Temporal Reasoning with Large Language Models Augmented by Evolving Knowledge Graphs

Anonymous authors
Paper under double-blind review

Abstract

Large language models (LLMs) demonstrate impressive capability in natural language understanding, yet they remain limited when reasoning over knowledge that evolves. A common remedy is to augment LLMs with knowledge graphs (KGs), which provide structured access to factual information. However, most existing approaches rely on a static snapshot of the KG and fail to account for temporal evolution and conflicting updates that naturally arise in real-world knowledge. To address these challenges, we present **EvoReasoner**, a temporal-aware multi-hop reasoning algorithm that integrates global-local entity grounding, multi-route decomposition, and time-sensitive scoring to support robust inference. Complementing this, we introduce **EvoKG**, a noise-resilient graph evolution module that continuously updates the KG from unstructured documents using confidence-aware contradiction handling and temporal trend tracking. We evaluate our framework on temporal QA benchmarks and a new end-to-end setting where the KG is dynamically updated from raw text. Our method consistently surpasses prompting-only and static/dynamic KG-augmented baselines, and notably enables an 8B-parameter model to achieve accuracy on par with a 671B model trained seven months later. These findings underscore the necessity of unifying temporal reasoning with KG evolution to ensure LLMs remain accurate and up-to-date. Our code and data are released at anonymous.4open.science/r/TREK-434C.

1 Introduction

Recent years have seen growing interest in combining large language models (LLMs) with knowledge graphs (KGs) to improve factual reasoning and knowledge coverage (Pan et al., 2024; Kau et al., 2024). Generally, this line of research aims to address a core limitation of LLMs: they are trained on static corpora and struggle to adapt to emerging knowledge efficiently. By interacting with structured KGs and utilizing their knowledge, LLMs can better handle complex reasoning tasks and answer queries involving up-to-date or structured knowledge (Wang et al., 2024; Ji et al., 2023).

Despite these advances, many existing KG-augmented approaches (Sun et al., 2024; Chen et al., 2024; LUO et al., 2024; Wang et al., 2025) rely on a *static snapshot of the KG* (e.g., a downloaded dump of Freebase (Bollacker et al., 2008) and Wikidata (Vrandečić & Krötzsch, 2014)), overlooking the *temporal dynamics* of evolving knowledge and factual noise inherent in real-world knowledge. Temporal metadata is often treated as auxiliary information rather than a core signal for inference (Chen et al., 2022; Yang et al., 2024b). To highlight these limitations in scenarios involving temporally evolving or shifting information, we analyze a question answering (QA) benchmark (Yang et al., 2024c) in which queries require reasoning over either slowly or rapidly changing facts (Figure 1).

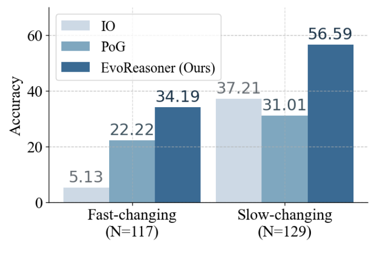

Figure 1: Question answering performance grouped by the required knowledge types.

The results show the following. (1) For questions involving **slow-changing facts**, prompting-based method (e.g., IO Prompt (Brown et al., 2020)) performs well, as internal model knowledge is often sufficient; in contrast, static-graph reasoning methods like Plan-on-Graph (PoG) (Chen et al., 2024) can be hindered by noisy or contradictory facts from the evolving KG. (2) For questions involving **fast-changing facts**, LLMs struggle due to outdated internal knowledge, and while PoG benefits from explicit KG updates, its lack of temporal awareness yields only limited improvements. This underscores the need for a framework such as our EvoReasoner, which is explicitly designed to reason over evolving knowledge.

These limitations stem from two challenges underlying knowledge evolution. First, KG facts subject to *exclusivity* constraints (e.g., birth date, primary affiliation) can be contradicted by newly added or conflicting information during updates. This necessitates *robust contradiction resolution* strategies that account for context, source reliability, and temporal cues, rather than treating all assertions as equally valid. Second, *non-exclusive* facts (e.g, jobs, events) naturally evolve over time, yet existing methods frequently overlook temporal ordering, limiting their effectiveness on time-sensitive queries.

Nevertheless, addressing these challenges demands more than improved reasoning: it requires a unified solution that *jointly* performs temporal reasoning and maintains a consistent, evolving knowledge graph. We propose a unified framework that integrates a multi-hop temporal reasoning algorithm with a temporal-aware, noise-tolerant KG evolution mechanism (illustrated in Figure 2). Unlike prior works that treat KG retrieval as a static augmentation layer, our approach tightly couples temporal reasoning with dynamic KG construction and updates. It consists of two key components: (i) **EvoReasoner**, a temporal multi-hop reasoning algorithm that performs multi-route decomposition, global-local entity grounding, and temporal-aware scoring; and (ii) **EvoKG**, a noise-resilient KG evolution method that constructs and updates the graph from unstructured documents. EvoKG applies confidence-based contradiction resolution and temporal trend tracking to maintain robustness and consistency over time.

We experimentally evaluate EvoReasoner on temporal question answering (QA) benchmarks and a novel end-to-end setting where EvoKG dynamically updates the KG from raw documents. By integrating missing or updated facts, our framework significantly enhances LLM reasoning, outperforming state-of-the-art static/dynamic KG and LLM-only methods with accuracy gains of up to **14.0%** in temporal reasoning and **8%** in evolving KG tasks. Notably, our approach **closes the gap between small and large models**. For example, a compact LLaMA 3.1–8B (Grattafiori et al., 2024) model, trained in December 2023 and run on a single consumer GPU, improves accuracy from **18.6** to **37.0%** after KG updates, comparable to directly prompting a much larger 671B DeepSeek-V3 (Liu et al., 2024) model (38.3% accuracy) trained seven months later. This further highlights the effectiveness of our proposed EvoKG framework in improving the reasoning accuracy of LLMs.

Our contributions can be summarized as follows:

- **Temporal reasoning method.** We introduce EvoReasoner, the first method that combines multi-route decomposition, context-informed global initalization, and temporal-aware local exploration to enhance LLM reasoning over evolving knowledge graphs. Unlike prior work, EvoReasoner enables precise temporal grounding and robust multi-hop inference under ambiguous or time-sensitive conditions, leading to significantly improved reasoning accuracy.
- **Temporal KG evolution method.** We propose EvoKG, a KG construction and update method that resolves factual contradictions and models the temporal progression of non-static facts. This enables the KG to remain accurate and consistent over time, supporting reliable reasoning in dynamic real-world scenarios.
- **Evaluation across static and temporal dimensions.** We evaluate our method on temporal QA benchmarks and a novel end-to-end setting where KGs are updated from raw documents, allowing us to assess the reasoning capability in a temporal setting and how knowledge evolution contributes to downstream QA performance. Our method consistently outperforms both LLM-only and KG-based reasoning baselines across all settings.

2 Preliminaries

Definition 1 (Attributed Knowledge Graph) *An attributed knowledge graph is a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{R}, \mathcal{P})$, where each node $v \in \mathcal{V}$ and each edge $e \in \mathcal{E}$ has a type given by $\tau(v) : \mathcal{V} \to \mathcal{A}$ and $\phi(e) : \mathcal{E} \to \mathcal{R}$, respectively. In addition, each node and edge is associated with a property map $\pi : \mathcal{V} \cup \mathcal{E} \to \mathcal{P}$, where $\mathcal{P}$ denotes the space of key-value pairs. These property maps allow nodes and edges to store arbitrary attributes in the form of a hashmap.*

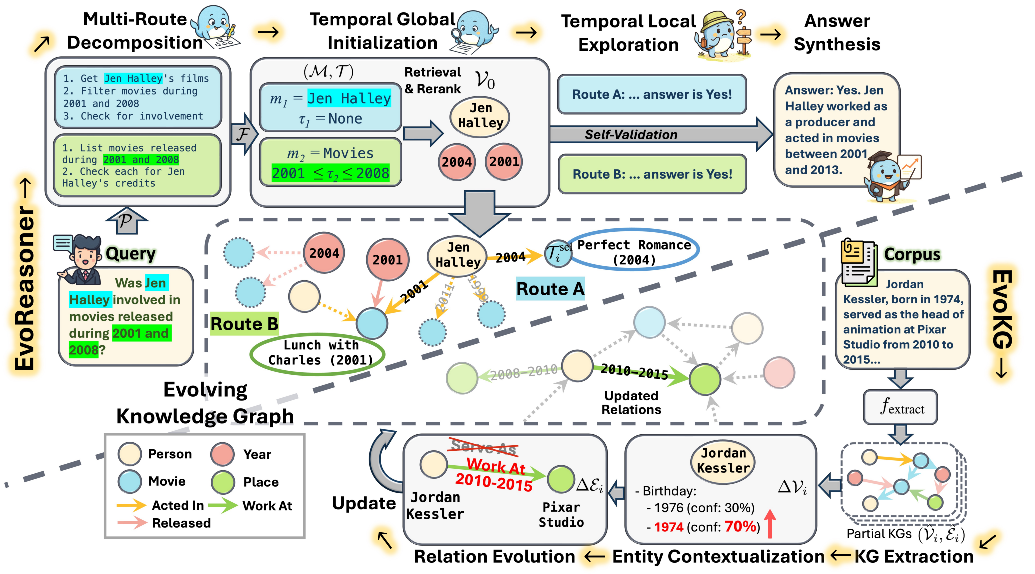
Multi-Route Decomposition
Temporal Global Initialization
Temporal Local Exploration
Answer Synthesis
1. Get Jen Halley's films
2. Filter movies during 2001 and 2008
3. Check for involvement
1. List movies released during 2001 and 2008
2. Check each for Jen Halley's credits
Retrieval & Rerank
Route A: ... answer is Yes!
Self-Validation
Route B: ... answer is Yes!
Answer: Yes. Jen Halley worked as a producer and acted in movies between 2001 and 2013.
EvoReasoner
Query
Was Jen Halley involved in movies released during 2001 and 2008?
Route A
Route B
Perfect Romance (2004)
Lunch with Charles (2001)
Evolving Knowledge Graph
Updated Relations
Corpus
Jordan Kessler, born in 1974, served as the head of animation at Pixar Studio from 2010 to 2015...
EvoKG
Person
Year
Movie
Place
Acted In
Work At
Released
Update
~~Serve As~~
Work At 2010-2015
Jordan Kessler
Pixar Studio
Birthday: 1976 (conf: 30%) 1974 (conf: 70%)
Partial KGs
Relation Evolution
Entity Contextualization
KG Extraction

