**	.166	**.294**
gpt-4o-mini	Few-Shot	.302	.288	.220	.286
	Reasoning	.264	**.324**	.236	.270
	TKG	.306	.256	.201	.280
	RASTeR	**.319**	.315	**.262**	**.311**
Llama-3.1-8B-Instruct	Few-Shot	.087	.124	.069	.090
	Reasoning	.217	.227	.135	.205
	TKG	**.266**	.227	.135	**.238**
	RASTeR	.253	**.231**	**.182**	**.238**

Table 1: Acc averaged across subset, and eval-context for each model and prompting strategy.

Model	Prompt	MQA	TR	TSQA	Avg
gemma-3-12b-it	Few-Shot	.238	.004	.010	.161
	Reasoning	.110	.026	.028	.082
	TKG	.235	.004	.010	.159
	RASTeR	**.305**	**.052**	**.098**	**.228**
gpt-4o-mini	Few-Shot	.190	.018	.044	.137
	Reasoning	.174	**.116**	.120	.155
	TKG	.211	.014	.030	.148
	RASTeR	**.253**	.091	**.164**	**.171**
Llama-3.1-8B-Instruct	Few-Shot	.019	.000	.002	.013
	Reasoning	.090	.000	.002	.060
	TKG	.179	.012	.018	.124
	RASTeR	**.209**	**.050**	**.102**	**.165**

Table 2: Acc averaged across subset for Irrelevant Context Evaluations Only.

The prompt for this baseline is in Figure 10 in the Appendix.

Reasoning. In the reasoning approach, we ask the model to follow the following reasoning chain (**1**) restate the question, (**2**) evaluate the relevance of the context, (**3**) quote supporting evidence, (**4**) reason towards an answer, and (e) use the reasoning to come to a final conclusion. Basically, this is a structured chain-of-thought-like prompt (Sultan et al., 2024) for TQA. The full prompt can be seen in Figure 11 in the Appendix.

Simple TKG. In this approach, the model first extracts entities from the context and uses them to construct a structured TKG composed of timestamped relational tuples. It then answers the question using only the generated TKG, encouraging structured reasoning and temporal grounding without additional agentic steps. Unlike the simple TKG baseline, which directly constructs a TKG from the context without evaluating its relevance or consistency, our method introduces agentic reasoning steps. These include checking whether the context is relevant or altered, correcting temporal inconsistencies, and iteratively building a TKG conditioned on previous outputs, resulting in a more robust and context-sensitive reasoning process. The full prompt is in Figure 12 in the Appendix.

Results. Table 1 shows the average contains accuracy on the MQA, TR, and TSQA datasets. RASTeR demonstrates consistent robustness across MQA, TR, and TSQA. It generalizes well across Gemma (gemma-3-12b-it) , GPT (gpt-4o-mini) , and Llama (Llama-3.1-8B-Instruct) with an average improvement in accuracy from .293, .286, and .205 to .294, .311, and .238, respectively. These scores are averaged across all four context types: relevant (C_r), irrelevant C_i, altered (C_a), and no context (C_0). For Gemma and LLaMA, TKG ties for best average score. Overall, this shows strong robustness to noisy RAG contexts compared to standard baseline methods.

Next, we report how our system works in a worst-case setting, when evaluated only on the irrelevant context in Table 2. On average, RASTeR consistently outperforms other methods, particularly on open-source models. RASTeR with Gemma scores on average $\sim$7% better (.228) than the runner-up (.161). Likewise, LLaMA (.165) handles random context on average $\sim$6% better than its runner-up (.124). Furthermore, in the irrelevant evaluation setting, our method is the **dominant prompting strategy across nearly every dataset and model combo**. The only exceptions being gpt + TR, where reasoning is higher RASTeR (.116 vs. .091)

In practice, RAG systems often surface both relevant and irrelevant content. The context is *generally* never completely relevant nor completely irrelevant. To simulate this, we manipulate TSQA by inserting n irrelevant contexts on each side of the golden context (e.g., for $n = 3$: *irr*, *irr*, *irr*, *rel*, *irr*, *irr*, *irr*; where '*irr*' is an unrelated distractor and '*rel*' is the true relevant context). Intuitively, the model needs to identify the relevant context within many noisy contexts. This is particularly difficult given that language models generally "lose" information in the middle (Liu et al., 2024; Zhang et al., 2025). Descriptive statistics for this experiment can be found in Figure 6 in the Appendix. Figure 3 shows the overall findings of our experiments. Overall, RASTeR remains highly effective under this setup, maintaining strong performance despite the presence of distractors. At each n, RASTeR achieves the highest performance. In fact, with forty distractors ($n = 20$), RASTeR, with an accuracy of 74%, outperforms (by at least 12%) all

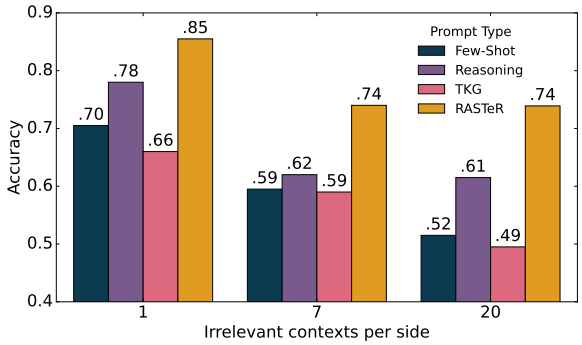

Figure 3: GPT accuracy as the number of distractors (irrelevant contexts) increases around a single relevant passage. All contexts have a relevant passage.

Model	Prompt Type	HardParallel	HardSerial	Avg
gemma-3-12b-it	Few-Shot	.085	.117	.101
	Reasoning	.146	.151	.149
	TKG	**.341**	**.408**	**.375**
	RASTeR	**.391**	**.355**	**.373**
gpt-4o-mini	Few-Shot	.267	.317	.292
	Reasoning	.190	.164	.177
	TKG	**.533**	**.551**	**.542**
	RASTeR	**.251**	**.331**	**.291**
LLaMA	Few-Shot	.109	.130	.120
	Reasoning	.197	.192	.195
	TKG	**.275**	**.274**	**.275**
	RASTeR	**.213**	**.256**	**.235**

Table 3: Acc across the UTQA hard subsets using relevant context only.