Figure 2: The overall framework, consisting of two components: EVOREASONER for multi-hop temporal reasoning and EVOKG for document-driven KG evolution. The top pipeline shows how a query is decomposed, grounded to temporal entities, explored locally, and synthesized into an answer. The bottom pipeline illustrates how external documents update the KG via entity and relation contextualization with confidence-aware resolution and temporal evolution.

Problem Formulation. Let $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t, \pi_t)$ denote the KG at time t, where π_t stores attribute metadata for nodes and edges, including temporal information. Each edge $e \in \mathcal{E}_t$ may be associated with a temporal validity interval $[\tau_{\text{start}}(e), \tau_{\text{end}}(e)] \subseteq \mathbb{R} \cup \{\perp\}$, where $\perp$ denotes an unknown or open-ended temporal boundary, indicating when the associated fact holds true. These start and end times are stored in the property map: $\tau_{\text{start}}(e) = \pi_t(e)[\texttt{valid from}], \tau_{\text{end}}(e) = \pi_t(e)[\texttt{valid until}]$.

Given a natural language question x, potentially with an explicit or implicit temporal scope (e.g., an explicit temporal question could be "Who was the president of the U.S. in 2008?", while an implicit temporal question could be "Who was the president when the financial crisis happened?"), the goal is to identify a set of entities $\mathcal{A} \subseteq \mathcal{V}_t$ by reasoning over the temporally relevant subgraph of $\mathcal{G}_t$. These entities are then used by the LLM to formulate the final response to the user query.

3 MULTI-HOP TEMPORAL REASONING OVER EVOLVING KGS

We introduce a novel three-stage inference framework, EVOREASONER, as illustrated in Figure 2: (1) **multi-route decomposition** for diverse and robust exploration; (2) **global initialization** for temporal-aware query grounding; and (3) **local exploration** with temporal information. These stages yield multiple reasoning paths, which are aggregated to produce the final answer.

3.1 MULTI-ROUTE DECOMPOSITION

We first decompose the original query x into multiple semantic reasoning routes (e.g., Route A and B in Figure 2), each representing a distinct interpretation or plan for answering the question. This decomposition improves both robustness and recall by accounting for the structural diversity and noise often present in evolving KGs. The resulting routes serve two purposes: (1) they provide ***soft reasoning guidance*** to direct graph traversal, and (2) offer a ***self-validation*** mechanism—if multiple routes converge to the same answer, the result is more likely to be reliable. As different routes vary in reasoning efficiency (e.g., number of hops or breadth of search), we prioritize concise and targeted ones, starting from the most efficient and expanding only when necessary to establish consensus.

Let $\mathcal{R} = \{r_1, \ldots, r_J\}$ denote the set of candidate decomposition routes, where each route r_j consists of a sequence of subgoals $r_j = \{x_1^{(j)}, x_2^{(j)}, \ldots, x_{T_j}^{(j)}\}$, where T_j is the length of route r_j. We define a route planning function $\mathcal{P}$ (see Appendix E for prompt) that maps the query to these candidate plans $x \xrightarrow{\mathcal{P}} \mathcal{R}$. To evaluate the cost of each route, we propose a multi-objective cost function:

$$\text{Cost}(r_j) = \sum_{t=1}^{T_j} \psi(x_t^{(j)}), \text{ where } \psi(x_t^{(j)}) = (b_t \cdot n_t)^{h_t}.$$

$\psi(x_t^{(j)})$ reflects traversal complexity of subgoal $x_t^{(j)}$, based on: b_t, the number of candidate relation types from the current entity (branching factor); n_t, the number of entities reachable through those relations; and h_t, the estimated number of hops required to complete the subgoal.

This formulation encourages the selection of reasoning plans that are narrower and shorter, reducing the chance of semantic drift during exploration. To balance efficiency and diversity, we retain the top-N lowest-cost and semantically distinct plans: $\mathcal{R}^* = \text{TopK}_N\left(\{r_j \in \mathcal{R} : -\text{Cost}(r_j)\}\right)$. These selected routes are used to guide the traversal process, offering complementary reasoning strategies that can be aggregated for answer synthesis and justification.

3.2 Temporal Contextualized Global Initialization

Given the decomposed routes, our method begins by identifying topic entities in $\mathcal{G}_t$ that are semantically and temporally aligned with the query x and its subgoals $\{x_t^{(j)}\}$. Prior works (e.g., Yang et al. (2024c); Sun et al. (2024)) typically extract topic entities directly using simple string matching, without verifying their presence in the KG or accounting for the temporal context. While this may suffice in static and clean KGs, it fails in evolving and noisy KGs. For example, a query mentioning "Pixar Company" cannot be grounded if the KG only contains the entity "Pixar Animation Studio." Similarly, "Oscar Awards" lacks temporal specificity unless linked to one of its 100 historical variants.

To address these challenges, we propose a *context-aware global initialization* strategy that leverages both semantic similarity and temporal signal. This ensures robust grounding to valid entities, even in cases of naming ambiguity or temporally-indexed duplicates, and avoids the common failure case where reasoning begins from non-existent or irrelevant nodes. Let $\mathcal{F}$ represent a query analysis function that produces two aligned outputs, where $\mathcal{M}$ is a disjoint set of entity mentions in the query and $\mathcal{T}$ is the corresponding set of temporal contexts (e.g., "between 2001 and 2008" in Figure 2):

$$\mathcal{F}(x) \to (\mathcal{M}, \mathcal{T}), \quad \mathcal{M} = \langle m_1, \ldots, m_n \rangle, \quad \mathcal{T} = \langle \tau_1, \ldots, \tau_n \rangle.$$

Each pair (m_i, τ_i) is jointly embedded into a context-aware vector representation: $\boldsymbol{v}_{m_i}^{\tau} = \text{Enc}(m_i, \tau_i)$, where $\text{Enc}(\cdot, \cdot)$ is a pre-trained transformer encoder that incorporates both textual and temporal inputs (e.g., sentence-transformers with timestamp-augmented prompts; see Appendix E). This embedding is then matched against all KG entity embeddings $\{\boldsymbol{v}_e \mid e \in \mathcal{V}_t\}$. To retrieve grounding candidates from the KG, we use cosine similarity to select the top-k most similar nodes: $\mathcal{C}_i = \text{TopK}_k\left(\left\{e \in \mathcal{V}_t \ : \ \text{sim}(\boldsymbol{v}_{m_i}^{\tau}, \boldsymbol{v}_e)\right\}\right)$, where $\text{TopK}_k(\cdot)$ returns the k highest-scoring entities under cosine similarity $\text{sim}(\cdot, \cdot)$, and k is a predefined hyperparameter (e.g., $k = 5$).

Each retrieved candidate $e_j \in \mathcal{C}_i$ is further verified using a relevance scoring function (see Appendix E for the prompt template), $s_{\text{init}}(e_j \mid x, \tau_i) \in [0, 1]$, which estimates the likelihood that a KG entity e_j is the correct grounding for m_i, conditioned on the full question x and its associated temporal context τ_i. This allows for disambiguation among semantically similar or temporally conflicting entities. Let w, the reasoning width, be the number of final candidates retained per entity mentions in the query. The final set of anchor entities is selected as

$$\mathcal{V}_0 = \bigcup_i \text{TopK}_w\left(\{e_j \in \mathcal{C}_i : s_{\text{init}}(e_j \mid x, \tau_i)\}\right).$$

3.3 Temporal-Aware Local Exploration

After global initialization, our approach performs beam search-based local exploration to traverse the KG. At each step, neighboring relations and entities are scored for relevance to the query. Prior methods often overlook the temporal context, which is critical in domains like sports or politics where entities are linked to numerous time-stamped events. Ignoring temporal signals can lead to irrelevant exploration or reliance on random sampling (Sun et al., 2024), missing key evidence.

We address this by introducing a temporal-aware strategy that reranks neighbors based on both semantic and temporal alignment with the current subgoal. This enables more focused and accurate multi-hop traversal through time-relevant paths. In particular, at step i of a reasoning route r_j, given a current entity $v_t \in \mathcal{V}_i$, we iteratively perform the following three actions:

1. Relation Selection: We retrieve all outgoing relations $r \in \mathcal{R}_i(v_t)$ and the relevance scoring function $s_{\text{rel}}(r \mid x, \{x_t^{(k)}\}_{t=1}^{T_k})$ scores them for relevance to the current query and subgoals. The top-ranked relations are selected as follows: $\mathcal{R}_i^{\text{sel}} = \text{TopK}_w\left(\left\{r \in \mathcal{R}_i(v_i) : s_{\text{rel}}(r \mid x, \{x_t^{(k)}\})\right\}\right)$, where w is the reasoning width.

2. Entity Expansion: For each relation $r \in \mathcal{R}_i^{\text{sel}}$, we retrieve connected triplets of the form $(v_i, r, v') \in \mathcal{E}_i$, forming a set of candidate triplets: $\mathcal{T}_i = \{(v_i, r, v') \in \mathcal{E}_i : r \in \mathcal{R}_i^{\text{sel}}\}$.

3. Temporal-Aware Reranking: We perform semantic reranking of triplets using temporal and contextual signals. Each triplet $(v_i, r, v') \in \mathcal{T}_i$ is first verbalized into a natural language string (e.g., "Team A played against Team B in 2020") incorporating both its semantic content and temporal validity interval $[\tau_{\text{start}}, \tau_{\text{end}}] \subseteq \mathbb{R} \cup \{\perp\}$, where $\perp$ denotes an unknown time. Then, the temporal-aware similarity score is computed as $s_{\text{triplet}}(x, (v_i, r, v')) = \text{sim}\,(x, (v_i, r, v'))$, i.e., the similarity between embeddings of x and the verbalized triplet. We select the top-ranked triplets as follows:

$$\mathcal{T}_i^{\text{sel}} = \text{TopK}_w\left(\{t \in \mathcal{T}_i : s_{\text{triplet}}(x, (v_t, r, v'))\}\right), \; \mathcal{V}_{i+1} = \{v' \mid (v_i, r, v') \in \mathcal{T}_i^{\text{sel}}\}.$$

The destination entities in $\mathcal{V}_{i+1}$ from the selected triplets define the next search frontier. The reasoning proceeds until either (1) a stopping criterion is met (i.e., the answer is found or the maximum depth is reached), or (2) no high-confidence edges are available for expansion.

3.4 Answer Synthesis and Justification

Once valid paths are collected, we synthesize a final answer by aggregating high-confidence paths. Each path $\mathcal{P} = \{(r_1, v_1), \ldots, (r_L, v_L)\}$, starting from anchor v_0, is assigned a confidence score:

$$\text{Conf}(\mathcal{P}) = s_{\text{init}}(v_0) \cdot \prod_{i=1}^{L} s_{\text{rel}}(r_i) \cdot s_{\text{triplet}}(x, (v_{i-1}, r_i, v_i)),$$

where $s_{\text{init}}(v_0)$ is the global initialization score from Section 3.2, $s_{\text{rel}}(r_i)$ scores relation relevance at step i, and s_{triplet} captures the temporal and semantic alignment of the traversed edge. For interpretability, we optionally visualize these reasoning paths as annotated subgraphs.

Majority Voting Across Reasoning Routes. Let $\mathcal{R}^* = \{r_1, \ldots, r_N\}$ denote the top-ranked reasoning routes, each yielding a predicted answer $a_i \in \mathcal{V}_t$ with confidence score $\text{Conf}(\mathcal{P}_i)$. The final answer is selected via weighted voting: $a^* = \arg\max_{a \in \mathcal{A}} \sum_{i : a_i = a} \text{Conf}(\mathcal{P}_i)$. With confidence scores, we improve robustness by consolidating predictions from multiple plausible reasoning paths, reducing the impact of individual errors or noisy subgoals.

4 Noise-Resilient Knowledge Graph Evolution

We present EvoKG, a temporal-aware framework for constructing and updating knowledge graphs from unstructured text. Our proposed approach aims to resolve the following two challenges:

1. Contradiction and Noise in Static Knowledge. Certain relations (e.g., birth date and primary affiliation) are *exclusive*: only one value should hold at a time. Let $\mathcal{R}_{\text{excl}} \subset \mathcal{R}$ denote the set of such relations. Updates to these facts can introduce contradictions, especially when sources are noisy or inconsistent. We model exclusive relations as entity-level properties via the map $\pi : \mathcal{V} \to \mathcal{P}$, where each relation $r \in \mathcal{R}_{\text{excl}}$ is associated with a set of candidate values

$$\pi(v)[r] = \{(o_1, C(o_1), \text{ctx}_1), \ldots, (o_k, C(o_k), \text{ctx}_k)\},$$

where o_i is a candidate value, $C(o_i) \in [0, 1]$ reflects confidence (based on frequency and recency), and $\text{ctx}_i \in \mathcal{C}$ captures the extraction context. Unlike prior approaches (Lu et al., 2025) that overwrite conflicting values, EvoKG retains multiple candidates with associated confidence, enabling context-sensitive reasoning and improving robustness to noise.

2. Evolving Trends in Temporal Knowledge. Non-exclusive relations (e.g., "works as" and "acted in") naturally change over time. Let $\mathcal{R}_{\text{non-excl}} = \mathcal{R} \setminus \mathcal{R}_{\text{excl}}$ denote this set of temporally evolving relations. For $r \in \mathcal{R}_{\text{non-excl}}$, we model temporal dynamics via edges $e = (v_s, r, v_t, [t_{\text{from}}, t_{\text{to}}]) \in \mathcal{E}$, where $t_{\text{from}}, t_{\text{to}} \in \mathbb{R} \cup \{\perp\}$ represent the validity interval. The interval is stored in the property map:

$$\phi(e) \in \mathcal{R}_{\text{non-excl}}, \quad \pi(e)[\texttt{valid from}] = \tau_{\text{start}}, \quad \pi(e)[\texttt{valid until}] = \tau_{\text{end}}.$$

This temporal representation allows EvoKG to preserve the full evolution of entity relations, rather than collapsing them into static facts. For instance, the entity Obama may simultaneously hold (Obama, `works as`, Senator, [2004, 2009]) and (Obama, `works as`, President, [2009, 2017]).

To address these two challenges, we develop a two-stage evolution framework: (1) **entity contextualization** to address static contradiction and ambiguity in extracted nodes; and (2) **relation evolution** to model time-sensitive relational changes.

As shown in Figure 2, given an input document corpus $\mathcal{D} = \{d_1, \ldots, d_N\}$, we preprocess it into text chunks d_i and apply a KG extraction function f_{extract} to produce a partial KG. The resulting triples are then aligned and merged with the current KG, $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t, \pi_t)$, via a merge function f_{merge} consisting of the entity contextualization and relation evolution, and produces

$$d_i \xrightarrow{f_{\text{extract}}} \hat{\mathcal{G}}_i = (\hat{\mathcal{V}}_i, \hat{\mathcal{E}}_i) \xrightarrow{f_{\text{merge}}} (\Delta\mathcal{V}_i, \Delta\mathcal{E}_i) \xrightarrow{\text{insert}} \mathcal{G}_{t+1},$$

where $\Delta\mathcal{V}_i$ and $\Delta\mathcal{E}_i$ are the sets of new or updated nodes and edges, respectively. This process may be repeated across document batches for comprehensive coverage.

4.1 Entity Contextualization

Document-extracted entities often exhibit lexical ambiguity (e.g., "Pixar Company" vs. "Pixar Animation Studio") or naming collisions (e.g., movies with the same title across years). To maintain KG quality and ensure effective downstream reasoning, we perform a two-stage process:

1. Contextual Alignment: Given extracted entities $\hat{\mathcal{V}} = \{\hat{v}_1, \ldots, \hat{v}_n\}$, each with a name, description, and property map $\pi(\hat{v}_i)$, we aim to align $\hat{v}_i$ to a node $v \in \mathcal{V}$ or insert it as a new node. Each candidate is jointly encoded over its type, name, and description, and matched against existing KG nodes:

$$\boldsymbol{v}_i = \text{Enc}(\tau(\hat{v}_i), \text{name}(\hat{v}_i), \text{desc}(\hat{v}_i)),\ \mathcal{A}_i = \text{TopK}_k\left(\{v \in \mathcal{V} \mid \text{sim}(\boldsymbol{v}_i, \boldsymbol{v})\}\right).$$

Each $v' \in \mathcal{A}_i$ is then reranked by a contextual scoring function $s(\hat{v}_i, v') \in [0, 1]$, and the best match is selected: $v^* = \arg\max_{v' \in \mathcal{A}(\hat{v})} s(\hat{v}_i, v')$. If $s(\hat{v}_i, v^*) < \theta$, then $\hat{v}_i$ is inserted as a new node.