Ablation	**irrelevant**	**avg**
RASTeR	**.300**	**.388**
w/o DateFix	.263	.375
w/o TKG	.275	.360
w/o DetRel	.212	.325

Table 4: Ablation results showing accuracy for *irrelevant* context and the overall average across all context types.

other prompting strategies' performance at fourteen distractors ($n = 7$). These results demonstrate that our system can robustly reason over long contexts with numerous distractors. This finding is highly impactful, given that we show that careful engineering of how context is handled, even at a low, nearly artificial level, can generalize to more realistic scenarios that are experienced in practice. Moreover, the results in this experiment are stronger than the artificial experiments.

In Table 3, we report the results of the USQA dataset. Intuitively, we are evaluating generalization to unseen data, i.e., information the model has never seen during pretraining. At a high level, we hypothesize that using the TKG is crucial for improved temporal reasoning when the temporal context wasn't observed during pretraining. While RASTeR incorporates a TKG, it may not consistently outperform the TKG baseline alone, as RASTeR's reasoning and TKG components are decoupled to better handle noisy context. In contrast, the TKG baseline reasons and answers within a single prompt. We find that RASTeR outperforms the reasoning and few-shot baselines across all models and metrics, confirming that incorporating a TKG, even in a modular setup, substantially enhances generalization to novel temporal contexts. For instance, using the gemma-3-12b-it model, RASTeR achieves an average accuracy of 0.373 compared to only 0.149 for the reasoning baseline and 0.101 for few-shot prompting. This trend holds across other models as well, such as LLaMA, where RASTeR improves from 0.120 (few-shot) and 0.195 (reasoning) to 0.235. Although RASTeR does not always exceed the decoupled TKG baseline, its consistent advantage over non-TKG methods demonstrates the importance of explicitly structured temporal representations even in modular reasoning pipelines. This result suggests that future work can explore how to better link the reasoning answerer and the actual TKG generation (e.g., through iterative TKG generation and answering, in a back-and-forth framework).

Ablation. To assess the contribution of each component in RASTeR, we conducted an ablation study by evaluating three modified variants of the pipeline: (**1**) *w/o DateFix*, which disables the context corrector responsible for resolving temporally inconsistent information; (**2**) *w/o TKG*, which removes the TKG constructor and relies entirely on natural language rather than structured graphs; and (**3**) *w/o DetRel*, which bypasses relevance assessment by treating all input context as relevant. Each ablation was tested against the full pipeline on a randomly sampled subset spanning all datasets and subsets. Descriptive statistics for this subset appear in Table 7 (Appendix A.2]). Overall, the full RASTeR pipeline achieves the highest average accuracy (.388), outperforming all ablations. In the irrelevant context setting, RASTeR also obtains the best performance (.300), indicating that both the TKG and context relevance agents contribute meaningfully to robustness under noisy retrieval. Notably, removing the relevance detector (*w/o DetRel*) leads to the largest drop in performance, especially

		Predicted Context Type		
		SA	NO/IRR	REL
Eval Context	No	0.0%	100.0%	0.0%
	Irrelevant	2.3%	90.1%	7.6%
	Relevant	2.6%	4.8%	92.6%
	Slightly Altered	77.1%	8.5%	14.3%

Table 5: Confusion matrix of RASTeR's predicted context type (columns) versus true context type (rows).

in the irrelevant context setting, suggesting that misclassifying noisy inputs as relevant can significantly degrade reasoning. These results highlight the importance of both structured temporal representation and selective context filtering in improving temporal QA robustness. Full ablation results are shown in Table 10 (Appendix A.4).

Determining Relevance Error Analysis. Looking at Table 5, we observe distinct patterns in how relevance is predicted across different eval contexts. When evaluated with no context, the model perfectly classifies the context as `NO / IRR`, 100% of the time. When presented with irrelevant context, the model sometimes mistakenly labels the context as `REL` (7.6%) or less frequently as `SA` (2.3%) . This is followed by evaluating on relevant context, in which the model most frequently misclassified the context as `NO / IRR` (4.8% of the time). There is a big drop in performance when evaluated on `slightly altered`, relevance errors are more evenly split. While 72.6% of predictions are correctly labeled as `SA`, the 14.3% mislabeled as `REL` and 8.5% as *NO / IRR* suggest that identifying slightly altered context remains challenging and leaves room for improvement.

Types of General Errors. A common issue occurs when the model attempts to infer an answer even when the gold label is unanswerable. Our prompting pipeline encourages models to reason and guess in the absence of explicit evidence, which improves performance in no-context and irrelevant-context settings. However, this behavior can produce errors in settings where abstaining is preferred. An example is provided below.

QUESTION: What job did Mary have in 2010?
CONTEXT: In 2009, Mary was a teacher at Lincoln High School. In 2011, she became a school principal.
INCORRECT REASONING: Mary's 2010 job is not explicitly stated. But since she was a teacher in 2009 and only became a principal in 2011, it is inferred she remained a teacher in 2010. **GROUND TRUTH:** unanswerable **PREDICTION:** Lincoln High School

Finally, temporal reasoning remains one of the most challenging categories of errors. To highlight these issues, MQA's counterfactual questions require models to answer based on a hypothetical that directly contradicts the context. These questions test whether models adhere to the "what-if" condition rather than relying on factual timelines. An example of a subset-specific reasoning error can be found below:

QUESTION: What school did Henry go to from 2008 to 2010, if Henry didn't graduate from Rice High School until 2011?
CONTEXT: Henry started at Rice High School in 2004. In 2008, he graduated and enrolled at Brown University. He completed his studies there in 2015.
INCORRECT REASONING: The timeline shows Henry enrolled at Brown University in 2008, which implies he attended it from 2008 to 2010. Since no other school is mentioned, Brown is inferred as the correct answer.
GROUND TRUTH: Rice High School
PREDICTION: Brown University

These examples illustrate the need for finer-grained evaluation and improved handling of temporal and counterfactual reasoning in large language models.