2. Contextual Merging: Suppose v^* is the aligned node and $\hat{p} = (r, o, \text{ctx})$ is a candidate property to be merged. For exclusive relations $r \in \mathcal{R}_{\text{excl}}$, the updated property set is

$$\pi(v^*)[r] \leftarrow \pi(v^*)[r] \cup \{(o, C(o), \text{ctx})\},\ C(o) = \frac{\delta \cdot f(o)}{1 + e^{-\gamma \cdot \Delta t(o)}},$$

where $C(o) \in [0, 1]$ is the confidence score, with $f(o)$ as the normalized frequency of the value o among prior extractions, $\Delta t(o)$ as the time elapsed since o was last observed (temporal decay), and γ and δ as hyperparameters. Each value o is associated with a context ctx $\in \mathcal{C}$, e.g., document span, region, or temporal window. These values are preserved in the KG and may later be aggregated or filtered via *context-aware property grouping*: in the example of the person "Jordan Kessler" (Figure 2), if the correct birthday 1974 is seen in 7 sources and an incorrect value of 1976 is seen three times, the former achieves a higher frequency and confidence score, ensuring robustness to noise.

4.2 Relation Evolution

Given a set of extracted candidate relations $\hat{\mathcal{E}} = \{(\hat{v}_s, \hat{r}, \hat{v}_t)\}$, the goal is to align or insert each relation into the KG, based on the relation schema $\{(\tau(\hat{v}_s), \phi(\hat{r}), \tau(\hat{v}_t))\}$ and semantics. To perform relation evolution, we propose a two-stage process:

1. Synonym Matching: We first normalize relation types by matching relation schemas. Each candidate relation schema $\hat{s} = (\tau(\hat{v}_s), \phi(\hat{r}), \tau(\hat{v}_t))$ is embedded as $\hat{\boldsymbol{s}} = \text{Enc}(\hat{s})$. We compare it against KG schema embeddings $\{\boldsymbol{s}_r\}_{r \in \mathcal{R}}$ and select the best match: $r^* = \arg\max_{r' \in \mathcal{R}} \text{sim}(\hat{\boldsymbol{s}}, \boldsymbol{s}_{r'})$. We treat $\phi(\hat{r}) \equiv \phi(r^*)$ as a synonym match when $\max s(\hat{\mathbf{s}}, \mathbf{s}_{r^*}) > \theta$, where θ is a preset threshold.

2. Contextual Alignment and Temporal Merging: Assuming the entity alignments $\hat{v}_s \rightarrow v_s$ and $\hat{v}_t \rightarrow v_t$ (from Section 4.1) are complete, we form the candidate edge: $e = (v_s, r^*, v_t, [\tau_{\text{start}}, \tau_{\text{end}}])$, where the temporal range is derived from the context of extraction.

For non-exclusive relations $r^* \in \mathcal{R}_{\text{non-excl}}$, we insert the edge directly, preserving all temporally distinct facts. For exclusive relations $r^* \in \mathcal{R}_{\text{excl}}$, we apply the same confidence-based contradiction handling as in entity merging (see Section 4.1).

4.3 Efficiency Analysis

Let n and m be the number of extracted entities and relations, respectively, and N and M be the number of nodes and edges in the KG, respectively. KG extraction requires one LLM call with cost $O(n+m)$; entity and relation alignment use HNSW-based similarity search (Malkov & Yashunin, 2018) with sublinear complexity $O(n \log N + m \log M)$; merging performs lightweight attribute aggregation with cost $O(n+m)$; and insertion also costs $O(n+m)$. The total complexity is $O(n \log N + m \log M)$. In addition, the evolution uses a constant number of LLM calls per corpus (five in our setup: one for extraction, two for alignment, two for merging), making the pipeline lightweight and practical for continuous large-scale KG updates.

5 Experiments

We evaluate TREK on temporal QA benchmarks and a novel end-to-end setting where the KG is dynamically updated from raw documents by EvoKG. Our experiments are designed to answer two key questions: **Q1**: *Can* EvoReasoner *perform accurate and robust multi-hop reasoning over a temporal knowledge graph?* and **Q2**: *Can* EvoKG *improve question answering accuracy by evolving an incomplete or outdated KG with new information from raw documents?*

Datasets. (1) **KGQA.** To assess multi-hop temporal-reasoning performance, we use two KGQA benchmarks: (a) `TimeQuestions` (Jia et al., 2021), a multi-hop factoid QA dataset over Wikidata requiring answers grounded in specific time periods; and (b) `MultiTQ` (Chen et al., 2023), a large temporal KGQA dataset consisting of 500K unique question-answer pairs, requiring reasoning over multi-hop temporal facts, with varying temporal granularities (day, month, and year), and often involves multiple constraints in the question. 5,000 samples are randomly selected from it and divided into 5 test splits. (2) **End-to-End QA with KG Evolution.** We use two domains from the CRAG benchmark (Yang et al., 2024c): (a) `Movie`, based on IMDB, with questions about films, actors, and awards; and (b) `Sports`, covering basketball and soccer matches across major leagues. Each CRAG question is paired with ground-truth answers, a web document corpus, and a mock domain-specific KG. To simulate real-world KG incompleteness, we downsample key entity types (e.g., only 60% of `Movie` and `Person` nodes are retained in the `Movie` dataset), requiring recovery through document-driven KG updates.

Baselines. We use five state-of-the-art LLMs. We compare our approach against nine baselines grouped into three categories: (1) **LLM-only methods**, including IO Prompt (IO) (Brown et al., 2020), Chain-of-Thought (CoT) (Wei et al., 2022), Self-Consistency (SC) (Wang et al., 2023), and Retrieval-Augmented Generation (RAG) (Lewis et al., 2020); (2) **KG-enhanced methods**, including 1-hop KG (Yang et al., 2024c) (augmenting the LLM with facts from 1-hop KG neighbors of the topic entities) and RAG + 1-hop KG (Yang et al., 2024c); and (3) **Multi-hop graph reasoning methods**, including state-of-the-art approaches on static KG, Think-on-Graph (Sun et al., 2024), Plan-on-Graph (Chen et al., 2024), and on dynamic KGs, TimeR^4 (Qian et al., 2024). We report the average and standard deviation over 5 runs for all models.

5.1 Experimental Results

A1: Comparative Results on Temporal Reasoning. Table 1 summarizes our results on temporal QA benchmarks. Across all datasets and model scales, EvoReasoner consistently outperforms strong baselines. It achieves up to **14.0%** and **4.6%** absolute accuracy improvement over the best baseline in `TimeQuestions` and `MultiTQ`, respectively. Moreover, the performance improves further with larger LLMs, indicating that more capable models better follow EvoReasoner's decomposition strategy, rank relevant relations, and explore local graph neighborhoods. Interestingly, although Qwen 2.5 and LLaMA 3.3 are similar in size, the accuracy of their LLM-only methods diverges, potentially due to differences in the pretraining of these models. However, once grounded in the KG through EvoReasoner, their performance converges. This suggests that EvoReasoner mitigates discrepancies in internal knowledge by anchoring inference in external, structured signals.

On `TimeQuestions`, EvoReasoner improves absolute accuracy by 15–23% compared to the best LLM-only baseline, and by 7–14% over the best KG-enhanced and graph reasoning baselines. For example, on Qwen 2.5–72B, our method improves accuracy from 44.84% (best LLM-only) to 67.76% (+22.9%) and from 56.71% (best KG-based method) to 67.76% (+11.1%). Similarly,

Table 1: Temporal KGQA results (accuracy, %). RAG is not applicable since the datasets do not contain documents.

	Datasets →	`TimeQuestions`					`MultiTQ`				
	Methods↓	Deep-Seek V3 671B	Qwen 2.5 72B	LLaMA 3.3 70B	GPT-4o mini	LLaMA 3.1 8B	Deep-Seek V3 671B	Qwen 2.5 72B	LLaMA 3.3 70B	GPT-4o mini	LLaMA 3.1 8B
LLM-Only	IO	54.36$_{\pm0.57}$	36.31$_{\pm0.16}$	48.56$_{\pm0.37}$	35.73$_{\pm0.28}$	28.04$_{\pm0.26}$	5.20$_{\pm0.90}$	0.50$_{\pm0.17}$	4.20$_{\pm0.46}$	1.37$_{\pm0.45}$	2.92$_{\pm0.48}$
	CoT	50.55$_{\pm0.54}$	44.84$_{\pm0.27}$	53.41$_{\pm0.08}$	41.99$_{\pm0.11}$	38.68$_{\pm0.21}$	2.83$_{\pm0.40}$	0.94$_{\pm0.42}$	4.22$_{\pm0.71}$	5.10$_{\pm0.20}$	6.42$_{\pm0.56}$
	SC	49.49$_{\pm2.23}$	44.30$_{\pm0.15}$	53.80$_{\pm0.20}$	42.16$_{\pm0.30}$	39.98$_{\pm0.33}$	2.43$_{\pm0.71}$	0.76$_{\pm0.29}$	4.34$_{\pm0.40}$	5.03$_{\pm0.40}$	6.12$_{\pm0.90}$
KG-Based	1-hop KG	37.25$_{\pm1.05}$	26.97$_{\pm.31}$	42.27$_{\pm0.09}$	31.55$_{\pm0.16}$	33.75$_{\pm0.27}$	7.57$_{\pm0.95}$	7.14$_{\pm1.22}$	8.28$_{\pm0.99}$	5.87$_{\pm1.16}$	5.06$_{\pm0.35}$
	Think-on-Graph	32.83$_{\pm2.24}$	33.07$_{\pm0.23}$	48.89$_{\pm0.27}$	32.19$_{\pm0.59}$	16.50$_{\pm0.52}$	28.90$_{\pm3.66}$	23.35$_{\pm1.76}$	25.86$_{\pm0.75}$	19.57$_{\pm2.80}$	3.90$_{\pm0.41}$
	Plan-on-Graph	35.16$_{\pm0.91}$	32.28$_{\pm0.11}$	48.58$_{\pm0.16}$	32.27$_{\pm0.26}$	27.58$_{\pm0.15}$	29.90$_{\pm2.81}$	25.16$_{\pm0.80}$	28.08$_{\pm1.04}$	19.77$_{\pm3.69}$	11.04$_{\pm0.58}$
	TimeR[4]	60.19$_{\pm 0.33}$	56.71$_{\pm 0.25}$	56.66$_{\pm 0.17}$	56.82$_{\pm 0.11}$	42.15$_{\pm 2.60}$	37.13$_{\pm 0.48}$	35.55$_{\pm 0.10}$	33.57$_{\pm 0.09}$	32.47$_{\pm 0.49}$	24.54$_{\pm 0.12}$
	EvoReasoner	**67.25$_{\pm1.90}$**	**67.76$_{\pm0.28}$**	**68.63$_{\pm0.27}$**	**65.42$_{\pm0.20}$**	**56.13$_{\pm0.12}$**	**39.07$_{\pm2.26}$**	**39.45$_{\pm0.38}$**	**35.30$_{\pm1.18}$**	**34.03$_{\pm2.27}$**	**29.15$_{\pm2.15}$**

Table 2: End-to-end experiment results (accuracy, %) across LLMs of different scales.