5 Conclusion

TQA presents persistent challenges for LLMs, particularly when retrieved context is irrelevant, misleading, or missing. We introduced RASTeR, a modular, agentic framework that separates context evaluation from answer generation. By assessing context quality, constructing structured TKGs, and correcting inconsistencies, RASTeR enables more robust and temporally grounded reasoning. Across four temporal QA datasets and three LLMs, RASTeR consistently improves accuracy in noisy and distractor-heavy settings while maintaining strong performance in ideal conditions. In needle-in-the-haystack scenarios, it not only outperforms alternatives but also degrades more gracefully as distractors increase. In future work, we plan to extend RASTeR to support multi-hop temporal reasoning and questions with multiple temporally valid answers. We also aim to broaden our robustness analysis beyond date shifts to include perturbations such as entity substitutions and relation modifications, better characterizing model sensitivity to noisy temporal input. Furthermore, we aim to investigate how to more effectively integrate TKG generation and the reasoner answerer for improved performance on unseen temporal reasoning questions.

6 Limitations

Despite our best efforts to develop a comprehensive and robust framework for temporal question answering, several limitations persist. RASTeR uses *slightly* more resources than traditional prompting. While RASTeR's agent-based architecture introduces multiple prompting steps per query, we found the overall overhead to be manageable in practice. On average, each full query involves 3–4 calls to the underlying LLM, with total token usage ranging from 2.5x to 3x that of a single monolithic prompt. However, because the number of agent calls is fixed and does not scale with input length or number of retrieved documents, the additional cost remains minimal and predictable across queries. This fixed modular structure ensures stable inference time and simplifies deployment planning. RASTeR has not been evaluated on datasets with gold-standard temporal graphs, leaving the accuracy of its generated knowledge graphs unverified. While the framework is practical in retrieval-based settings, it underperforms on tasks requiring abstract generalization, where simpler prompting strategies may suffice. Moreover, although RASTeR prompts with structured temporal knowledge, it does not yet leverage deeper architectural integration, such as graph neural networks or instruction-tuned models, which may offer more effective handling of complex temporal relationships.

References

Ashutosh Bajpai, Aaryan Goyal, Atif Anwer, and Tanmoy Chakraborty. 2024. Temporally consistent factuality probing for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15864–15881, Miami, Florida, USA. Association for Computational Linguistics.

Himanshu Beniwal, Dishant Patel, Kowsik Nandagopan D, Hritik Ladia, Ankit Yadav, and Mayank Singh. 2024. Remember this event that year? assessing temporal information and understanding in large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 16239–16348.

Kai Chen, Ye Wang, Yitong Li, Aiping Li, Han Yu, and Xin Song. 2024. A unified temporal knowledge graph reasoning model towards interpolation and extrapolation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 117–132, Bangkok, Thailand. Association for Computational Linguistics.

Wenhu Chen, Xinyi Wang, William Yang Wang, and William Yang Wang. 2021. A dataset for answering time-sensitive questions. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.

Chen Cheng, Xinzhi Yu, Haodong Wen, Jingsong Sun, Guanzhang Yue, Yihao Zhang, and Zeming Wei. 2024. Exploring the robustness of in-context learning with noisy labels. In *ICLR 2024 Workshop on Reliable and Responsible Foundation Models*.

Zheng Chu, Jingchang Chen, Qianglong Chen, Weijiang Yu, Haotian Wang, Ming Liu, and Bing Qin. 2023. Timebench: A comprehensive evaluation of temporal reasoning abilities in large language models. *arXiv preprint arXiv:2311.17667*.

Feiteng Fang, Yuelin Bai, Shiwen Ni, Min Yang, Xiaojun Chen, and Ruifeng Xu. 2024. Enhancing noise robustness of retrieval-augmented language models with adaptive adversarial training. *Preprint*, arXiv:2405.20978.

Yifu Gao, Linbo Qiao, Zhigang Kan, Zhihua Wen, Yongquan He, and Dongsheng Li. 2024a. Two-stage generative question answering on temporal knowledge graph using large language models. *arXiv preprint arXiv:2402.16568*.

Yifu Gao, Linbo Qiao, Zhigang Kan, Zhihua Wen, Yongquan He, and Dongsheng Li. 2024b. Two-stage generative question answering on temporal knowledge graph using large language models. *Preprint*, arXiv:2402.16568.

Raphael Gruber, Abdelrahman Abdallah, Michael Färber, and Adam Jatowt. 2024a. Complextempqa: A large-scale dataset for complex temporal question answering. *Preprint*, arXiv:2406.04866.

Raphael Gruber, Abdelrahman Abdallah, Michael Färber, and Adam Jatowt. 2024b. Complextempqa: A large-scale dataset for complex temporal question answering. *Preprint*, arXiv:2406.04866.

Haoyu Han, Yu Wang, Harry Shomer, Kai Guo, Jiayuan Ding, Yongjia Lei, Mahantesh Halappanavar, Ryan A. Rossi, Subhabrata Mukherjee, Xianfeng Tang, Qi He, Zhigang Hua, Bo Long, Tong Zhao, Neil Shah, Amin Javari, Yinglong Xia, and Jiliang Tang. 2025. Retrieval-augmented generation with graphs (graphrag). *Preprint*, arXiv:2501.00309.

Bolei He, Nuo Chen, Xinran He, Lingyong Yan, Zhenkai Wei, Jinchang Luo, and Zhen-Hua Ling. 2024. Retrieving, rethinking and revising: The chain-of-verification can improve retrieval augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10371–10393, Miami, Florida, USA. Association for Computational Linguistics.

Qianyi Hu, Xinhui Tu, Cong Guo, and Shunping Zhang. 2025a. 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.

Yunhai Hu, Yilun Zhao, Chen Zhao, and Arman Cohan. 2025b. Mcts-rag: Enhancing retrieval-augmented generation with monte carlo tree search. *Preprint*, arXiv:2503.20757.

Yeonjun In, Sungchul Kim, Ryan A. Rossi, Mehrab Tanjim, Tong Yu, Ritwik Sinha, and Chanyoung Park. 2025. Diversify-verify-adapt: Efficient and robust retrieval-augmented ambiguous question answering. In *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)*, pages 1212–1233, Albuquerque, New Mexico. Association for Computational Linguistics.

Yunseok Jang, Yale Song, Youngjae Yu, Youngjin Kim, and Gunhee Kim. 2017. Tgif-qa: Toward spatio-temporal reasoning in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2758–2766.

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*, CIKM '18, page 1807–1810, New York, NY, USA. Association for Computing Machinery.

Hyuhng Joon Kim, Youna Kim, Cheonbok Park, Junyeob Kim, Choonghyun Park, Kang Min Yoo, Sanggoo Lee, and Taeuk Kim. 2024. Aligning language models to explicitly handle ambiguity. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1989–2007, Miami, Florida, USA. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12.