	Datasets →	`Movie`					`Sports`				
	Methods↓	Deep-Seek V3 671B	Qwen 2.5 72B	LLaMA 3.3 70B	GPT-4o mini	LLaMA 3.1 8B	Deep-Seek V3 671B	Qwen 2.5 72B	LLaMA 3.3 70B	GPT-4o mini	LLaMA 3.1 8B
LLM-Only	IO	56.22$_{\pm0.10}$	34.27$_{\pm0.32}$	51.75$_{\pm0.30}$	38.76$_{\pm0.31}$	34.16$_{\pm0.31}$	38.25$_{\pm0.36}$	23.78$_{\pm0.20}$	28.55$_{\pm0.56}$	26.64$_{\pm0.23}$	18.60$_{\pm0.27}$
	CoT	56.52$_{\pm0.37}$	37.27$_{\pm0.43}$	59.75$_{\pm 0.52}$	45.37$_{\pm0.20}$	40.46$_{\pm0.59}$	44.63$_{\pm0.62}$	29.30$_{\pm0.13}$	32.15$_{\pm0.61}$	32.01$_{\pm0.23}$	26.96$_{\pm0.79}$
	SC	57.05$_{\pm0.37}$	37.52$_{\pm0.63}$	59.15$_{\pm0.54}$	45.49$_{\pm0.47}$	43.61$_{\pm1.99}$	45.48$_{\pm0.94}$	29.44$_{\pm0.44}$	33.27$_{\pm0.51}$	32.09$_{\pm0.36}$	26.49$_{\pm0.79}$
	RAG	32.39$_{\pm0.61}$	29.13$_{\pm0.27}$	30.94$_{\pm0.44}$	27.55$_{\pm0.20}$	27.04$_{\pm0.38}$	39.02$_{\pm0.62}$	34.53$_{\pm0.38}$	34.91$_{\pm0.27}$	33.02$_{\pm0.14}$	29.35$_{\pm0.31}$
Original KG	1-hop KG	25.13$_{\pm0.35}$	15.75$_{\pm0.13}$	16.35$_{\pm0.10}$	18.29$_{\pm0.20}$	17.59$_{\pm0.20}$	32.71$_{\pm1.24}$	12.24$_{\pm0.13}$	16.21$_{\pm0.59}$	15.65$_{\pm0.40}$	10.49$_{\pm0.12}$
	RAG + 1-hop KG	33.10$_{\pm1.06}$	31.86$_{\pm0.13}$	31.08$_{\pm0.65}$	28.50$_{\pm0.31}$	27.54$_{\pm0.30}$	36.92$_{\pm1.21}$	33.78$_{\pm0.21}$	32.99$_{\pm0.51}$	30.37$_{\pm0.23}$	28.46$_{\pm0.35}$
	Think-on-Graph	23.42$_{\pm0.51}$	20.53$_{\pm1.02}$	33.98$_{\pm0.99}$	20.83$_{\pm0.37}$	15.36$_{\pm0.93}$	9.89$_{\pm0.36}$	8.69$_{\pm0.51}$	11.73$_{\pm1.16}$	9.42$_{\pm0.67}$	2.20$_{\pm0.27}$
	Plan-on-Graph	21.83$_{\pm0.45}$	15.19$_{\pm0.49}$	32.71$_{\pm0.63}$	12.86$_{\pm0.27}$	17.88$_{\pm0.48}$	15.26$_{\pm0.94}$	8.27$_{\pm0.46}$	15.65$_{\pm0.87}$	12.38$_{\pm0.47}$	4.63$_{\pm0.35}$
	EvoReasoner	61.77$_{\pm 1.45}$	50.55$_{\pm 0.51}$	58.90$_{\pm1.90}$	52.45$_{\pm 1.44}$	45.80$_{\pm 0.29}$	53.79$_{\pm 0.59}$	40.14$_{\pm 0.63}$	42.48$_{\pm 0.98}$	44.39$_{\pm 0.84}$	33.36$_{\pm 1.08}$
Evolved KG	1-hop KG	21.47$_{\pm1.27}$	20.18$_{\pm0.12}$	23.36$_{\pm0.22}$	20.18$_{\pm0.18}$	22.51$_{\pm0.26}$	13.40$_{\pm3.00}$	5.75$_{\pm0.13}$	7.66$_{\pm0.20}$	8.96$_{\pm0.27}$	4.86$_{\pm0.26}$
	RAG + 1-hop KG	34.28$_{\pm0.51}$	33.91$_{\pm0.20}$	34.97$_{\pm0.27}$	28.97$_{\pm0.20}$	30.80$_{\pm0.18}$	31.62$_{\pm1.10}$	29.63$_{\pm0.10}$	32.80$_{\pm0.27}$	28.50$_{\pm0.40}$	26.45$_{\pm0.30}$
	Think-on-Graph	25.09$_{\pm1.17}$	31.82$_{\pm0.34}$	40.21$_{\pm0.51}$	24.60$_{\pm0.18}$	23.65$_{\pm1.37}$	24.92$_{\pm2.86}$	23.55$_{\pm0.56}$	22.29$_{\pm0.27}$	15.81$_{\pm1.41}$	10.28$_{\pm1.10}$
	Plan-on-Graph	23.78$_{\pm0.45}$	31.19$_{\pm0.40}$	41.17$_{\pm0.46}$	24.54$_{\pm0.54}$	24.71$_{\pm0.87}$	28.04$_{\pm1.98}$	23.46$_{\pm0.51}$	28.08$_{\pm0.85}$	16.04$_{\pm1.05}$	13.46$_{\pm0.97}$
	EvoReasoner	**63.13$_{\pm0.10}$**	**57.84$_{\pm0.70}$**	**64.85$_{\pm0.71}$**	**57.52$_{\pm0.53}$**	**51.11$_{\pm0.79}$**	**55.61$_{\pm0.33}$**	**47.90$_{\pm0.23}$**	**51.17$_{\pm0.62}$**	**51.64$_{\pm1.98}$**	**37.06$_{\pm1.58}$**

on LLaMA 3.1-8B, EvoReasoner improves accuracy from 39.98% (LLM-only) and 42.15% (KG-based) to 56.13%, yielding relative gains of +16.2% and +14.0%, respectively.

On the more challenging `MultiTQ` dataset, EvoReasoner maintains its lead despite lower overall accuracy due to higher temporal ambiguity. It improves accuracy by 23–38% compared to the best LLM-only baseline, and outperforms the best KG-enhanced and reasoning baselines by 2–5%.

A2: EvoKG Boosts Reasoning Performance. Table 2 presents QA accuracy before and after applying EvoKG. Due to space constraints, we also present separate KG evolution effectiveness experiment results in Appendix D.2. All multi-hop graph reasoning models, including EvoReasoner, PoG, and ToG, benefit from KG updates, with accuracy gains of 2–8%. In contrast, 1-hop KG and RAG+1-hop KG baselines show limited or even negative gains, especially on `Sports`. This is due to their reliance on indiscriminate context concatenation: 1-hop KG retrieves all neighboring facts until the context window is full. In domains like sports, where a single team can accumulate hundreds of historical matches over multiple seasons, this dense retrieval overwhelms the model and introduces irrelevant information, ultimately degrading performance.

EvoReasoner not only outperforms all baselines on the original KG—as shown in the temporal QA task—but also benefits significantly from KG updates. In both domains, its performance increases by 2–8% after applying EvoKG, validating the quality of the constructed facts. For example, in the `Movie` dataset with GPT-4o-mini, EvoReasoner improves accuracy from 52.45% (original KG) to 57.52% (updated KG), a +5% boost. Moreover, with the help of EvoKG, a small LLaMA 3.1–8B model trained in December 2023 achieves an 18.6 → 37.0% (+18.4%) accuracy gain, which is comparable to directly prompting a 671B DeepSeek-V3 model (38.3%) trained in July 2024. This demonstrates that our framework enables strong QA performance even for smaller models by evolving and grounding the KG appropriately, and closes the gap between the small and large models.