Hector Llorens, Nathanael Chambers, Naushad UzZaman, Nasrin Mostafazadeh, James Allen, and James Pustejovsky. 2015. Semeval-2015 task 5: Qa tempeval-evaluating temporal information understanding with question answering. In *proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 792–800.

Jasmine Lorenzini, Hanspeter Kriesi, Peter Makarov, and Bruno Wüest. 2022. Protest event analysis: Developing a semiautomated nlp approach. *American Behavioral Scientist*, 66(5):555–577.

Kelvin Luu, Daniel Khashabi, Suchin Gururangan, Karishma Mandyam, and Noah A. Smith. 2022. Time waits for no one! analysis and challenges of temporal misalignment. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5944–5958, Seattle, United States. Association for Computational Linguistics.

Seong-Il Park, Seung-Woo Choi, Na-Hyun Kim, and Jay-Yoon Lee. 2024. Enhancing robustness of retrieval-augmented language models with in-context learning. *Preprint*, arXiv:2408.04414.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. *arXiv preprint arXiv:2005.04611*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Yifu Qiu, Zheng Zhao, Yftah Ziser, Anna Korhonen, Edoardo M Ponti, and Shay B Cohen. 2023. Are large language models temporally grounded? *arXiv preprint arXiv:2311.08398*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

Apoorv Saxena, Soumen Chakrabarti, and Partha Talukdar. 2021. 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.

Sagi Shaier, Lawrence E. Hunter, and Katharina von der Wense. 2024. Desiderata for the context use of question answering systems. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*. Long Paper.

Md Arafat Sultan, Jatin Ganhotra, and Ramón Fernandez Astudillo. 2024. Structured chain-of-thought prompting for few-shot generation of content-

grounded qa conversations. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 16172–16187.

Weiyi Sun, Anna Rumshisky, and Ozlem Uzuner. 2013. Temporal reasoning over clinical text: the state of the art. *Journal of the American Medical Informatics Association*, 20(5):814–819.

Qingyu Tan, Hwee Tou Ng, and Lidong Bing. 2023a. Towards benchmarking and improving the temporal reasoning capability of large language models. *arXiv preprint arXiv:2306.08952*.

Qingyu Tan, Hwee Tou Ng, and Lidong Bing. 2023b. Towards benchmarking and improving the temporal reasoning capability of large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14820–14835, Toronto, Canada. Association for Computational Linguistics.

Qingyu Tan, Hwee Tou Ng, and Lidong Bing. 2024. Towards robust temporal reasoning of large language models via a multi-hop QA dataset and pseudo-instruction tuning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 6272–6286, Bangkok, Thailand. Association for Computational Linguistics.

Md Nayem Uddin, Amir Saeidi, Divij Handa, Agastya Seth, Tran Cao Son, Eduardo Blanco, Steven Corman, and Chitta Baral. 2025. UnSeenTimeQA: Time-sensitive question-answering beyond LLMs' memorization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL 2025), Volume 1: Long Papers*, pages 1873–1913, Vienna, Austria. Association for Computational Linguistics.

Sumithra Velupillai, Danielle L Mowery, Samir Abdelrahman, Lee Christensen, and Wendy Chapman. 2015. BluLab: Temporal information extraction for the 2015 clinical TempEval challenge. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 815–819, Denver, Colorado. Association for Computational Linguistics.

Jonas Wallat, Abdelrahman Abdallah, Adam Jatowt, and Avishek Anand. 2025. A study into investigating temporal robustness of LLMs. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 15685–15705, Vienna, Austria. Association for Computational Linguistics.

Jonas Wallat, Adam Jatowt, and Avishek Anand. 2024. Temporal blind spots in large language models. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 683–692.

Han Wang, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. 2025. Retrieval-augmented generation with conflicting evidence. *Preprint*, arXiv:2504.13079.

Jiexin Wang, Adam Jatowt, and Masatoshi Yoshikawa. 2022. Archivalqa: a large-scale benchmark dataset for open-domain question answering over historical news collections. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3025–3035.

Yuqing Wang and Yun Zhao. 2024. Tram: Benchmarking temporal reasoning for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Yifan Wei, Yisong Su, Huanhuan Ma, Xiaoyan Yu, Fangyu Lei, Yuanzhe Zhang, Jun Zhao, and Kang Liu. 2023. MenatQA: A new dataset for testing the temporal comprehension and reasoning abilities of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1434–1447, Singapore. Association for Computational Linguistics.

Feifan Wu, Lingyuan Liu, Wentao He, Ziqi Liu, Zhiqiang Zhang, Haofen Wang, and Meng Wang. 2024. Time–sensitive retrieval–augmented generation for question answering. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM)*, pages 2544–2553.

Jiaying Wu and Bryan Hooi. 2025. Chain-of-timeline: Enhancing LLM zero-shot temporal reasoning with SQL-style timeline formalization. In *Workshop on Reasoning and Planning for Large Language Models*.

Yuwei Xia, Mengqi Zhang, Qiang Liu, Shu Wu, and Xiao-Yu Zhang. 2022. MetaTKG: Learning evolutionary meta-knowledge for temporal knowledge graph reasoning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7230–7240, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. 2021. Next-qa: Next phase of question-answering to explaining temporal actions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9777–9786.

Xunjian Yin, Baizhou Huang, and Xiaojun Wan. 2023. ALCUNA: Large language models meet new knowledge. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1397–1414, Singapore. Association for Computational Linguistics.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024a. Making retrieval-augmented language models robust to irrelevant context. In *International Conference on Learning Representations (ICLR)*.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024b. Making retrieval-augmented language models robust to irrelevant context. *Preprint*, arXiv:2310.01558.

Junhao Zhang, Richong Zhang, Fanshuang Kong, Ziyang Miao, Yanhan Ye, and Yaowei Zheng. 2025. Lost-in-the-middle in long-text generation: Synthetic dataset, evaluation framework, and mitigation. *arXiv preprint arXiv:2503.06868*.

Qinggang Zhang, Junnan Dong, Hao Chen, Daochen Zha, Zailiang Yu, and Xiao Huang. 2024. Knowgpt: Knowledge graph based prompting for large language models. *Preprint*, arXiv:2312.06185.