5.2 Ablation Studies on EvoReasoner

We conduct studies on `TimeQuestions` and `Sports`, containing knowledge up to Mar. 2024, to assess the contribution of components in EvoReasoner. All results are reported using LLaMA

3.3 (70B) and LLaMA 3.1 (8B), both with a knowledge cutoff of Dec. 2023, to isolate the effect of reasoning over knowledge in KGs (additional results using different LLMs are available in the Appendix).

Impact of Sub-components. Table 3 shows that removing any sub-component leads to significant accuracy drops: (1) removing multi-route decomposition causes the largest degradation (up to 12%), confirming the importance of soft reasoning guidance for reasoning over KGs; (2) without global initialization, performance drops by 4–9%, showing the need for accurate entity grounding; and (3) removing the temporal-aware reranking component consistently leads to about a 1% drop in accuracy.

Table 3: Ablation studies on EvoReasoner (accuracy, %).

Datasets →	`TimeQuestions`		`Sports`	
Methods↓	LLaMa 3.3 70B	LLaMA 3.1 8B	LLaMa 3.3 70B	LLaMA 3.1 8B
EvoReasoner	**68.63**$_{\pm 0.27}$	**56.13**$_{\pm 0.12}$	**51.17**$_{\pm 0.62}$	**37.06**$_{\pm 1.58}$
w/o Route Decomposition	67.60$_{\pm 0.55}$	43.98$_{\pm 0.61}$	47.90$_{\pm 0.76}$	26.52$_{\pm 1.27}$
w/o Global Initialization	64.12$_{\pm 0.30}$	52.35$_{\pm 0.79}$	42.46$_{\pm 0.84}$	32.67$_{\pm 1.62}$
w/o Time-Aware Local Rerank	67.58$_{\pm 0.29}$	54.57$_{\pm 0.65}$	50.79$_{\pm 1.05}$	35.78$_{\pm 1.16}$

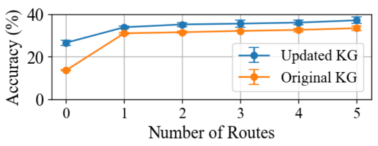

Figure 3: Route-wise performance.

Diverse Routes Improve Accuracy. Figure 3 reports route-wise QA performance in `Sports` on the LLaMA 3.1-8B model. Accuracy improves steadily with more routes, validating that route diversity enhances KG coverage and reasoning robustness. These results underscore the value of both flexible multi-path reasoning and temporal grounding in achieving strong QA performance.

6 Related Work

KG Reasoning. Reasoning over knowledge graphs (KGs) involves drawing structured inferences to support tasks like question answering and decision-making (Pan et al., 2024). Existing methods typically fall into two categories: (1) embedding-based approaches and (2) LLM-driven approaches. Embedding-based methods encode entities and relations into dense vectors to retrieve relevant subgraphs. Recent work such as SR (Zhang et al., 2022), UniKGQA (Jiang et al., 2023c), and ReasoningLM (Jiang et al., 2023b) adopt this strategy, but may struggle with noise and limited interpretability. LLM-driven approaches leverage large language models to guide or perform reasoning. StructGPT (Jiang et al., 2023a) generates structured queries, ToG (Sun et al., 2024) identifies salient subgraphs, and PoG (Chen et al., 2024) decomposes queries into multi-step plans. KG-Agent (Jiang et al., 2024) extends this with tool-augmented traversal. However, most approaches assume a static KG and treat temporal attributes as metadata, limiting their ability to handle evolving knowledge.

KG Extraction. Traditional KG construction techniques often rely on rule-based extraction from curated resources (e.g., YAGO (Suchanek et al., 2007)), but these approaches are brittle and hard to scale. More recent efforts turn to LLMs for end-to-end KG extraction from raw text. Techniques such as JointIE (Qiao et al., 2022), EXTRACT (Zhang & Soh, 2024), and UP-KG (Hatem et al., 2024) formulate KG extraction as a text-to-graph generation task. These techniques demonstrate promising performance but often overwrite existing knowledge without accounting for temporal drift or contradictions. Our work builds on this line of work but introduces a temporal-aware, contradiction-tolerant evolution process, allowing the KG to store multiple values, track confidence, and maintain historical facts over time.

7 Conclusion

We present a unified framework for temporal reasoning over evolving knowledge graphs, that combines a multi-hop reasoning module (EvoReasoner) with a temporal-aware KG evolution method (EvoKG). By jointly addressing contradiction resolution and temporal trend modeling, our method enables accurate and robust reasoning in dynamic knowledge environments. Comprehensive experiments on temporal QA and end-to-end KG update benchmarks show substantial improvements over LLM-only and KG-enhanced baselines. Although our framework demonstrates strong performance, it assumes access to clean temporal cues and may face scalability challenges on very large corpora. Addressing these limitations is an interesting direction for future work.

Reproducibility Statement

We have taken several steps to ensure the reproducibility of our work. The formal definitions of temporal reasoning and KG evolution are presented in Section 2, with complete algorithmic descriptions of EvoReasoner and EvoKG in Sections 3 and 4. Detailed experimental settings, evaluation protocols, and ablation studies are provided in Section 5 and the Appendix, covering all temporal QA and end-to-end KG evolution benchmarks. Full datasets, preprocessing steps, and evaluation splits (TimeQuestions, MultiTQ, and CRAG) are publicly released by their original authors and are properly documented in the supplementary material. We provide an anonymized codebase and implementation scripts, including prompts, training/evaluation pipelines, and KG update utilities needed to reproduce all tables and figures in the main text. These resources collectively ensure that our theoretical formulations and empirical results can be independently validated by the community.

Ethics Statement

This work builds exclusively on publicly-available datasets and open-source large language models, without involving human subjects, private information, or sensitive data. Our focus is on temporal reasoning and evolving knowledge graphs for factual QA, which has potential applications in domains such as scientific discovery and legal/financial analysis. While our framework aims to improve the reliability and timeliness of knowledge-grounded reasoning, it does not guarantee factual correctness and should not be misused in high-stakes decision-making without appropriate human oversight. As with any system that updates knowledge from external text, there remains a risk of propagating biased or adversarial content if data sources are not carefully curated. We encourage responsible use of our methods with attention to source verification, transparency, and fairness. Large language models were also used to assist in refining the writing of this paper, but all scientific ideas, methodology, experiments, and conclusions were developed by the authors.

References

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pp. 1247–1250, 2008.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:1877–1901, 2020.

Kai Chen, Ye Wang, Yitong Li, and Aiping Li. Rotateqvs: Representing temporal information as rotations in quaternion vector space for temporal knowledge graph completion. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5843–5857, 2022.

Liyi Chen, Panrong Tong, Zhongming Jin, Ying Sun, Jieping Ye, and Hui Xiong. Plan-on-graph: Self-correcting adaptive planning of large language model on knowledge graphs. In *Advances in neural information processing systems (NeurIPS)*, 2024.

Ziyang Chen, Jinzhi Liao, and Xiang Zhao. 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)*, pp. 11378–11392, 2023.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Shahenda Hatem, Ghada Khoriba, Mohamed H Gad-Elrab, and Mohamed ElHelw. Up to date: Automatic updating knowledge graphs using llms. *Procedia Computer Science*, 244:327–334, 2024.

IBM. slate-125m-english-rtrvr model card, November 2024. URL `https://dataplatform.cloud.ibm.com/docs/content/wsj/analyze-data/fm-slate-125m-english-rtrvr-v2-model-card.html?context=wx`. Accessed: 2025-05-13.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12): 1–38, 2023.

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

Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen. StructGPT: A general framework for large language model to reason over structured data. In *Conference on Empirical Methods in Natural Language Processing*, pp. 9237–9251, 2023a.

Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, Yaliang Li, and Ji-Rong Wen. ReasoningLM: Enabling structural subgraph reasoning in pre-trained language models for question answering over knowledge graph. In *Conference on Empirical Methods in Natural Language Processing*, pp. 3721–3735, 2023b.

Jinhao Jiang, Kun Zhou, Xin Zhao, and Ji-Rong Wen. UniKGQA: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph. In *The Eleventh International Conference on Learning Representations*, 2023c.

Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, Yang Song, Chen Zhu, Hengshu Zhu, and Ji-Rong Wen. KG-Agent: An efficient autonomous agent framework for complex reasoning over knowledge graph. *arXiv preprint arXiv:2402.11163*, 2024.

Amanda Kau, Xuzeng He, Aishwarya Nambissan, Aland Astudillo, Hui Yin, and Amir Aryani. Combining knowledge graphs and large language models. *arXiv preprint arXiv:2407.06564*, 2024.

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

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Yifan Lu, Yigeng Zhou, Jing Li, Yequan Wang, Xuebo Liu, Daojing He, Fangming Liu, and Min Zhang. Knowledge editing with dynamic knowledge graphs for multi-hop question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 24741–24749, 2025.

LINHAO LUO, Yuan-Fang Li, Reza Haf, and Shirui Pan. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *International Conference on Learning Representations (ICLR)*, 2024.

Yu A Malkov and Dmitry A Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836, 2018.

Belinda Mo, Kyssen Yu, Joshua Kazdan, Proud Mpala, Lisa Yu, Chris Cundy, Charilaos Kanatsoulis, and Sanmi Koyejo. Kggen: Extracting knowledge graphs from plain text with language models. *arXiv preprint arXiv:2502.09956*, 2025.