Xingmeng Zhao and Anthony Rios. 2024. Utsa-nlp at chemotimelines 2024: Evaluating instruction-tuned language models for temporal relation extraction. In *Proceedings of the 6th Clinical Natural Language Processing Workshop*.

Chenghao Zhu, Nuo Chen, Yufei Gao, Yunyi Zhang, Prayag Tiwari, and Benyou Wang. 2025. Is your LLM outdated? a deep look at temporal generalization. In *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)*, pages 7433–7457, Albuquerque, New Mexico. Association for Computational Linguistics.

Linchao Zhu, Zhongwen Xu, Yi Yang, and Alexander G. Hauptmann. 2017. Uncovering the temporal context for video question answering. *Int. J. Comput. Vision*, 124(3):409–421.

A Appendix

A.1 Metric Formalization

In any NLP applications, due to the diverse nature of natural langue, determining the correctness of a prediction is always challenging. To highlight this challenge, Figure 4 shows how the model output can be marked incorrect by both exact match (EM) and contains accuracy (Acc), despite being semantically correct. Having a variety of evaluation metrics allows us to get a better picture of model performance measured by partial matches, and more strict criteria.

QUESTION: John O. Moseley was an employee for whom from Mar 1936 to Dec 1938?
OUTPUT: central state college
GROUND TRUTH: central state teachers college

Figure 4: An example where the model output is semantically correct but fails EM and Acc.

To define our evaluation metrics formally, Let $\hat{a}$ be the predicted answer and let $A = \{a_1, a_2, \ldots, a_n\}$ denote the set of gold reference answers. Let W_x represent the multiset of words in answer x.

Exact Match (EM) EM measures whether the ground truth is *exactly identical* to the prediction. (e.g. "Border Collie" is identical to "Border Collie")

$$\text{EM} = \mathbf{1}[\hat{a} \in A]$$

EM returns 1 if the predicted answer exactly matches any gold answer, and 0 otherwise.

Contains Accuracy (Acc) Acc measures whether the ground truth *is contained* in the prediction. (e.g. "Border Collie" is contained in "The dog is a Border Collie")

$$\text{Acc} = \mathbf{1}[\exists a \in A \text{ such that } a \subseteq \hat{a}]$$

Acc returns 1 if any gold answer is a substring of the predicted answer, and 0 otherwise.

Word-Level F1 F1 is the most flexible metric. It measures the maximum word overlap between the predicted and gold answers by computing the harmonic mean of precision and recall. For each $a \in A$, we compute:

$$\text{Precision} = \frac{|W_{\hat{a}} \cap W_a|}{|W_{\hat{a}}|}, \quad \text{Recall} = \frac{|W_{\hat{a}} \cap W_a|}{|W_a|}$$

$$\text{F1} = \max_{a \in A} \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

For example: if the predicted answer is "central state college" and the gold answer is "central state teachers college", the prediction shares three words with the gold answer. Precision is 1 (3 out of 3 words), recall is .75 (3 out of 4 words), and F1 = $\frac{2 \cdot 1 \cdot .75}{1 + .75} = .857$.

A.2 Descriptive Statistics

In Table 6, we report the average number of words per context and the number of samples (n) for all datasets used in our experiments. The full MQA dataset was included but is substantially smaller than the other datasets. Full subsets of UTQA were also used, though we excluded the easy and medium settings, as they were less challenging and required minimal reasoning compared to the hard subsets. Among all subsets, HS n_20 had the highest average word count, with nearly 5.5k words. This is due to the relevant context being surrounded by forty distractors. The TSQA subsets

Dataset	Subset	Avg. Words	n
MQA	counterfactual	82.38	112
MQA	order	80.44	182
MQA	scope	82.38	112
MQA	scope_expand	75.91	176
UTQA	hardSerial	140.84	2700
UTQA	hardParallel	140.47	2700
HS	n_1	399.94	200
HS	n_3	904.67	200
HS	n_5	1503.09	200
HS	n_7	2021.04	200
HS	n_20	5494.45	200
TR	l2	128.29	250
TR	l3	141.08	250
TSQA	easy	2041.00	250
TSQA	hard	1827.08	250

Table 6: Average word count of relevant context and number of samples (n) per subset.

also had long contexts, making them the second most verbose in terms of average word count.

For ablations, we used a subset of 80 randomly selected rows sampled from our three main datasets. Half of the rows came from TR, while the other half were drawn from MQA and TSQA. Table 7 summarizes the row counts and proportions for each subset included in the ablation.

Dataset	Subset	Count	Proportion (%)
MQA	counterfactual	7	8.8
	order	6	7.5
	scope	5	6.2
	scope_expand	7	8.8
TR	l2	16	20.0
	l3	16	20.0
TSQA	easy	14	17.5
	hard	9	11.2
Total	–	**80**	**100.0**

Table 7: Descriptive statistics of combined data subsets used for ablations

A.3 Expanded Results

In addition to Acc show in Table 1, we report EM in Table 8 and F1 scores in Table 9. While Gemma with the Few-Shot prompt slightly outperforms RASTeR in terms of EM (by 0.8%), RASTeR consistently performs better on both Acc and F1. In fact, RASTeR shows the strongest gains when evaluated through the lens of F1. For example, RASTeR combined with LLaMA achieves a full 7% improvement over the next-best average F1 score.

Model	Prompt Type	MQA	TR	TSQA	Avg
gemma-3-12b-it	Few-Shot	**.306**	.257	**.150**	**.272**
	Reasoning	.191	.266	.136	.194
	TKG	.291	.240	.142	.258
	RASTeR	.291	**.276**	.140	.264
gpt-4o-mini	Few-Shot	.268	.288	.188	.258
	Reasoning	.210	**.318**	.195	.226
	TKG	.273	.253	.177	.254
	RASTeR	**.287**	.303	**.221**	**.287**
LLaMA	Few-Shot	.056	.122	.060	.068
	Reasoning	.088	.193	.095	.107
	TKG	.178	.153	.093	.159
	RASTeR	**.213**	**.222**	**.148**	**.204**

Table 8: Exact Match (EM) averaged across subset, and eval-context for each model and prompting strategy.

Model	Prompt Type	MQA	TR	TSQA	Avg
gemma-3-12b-it	Few-Shot	.364	.321	.206	.331
	Reasoning	.248	.317	.211	.253
	TKG	.345	.304	.205	.315
	RASTeR	**.368**	**.383**	**.223**	**.346**
gpt-4o-mini	Few-Shot	.343	.338	.270	.330
	Reasoning	.292	.382	.285	.306
	TKG	.344	.306	.248	.322
	RASTeR	**.359**	**.380**	**.317**	**.364**
LLaMA	Few-Shot	.085	.171	.090	.100
	Reasoning	.129	.239	.135	.148
	TKG	.225	.208	.133	.207
	RASTeR	**.285**	**.300**	**.224**	**.277**

Table 9: F1 Score averaged across subset, and eval-context for each model and prompting strategy.

We believe RASTeR's strong performance under F1 is due to the metric's sensitivity to partial overlap. Predictions are often semantically correct but do not match the gold answer word-for-word, especially when context is missing. Since RASTeR is designed to filter, correct, and reason over noisy context, it excels in settings where exact matches are unlikely but partial correctness is common.

A.4 Error Analysis

Table 5 presents a confusion matrix showing how our relevance reasoner classified different types of context. Note that the pipeline treats both no context and irrelevant context as equivalent, so both the relevance reasoner *should* label them as **NO / IRR**. As discussed in the main paper, the greatest area for improvement lies in detecting slightly altered contexts, which are only correctly identified 77.1% of the time.

We evaluate the contribution of individual com-

Ablation	none	irrelevant	relevant	slightly altered	avg
w/o DateFix	**.275**	.263	**.762**	.200	.375
w/o TKG	**.275**	.275	.700	.188	.360
w/o DetRel	.150	.212	**.762**	.175	.325
nothing ablated	.263	**.300**	.738	**.250**	**.388**

Table 10: Ablation results showing accuracy across different context types (none, irrelevant, relevant, slightly altered). Each row removes a specific module from the full pipeline to assess its contribution.

ponents in our agentic system by systematically removing steps and comparing the performance of the full system to these ablated variants. As can be seen in Table 5, Our full model achieves the highest average accuracy across context types, driven by strong performance in the `none`, `relevant`, and `slightly altered` settings. It also maintains a competitive score in the `relevant` condition, demonstrating balanced robustness across evaluation scenarios.

A.5 Prompts

Both our method and baseline prompts used few-shot examples. To ensure a fair, apples-to-apples comparison, we kept the examples as consistent as possible by using the same set of AlphaGo-related questions and contexts[3], randomly selected once and reused throughout. When relevant, we included examples with relevant, irrelevant, slightly-altered, and no context to test model robustness across conditions. Notably, for the Irrelevant Answerer shown in Figure 9 , we include only a no-context example, as its pipeline never permits prompting with any other context type. Further details are provided below.

A.5.1 RASTeR Prompts

Relevance Reasoner

To assess the relevance of a given context, we prompt the model to perform five steps using both the question and the context: (**1**) Identify the main entity in the question; (**2**) Determine whether this entity appears in the context; (**3**) If the context uses pronouns instead of explicit names (e.g., he/she/they instead of 'Abraham Lincoln'), assess whether the pronouns plausibly refer to the identified entity; (**4**) Evaluate the temporal validity of any dates in the context across four dimensions; (**5**) Based on this evaluation, decide whether date correction is needed. We included five few shot examples for the relevance reasoner: (**1**) a typical example; (**2**) an example with a longer context window; (**3**) an example with some noisy context, (**4**); an example with longer context and noise; and (**5**) a counterfactual example. The exact prompt is shown in Figure 5.

TKG Constructor

To incrementally construct a TKG, we prompt the model with a slice of historical context and all previously constructed nodes. The model is asked to identify new temporal facts from the context slice that are not already present in the prior graph. Then convert those facts into structured TKG nodes. Each output node includes: (**1**) a supporting quote from the context; (**2**) subject and object entities; (**3**) their relation; (**4**) a start and end time; and (**5**) a reformatted sentence that is grammatically correct, time-grounded, and follows specific templates. The model is explicitly instructed to infer plausible dates when none are stated, use qualifiers like "around" when necessary, and avoid duplicating existing facts. We included two Few-Shot examples to guide the TKG Constructor: (**1**) an example with no former TKG; and (**2**) an example with a starting TKG. The exact prompt is shown in Figure 6.

TKG Date Completion

When the Relevance Reasoner determines that a context requires correction, we manually remove the `starttime` and `endtime` from each node in the TKG, replacing them with placeholder values X and Y. The model is then prompted to (**1**) infer plausible temporal bounds using historical knowledge or contextual cues, and (**2**) generate a natural, grammatically correct sentence that incorporates the subject (`e1`), relation (`rel`), object (`e2`), and the inferred timeframe. The output must include both the completed sentence and the recovered temporal fields in a structured format. This step allows the model to use its internal knowledge to infer temporal boundaries, enabling accurate correction of incomplete TKG facts. We included a single few shot example to guide the model. The exact prompt is shown in Figure 7.

Relevant Answerer

After the construction of the TKG, we prompt the model with a question and the TKG. The model is instructed to perform three steps: (**1**) select the node(s) from the TKG that are temporally relevant and contain information necessary to answer the

[3]We selected the topic AlphaGo randomly; it does not confer any advantage to our method or the baselines and serves solely to ensure consistency across examples.

System Prompt

You will be given a question and a context. Your job is to carefully evaluate the context using the steps below:
1. Question Entity: Identify the main entity of the question.

2. Does the Entity appear in the context?

3. Do Pronouns Refer to the Entity?
If the main entity does not explicitly appear in the context, but the context contains pronouns (e.g., 'he", she", it"), can these pronouns be reasonably inferred to refer to the entity in question?
Options: True / False / None (if not applicable)

4. Evaluation of Context Dates:

- **a. Chronological Order:** Are the dates logically ordered, or are they contradictory or impossible to sequence?
- **b. Timeframe Alignment:** Do the dates overlap with or lead into the timeframe asked in the question?
- **c. Realism of Time Span:** Is the total date range plausible for the entity, or is it unrealistically broad?
- **d. Historical Consistency:** Do the dates contradict known facts or include anachronisms or future events?

5. Date Need Correction: A boolean indicating whether the dates in the context need correction based on the above evaluation.