OpenAI. Gpt-4o mini: advancing cost-efficient intelligence, July 2024. URL `https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/`. Accessed: 2025-05-13.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3580–3599, 2024.

Xinying Qian, Ying Zhang, Yu Zhao, Baohang Zhou, Xuhui Sui, Li Zhang, and Kehui Song. Timer4: 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*, pp. 6942–6952, 2024.

Bo Qiao, Zhuoyang Zou, Yu Huang, Kui Fang, Xinghui Zhu, and Yiming Chen. A joint model for entity and relation extraction based on bert. *Neural Computing and Applications*, pp. 1–11, 2022.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pp. 697–706, 2007.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *International Conference on Learning Representations (ICLR)*, 2024.

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

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. Knowledge editing for large language models: A survey. *ACM Computing Surveys*, 57(3):1–37, 2024.

Song Wang, Junhong Lin, Xiaojie Guo, Julian Shun, Jundong Li, and Yada Zhu. Reasoning of large language models over knowledge graphs with super-relations. *arXiv preprint arXiv:2503.22166*, 2025.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems (NeurIPS)*, 35:24824–24837, 2022.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

Jinfa Yang, Xianghua Ying, Yongjie Shi, and Bowei Xing. Tensor decompositions for temporal knowledge graph completion with time perspective. *Expert Systems with Applications*, 237:121267, 2024b.

Xiao Yang, Kai Sun, Hao Xin, Yushi Sun, Nikita Bhalla, Xiangsen Chen, Sajal Choudhary, Rongze Gui, Ziran Jiang, Ziyu Jiang, et al. Crag-comprehensive rag benchmark. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:10470–10490, 2024c.

Bowen Zhang and Harold Soh. Extract, define, canonicalize: An llm-based framework for knowledge graph construction. *arXiv preprint arXiv:2404.03868*, 2024.

Jing Zhang, Xiaokang Zhang, Jifan Yu, Jian Tang, Jie Tang, Cuiping Li, and Hong Chen. Subgraph retrieval enhanced model for multi-hop knowledge base question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5773–5784, 2022.

A LLM Usage

Large language models (LLMs) were employed in a limited and transparent manner during the preparation of this manuscript. Specifically, LLMs were used to assist with linguistic refinement, style adjustments, and minor text editing to improve clarity and readability. They were not involved in formulating the research questions, designing the theoretical framework, conducting experiments, or interpreting results. All scientific contributions—including conceptual development, methodology, analyses, and conclusions—are the sole responsibility of the authors.

B Broader Impacts

This work contributes to advancing temporal reasoning over evolving knowledge graphs, with potential positive applications in areas such as personal assistants, scientific discovery, legal research, and financial intelligence. By enabling more accurate and context-aware information retrieval, our framework could improve decision-making in dynamic, real-world environments.

However, several risks must also be considered. First, if the KG update mechanism integrates adversarial or biased sources, the system could amplify misinformation. Second, overreliance on automatically updated knowledge bases could reduce human oversight in high-stakes domains. Finally, evolving KGs might be misused for surveillance or automated profiling if applied to personal or sensitive data without consent. Mitigating these risks will require careful data curation, source verification, and transparency in how facts are scored, selected, and surfaced to end users.

C Additional Experiment Details

Datasets. We summarize the statistics of all datasets used in Table 5.

Table 4: Statistics of datasets.

Dataset	# Entities	# Relations	# Entity Types	# Relation Types	QA Pairs	Corpus
`Movie` (Original KG)	265,201	461,772	5	7	565	2,825
`Movie` (Evolved KG)	345,539	566,328	454	7,590	565	2,825
`Sports` (Original KG)	72,442	279,720	5	6	428	2,140
`Sports` (Evolved KG)	131,468	401,662	443	9,151	428	2,140
`TimeQuestions`	113,716	229,767	3	884	3,237	–
`MultiTQ`	10,742	166,760	3	251	5,000	–

Models. We evaluate our methods using five state-of-the-art LLMs, ordered by parameter size: DeepSeek V3 (671B) (Liu et al., 2024), Qwen 2.5 (72B) (Yang et al., 2024a), LLaMA 3.3 (70B) (Grattafiori et al., 2024), GPT-4o-mini (OpenAI, 2024), and LLaMA 3.1 (8B) (Grattafiori et al., 2024). All models follow the same QA protocol for fair comparison. We use a standard lightweight sentence-transformers model (IBM/slate-125m-english-rtrvr-v2 (IBM, 2024)) to generate semantic embeddings. We performed the KG updating using the LLaMA 3.3 (70B) model. All the models are accessible through online API providers, such as OpenAI, DeepSeek, and OpenRouter.

D Additional Experiment Results

D.1 Parameter Sensitivity

We perform ablation by varying the search width and analyze its impact in the following Table 1.

Table 5: Ablation of reasoning search depth (d) vs. width (w) on the evolved Sports KG.

	$d = 1$	$d = 2$	$d = 3$
$w = 1$	$42.99_{\pm 0.40}$	$44.55_{\pm 0.36}$	$44.78_{\pm 0.82}$
$w = 5$	$43.38_{\pm 11.07}$	$49.84_{\pm 2.01}$	$47.43_{\pm 4.25}$
$w = 10$	$43.81_{\pm 10.68}$	$49.69_{\pm 0.94}$	$50.16_{\pm 3.84}$
$w = 20$	$46.82_{\pm 1.21}$	$51.34_{\pm 0.62}$	$51.25_{\pm 0.71}$
$w = 30$	$50.70_{\pm 0.62}$	$51.99_{\pm 0.13}$	$51.64_{\pm 0.79}$

From the results, we observe that increasing the search width for the Relation Selection and Entity Expansion component significantly boosts performance—yielding up to 8% improvement—highlighting the importance of wider exploration in local reasoning.

D.2 KG Construction Effectiveness

To evaluate our framework's ability to reconstruct the KG after downsampling, we measure the fraction of verifiable facts recovered from the corpus, defined as:

$$\text{Recovery Rate} = \frac{\text{Number of facts recovered from corpus}}{\text{Number of verifiable facts in corpus}}.$$

We conducted this evaluation on the most recent MINE benchmark (Mo et al., 2025), which provides 106 document corpora, each associated with 15 manually verified facts. A fact is considered successfully recovered if it can be matched to triplets in the constructed KG.

As shown in Table 6, our framework achieves a 3% higher recovery rate compared to the state-of-the-art KG construction method KGGen, demonstrating that our document-driven KG update process is effective in recovering missing facts.

Table 6: KG reconstruction ablation study.

Methods	OpenIE	GraphRAG	KGGen	EvoKG (Ours)
Facts Captured (Average on the 106 corpus)	29.36%	47.08%	66.46%	**71.36%**

From the results, we observe that EvoKG recovers a higher proportion of verifiable facts than all baselines, outperforming the prior state-of-the-art KGGen by approximately 5%, highlighting the effectiveness of our KG reconstruction pipeline.

D.3 Ablation Studies on More LLMs

We additionally tested the effectiveness of components in EvoReasoner using Qwen 2.5–70B, a LLM from another family, and observed consistent trends: removing key components still caused significant performance drops, confirming the robustness of our method across LLM families.

Table 7: Ablation studies on EvoReasoner.

Datasets	TimeQuestions	Sports (Original KG)	Sports (Evolved KG)
EvoReasoner	**67.76**$_{\pm 0.28}$	**40.14**$_{\pm 0.63}$	**47.90**$_{\pm 0.23}$
w/o Route Decomposition	56.04$_{\pm 1.50}$	18.81$_{\pm 0.98}$	35.95$_{\pm 0.60}$
w/o Global Initialization	60.46$_{\pm 0.35}$	38.63$_{\pm 0.96}$	41.12$_{\pm 1.01}$
w/o Time-Aware Local Rerank	67.05$_{\pm 0.04}$	39.43$_{\pm 1.01}$	46.78$_{\pm 0.32}$

D.4 Empirical Efficiency and Scalability Analysis

This appendix complements the theoretical complexity analysis in Section 4 by providing empirical measurements of runtime, memory usage, token cost, and scaling behavior for both the KG-evolution stage (EvoKG) and the KG-reasoning stage (EvoReasoner). All experiments were conducted using identical hardware and decoding settings across baselines to ensure fair comparison. Specifically, evaluations were performed on a cloud server equipped with a single NVIDIA H100 GPU and running the LLaMA-3.1-8B model.

D.4.1 KG Evolution Efficiency

Per-Document Extraction Latency. To assess the efficiency of KG evolution, we measure the per-document extraction time across all corpora. Figure 4 plots the KG-update latency as a function of document length (in characters). The results exhibit a clear positive trend: longer documents incur proportional increases in processing time. This is expected, as extraction latency is dominated by the LLM forward pass, whose computational cost grows with input token count. We observe no evidence of superlinear blow-up. Instead, the empirical trend aligns with the theoretical $O(n \log n + m \log m)$ complexity of EvoKG's alignment and merge operations. Variance in the vertical direction is attributable to LLM-provider latency: extraction is executed via the OpenRouter API, whose backend routing introduces heterogeneity in response times that is unrelated to algorithmic complexity.