User Prompt

Please evaluate the following context by following these steps:

1. Question Entity
2. Does the Entity appear in the context?
3. Do the Pronouns Refer to the Entity?
4. Evaluation of Context Dates
5. Date Need Correction

HERE IS THE QUESTION:
`{question}`
HERE IS THE CONTEXT:
`{context}`

Figure 5: System and user determining the relevance of a provided context.

question; (**2**) explain how the selected node(s) support the answer, including reasoning over temporal relationships such as before/after conditions; and (**3**) provide a confident, direct answer based on the evidence, or make an educated guess using indirect cues if no explicit answer is available. This step leverages the structured context encoded in the TKG to produce grounded, time-aware answers. We included six Few-Shot examples to help guide the models reasoning: (**1**) an example with relevant context; (**2**) and example without context; (**3**) an example with random context; (**4**) and example with slightly altered context; (**5**) an example showing when pronouns correctly refer to the entity in the question; and (**6**) an example showing when the pronouns do not refer to the entity in the question. The exact prompt is shown in Figure 8.

Irrelevant Answerer

When the context is determined to be irrelevant, we discard the context, and prompt the model to answer questions using only its internal knowledge. The model is guided through a 3-step reasoning process: (**1**) restate the question to clarify what is being asked; (**2**) reason toward an answer using general world knowledge; and (**3**) provide a final answer in a clear, structured format. By discarding the context, we eliminate distractors and evaluate the model's ability to interpret and answer temporal questions without relying on a retrieved context.

We provide a single Few-Shot example to help guide the model's reasoning. The exact prompt is shown in Figure 9.

A.5.2 Baseline Prompts

All baseline prompts have four Few-Shot examples to help guide their reasoning: (**1**) an example with relevant context; (**2**) an example without context; (**3**) an example with random context; (**4**) an example with slightly altered context.

Few-Shot
Our first baseline evaluates model performance using a minimal prompt that mirrors common few-shot setups. The model is given a question and a corresponding context and is instructed to respond concisely using the format: `The answer is X`. Unlike our structured approaches, this prompt includes no explicit reasoning steps or guidance for interpreting the context. It serves to benchmark how well the model can extract answers when given relevant input, and how it performs in the presence of no or irrelevant context without any reasoning scaffolding. The exact prompt is shown in Figure 10.

Reasoning
Our second baseline introduces a structured 5-step reasoning process to guide the model through question answering. Given a question and a context, the model is instructed to (**1**) restate the question to clarify its intent; (**2**) assess whether the context is relevant; (**3**) quote specific evidence from the context, or indicate `NONE` if no useful information is found; (**4**) reason toward an answer using either the provided evidence or its own internal knowledge; and (**5**) produce a final answer in the format: `The answer is X`. This format encourages explicit reasoning and evidence grounding. The exact prompt is shown in Figure 11.

Simple TKG
This baseline a non-iterative TKG construction without the full multi-agent pipeline. The model is prompted to (**1**) extract all entities from the context, including people, places, roles, and other named concepts; (**2**) construct a TKG; and (**3**) answer the question based on the constructed TKG using the standard format: `The answer is X`. The model is allowed to correct factual inconsistencies in the context or fall back on internal knowledge when context is irrelevant. This prompt provides a basic measure of how well the model can extract temporal structure and reason over it in a single pass. The exact prompt is shown in Figure 12.

System Prompt

You will be presented with a slice of historical context and a previously constructed temporal knowledge graph (TKG). Your task is to identify **new temporal facts** from the current context and output them as TKG nodes.
Do not repeat facts already included in the previous TKG.

Each node should include the following fields:

- **quote**: a verbatim snippet or sentence from the context that supports the node's validity
- **e1**: subject entity (e.g., person, organization)
- **e2**: object entity (e.g., location, role, other person)
- **rel**: the relation between e1 and e2
- **starttime**: when the relation began
- **endtime**: when the relation ended
- **reformatted**: a rewritten sentence that:
  - rearranges the quote to follow the order: time(s), e1, rel, e2
  - is grammatically correct and includes only e1, e2, rel, and times
  - **must include temporal information** (date, year, or month); if not explicit, infer it
  - uses qualifiers like 'around" or roughly" when inferring time
  - follows these example templates:
    - `On {starttime}, {e1} was {rel} {e2}`
    - `Between {starttime} and {endtime}, {e1} was {rel} {e2}`
    - `Roughly in {starttime}, {e1} was {rel} {e2}`

Format your output as a list of dictionaries:

```
[ { "quote": "...", "e1": "...", "e2": "...", "relation": "...", "starttime": "...",
"endtime": "...", "reformatted": "..." }, ... ]
```

Notes:

- Begin with an empty TKG or 'NONE" on the first slice.
- Only include new nodes clearly grounded in the current context.
- Use short, direct quotes.
- Do not repeat nodes from the former TKG.
- Preferred time formats: 'YYYY-MM-DD", then YYYY", Month YYYY", or UNKNOWN".
- You may extract overlapping or nested events if they are distinct.
- Use only double quotes in your answer (no single quotes).

User Prompt Header

Construct new TKG nodes using the provided **context** and **former_tkg**.
Avoid duplicating facts already extracted. Output only new nodes relevant to the current slice of context.

Reminder: The `"reformatted"` quote should be a grammatically correct sentence that includes a specific date, year, month, or timespan. If not explicitly stated in the context, *infer it using surrounding information*. In such cases, use terms like 'around" or roughly".

HERE IS THE CONTEXT:
`{context}`

HERE IS THE FORMER TKG:
`{former_tkg}`

Figure 6: System and user prompt for generating new temporal knowledge graph (TKG) slices of a provided context.

System Prompt

You are given a temporal knowledge graph (TKG) triple with missing `starttime` and `endtime`.

Your task is to:

1. Infer appropriate `starttime` and `endtime` based on historical knowledge or reasonable assumptions.
2. Write a grammatically correct and natural-sounding sentence that incorporates:
   - the subject (`e1`)
   - the relationship (`rel`)
   - the object (`e2`)
   - and the inferred temporal range

You are allowed to rephrase the sentence as long as all elements are included and the timeframe is clearly conveyed.

Your output should follow this format:

```
COMPLETE SENTENCE: [your complete sentence]
STARTIME: [YYYY-MM-DD]
ENDTIME: [YYYY-MM-DD]
```

Example:

INPUT:
```
{"e1": "Arseny Dmitrievich Mironov", "e2": "USSR State Prize", "rel": "recipient of",
"starttime": X, "endtime": Y}
```

OUTPUT:
```
COMPLETE SENTENCE: Arseny Dmitrievich Mironov received the USSR State Prize in 1976.
STARTIME: 1976-01-01
ENDTIME: 1976-12-31
```

User Prompt Header

Given the following TKG triple with missing temporal information, fill in the `starttime` and `endtime`, and write a complete, natural-sounding sentence.

You must:

- Include `e1`, `rel`, and `e2`
- Clearly indicate the time period
- Use correct grammar and phrasing

Format:

```
COMPLETE SENTENCE: ...
STARTIME: ...
ENDTIME: ...
```

HERE IS YOUR TRIPLE:
`{triple}`

Figure 7: System and user prompt for inferring missing temporal values in a temporal knowledge graph triple.

System Prompt

You are given a question and a temporal knowledge graph (TKG). Your job is to answer the question using the TKG to assist you.

Please follow these steps:
1. Select Supporting Nodes:

- From the TKG, return the node(s) that provide the information necessary to answer the question.
- You may include one or more nodes.
- Only include nodes that are temporally relevant to the question.
- You must consider the time frame mentioned in the question.
- If multiple matching nodes exist, include them all.

2. Explain Your Reasoning:

- Justify how the node(s) support your answer.
- If no node is directly about the question, you may infer the answer from strong contextual clues.
- For *before/after* questions, identify the event that occurred immediately before or after the referenced one.
- **Example:**
  Context: Dan attended high school from 2010–2014, undergrad from 2014–2018, a master's from 2023–2024, and began a PhD in 2024.
  Question: What did Dan do after high school?
  Reasoning: Dan completed undergrad, a master's, and began a PhD after high school. However, undergrad was immediately after, so it is the correct answer.

3. Answer the Question:

- Respond in the format: `The answer is X`
- If no nodes provide a direct answer, use indirect evidence to make an educated guess.
- For instance, political roles, awards, institutions, or cities may imply nationality or affiliation.
- Your answer should be confident and definite.

Note: Use only double quotes in your answer. Do not use single quotes.

User Prompt Header

Given a question and a temporal knowledge graph (TKG), answer the question using the TKG to assist you.
Follow these steps:

1. Select Supporting Nodes
2. Explain Your Reasoning
3. Answer the Question

If no nodes directly provide information to answer the question, use indirect evidence to make an educated guess.

HERE IS YOUR QUESTION:
`{question}`

HERE IS THE TKG:
`{TKG}`

Figure 8: System and user prompts for answering temporal questions using a temporal knowledge graph (TKG).

System Prompt

You are a helpful and precise assistant. When given a question, respond clearly and concisely, only using relevant information. To help you arrive at the correct answer, follow this 3-step reasoning process:

1. Restate the Question
Rephrase the original question to clarify what is being asked.

2. Reason
Rely on your general knowledge to reason toward an answer.

3. Answer
Provide your final answer using the format: `The answer is X.`

Note: Only use double quotes in your answer. Do not use single quotes.

User Prompt Header

Please answer the question using the provided context.
Follow the 3-step reasoning process above, and end with a final answer in Step 3.

HERE IS THE QUESTION:
`{question}`

Figure 9: System and user prompts for answering questions without context using a structured 3-step reasoning process.

System Prompt

You are a helpful and precise assistant. When given a context and a question, respond clearly and concisely, only using relevant information.

Please use the format: `The answer is X` where `X` is your answer.

Example formatting:

```
The answer is Michael Phelps.
```

User Prompt Header

Please answer the question given the context.
Your response should follow the format: `The answer is X`

HERE IS THE QUESTION:
`{question}`

HERE IS THE CONTEXT:
`{context}`

Figure 10: System and user prompts for answering questions with concise, format-specific responses.

System Prompt

You are a helpful and precise assistant. When given a context and a question, respond clearly and concisely, only using relevant information. To help you arrive at the correct answer, follow this 5-step reasoning process:

1. Restate the Question
Rephrase the original question to clarify what is being asked.

2. Evaluate Context Relevance
Determine whether the provided context contains information that is useful for answering the question.

3. Quote Supporting Evidence
Copy and paste the exact portion(s) of the context that support your answer.
If no useful evidence exists, write `NONE`.

4. Reason
If the context is relevant, use the quoted evidence to logically derive the answer.
If the context is not relevant, rely on your general knowledge to reason toward an answer.

5. Answer
Provide your final answer using the format: `The answer is X.`

User Prompt Header

Please answer the question using the provided context.
Follow the 5-step reasoning process above, and end with a final answer in Step 5.

HERE IS THE QUESTION:
`{question}`

HERE IS THE CONTEXT:
`{context}`

Figure 11: System and user prompts for answering questions using a structured 5-step reasoning process.

System Prompt

You are a helpful and precise assistant. You will be presented with some context and a question. Your job has two parts.

First: Identify all entities in the context, including places, names, occupations, and things.
List them using the following format, wrapped in triple backticks. Do not skip any entities.

```
e1. Yoko Ono
e2. Businessman
e3. Europe
```

Second: Construct a Temporal Knowledge Graph (TKG) based on the context using the identified entities.
Each TKG node should include the following fields:

- `Entity1`
- `Entity2`
- `Relation`
- `Timestamp`

Format:

```
[
{'Entity1': '...', 'Entity2': '...', 'Relation': '...', 'Timestamp': '...'},
...
]
```

Additional Instructions:

- If the context is partially incorrect, correct the information before building that part of the TKG.
- If the context is irrelevant or marked as `NONE`, discard it and use your internal knowledge instead.

Once the TKG is complete, use it to answer the question. Respond concisely using the format: `The answer is X.`

Example formatting:

```
The answer is Michael Phelps.
```

User Prompt Header

Build a temporal knowledge graph (TKG) to help answer the question using the provided context.
The TKG should be a list of nodes, each with `Entity1`, `Entity2`, `Relation`, and `Timestamp` fields.

Once the TKG is complete, use it to answer the question.
Your answer should follow the format: `"The answer is X"`

HERE IS THE QUESTION:
`{question}`

HERE IS THE CONTEXT:
`{context}`

Figure 12: System and user prompts for entity extraction, temporal knowledge graph construction, and question answering.