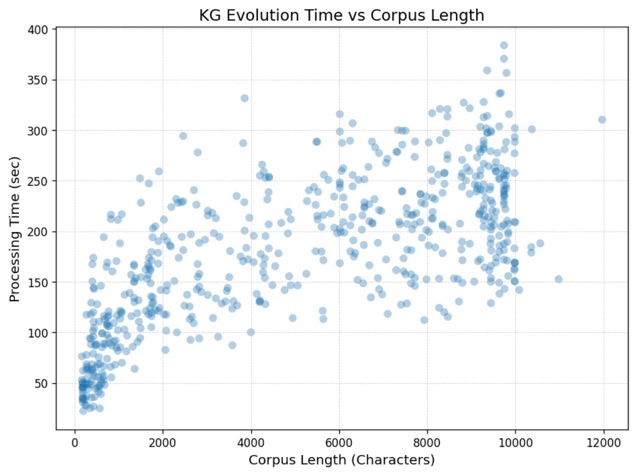

Figure 4: Per-document KG evolution latency as a function of corpus length (in characters). Longer documents incur proportionally higher extraction time, and the trend follows the expected near-linear scaling dominated by LLM token processing.

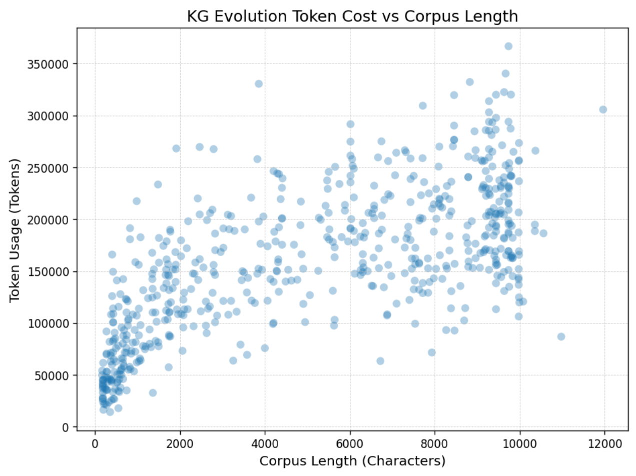

Figure 5: Token usage during KG extraction plotted against corpus length. Token cost grows approximately linearly with input size due to proportional input tokens and a fixed number of extraction calls per document. The smooth scaling pattern confirms that EvoKG's update cost remains predictable and stable across document sizes.

To clarify practical deployment: although Figure 4 reports per-document latency, EvoKG supports data-parallel extraction. When run in parallel, the effective end-to-end KG-update time is substantially lower than the sum of individual extraction times.

Token Cost Scaling. Figure 5 presents token usage versus document length. Token consumption grows approximately linearly with the size of the input, reflecting two factors: (1) input tokens are proportional to corpus length; and (2) output tokens consist of extracted triples and temporal metadata. Because EvoKG performs a fixed number of LLM calls per document, token usage is predictable and stable.

Together, Figure 4 and 5 empirically validate the scalability of EvoKG and demonstrate that both latency and token cost grow smoothly and predictably with corpus size.

D.5 Licenses for Existing Assets

We use the following models and assets in this work, all of which are properly cited and used under their respective licenses:

- **DeepSeek V3 (671B)**: Accessed via DeepSeek API, under the DeepSeek open-source model license available at `https://github.com/deepseek-ai/DeepSeek-LLM/blob/main/LICENSE-MODEL`.
- **Qwen 2.5 (72B)**: Accessed via OpenRouter, based on Qwen model family released by Alibaba under a commercial-use friendly license.
- **LLaMA 3.3 & 3.1 (70B, 8B)**: Accessed via OpenRouter API, follows Meta's LLaMA 3 license available at `https://ai.meta.com/llama/license/`.
- **GPT-4o-mini**: Accessed through OpenAI API, used in accordance with the terms of service and usage policies listed at `https://openai.com/policies/terms-of-use`.
- **IBM Slate-125M-english-rtrvr-v2:** Released by IBM under Apache 2.0 License via `https://huggingface.co/ibm-granite/granite-embedding-125m-english`.

All datasets used in the paper (TimeQuestions, MultiTQ, CRAG benchmark) are publicly released by their authors for research purposes and properly cited in the paper.

Compute Resources. All experiments involving large language models were conducted via public API providers. Specifically, we accessed models through the OpenAI and OpenRouter platforms, which internally manage GPU compute infrastructure. As such, we do not control the underlying hardware but follow their standard rate limits and access conditions.

All preprocessing, knowledge graph construction, and hosting were performed on a cloud server equipped with an Intel Xeon E5-2698 CPU (40 cores, 2.2 GHz) and 500GB of main memory. Each full run of a QA benchmark (including reasoning) takes less than 2–3 hours, depending on model size and API latency. The KG update takes longer, which requires processing more than 3,400 corpora with an average length of 10,000 words, and usually can be accomplished within 24 hours. No dedicated GPU resources were used beyond the API endpoints.

E PROMPT TEMPLATES

E.1

Prompt Templates for Route Planning

You are a helpful assistant who is good at answering questions in the domain domain by using knowledge from an external knowledge graph. Before answering the question, you need to break down the question so that you may look for the information from the knowledge graph in a step-wise operation. Hence, please break down the process of answering the question into as few sub-objectives as possible based on semantic analysis. A query time is also provided; please consider including the time information when applicable.

There can be multiple possible route to break down the question, aim for generating route possible routes. Note that every route may have a different solving efficiency, order the route by their solving efficiency.
Return your reasoning and sub-objectives as multiple lists of strings in a flat JSON of format: "reason": "...", "routes": [[<a list of sub-objectives>], [<a list of sub-objectives>], ...]. (TIP: You will need to escape any double quotes in the string to make the JSON valid)

-Example-
Q: Which of the countries in the Caribbean has the smallest country calling code?
Query Time: 03/05/2024, 23:35:21 PT
Output: { "reason": "The most efficient route involves directly identifying Caribbean countries and their respective calling codes, as this limits the scope of the search. In contrast, routes that involve broader searches, such as listing all country calling codes worldwide before filtering, are less efficient due to the larger dataset that needs to be processed. Therefore, routes are ordered based on the specificity of the initial search and the subsequent steps required to narrow down to the answer.",
"routes": [["List all Caribbean countries", "Determine the country calling code for each country", "Identify the country with the smallest calling code"], ["Identify Caribbean countries", "Retrieve their country calling codes", "Compare to find the smallest"], ["Identify the smallest country calling code globally", "Filter by Caribbean countries", "Select the smallest among them"], ["List all country calling codes worldwide", "Filter the calling codes by Caribbean countries", "Find the smallest one"]] }

Q: <query>
Query Time: <query time>
Output Format (flat JSON): {{"reason": "...", "routes": [[<a list of sub-objectives>], [<a list of sub-objectives>], ...]}}
Output:

Prompt Templates for Context-Aware Global Initialization

-Goal-
You are presented with a question in the domain domain, its query time, and a potential route to solve it.

1) Determine the topic entities asked in the query and each step in the solving route. The topic entities will be used as source entities to search through a knowledge graph for answers. It's preferrable to mention the entity type explictly to ensure a more precise search hit.

2) Extract those topic entities from the query into a string list in the format of ["entity1", "entity2", ...].
Consider extracting the entities in an informative way, combining adjectives or surrounding information.
A query time is provided - please consider including the time information when applicable.

NEVER include ANY EXPLANATION or NOTE in the output, ONLY OUTPUT JSON

-Examples-
<few-shot examples>

Question: <query>
Query Time: <query time>
Solving Route: <route>
Output Format: ["entity1", "entity2", ...]
Output:

Prompt Templates for Global Initialization Relevance Scoring

-Goal-
You are presented with a question in the domain domain, its query time, a potential route to solve it, and a list of entities extracted from a noisy knowledge graph.
The goal is to identify all possible relevant entities to answering the steps in the solving route and, therefore, answer the question.
You need to consider that the knowledge graph may be noisy and relations may split into similar entities, so it's essential to identify all relevant entities.
The entities' relevance would be scored on a scale from 0 to 1 (use at most 3 decimal places, and remove trailing zeros; the sum of the scores of all entities is 1).

-Steps-
1. You are provided a set of entities (type, name, description, and potential properties) globally searched from a knowledge graph that most similar to the question description, but may not directly relevant to the question itself.
Given in the format of "ent_i: (<entity type>: <entity name>, desc: "description", props: {key: [val_1 (70%, ctx:"context"), val_2 (30%, ctx:"context")], ...})"
where "i" is the index, the percentage is confidence score, "ctx" is an optional context under which the value is valid. Each property may have only a single value, or multiple valid values of vary confidence under different context.

2. Score *ALL POSSIBLE* entities that are relevant to answering the steps in the solving route and therefore answering the question, and provide a short reason for your scoring.
Return its index (ent_i) and score into a valid JSON of the format: {"reason": "reason",

"relevant entities": {"ent_i": 0.6, "ent_j": 0.3, ...}}. (TIP: You will need to escape any double quotes in the string to make the JSON valid)

NEVER include ANY EXPLANATION or NOTE in the output, ONLY OUTPUT JSON

-Examples-
<few-shot examples>

Question: <query>
Query Time: <query time>
Solving Route: <route>
Entities: <topk entities str>

Output Format (flat JSON): {"reason": "reason", "relevant_entities": {"ent_i": 0.6, "ent_j": 0.3, ...}}
Output: