REASONING OR RETRIEVAL? DIAGNOSING LLM FAILURES ON TEMPORAL KNOWLEDGE GRAPHS WITH GROUND-TRUTH EVIDENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

When an LLM fails at multi-hop reasoning over an event history, is the model unable to reason or did retrieval fail to provide the evidence it needed? Existing benchmarks cannot disentangle these explanations because the events that caused an outcome are unknown. We introduce a controllable synthetic temporal knowledge graph (TKG) generator that constructs event histories from explicit multi-hop temporal patterns, calibrates them to real event statistics, and records the ground-truth evidence behind every event. This lets us directly separate retrieval from reasoning failures and measure how well retrieval reaches the evidence a query requires. Benchmarking seven LLMs against graph- and rule-based forecasters under a range of retrieval strategies, we find that strong LLMs can overcome increasing hop length when given the right context (the pattern's ground-truth evidence together with one solved example) while neither suffices alone. For example, on 3-hop patterns, GPT-5.6-terra reaches 3.7× the Hits@1 of the GNN forecaster RE-Net. The dominant obstacle is instead retrieval. Under realistic retrieval methods, 82–98% of errors on 2- and 3-hop patterns occur because necessary evidence never reaches the model, and the best existing and proposed retrievers close at most 39% of the gap to the right context. Models remain vulnerable to frequency and recency shortcuts even when the correct evidence is present. Together, these results recast multi-hop temporal reasoning as primarily a retrieval problem. Strong LLMs can often perform the reasoning, but realizing that capability requires retrieval methods that reliably surface the evidence that matters.

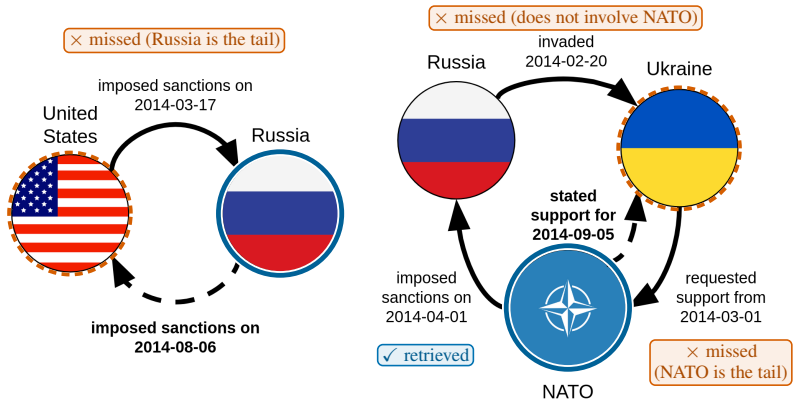
1 INTRODUCTION

Large Language Models (LLMs) are increasingly asked to reason over histories of events: forecasting future events on temporal knowledge graphs (TKGs) (Lee et al., 2023; Liao et al., 2023), answering questions over them (Tan et al., 2026), and serving as the long-term memory for LLM agents (Rasmussen et al., 2025). Given a context of relevant facts, an LLM infers and applies the pattern that predicts the next fact. Yet LLMs struggle with multi-hop reasoning (Wu et al., 2024b; Khodadad et al., 2025; Yang et al., 2025), and multi-hop queries are common in TKG benchmarks. In 45% of ICEWS14 test queries, the answer is not a direct neighbor of the query entity in recent history, and Qwen3-32B's Hits@3 drops from 0.68 on direct neighbor queries to 0.13 on these (Figure 8).

When an LLM fails on such a query, the cause is ambiguous. The model may be unable to compose the evidence, or retrieval may never have supplied the required evidence to make a correct prediction. Consider the 3-hop pattern in Figure 1. To predict that NATO will state support for Ukraine, a model must see that Ukraine requested support *from* NATO, but a naive retriever that collects NATO's own actions never returns that fact. On existing benchmarks, the events that caused an outcome are unknown, so accuracy cannot tell us whether better reasoning or better retrieval is needed, and existing synthetic benchmarks are not calibrated to real event data (Fatemi et al., 2025; Bellomarini et al., 2022).

We build a novel synthetic TKG generator from temporal motifs (Milo et al., 2002; Kovanen et al., 2011), the recurring multi-hop patterns in which earlier events trigger later ones. Our generator controls the hop length of patterns, is calibrated to the statistics of ICEWS14 (Boschee et al., 2015),

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



Query: (Russia, imposed sanctions on, **?**, 2014-08-06)
Head-entity retrieval returns: *nothing*

Query: (NATO, stated support for, **?**, 2014-09-05)
Head-entity retrieval returns: (NATO, imposed sanctions on, Russia)

Figure 1: Two multi-hop patterns from 2014 events. **Left:** 1-hop pattern in which the U.S. imposed sanctions on Russia, which imposed reciprocal sanctions. **Right:** 3-hop pattern in which Russia invaded Ukraine, which requested support from NATO, which imposed sanctions on Russia and later stated support for Ukraine. The overlay shows the limitation of a naive entity-history retriever. For each query (dashed edge, query entity circled in solid blue), it collects facts whose head is the query entity (✓ retrieved) and misses the rest of the evidence (× missed). In both cases, the ground-truth answer (dashed circle) never enters the context, so the query cannot be answered no matter how well a model reasons.

and records, for every event, the earlier facts that caused it. This ground-truth evidence lets us score any retriever against true evidence, attribute each error to retrieval or reasoning, and derive which evidence a retrieval strategy can reach at all. The last analysis reveals blind spots before any model is run. Naive entity-history retrieval cannot reach the correct answer for 7 of 12 pattern shapes, including both examples in Figure 1.

Our experiments with seven LLMs show that the main bottleneck for multi-hop reasoning is retrieval, not reasoning. Given a pattern's ground-truth evidence together with one solved example, strong LLMs overcome hop length and far outperform graph-based forecasters. On 3-hop patterns, GPT-5.6-terra reaches 3.7 times the Hits@1 of RE-Net. Under realistic retrieval, 82–98% of errors on multi-hop patterns are retrieval failures, and the best existing and new retrievers close at most 39% of the gap to this ideal context. Even with the complete evidence in view, models lean on frequency and recency cues. When those cues point away from the correct answer, all six models we test with distractors lose 67–96% of their 3-hop accuracy. Closing this gap calls for retrievers that learn which facts matter for multi-hop reasoning.

Our contributions are:

- A novel synthetic TKG generator with ground-truth evidence, calibrated to real event data, and a suite of pre-built benchmarks from it (Section 3).[1]
- Evidence diagnostics that split each error into a retrieval or a reasoning failure, and a reachability analysis of which evidence each retrieval strategy can reach (Section 4.1 and Section 4.2).
- Three novel label-free retrievers that target the exposed failure modes, including one trained on ground-truth evidence that transfers to longer patterns (Section 4.3).
- A benchmark of seven LLMs, graph-based and rule-based forecasters, and prior retrieval strategies, with controlled distractors that reveal cue reliance (Section 5).

[1]Code and data: https://anonymous.4open.science/r/reasoning-or-retrieval

2 Related Work

TKG forecasting. Rule learners mine temporal rules from random walks over the graph (Liu et al., 2022; Li et al., 2023; Kiran et al., 2023), several with a confidence that depends on the time elapsed since the rule body (Omran et al., 2019; Xiong et al., 2023; Huang et al., 2024; Gastinger et al., 2026). GNN-based forecasters model the evolving graph with recurrent or temporal message-passing modules (Jin et al., 2020; Wu et al., 2020; Li et al., 2022a; Xu et al., 2023b) or exploit repeated facts (Zhu et al., 2021). LLM-based forecasters textualize the history and predict in context (Xu et al., 2023a; Lee et al., 2023; Xia et al., 2024; Liao et al., 2023), and can match task-specific GNNs even when entity and relation names are obfuscated (Lee et al., 2023). Even outside TKGs, LLM accuracy degrades with the number of reasoning hops and often rests on shortcuts (Biran et al., 2024; Wu et al., 2024b; Yang et al., 2025; Khodadad et al., 2025).

Retrieval for LLM-based TKG reasoning. Because context windows are small relative to the complete history of a TKG, recent methods differ mainly in *which* facts they retrieve. Rule-based retrieval grounds mined temporal rules from the query entity (Liao et al., 2023; Akgül et al., 2026), or uses LLM-generated rules that are updated over time (Wang et al., 2024b). Multi-hop history is explored step-by-step with LLM selection (Xia et al., 2024) or by time-monotonic path expansion (Tan et al., 2026). Other methods let a GNN propose candidates and evidence paths that an LLM reranks or edits (Li et al., 2026; Fan et al., 2026). Graph-based RAG more broadly propagates relevance over the graph, e.g., with personalized PageRank (Haveliwala, 2002; Gutiérrez et al., 2024) or over time-aligned rule graphs (Zhu et al., 2025; Han et al., 2025), or lets the LLM traverse the graph iteratively (Sun et al., 2024; Luo et al., 2024). All of these are evaluated on benchmarks where the facts that directly caused an event are unknown, so their retrieval quality can only be judged indirectly through end-task accuracy. Our generator records the ground-truth evidence for every query, which lets us score each family of retrievers directly (Section 4).

Synthetic and diagnostic benchmarks. TKG forecasters are evaluated on subsets of real event corpora such as ICEWS and GDELT (Boschee et al., 2015; Leetaru & Schrodt, 2013), where the causes of an event are unknown. Synthetic temporal data has been generated to benchmark temporal rule reasoners and to test whether rule learners recover gold rules (Bellomarini et al., 2022; Wang et al., 2024a), and to study LLM temporal question answering without pre-training leakage (Fatemi et al., 2025), Cattaneo et al. (2026) use ground-truth answer subgraphs to train and evaluate retrievers on static KGs, and multi-hop QA datasets annotate evidence chains over documents (Ho et al., 2020; Trivedi et al., 2022). We instead generate TKGs for forecasting from explicit multi-hop temporal patterns, calibrate them to real event statistics, and use the recorded evidence to separate retrieval from reasoning failures. Appendix G extends this discussion of related work, including controllable benchmarks, temporal motifs, and case-based reasoning.

3 A Synthetic Benchmark of Temporal Patterns

3.1 Temporal Knowledge Graphs and Patterns

A temporal knowledge graph (TKG) $\mathcal{G} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$ is a set of events (h, r, t, τ) with head and tail entities $h, t \in \mathcal{E}$, relation $r \in \mathcal{R}$, and discrete timestamp $\tau \in \mathcal{T}$. We study forecasting as tail prediction. Given a query $q = (h, r, ?, \tau')$ and only the events before τ', predict the missing tail.

Intuitively, a pattern defines a minimal multi-hop reasoning task, a small set of temporally ordered facts whose joint occurrence implies a future event. A **temporal pattern** $p \in \mathcal{P}$ of hop length k is a tuple $p = (A_p, c_p, \Delta_p)$ with antecedents $A_p = [a_1, \ldots, a_k]$, consequence c_p, and lag constraints $\Delta_p = [\delta_1, \ldots, \delta_k]$, where each $\delta_i = (\delta_i^-, \delta_i^+)$ is an integer interval. Antecedents and consequence are all triples $a_i = (x_i, r_i, y_i)$ and $c_p = (x_c, r_c, y_c)$ over placeholders $x_*, y_* \in \mathcal{V}$ and concrete relations $r_* \in \mathcal{R}$. An **instantiation** $(\pi, \{\tau\}_{1:k+1})$ of p maps placeholders to entities, $\pi : \mathcal{V} \to \mathcal{E}$, such that

$$(\pi(x_i), r_i, \pi(y_i), \tau_i) \in \mathcal{G} \;\; (1 \le i \le k), \qquad (\pi(x_c), r_c, \pi(y_c), \tau_{k+1}) \in \mathcal{G}, \qquad \tau_{i+1} - \tau_i \in [\delta_i^-, \delta_i^+]. \tag{1}$$

In practice, we obtain the pattern set $\mathcal{P}$ by enumerating all 1-, 2-, and 3-hop templates over a small placeholder vocabulary and keeping those that pass structural validity checks, which yields the twelve general pattern shapes of Table 5 (Section A.1).

Dataset	Facts	Entities	Relations	Timestamps	Avg. degree	Gini (ent.)	Gini (rel.)	Queries
ICEWS14	90,730	7,128	230	365	25.5	0.85	0.88	7,371
ICEWS14-Synth	83,106	3,282	207	365	50.6	0.92	0.46	8,468
Only 1-hop	21,322	1,460	54	180	29.2	0.83	0.39	1,128
Only 2-hop	21,474	1,459	71	180	29.4	0.83	0.49	820
Only 3-hop	20,972	1,419	72	180	29.6	0.82	0.52	603

Table 1: Statistics of the real ICEWS14, its calibrated synthetic analogue, and the three hop datasets. Gini coefficients measure the skew of the entity and relation frequency distributions. Queries are all test-split facts for ICEWS14 and ICEWS14-Synth, and just test-split pattern consequences for the hop datasets.

3.2 Generator and Datasets

Starting from time $\tau = 0$, at each time step, our generator (i) injects complete instances of each pattern with a pattern-specific probability and (ii) emits a consequence whenever all antecedents of a pattern are already present in the history (Algorithm 1, Appendix A). It records the pattern instance behind every consequence, which gives each query its *ground-truth evidence*. The generator controls the hop length of the patterns, the temporal dispersion of their elements, the imbalance of entity and relation distributions (via sampling weights), and the size and density of the graph. Realism comes from reproducing the structure and statistics of real event graphs, not from entity- or relation-level semantics (we use numeric identifiers for all entities and relations). When creating a synthetic version of an existing TKG, we calibrate the generator with Bayesian optimization so that the synthetic TKG matches the original in its descriptive statistics and in the accuracy of recency and frequency heuristics (Section A.2). For ICEWS14 (Boschee et al., 2015), the resulting ICEWS14-Synth has similar statistics (Table 1) and ranks predictive methods like the original (over 37 model–method pairs, Kendall's $\tau = 0.50$, Pearson $r = 0.66$, Appendix B). Our main experiments use three datasets that contain only 1-, 2-, or 3-hop patterns and otherwise have similar statistics (Table 1). Seven further datasets vary the time lag between antecedents or the skew of the entity and relation distributions. Both change accuracy much less than hop length, so we report them in Section F.1.

4 Retrieval Strategies and Evidence Diagnostics

An LLM-based forecaster only sees the facts it is given, so its accuracy depends on retrieval as well as reasoning. On real TKGs these dependencies are hard to separate, because the facts that caused an event are unknown. Our generator records, for every consequence, the antecedent facts that produced it, i.e., its *ground-truth evidence*. This lets us (i) score any retriever directly against that evidence, (ii) determine analytically which pattern shapes a retriever can and cannot cover, and (iii) split every prediction error into a retrieval failure or a reasoning failure.

4.1 Evidence Diagnostics

Let $q = (h, r, ?, \tau)$ be a query whose true tail, the ground-truth answer, is $t^\star$. The query was generated by one or more pattern instantiations with antecedent sets $S_1, \ldots, S_m$ (Section 3.1), and C is the set of facts that a retriever places in the context of an LLM-based forecaster. We measure how much of the ground-truth evidence C contains with three quantities:

$$\mathrm{EvRec}(q, C) = \max_j \frac{|S_j \cap C|}{|S_j|}, \quad \mathrm{FullEv}(q, C) = \mathbb{1}\left[\exists j : S_j \subseteq C\right], \quad \mathrm{AnsIn}(q, C) = \mathbb{1}\left[t^\star \in \mathrm{ent}(C)\right], \tag{2}$$

where $\mathrm{ent}(C)$ is the set of entities that appear in C. *Evidence recall* (EvRec) is the fraction of a pattern instance's antecedents that the context contains, for the best-covered instance. *Full evidence* (FullEv) indicates that the context contains every antecedent of at least one instance, which is what a model needs to predict the answer from its pattern. *Answer in context* (AnsIn) indicates that the answer appears in the context at all. Because candidates are restricted to entities in the context, $\mathrm{AnsIn} = 0$ makes a query unanswerable regardless of the model. We label each incorrect prediction (Hits@3 $= 0$) as a *reasoning failure* if $\mathrm{FullEv} = 1$, and as a *retrieval failure* otherwise, further split by whether the answer is in the context. For retrievers that select in-context demonstrations, we also

report *pattern-identification accuracy*, the fraction of demonstrations generated by the same pattern as the query. These diagnostics require ground-truth evidence and cannot be measured on standard TKG benchmarks.

4.2 Reachability of Pattern Evidence

Before running any model, pattern definitions already determine which evidence a retrieval strategy *can* reach from the query. For a pattern whose consequence is the query ($h = \pi(x_c)$, $t^\star = \pi(y_c)$), we consider only the pattern's own antecedents and ask which of them a strategy can reach, starting from h: **N-E** (facts with head h, naive entity retrieval), **N-P** (facts with head h and relation r, naive pair retrieval), **TM↑ / TM↓** (paths from h whose timestamps are non-decreasing / non-increasing moving away from h), and undirected breadth-first search, for which we report the depth needed to reach the answer and all of the evidence. TM↑ matches the cyclic rule bodies of TLogic ($\tau_1 \leq \cdots \leq \tau_l$), which GenTKG also uses for retrieval (Liu et al., 2022; Liao et al., 2023). Table 2 gives the result for every shape in our taxonomy (Table 5) in its canonical orientation and time order ($\tau_1 < \tau_2 < \tau_3$). Section C.3 computes the same quantities empirically for every pattern in our datasets.

Three consequences follow directly from the definitions. First, head-anchored retrieval (N-E) cannot see the answer in 7 of the 12 shapes, including every 3-hop shape except premise re-derivation, which makes those queries unanswerable from pattern evidence under our candidate restriction. Second, no single time direction suffices. TM↑ paths, the form of TLogic's cyclic rule bodies, cannot cover cyclic entailment or bridging composition, while TM↓ paths miss the transitive-closure family. Rule mining empirically confirms this prediction. TLogic recovers none of the bridge, bridging-composition, or cyclic-entailment patterns in our datasets, but 79% of transitive closures (Section C.4, Table 8). Rule learners that relax the time order (Kiran et al., 2023) or allow acyclic rules (Li et al., 2023) are not bound by this particular limit. Third, the *bridge* shape is unreachable from the query head at any depth, because its second antecedent touches only the answer. Only retrieval anchored on candidate answers can recover it, which motivates the candidate-anchored strategies below. Appendix D states these three observations as propositions with proofs, and adds a budget bound. A retriever with no pattern signal to separate ground-truth evidence from other facts, choosing B facts from a neighborhood of N_k facts, contains the full evidence of a k-hop pattern with probability at most $(B/N_k)^k$.

Hops	Shape	N-E	N-P	TM↑	TM↓	BFS depth (ans./all)	Needs answer anchor
1	Self-entailment	●	●	●	●	1 / 1	
1	Inverse	–	–	●	●	1 / 1	
1	Reflexivity	●	●	●	●	0 / 1	
2	Transitive closure	○	–	●	○	2 / 2	
2	Reinforcement of premise	◐	◐	●	◐	1 / 2	
2	Bridge	–	–	○	○	– / –	✓
2	Multi-derivation	●	–	●	●	1 / 1	
3	Multi-step transitive closure	○	–	●	○	3 / 3	
3	Shortcut entailment	○	–	●	○	2 / 3	
3	Premise re-derivation	◐	◐	●	◐	1 / 2	
3	Bridging composition	○	–	○	○	3 / 3	
3	Cyclic entailment	–	–	○	●	3 / 3	

Table 2: Which of a pattern's antecedents each retrieval strategy can reach from the query head, for every pattern shape in canonical orientation and time order. Icons: antecedents are solid, consequence is dashed (full drawings in Table 5). ●: all antecedents and answer. ◐: some antecedents and the answer. ○: some antecedents but no answer. –: none. N-P assumes the consequence relation is new. If it copies an antecedent relation, N-P reaches that antecedent. BFS depth is the number of undirected hops from the head needed to reach the answer / all antecedents, and – means unreachable from the head. For reflexivity the answer is the query head itself (depth 0).

4.3 RETRIEVERS

We evaluate retrievers from five families, ordered in Table 6 by how much they know about the generating pattern. *Local* retrievers return the query head's recent history (N-E, N-P). *Multi-hop structural* retrievers expand paths from the head (breadth-first search, MemoTime- and HippoRAG-style). *Rule-based* retrievers ground mined temporal rules from the head (GenTKG- and RECIPE-style), *case-based* retrievers add solved past cases as demonstrations, and *candidate-anchored* retrievers also expand from candidate answers proposed by a GNN (SKER-style). Each retriever fills a budget of $B = 25$ facts from before the query time τ, and anything learned from data is fit on the training split only. We also compare against label-informed analogies AH, AR, and BAR, which leak queries' generating patterns, so, like OICL, they serve only as references. Appendix C details our reimplementations of prior retrieval methods (marked "-style").

We introduce three label-free retrievers, each built around a property that our benchmark makes measurable. MOTIFCASE recovers the generating pattern from structure alone. It compares the time-ordered, lag-binned, role-anonymized paths between the query head and each candidate with those of past cases of the same relation, and uses the closest cases as demonstrations. Causal anonymous walks (Wang et al., 2021) and G2S (Bai et al., 2025) use similar anonymized structure as a model input, whereas MOTIFCASE uses it as a key for retrieving cases. BALBUNDLE anchors retrieval on candidate answers, the only way to reach bridge evidence (Table 2), and gives every candidate an equal, labeled evidence bundle, which lessens frequency bias. EVRANK is a gradient-boosted classifier trained on the generator's ground-truth evidence with identifier-free features, which lets us test transfer from shorter to longer patterns. Section C.1 provides more details about the implementation of our introduced retrievers. We also evaluate lag-aware rule grounding, following prior rule learners with time-dependent confidence (Omran et al., 2019; Xiong et al., 2023; Huang et al., 2024; Gastinger et al., 2026).

5 EXPERIMENTS

5.1 SETUP

Evaluation. We query only facts that a pattern generated as a consequence, because only these have ground-truth evidence from our generator. So queries are test-split consequences (1,128, 820, and 603 samples on the 1-, 2-, and 3-hop datasets). For each query $(h, r, ?, \tau)$, models rank candidate entities, the entities in its context plus the query head. LLMs see seeded samples of 500 queries per dataset (300 for GPT-4.1 and 200 for the reasoning API models). We report Hits@3, or Hits@1 where Hits@3 saturates, with 95% bootstrap confidence intervals. A second seeded sample for three open models falls inside the first sample's interval (Hits@3) for 151 of 155 conditions. Because the candidate set differs between conditions (an oracle context names only 1.9–3.5 entities on average), we also always report the score of a random ranking over the same candidates.

Models. We evaluate seven LLMs from four families: GPT-4.1, GPT-5-mini, and the reasoning model GPT-5.6-terra (closed-source), gpt-oss-20b (open, reasoning), Qwen3-8B and Qwen3-32B (open), and Llama-3.1-8B (open). Baselines are the graph-based methods RE-Net (Jin et al., 2020) and CyGNet (Zhu et al., 2021), and frequency and recency heuristics over the entity (F-E, R-E) or entity–relation (F-P, R-P) history. LLM contexts are given at most 25 facts, and each entity is written with a short index label that we use when scoring (e.g., $21 : [0.\ 1071, 20, 1.\ 1493]$), which avoids tokenization confounding. Open models are scored exactly, by the log-probability of each candidate's label, and closed-source and reasoning models write a ranked list. The two can disagree, so we also compare models only under one scoring method, on the same 300 queries (Table 13).

Context conditions. **Naive** contexts hold the query head's history (N-E) or its history with the query relation (N-P). **Retriever** contexts come from the strategies of Section 4.3. **Oracle in-context learning** (OICL) gives the model exactly the ground-truth evidence of the query. A **solved example** is an earlier instance of the same pattern, with its own antecedents and consequence. We add it to the OICL context (OICL+ex) or to the naive context (N-E+ex). To test whether models use the evidence or rely on heuristics, we add up to 100 distractor facts to the OICL+ex context, with the ground-truth evidence always present. Distractors are random facts, near-misses that share the evidence's entities or relations, near-misses without the answer (which removes the frequency cue), or cue-adversarial

	1-hop				2-hop				3-hop			
Model	N-E	N-E+ex	OICL	OICL+ex	N-E	N-E+ex	OICL	OICL+ex	N-E	N-E+ex	OICL	OICL+ex
Llama-3.1-8B	0.12	0.08	0.54	0.28	0.07	0.12	0.35	0.27	0.07	0.07	0.29	0.18
Qwen3-8B	0.12	0.06	0.53	0.24	0.08	0.04	0.35	0.27	0.06	0.05	0.27	0.21
Qwen3-32B	0.25	0.31	0.50	0.55	0.16	0.19	0.32	0.43	0.12	0.15	0.29	0.34
gpt-oss-20b	**0.35**	0.30	0.51	0.77	0.27	0.26	0.32	0.65	0.12	0.16	**0.31**	0.60
GPT-5-mini	**0.35**	0.25	**0.55**	0.71	**0.31**	0.29	0.32	0.57	0.11	0.12	0.22	0.47
GPT-4.1	0.34	**0.40**	0.50	0.86	0.24	0.28	0.26	0.67	0.12	0.15	0.28	0.63
GPT-5.6-terra	**0.35**	0.33	0.54	**0.96**	0.28	**0.33**	0.29	**0.76**	**0.13**	**0.17**	0.24	**0.77**
Random ranking	0.10	0.07	**0.55**	0.28	0.07	0.05	**0.38**	0.20	0.06	0.04	0.30	0.16

Table 3: Separating the retrieval and reasoning interventions (Hits@1 on the hop datasets). N-E: the query head's entity history. N-E+ex: N-E plus a solved earlier instance of the query's pattern (the reasoning intervention). OICL: exactly the ground-truth evidence (the retrieval intervention). OICL+ex: the ground-truth evidence plus the same solved example (both interventions). **Bold** marks the best value in each column and underlining the second best, with the random ranking included, so a marked random cell means that no model beats chance. Last row: Hits@1 of a random ranking over the same candidates (oracle contexts contain only 2–4 entities, so Hits@3 is saturated there). Graph baselines (all entities as candidates) reach Hits@1 of 0.46/0.31/0.21 (RE-Net) and 0.41/0.26/0.17 (CyGNet). All ten datasets are in Table 11.

facts that make both the most frequent and the most recent entity differ from the answer. The last three use the known answer and are diagnostic interventions, not realistic retrieval methods. Appendix E defines every condition, including label-informed analogy methods that use the generating pattern, and shows example prompts.

5.2 Results

Q1: What does it take to overcome hop length? With naive entity history, accuracy falls as patterns lengthen. GPT-5.6-terra's Hits@1 drops from 0.35 to 0.28 and 0.13 from 1- to 3-hop patterns, and every other model follows the same trend (Table 3). The reachability analysis explains much of this drop, because head-anchored retrieval cannot reach the answer of any 3-hop shape except premise re-derivation (Section 4.2). Table 3 separates two interventions. Giving a model exactly the ground-truth evidence (OICL, the retrieval intervention) leaves every model at chance. Without knowing the pattern, the model cannot tell which entity the consequence points to. Adding a solved example of the pattern to the naive context (N-E+ex, the reasoning intervention) barely helps either. It changes Hits@1 by -0.10 to $+0.06$ across models and hop lengths, because the entity history rarely contains the evidence that the example would teach the model to use. The two together (OICL+ex) overcome hop length, but only for strong models. GPT-5.6-terra reaches Hits@1 of 0.96, 0.76, and 0.77 on 1-, 2-, and 3-hop patterns, GPT-4.1 0.86, 0.67, and 0.63, and gpt-oss-20b 0.77, 0.65, and 0.60. These scores are far above chance (0.28, 0.20, 0.16) and above RE-Net (0.46, 0.31, 0.21), which GPT-5.6-terra beats 3.7 times over on 3-hop patterns. The 8B models stay near chance even with both interventions. The ground-truth evidence plus one solved example is therefore the context under which multi-hop forecasting becomes solvable, and should be the goal for realistic retrieval methods.

Q2: Do models reason over the evidence or follow heuristic cues? Figure 2 adds 50 distractor facts to the OICL+ex context, with the ground-truth evidence and the solved example still present. Random distractors lower every model's Hits@3 but leave it far above chance. Only GPT-5.6-terra (0.91, 0.73, and 0.55 on 1-, 2-, and 3-hop patterns) beats a heuristic that ranks entities by their most recent appearance in the context (0.76, 0.61, and 0.49). Distractors that turn the frequency and recency cues against the answer are far more damaging. Cue-adversarial facts cut the 3-hop Hits@3 of all six models we test with distractors by 67–96% (Llama-3.1-8B, scored exactly: 0.52 to 0.08). GPT-5.6-terra falls from 0.89 to 0.27, gpt-oss-20b from 0.72 to 0.24, and Qwen3-32B from 0.63 to 0.08, below the random-ranking baseline of 0.09. Paired comparisons on the same distractor draws show that both cues matter, and gpt-oss-20b, the only model that ignores recency, still loses 0.13–0.15 to the frequency cue alone (Figure 9). Even with the complete evidence in view, current LLMs lean on frequency and recency shortcuts rather than identifying and composing a pattern from noisy contexts.

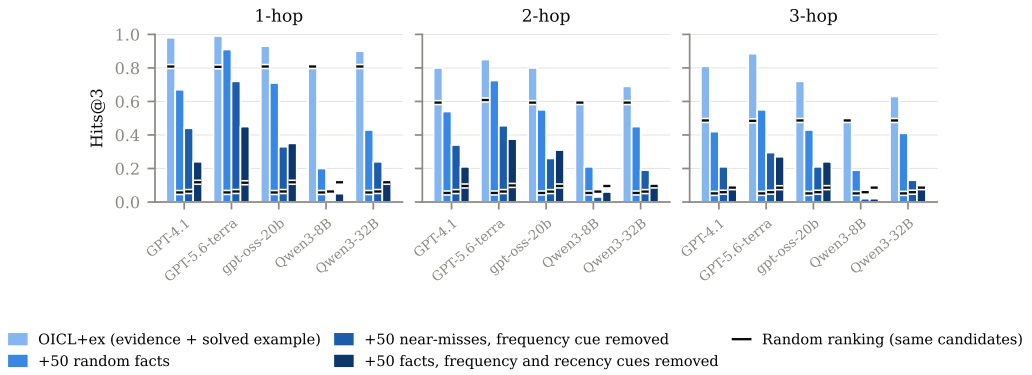


Figure 2: Frequency and recency cue reliance. Hits@3 with the ground-truth evidence and one solved example (OICL+ex, lightest bar), and after adding 50 facts that are random, near-misses with the frequency cue removed, or cue-adversarial (frequency and recency both point away from the answer). Black ticks: random ranking over the same candidates. All models write ranked lists on the same 300 queries (Table 13), except GPT-5.6-terra (subset of 200 queries for cost). The full sweep over the number of distractors, including the other models, is in Figure 9.

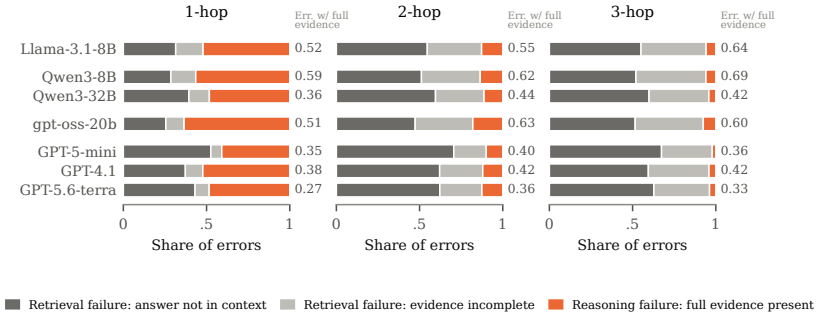


Figure 3: Sources of error with realistic, label-free retrieval, pooled over retrieval methods. Each bar splits a model's errors into retrieval failures (answer absent from the context, or evidence incomplete) and reasoning failures (full evidence present). Numbers: error rate when the full evidence is in the context, to put the otherwise normalized shares in context.

Q3: Where do errors come from under realistic retrieval? Figure 3 splits every error made with realistic, label-free retrieval into retrieval failures, where the context lacks part of the ground-truth evidence, and reasoning failures, where the full evidence is present. It pools 15 label-free retrievers (Table 6) for Llama-3.1-8B, both Qwen3 models, and GPT-4.1, and 3–7 for the others. Retrieval failures account for 37–59% of errors on 1-hop patterns and for 82–98% on 2- and 3-hop patterns, for every model. Realistic retrievers deliver the complete evidence for only 4–20% of 2- and 3-hop queries. When the full evidence is in the context, the error rate of large models stays roughly flat as patterns lengthen (Qwen3-32B: 0.36, 0.44, and 0.42 for 1, 2, and 3 hops), whereas it rises for 8B models (Llama-3.1-8B: 0.52 to 0.64, Qwen3-8B: 0.59 to 0.69). Retrieval is therefore the dominant bottleneck, and compositional depth is a second difficulty that mainly affects smaller models.

Q4: Which retrievers reach the evidence? Figure 4 relates each retriever's full-evidence rate to its downstream accuracy. Across label-free retrievers, the rank correlation between the two is 0.81, 0.59, and 0.70 at 1, 2, and 3 hops, so predictive performance tends to follow evidence. The standard entity history delivers the full evidence for 44%, 8%, and 0% of 1-, 2-, and 3-hop queries. Multi-hop retrievers from prior work (breadth-first search and RECIPE-, HippoRAG-, and SKER-style) reach 0.25–0.29 at 2 hops and 0.09–0.14 at 3 hops. Our MOTIFCASE matches the best of these at 2 hops (0.29) and exceeds it at 3 hops (0.16). It also identifies the generating pattern of its demonstration for 35–58% of queries, against 12–18% for same-relation demonstrations. Our

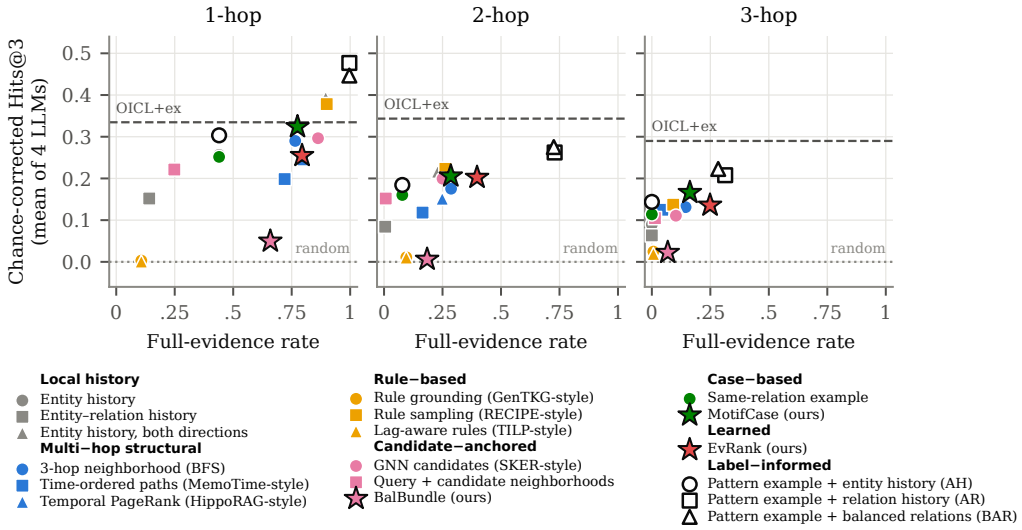


Figure 4: Evidence predicts accuracy. Each point is a retriever. x: the rate at which its context contains the full ground-truth evidence (retrieval only, all test queries). y: chance-corrected Hits@3, $(\text{Hits@3} - \text{random})/(1 - \text{random})$, where "random" is the Hits@3 of a random ranking over the same candidates, so 0 means chance. Values are averaged over Llama-3.1-8B, Qwen3-8B, Qwen3-32B, and GPT-4.1. Color: retriever family. Stars: our retrievers. Open markers: label-informed references. Dashed: OICL+ex. The rank correlation over label-free retrievers is 0.81, 0.59, and 0.70 for 1-, 2-, and 3-hop patterns.

EvRank, trained on ground-truth evidence, reaches the highest full-evidence rate (0.40 and 0.25), and 0.17 at 3 hops when trained only on 1- and 2-hop data. The label-informed AR reaches 0.73 and 0.31, which bounds what knowing the pattern is worth for retrieval. Our BalBundle removes the frequency shortcut (wrong answers follow the most frequent entity 17–19% of the time instead of 43–53%), but accuracy does not rise (Table 13). None of these methods closes the gap to OICL+ex on multi-hop patterns. Averaged over four LLMs, the best label-free retriever closes 34% of the gap in chance-corrected Hits@3 between entity history and OICL+ex at 2 hops (RECIPE-style) and 38% at 3 hops (MotifCase). Frontier reasoning models do not change this. For GPT-5.6-terra, MotifCase, its best retriever, closes 44%, 39%, and 27% of the gap at 1, 2, and 3 hops.

6 Conclusion

We introduced a controllable synthetic benchmark for multi-hop reasoning over TKGs that records the ground-truth evidence behind every event. Given the pattern's ground-truth evidence and one solved example, strong LLMs overcome hop length and far outperform graph-based forecasters. Under realistic retrieval, however, most errors on multi-hop patterns are retrieval failures, and existing and new retrievers close at most 39% of the gap to this ideal context. Retrieval is therefore the primary bottleneck for LLM-based TKG reasoning, and robustness to frequency and recency cues the next one. The most direct next step is to learn retrieval: representations of entities, relations, and facts trained, perhaps with our generator's supervision, to surface the evidence a multi-hop pattern needs. EvRank is a first step in that direction.

Limitations and future work. Our generator makes simplifying assumptions that future work can relax. Patterns span one to three hops, whereas real events may follow longer chains and combinations of patterns. Entities and relations are numeric identifiers, and every synthetic fact comes from a pattern, so richer semantics and unexplained background events are missing. Annotating the causes of real events could bring these diagnostics to real TKGs.

Reproducibility Statement

All code, generated benchmark datasets, and result summaries behind every table and figure are available in an anonymized repository. Its README gives the commands that regenerate the datasets, rerun the evaluation, and rebuild the tables and figures, and all random choices are seeded. The data-generating process is summarized in Appendix A (Algorithm 1) and the calibration procedure used to match ICEWS14 in Section A.2. The repository's `docs/supplement.pdf` gives pseudo-code for every generator component. Descriptive statistics for every generated dataset are given in Table 4, prompt formats and retrieval strategies in Appendix E, and complete Hits@3 results in Appendix F. Hits@1, Hits@10, and MRR with 95% confidence intervals are in the repository's result summaries.

AI Use Statement

We used generative AI tools in several parts of this work. AI coding assistants implemented parts of the evaluation pipeline, the reimplemented and proposed retrieval methods, and the analysis and plotting scripts. The authors specified these implementations, reviewed the code, and checked its outputs. The research questions, synthetic data generator, and experimental design are the authors' own. We have reviewed all AI-assisted work and take full responsibility for the content of this paper, including all code, claims, and results.

References

Ömer Faruk Akgül, Feiyu Zhu, Yuxin Yang, Rajgopal Kannan, and Viktor Prasanna. RECIPE-TKG: From sparse history to structured reasoning for LLM-based temporal knowledge graph completion. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2026.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

İbrahim Bahadır Altun and Ahmet Erdem Sarıyüce. Stochastic event prediction via temporal motif transitions. *arXiv preprint arXiv:2603.05874*, 2026.

Long Bai, Zixuan Li, Xiaolong Jin, Jiafeng Guo, Xueqi Cheng, and Tat-Seng Chua. G2S: A general-to-specific learning framework for temporal knowledge graph forecasting with large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, 2025.

Luigi Bellomarini, Markus Nissl, and Emanuel Sallinger. iTemporal: An extensible generator of temporal benchmarks. In *IEEE 38th International Conference on Data Engineering (ICDE)*, 2022.

Eden Biran, Daniela Gottesman, Sohee Yang, Mor Geva, and Amir Globerson. Hopping too late: Exploring the limitations of large language models on multi-hop queries. *arXiv preprint arXiv:2406.12775*, 2024.

Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. ICEWS coded event data. *Harvard Dataverse*, 12(2), 2015.

Alberto Cattaneo, Carlo Luschi, and Daniel Justus. Ground-truth subgraphs for better training and evaluation of knowledge graph augmented LLMs. *Transactions on Machine Learning Research*, 2026.

Rajarshi Das, Ameya Godbole, Nicholas Monath, Manzil Zaheer, and Andrew McCallum. Probabilistic case-based reasoning for open-world knowledge graph completion. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay-Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. Case-based reasoning for natural language queries over knowledge bases. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

Yufeng Du, Minyang Tian, Srikanth Ronanki, Subendhu Rongali, Sravan Bodapati, Aram Galstyan, Azton Wells, Roy Schwartz, Eliu A. Huerta, and Hao Peng. Context length alone hurts LLM performance despite perfect retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, 2025.

Shiqi Fan, Quanming Yao, Hongyi Nie, Wentao Ma, Zhen Wang, and Wen Hua. Bridging graph structure and knowledge-guided editing for interpretable temporal knowledge graph reasoning. *arXiv preprint arXiv:2601.21978*, 2026.

Bahare Fatemi, Mehran Kazemi, Anton Tsitsulin, Karishma Malkan, Jinyeong Yim, John Palowitch, Sungyong Seo, Jonathan Halcrow, and Bryan Perozzi. Test of time: A benchmark for evaluating LLMs on temporal reasoning. In *International Conference on Learning Representations*, 2025.

Julia Gastinger, Christian Meilicke, and Heiner Stuckenschmidt. CountTRuCoLa: Rule learning for interpretable temporal knowledge graph forecasting. In *International Semantic Web Conference*, 2026.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did Aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Jiale Han, Austin Cheung, Yubai Wei, Zheng Yu, Xusheng Wang, Bing Zhu, and Yi Yang. RAG meets temporal graphs: Time-sensitive modeling and retrieval for evolving knowledge. *arXiv preprint arXiv:2510.13590*, 2025.

Bing Hao, Minglai Shao, Zengyi Wo, Yunlong Chu, Yuhang Liu, and Ruijie Wang. LLMTM: Benchmarking and optimizing LLMs for temporal motif analysis in dynamic graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

Taher H. Haveliwala. Topic-sensitive PageRank. In *Proceedings of the 11th International Conference on World Wide Web*, pp. 517–526, 2002.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*, 2020.

Rikui Huang, Wei Wei, Xiaoye Qu, Shengzhe Zhang, Dangyang Chen, and Yu Cheng. Confidence is not timeless: Modeling temporal validity for rule-based temporal knowledge graph forecasting. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10783–10794, 2024.

Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. Recurrent event network: Autoregressive structure inference over temporal knowledge graphs. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6669–6683, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.541. URL https://aclanthology.org/2020.emnlp-main.541/.

Daniel Kaiser, Arnoldo Frigessi, Ali Ramezani-Kebrya, and Benjamin Ricaud. CogniLoad: A synthetic natural language reasoning benchmark with tunable length, intrinsic difficulty, and distractor density. *arXiv preprint arXiv:2509.18458*, 2025.

Mohammad Khodadad, Ali Shiraee Kasmaee, Mahdi Astaraki, Nicholas Sherck, Hamidreza Mahyar, and Soheila Samiee. Evaluating multi-hop reasoning in large language models: A chemistry-centric case study. *arXiv preprint arXiv:2504.16414*, 2025.

Rage Uday Kiran, Abinash Maharana, and Krishna Reddy Polepalli. A novel explainable link forecasting framework for temporal knowledge graphs using time-relaxed cyclic and acyclic rules. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 264–275, 2023.

Lauri Kovanen, Márton Karsai, Kimmo Kaski, János Kertész, and Jari Saramäki. Temporal motifs in time-dependent networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(11): P11005, 2011.

Dong-Ho Lee, Kian Ahrabian, Woojeong Jin, Fred Morstatter, and Jay Pujara. Temporal knowledge graph forecasting without knowledge using in-context learning. *arXiv preprint arXiv:2305.10613*, 2023.

Kalev Leetaru and Philip A Schrodt. GDELT: Global data on events, location, and tone, 1979–2012. In *ISA annual convention*, volume 2, pp. 1–49. Citeseer, 2013.

Bo Li, Bin Song, Tiantian He, and Yew-Soon Ong. A structural knowledge enhanced re-ranking method with large language models for temporal knowledge graph prediction. *Pattern Recognition*, pp. 113834, 2026.

Ningyuan Li, E Haihong, Shi Li, Mingzhi Sun, Tianyu Yao, Meina Song, Yong Wang, and Haoran Luo. TR-Rules: Rule-based model for link forecasting on temporal knowledge graph considering temporal redundancy. In *Findings of the association for computational linguistics: EMNLP 2023*, pp. 7885–7894, 2023.

Yujia Li, Shiliang Sun, and Jing Zhao. TiRGN: Time-guided recurrent graph network with local-global historical patterns for temporal knowledge graph reasoning. In *IJCAI*, pp. 2152–2158, 2022a.

Zixuan Li, Zhongni Hou, Saiping Guan, Xiaolong Jin, Weihua Peng, Long Bai, Yajuan Lyu, Wei Li, Jiafeng Guo, and Xueqi Cheng. HiSMatch: Historical structure matching based temporal knowledge graph reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, 2022b.

Ruotong Liao, Xu Jia, Yunpu Ma, and Volker Tresp. GenTKG: Generative forecasting on temporal knowledge graph. *arXiv preprint arXiv:2310.07793*, 2023.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

Yushan Liu, Yunpu Ma, Marcel Hildebrandt, Mitchell Joblin, and Volker Tresp. TLogic: Temporal logical rules for explainable link forecasting on temporal knowledge graphs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 4120–4127, 2022.

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *International Conference on Learning Representations*, 2024.

Pablo Sanchez Martin, Tarek Besold, and Priyadarshini Kumari. FRUNI and FTREE synthetic knowledge graphs for evaluating explainability. In *XAI in Action: Past, Present, and Future Applications*, 2023.

Christian Meilicke, Melisachew Wudage Chekol, Manuel Fink, and Heiner Stuckenschmidt. Reinforced anytime bottom up rule learning for knowledge graph completion. *arXiv preprint arXiv:2004.04412*, 2020.

Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. GSM-Symbolic: Understanding the limitations of mathematical reasoning in large language models. In *International Conference on Learning Representations*, 2025.

Minh-Vuong Nguyen, Linhao Luo, Fatemeh Shiri, Dinh Phung, Yuan-Fang Li, Thuy Vu, and Gholamreza Haffari. Direct evaluation of chain-of-thought in multi-hop reasoning with knowledge graphs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 2862–2883, 2024.

Pouya Ghiasnezhad Omran, Kewen Wang, and Zhe Wang. Learning temporal rules from knowledge graph streams. In *AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering*, 2019.

Ashwin Paranjape, Austin R. Benson, and Jure Leskovec. Motifs in temporal networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 601–610, 2017.

Mohammad Ramezanali, Mo Vazifeh, and Paolo Santi. seqBench: A tunable benchmark to quantify sequential reasoning limits of LLMs. *arXiv preprint arXiv:2509.16866*, 2025.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. Zep: A temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*, 2025.

Harshay Shah, Badih Ghazi, Yangsibo Huang, Ravi Kumar, Da Yu, and Chiyuan Zhang. Do language models robustly acquire new knowledge? In *AI That Keeps Up: NeurIPS 2025 Workshop on Continual and Compatible Foundation Model Updates*, 2025.

Yongxue Shan, Meihan Wu, Cundi Fang, Jie Peng, and Xiaodong Wang. Ontology-guided evidence path inference for multi-hop knowledge graph question answering. *arXiv preprint arXiv:2606.28076*, 2026.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed Chi, Nathanael Schärli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Sola Shirai, Debarun Bhattacharjya, and Oktie Hassanzadeh. Event prediction using case-based reasoning over knowledge graphs. In *Proceedings of the ACM Web Conference 2023*, 2023.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M. 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*, 2024.

Xingyu Tan, Xiaoyang Wang, Qing Liu, Xiwei Xu, Xin Yuan, Liming Zhu, and Wenjie Zhang. MemoTime: Memory-augmented temporal knowledge graph enhanced large language model reasoning. In *Proceedings of the ACM Web Conference 2026*, 2026.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

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

Dingmin Wang, Przemysław A. Wałęga, and Bernardo Cuenca Grau. MTLearn: Extracting temporal rules using datalog rule learners. In *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning*, 2024a.

Jiapu Wang, Kai Sun, Linhao Luo, Wei Wei, Yongli Hu, Alan Wee-Chung Liew, Shirui Pan, and Baocai Yin. Large language models-guided dynamic adaptation for temporal knowledge graph reasoning. In *Advances in Neural Information Processing Systems*, volume 37, 2024b.

Yanbang Wang, Yen-Yu Chang, Yunyu Liu, Jure Leskovec, and Pan Li. Inductive representation learning in temporal networks via causal anonymous walks. In *International Conference on Learning Representations*, 2021.

Jian Wu, Linyi Yang, Manabu Okumura, and Yue Zhang. MRKE: The multi-hop reasoning evaluation of LLMs by knowledge edition. *CoRR*, 2024a.

Jian Wu, Linyi Yang, Zhen Wang, Manabu Okumura, and Yue Zhang. CoFCA: A step-wise counterfactual multi-hop QA benchmark. *arXiv preprint arXiv:2402.11924*, 2024b.

Jiapeng Wu, Meng Cao, Jackie Chi Kit Cheung, and William L Hamilton. TeMP: Temporal message passing for temporal knowledge graph completion. *arXiv preprint arXiv:2010.03526*, 2020.

Yuwei Xia, Ding Wang, Qiang Liu, Liang Wang, Shu Wu, and Xiao-Yu Zhang. Chain-of-history reasoning for temporal knowledge graph forecasting. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 16144–16159, 2024.

Siheng Xiong, Yuan Yang, Faramarz Fekri, and James Clayton Kerce. TILP: Differentiable learning of temporal logical rules on knowledge graphs. In *International Conference on Learning Representations*, 2023.

Wenjie Xu, Ben Liu, Miao Peng, Xu Jia, and Min Peng. Pre-trained language model with prompts for temporal knowledge graph completion. *arXiv preprint arXiv:2305.07912*, 2023a.

Yi Xu, Junjie Ou, Hui Xu, and Luoyi Fu. Temporal knowledge graph reasoning with historical contrastive learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 4765–4773, 2023b.

Sohee Yang, Nora Kassner, Elena Gribovskaya, Sebastian Riedel, and Mor Geva. Do large language models perform latent multi-hop reasoning without exploiting shortcuts? In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 3971–3992, 2025.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Cunchao Zhu, Muhao Chen, Changjun Fan, Guangquan Cheng, and Yan Zhang. Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 4732–4740, 2021.

Zulun Zhu, Haoyu Liu, Mengke He, and Siqiang Luo. Right answer at the right time – temporal retrieval-augmented generation via graph summarization. *arXiv preprint arXiv:2510.16715*, 2025.

A Generator Details

This section collects the full generation algorithm, illustrations of pattern shapes and complexity dimensions, and descriptive statistics of all datasets.

Notation. We define sampling distributions over entities and relations, represented by non-negative weight functions $w_E : \mathcal{E} \to \mathbb{R}_{\geq 0}$ and $w_R : \mathcal{R} \to \mathbb{R}_{\geq 0}$. We write $x \sim W_E$ and $r \sim W_R$ for draws from the corresponding normalized distributions.

Finite pattern set $\mathcal{P}$ contains patterns $p \in \mathcal{P}$, each of which is a tuple $p = (A_p, c_p, \Delta_p)$ of hop-length k, as defined in Section 3.1. We associate each pattern p with two control parameters, forcing probability $\rho_{\text{force}}(p) \in [0, 1]$ and forcing frequency $n_{\text{force}}(p) \in \mathbb{N}$. At each time step, we perform $n_{\text{force}}(p)$ independent Bernoulli trials with success probability $\rho_{\text{force}}(p)$ to decide how many explicit instantiations of p to inject into the synthetic TKG.

We also maintain a label map $\ell : E \times R \times E \times T \to 2^{\mathcal{P}}$, where $\ell(h, r, t, \tau)$ is the set of patterns that generated the quadruple (h, r, t, τ). By convention $\ell(h, r, t, \tau) = \emptyset$ if no pattern is responsible (to allow for extensions of the synthetic TKG with "noise" facts). Finally, for a fact $f = (h, r, t, \tau)$ we write $\tau(f)$ for its time coordinate.

Algorithm 1 Synthetic multi-hop TKG generation (detailed)

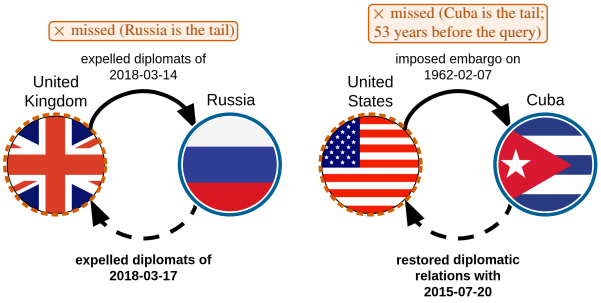
Require: Entity set E, relation set R, time horizon T, maximum hop length $H \in \{1, 2, 3\}$, template libraries $\{\tilde{\mathcal{P}}^{(k)}\}_{k=1}^{H}$, entity and relation weight functions w_E, w_R, time-lag profiles Lag, desired number of instantiated patterns $\{M_k\}_{k=1}^{H}$, forcing parameters for each pattern (probability ρ_{force} and frequency n_{force})

Ensure: Synthetic temporal knowledge graph G and instantiated pattern set $\mathcal{P}$

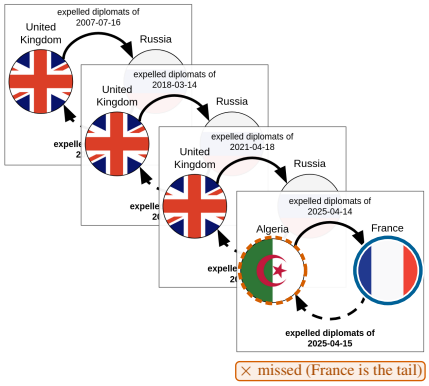
$\mathcal{P} \leftarrow \emptyset$; $G \leftarrow \emptyset$; $\ell \leftarrow$ empty label map ▷ initialize pattern set, graph, and pattern labels
for $k = 1$ to H **do** ▷ sample M_k concrete k-hop patterns
  for $m = 1$ to M_k **do**
    $\tilde{p} \leftarrow \textsc{SampleTemplate}(\tilde{\mathcal{P}}^{(k)})$ ▷ choose a k-hop template over placeholders
    $\pi \leftarrow \textsc{SampleEntityBinding}(w_E)$ ▷ map placeholders to concrete entities
    $\{r_i\} \leftarrow \textsc{SampleRelations}(\tilde{p}, w_R)$ ▷ instantiate relations in antecedents and consequence
    $\Delta_p \leftarrow \textsc{SampleLags}(\text{Lag}, k)$ ▷ choose a lag profile for this pattern
    $(\rho_{\text{force}}(p), n_{\text{force}}(p)) \leftarrow \textsc{SampleForcingParams}(k)$
    $p \leftarrow \textsc{InstantiatePattern}(\tilde{p}, \pi, \{r_i\}, \Delta_p)$ ▷ construct concrete pattern $p = (A_p, c_p, \Delta_p)$
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{p\}$
  end for
end for
for time $\tau = 1$ to $|T|$ **do** ▷ simulate the temporal KG over time
  for pattern $p \in \mathcal{P}$ **do**
    $N \sim \text{Binomial}\big(n_{\text{force}}(p), \rho_{\text{force}}(p)\big)$ ▷ number of forced instantiations at time τ
    for $j = 1$ to N **do** ▷ inject explicit pattern instances
      $(G, \ell) \leftarrow \textsc{InjectForcedInstance}(G, \ell, p, \tau)$
    end for ▷ emit consequences implied by existing antecedents
    $(G, \ell) \leftarrow \textsc{CloseImpliedConsequences}(G, \ell, p, \tau)$
  end for
end for
return $G, \mathcal{P}$

		$\lvert\mathcal{G}\rvert$	$\lvert E\rvert$	$\lvert R\rvert$	$\lvert T\rvert$	Avg. Deg	Avg. Events/T	Gini(E)	Gini(R)
ICEWS	ICEWS14	90,730	7,128	230	365	25.46	248.58	0.85	0.88
	ICEWS14-Synth	83,106	3,282	207	365	50.64	227.69	0.92	0.46
Hops	Only 1-hop	21,322	1,460	54	180	29.21	118.46	0.83	0.39
	Only 2-hop	21,474	1,459	71	180	29.44	119.30	0.83	0.49
	Only 3-hop	20,972	1,419	72	180	29.56	116.51	0.82	0.52
Time lag	$\mathcal{U}(1, 3)$	21,438	1,464	67	180	29.29	119.10	0.83	0.47
	$\mathcal{U}(1, 5)$	21,402	1,449	71	180	29.54	118.90	0.83	0.49
	$\mathcal{U}(1, 8)$	21,360	1,446	66	180	29.54	118.67	0.83	0.49
E/R dist	Ent-Γ Rel-Γ	21,246	1,662	62	180	25.57	118.03	0.50	0.48
	Ent-Γ Rel-$\mathcal{U}$	14,112	1,583	76	180	17.83	78.40	0.49	0.32
	Ent-$\mathcal{U}$ Rel-Γ	21,384	1,800	65	180	23.76	118.80	0.18	0.44
	Ent-$\mathcal{U}$ Rel-$\mathcal{U}$	21,396	1,800	80	180	23.77	118.87	0.18	0.30

Table 4: Descriptive statistics for datasets (and reference ICEWS14) across our synthetic generator runs.



(a) Time dispersion: **(left)** quick pattern in which the U.K. expelled Russian diplomats and 3 days later Russia did the same, **(right)** slow pattern in which the U.S. imposed an embargo on Cuba in 1962 and it took 53 years for diplomatic relations to be restored. For both queries (dashed edges), head-entity retrieval would return nothing.



(b) Imbalanced entity distribution: one pattern (Country A expels Country B's diplomats, followed by a reciprocal expulsion) instantiated three times with the U.K. and Russia and once with Algeria and France. For the last query, head-entity retrieval again would return nothing.

Figure 5: Real-world examples of two dimensions of pattern complexity controlled by our generator. As in Figure 1, the query head is circled in solid blue, the answer in a dashed orange circle, and × tags mark evidence that head-entity retrieval misses.

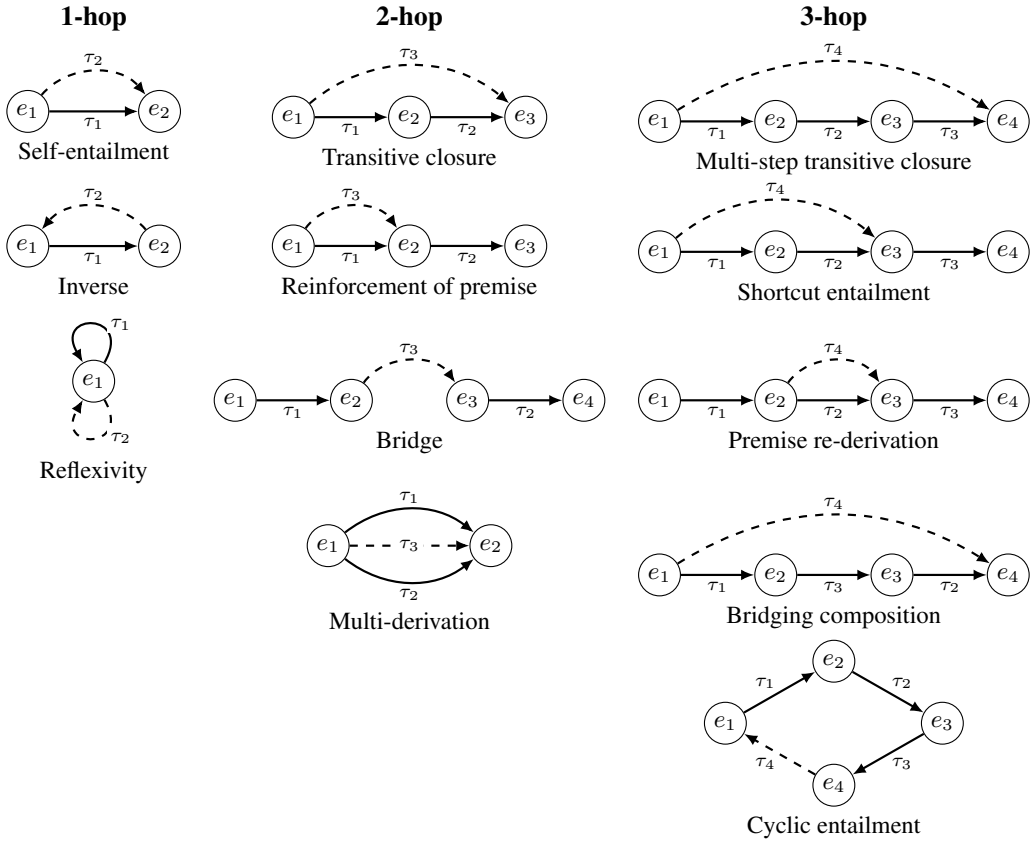


Figure 6: The general *shapes* of the patterns in our generator. Each panel shows entity connectivity only (relation labels are suppressed). τ_i gives the order of events. Antecedent facts are solid and the consequence is dashed.

A.1 Pattern Shapes and Control Parameters

Pattern Control Parameters. Each pattern $p = (A_p, c_p, \Delta_p)$ in our generator is associated with three families of control parameters: (i) *structural parameters* including hop length and entity overlap regime, (ii) *temporal parameters* defined by per-step lag intervals Δ_p, and (iii) *instantiation parameters* including forcing probability $\rho_{\text{force}}(p)$ and forcing frequency $n_{\text{force}}(p)$.

These parameters allow us to independently control the depth of relational composition (via hop length), the temporal dispersion of evidence (via lag intervals), and the density and salience of historical context (via instantiation controls). This disentanglement enables targeted evaluation of multi-hop reasoning behavior along a single axis at a time, as reflected in the experimental sweeps in Section 5.2 and Figure 10.

Relation to Mined KG Patterns. Although our generator does not enumerate all possible logical rules that may arise in real-world KGs, prior work on static and temporal pattern mining (Meilicke et al., 2020; Liu et al., 2022; Li et al., 2023) consistently recovers a small set of recurring structural motifs: transitive closures, inverses, reinforcements, bridges, and cyclic patterns. These motifs are directly represented in our pattern taxonomy (Table 5), up to temporal ordering and relation orientation.

Our goal is not exhaustive rule coverage, but controlled evaluation. By restricting to a finite yet expressive family of commonly observed motifs, we trade completeness for interpretability and experimental control, which is essential for isolating failure modes in multi-hop reasoning.

Pattern enumeration. Table 5 enumerates the structural *shape templates* used by our generator (1-, 2-, and 3-hops), which define the variable-sharing structure between antecedents and consequence. We obtain the finite pattern set $\mathcal{P}$ by exhaustively enumerating all 1-, 2-, and 3-hop templates over a small placeholder vocabulary and filtering them with structural validity checks. Constraints on entity reuse, optional prohibition of self-connections, and a connectivity requirement under which the antecedents and the consequence form a single cycle in the induced entity graph. On top of these shapes, the generator applies independent control regimes for entity reuse, relation reuse, temporal lags, and global validity constraints (for example, a single flag prohibits self-connections), so that one factor can be varied while the others are held fixed. The repository's `docs/supplement.pdf` gives these regimes as tables, together with pseudo-code for pattern enumeration, the validity checks, pattern instantiation, and the matching of satisfied antecedents.

A.2 Generator Calibration

To obtain ICEWS14-Synth we calibrate the generator hyperparameters so that the resulting synthetic TKG matches a reference TKG (ICEWS14) along simple descriptive statistics and heuristic baseline performance (pseudo-code in the repository's `docs/supplement.pdf`).

We use Bayesian optimization to search over hyperparameters controlling the number and type of patterns, their forcing parameters, and the entity/relation sampling distributions. For each proposed configuration, we run the generator once, compute descriptive statistics on the synthetic TKG (number of entities, relations, timestamps, edges, and degree quantiles), and evaluate the recency and frequency baselines. These measurements are combined into a scalar objective, and the optimizer (a TPE-based sampler) iteratively proposes new configurations that minimize this objective over a budget of 800 trials (Akiba et al., 2019).

The calibration objective has two terms. The first is a weighted sum of relative errors between synthetic and reference descriptive statistics, encouraging the generator to match the size, sparsity, and degree structure of the target TKG. The second term compares the Hits@K scores of recency and frequency baselines on synthetic and reference data, penalizing configurations whose heuristic baselines behave very differently from those on the original dataset. A single hyperparameter λ controls the trade-off between matching descriptive statistics and matching baseline behavior. In practice, this calibration yields synthetic TKGs that closely resemble ICEWS14 at both the structural and behavioral levels. The synthetic datasets of each TKG dimension family (number of hops, time lag, entity/relation distribution) are calibrated with the same method to similar descriptive statistics.

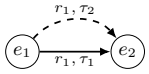
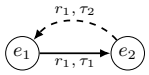
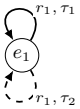
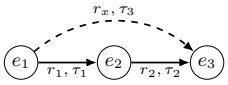
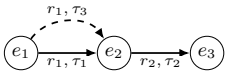
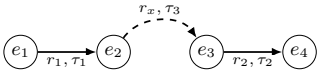
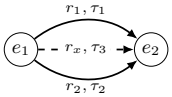
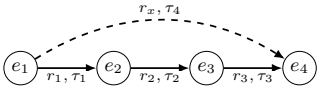
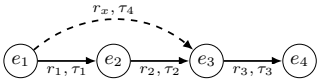
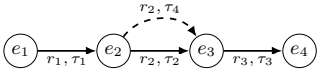
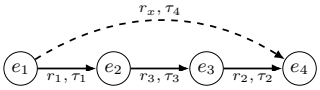
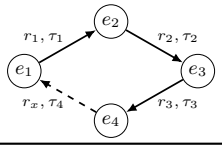
Hops	Shape	Description	Illustration	Logical form
1	**Self-entailment**	Antecedent and consequence share the same endpoints (e.g., reinforcement, repetition).		(e_1, r_1, e_2, τ_1) $\rightarrow (e_1, r_1, e_2, \tau_2)$
1	**Inverse**	Same endpoints, reversed direction (e.g., reciprocation, reply).		(e_1, r_1, e_2, τ_1) $\rightarrow (e_2, r_1, e_1, \tau_2)$
1	**Reflexivity**	At least one triple has $h = t$.		(e_1, r_1, e_1, τ_1) $\rightarrow (e_1, r_1, e_1, \tau_2)$
2	**Transitive closure**	Antecedents share one entity; the consequence closes the triangle by connecting the open ends.		$(e_1, r_1, e_2, \tau_1) \wedge$ (e_2, r_2, e_3, τ_2) $\rightarrow (e_1, r_x, e_3, \tau_3)$
2	**Reinforcement of premise**	Antecedents share one entity; the consequence repeats one antecedent edge.		$(e_1, r_1, e_2, \tau_1) \wedge$ (e_2, r_2, e_3, τ_2) $\rightarrow (e_1, r_1, e_2, \tau_3)$
2	**Bridge**	Antecedents are disjoint components; the consequence must connect them.		$(e_1, r_1, e_2, \tau_1) \wedge$ (e_3, r_2, e_4, τ_2) $\rightarrow (e_2, r_x, e_3, \tau_3)$
2	**Multi-derivation**	Both antecedents use the same unordered entity pair.		$(e_1, r_1, e_2, \tau_1) \wedge$ (e_1, r_2, e_2, τ_2) $\rightarrow (e_1, r_x, e_2, \tau_3)$
3	**Multi-step transitive closure**	Three antecedents form a length-3 path; the consequence closes the cycle by connecting the endpoints.		$\bigwedge_{i=1}^{3}(e_i, r_i, e_{i+1}, \tau_i)$ $\rightarrow (e_1, r_x, e_4, \tau_4)$
3	**Shortcut entailment**	Length-3 path; the consequence links an endpoint to an interior node.		$\bigwedge_{i=1}^{3}(e_i, r_i, e_{i+1}, \tau_i)$ $\rightarrow (e_1, r_x, e_3, \tau_4)$
3	**Premise re-derivation**	Length-3 path; the consequence repeats one earlier edge.		$\bigwedge_{i=1}^{3}(e_i, r_i, e_{i+1}, \tau_i)$ $\rightarrow (e_2, r_2, e_3, \tau_4)$
3	**Bridging composition**	Early antecedents are disjoint; a later antecedent joins them, and the consequence connects the endpoints.		$(e_1, r_1, e_2, \tau_1) \wedge$ (e_3, r_2, e_4, τ_2) $\wedge (e_2, r_3, e_3, \tau_3)$ $\rightarrow (e_1, r_x, e_4, \tau_4)$
3	**Cyclic entailment**	All four edges form a single simple cycle.		$\bigwedge_{i=1}^{3}(e_i, r_i, e_{i+1}, \tau_i)$ $\rightarrow (e_4, r_x, e_1, \tau_4)$

Table 5: Shape taxonomy of our synthetic data generator. This covers all of our valid pattern shapes, up to variation in relation orientation and temporal ordering of the quadruples. Antecedents are solid and the consequence is dashed. Edge labels give the relation and time.

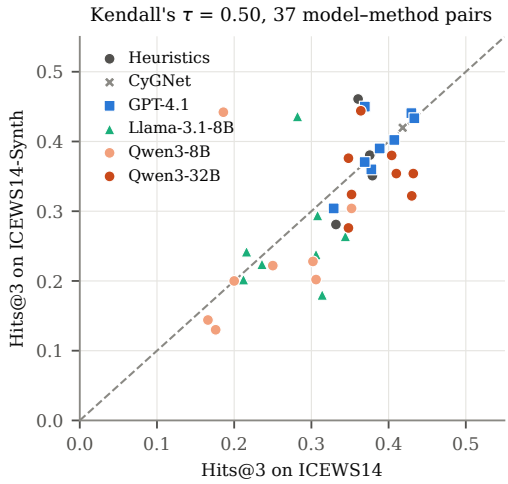


Figure 7: External validity. Hits@3 of each model and method on real ICEWS14 (x) against ICEWS14-Synth (y), for four LLMs over eight retrieval methods, the heuristic baselines, and a graph baseline CyGNet under identical label-free conditions. Dashed: identity.

B Validation of the Synthetic Data

As an external validity check on our generator, we compare model behavior on a real temporal event dataset with its synthetic analogue. To construct the synthetic analogue (ICEWS14-Synth), we calibrate generator hyperparameters so that the resulting synthetic TKG matches ICEWS14 along descriptive statistics and heuristic baseline behavior (Section A.2). We then run the same label-free conditions on both datasets (over all test facts, 7,371 queries on ICEWS14 and 8,468 on ICEWS14-Synth) and compare the resulting scores (Figure 7). Pooled over four LLMs, the heuristic baselines, and CyGNet (37 model–method pairs), scores on the two datasets agree (Kendall's $\tau = 0.50$, Pearson $r = 0.66$, both $p < 0.001$, mean absolute difference 0.05), and the agreement is stronger after subtracting each condition's random-ranking baseline (Kendall's $\tau = 0.60$, Pearson $r = 0.78$). CyGNet, which ranks all entities (not just candidates from retrieved context), reaches Hits@3 of 0.418 on ICEWS14 and 0.420 on ICEWS14-Synth. Per model, with eight retrieval methods each, agreement is strong for GPT-4.1 ($r = 0.75$, mean absolute difference 0.018) but slightly weaker for Qwen3-32B and Llama-3.1-8B. One systematic difference is self-loops. ICEWS14-Synth contains 28% self-loop facts and ICEWS14 none (Section 3.2). Regenerating ICEWS14-Synth without self-loops breaks the calibration (the graph grows to 161,000 facts with a median degree of 33), so we report the calibrated version and state this difference as a limitation.

Self-connections. Reflexive patterns, in which an antecedent or the consequence links an entity to itself, make up 91 of 149 1-hop patterns, 88 of 150 2-hop patterns, and 81 of 150 3-hop patterns, and in 60% of 1-hop queries (674 of 1,128) the answer is the query head itself. We keep them because removing them would change the calibrated statistics of every dataset, and because every method sees the same data, there is no confounding of the relative performance of our compared methods. The main consequence of their inclusion is that some 1-hop queries are easy for any method that ranks the query head highly, which the random-ranking baseline comparison accounts for. The generator can prohibit self-connections with a single flag.

Real names instead of identifiers. Replacing numeric identifiers with real ICEWS entity and relation names changes the Hits@3 of GPT-4.1 by at most 0.05 and of Qwen3-32B by at most 0.08 in any condition. The 8B models benefit more, mainly under random distractors (up to $+0.18$ for Qwen3-8B and $+0.21$ for Llama-3.1-8B). Names do not remove the cue reliance of Figure 2. For Llama-3.1-8B, making the cues point away from the answer still costs 0.07, 0.20, and 0.19 Hits@3 at 1, 2, and 3 hops with real names (0.11, 0.18, and 0.17 with identifiers).

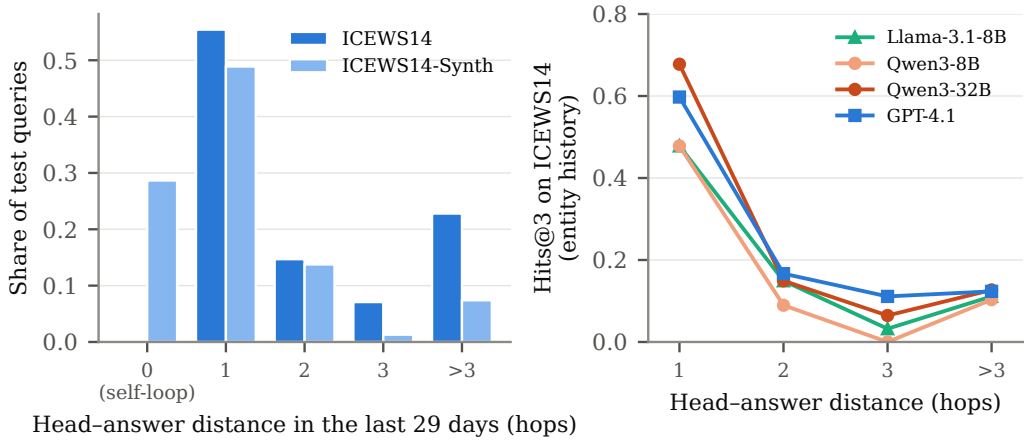


Figure 8: Multi-hop structure in real data. Left: distribution of the shortest head–answer distance over the facts of the preceding 29 days, for the test queries of ICEWS14 and ICEWS14-Synth (distance 0 means the answer is the head). Right: Hits@3 on ICEWS14 with entity-history context, by that distance.

Stability across query samples. A second seeded sample for Llama3.1-8B, Qwen3-8B, and Qwen3-32B falls inside the first sample's bootstrapped 95% confidence interval of Hits@3 for 151 of 155 conditions, with no difference in Hits@3 above 0.05.

Multi-hop structure in ICEWS14. To check that multi-hop queries occur in real benchmarks, we measure, for each test query, the shortest undirected path between the query head and the answer among the facts of the preceding 29 days (a reasonable timespan for the temporal dispersion of ICEWS14). On ICEWS14, the answer is directly linked to the head for 55% of test queries, two hops away for 15%, three hops away for 7%, and not connected within three hops for 23% (Figure 8). Note, this distance is only a lower bound on the hop length of the assumed underlying pattern. On ICEWS14-Synth, where the generating pattern is known, background facts and self-loops often link head and answer directly, so that only 8 of 735 test queries generated by 3-hop patterns have distance 3. For 45% of ICEWS14 queries, the answer is therefore not a direct neighbor of the head, so they require evidence beyond the head's direct neighborhood, and accuracy drops sharply for them. Pooled over these queries, Hits@3 with entity-history context falls from 0.68 to 0.13 for Qwen3-32B and from 0.60 to 0.14 for GPT-4.1. With entity-history context, GPT-4.1 reaches Hits@3 of 0.60, 0.17, and 0.11 at distances 1, 2, and 3, and Qwen3-32B 0.68, 0.15, and 0.06. Note, at distance 3 these rates rest on 18 and 31 queries.

C Retrieval Strategy Details

C.1 Our Label-Free Retrievers

We introduce three label-free retrievers, each built around a property that our benchmark makes measurable. We additionally evaluate *lag-aware* rule grounding, which ranks grounded rule bodies by how likely the consequence is to fall due at τ given the delays observed in training. The idea comes from prior rule learners with time-dependent confidence (Omran et al., 2019; Xiong et al., 2023; Huang et al., 2024; Gastinger et al., 2026). Our time-lag datasets let us test directly whether it improves retrieval.

MotifCase: case-based retrieval by temporal motif. The label-informed analogy methods AH, AR, and BAR (Appendix E) help because each demonstration comes from the query's pattern, but they obtain that pattern from generator labels. MotifCase recovers it from structure alone. For an entity pair (x, y) at time τ', it builds a signature from the paths of length at most 3 between x and y in a recent time window, together with the facts attached to either endpoint. Each path is written as

Family	Retrievers	Labels used	Anchor
Local	Entity (N-E), pair (N-P), undirected entity	none	query
Multi-hop structural	k-hop BFS, time-monotone paths (MemoTime-style (Tan et al., 2026)), temporal PPR (HippoRAG-style (Gutiérrez et al., 2024))	none	query
Statistical	Rule grounding (GenTKG-style (Liao et al., 2023), RECIPE-style (Akgül et al., 2026)), lag-aware rule grounding (TILP/CountTRuCoLa-style (Xiong et al., 2023; Gastinger et al., 2026)), **EvRank**	none at test time	query
Case-based	Same-relation demonstrations, **MotifCase**	none	query + cand.
Candidate-anchored	GNN-proposed rerank (SKER-style (Li et al., 2026)), bidirectional, **BalBundle**	none	query + cand.
Label-informed	AH, AR, BAR (Section 5.1)	pattern	query
Oracle	OICL	instance	–

Table 6: Retrieval strategies ordered by how much they know about the generating pattern. Ours in **bold**. "-style" marks our reimplementation of a prior method's retrieval component (Appendix C).

a sequence of (relation, direction, binned time lag) steps, and intermediate entities are replaced by anonymous roles. We index the signature of every training fact under its relation. At query time, we compute signatures between h and each nearby candidate entity, retrieve the past cases of relation r with the most similar signatures (IDF-weighted Jaccard), and present them, with their outcomes, as demonstrations. The signature resembles a causal anonymous walk (Wang et al., 2021) and the anonymous temporal structures of G2S (Bai et al., 2025). Those methods, however, use anonymized structure as a model input or a training format, whereas MotifCase uses it as a key for retrieving cases. Unlike case-based reasoning over static KGs (Das et al., 2020; 2021; Shirai et al., 2023), which compare entities by their relations and types, our cases are matched on time-ordered, lag-aware structure. Our benchmark also lets us measure directly whether the retrieved cases come from the right pattern.

BalBundle: equal evidence per candidate. Two observations motivate anchoring retrieval on candidate answers: (i) the bridge shape is reachable only from the answer side (Table 2) and (ii) LLMs can exploit how often an entity appears in the context (frequency cue bias) rather than the structure that connects it to the query. BalBundle takes the top-m candidates from cheap proposers (frequency and RE-Net) and retrieves, for each, the shortest paths linking it to h plus the facts attached to it. It gives every candidate the same share of the budget, and presents the evidence as labeled bundles in random order. GNN-proposed reranking (Li et al., 2026; Fan et al., 2026) and bidirectional retrieval (Shan et al., 2026) also use candidate-side evidence. Our tweak is to equalize it, so that comparing BalBundle with its unbalanced variant measures how much the model relies on frequency.

EvRank: a retriever trained on ground-truth evidence. Because the generator provides evidence labels, it can also serve as training data. EvRank is a gradient-boosted classifier that scores each candidate fact as evidence for the query. Its features contain no entity or relation identifiers. They cover structure (hop distance and direction relative to h, whether the fact lies on a time-monotone path), time (lag and recency rank), and relation statistics estimated on the training split. This lets us test generalization across dataset families (e.g., training on 1- and 2-hop patterns and testing on 3-hop). Cattaneo et al. (2026) train retrievers on ground-truth subgraphs for static KGQA. We study the temporal forecasting setting and transfer across pattern families.

C.2 Implementation Details

Shared setup. All retrievers see only facts with timestamps before τ and return at most $B = 25$ facts, plus up to B facts per in-context demonstration. Multi-hop retrievers are restricted to a window $[\tau - W, \tau)$, where W is the largest cumulative time lag of any pattern in the dataset, and they expand at most $F = 10$ of the most recent edges per node. Rule confidences, relation statistics, lag histograms, and learned parameters are estimated on the training split. Unless a serialization

Retriever	What we implement (hyperparameters)	What we leave out
Undirected entity	25 most recent facts with h as head or tail	–
k-hop BFS	Undirected BFS from h in $[\tau - W, \tau)$, depth $d \in \{1, 2, 3\}$, fan-out 10, ranked by depth, then recency	–
MemoTime-style	Simple paths from h of length ≤ 3 with non-decreasing (TM↑) or non-increasing (TM↓) times, fan-out 10, whole paths packed by recency of the outermost (TM↑) / innermost (TM↓) edge	LLM-selected operator toolkits, memory
HippoRAG-style	Personalized PageRank from h ($\alpha = 0.15$, 30 iterations) over time-decayed window edges, facts scored by decay $\times$ endpoint PPR mass	LLM extraction of an open KG
GenTKG-style	Groundings of the top-50 rules with head relation r from h (TLogic time order), facts ranked by confidence $\times$ decay	Few-shot instruction tuning
RECIPE-style	Body facts of the top-20 rule groundings, padded with recent undirected 1-hop history	Contrastive adapter fine-tuning, test-time semantic filtering
Same-relation demos	1 demonstration: most recent past fact with relation r (head $\neq h$ preferred) with its head's 25-fact history	–
Bidirectional	Depth-1 BFS from $h \cup 10$ most recent window facts of each of 10 proposed candidates (proposer: reciprocal-rank fusion of Frequency top-10 and RE-Net top-10), recency-ranked	–
MOTIFCASE	$L = 3$, lag bins $\{1, 2, 3\text{–}4, 5\text{–}8, 9\text{–}16, > 16\}$, 5 attached facts per endpoint, 2,000 cases per relation, similarity weights 1 / 0.5 / 0.25 (not tuned), top-3 case mean, $m = 5$ candidates, 1 demonstration, up to 50 candidates from 2-hop window + frequency top-10	–
BALBUNDLE	$m = 5$ candidates (same proposer as Bidirectional), per candidate $\lfloor B/m \rfloor$ facts: shortest h–candidate paths (≤ 3) + 2 attached facts, bundles in seeded random order, ranking restricted to the m candidates	–
SKER-style	RE-Net top-10 candidates, up to $\lfloor B/10 \rfloor$ facts per candidate from shortest h–candidate paths (≤ 3, window, most recent first), leftover budget from h's recent history, ranking restricted to the 10 candidates	Temporal-weighted GNN evidence selection, dual-identifier constraint
TILP/CountTRuCoLa-style	Groundings ranked by confidence $\times$ per-rule Laplace-smoothed delay histogram (relation-level backoff below 20 groundings)	Differentiable rule learning, parametric confidence functions

Table 7: Retrieval components we implement for prior methods and for our label-free retrievers.

ablation says otherwise, the context is shown as a flat, time-sorted list in the index-prompting format of Section 5.1.

Reimplemented baselines. Table 7 summarizes each retriever we implement, its hyperparameters, and what our reimplementations of prior methods leave out.

MOTIFCASE. The signature of an ordered entity pair (x, y) at time τ' has three parts: (i) the multiset of simple undirected paths of length at most $L = 3$ from x to y in $[\tau' - W, \tau')$, each written as a sequence of steps $(r_i, \mathrm{dir}_i, \mathrm{bin}(\tau' - \tau_i))$, (ii) the multiset of facts attached to y but not on any path, written as $(r, \mathrm{dir}, \mathrm{bin})$, (iii) the same for x. Lag bins are $\{1, 2, 3\text{–}4, 5\text{–}8, 9\text{–}16, > 16\}$, and intermediate entities are anonymous, so signatures contain no entity identifiers. We index up to 2,000 of the most recent training cases per relation. Similarity is an IDF-weighted Jaccard over path strings plus down-weighted Jaccard terms over the attached facts of each endpoint. A candidate's score is the mean of its top-3 case similarities. The demonstrations are the most similar cases for the top-scoring candidate. The query context contains the facts that realize the signatures of the top-m candidates. We also report MOTIFCASE's standalone ranking of candidates (no LLM) and ablations that remove lag bins or endpoint facts.

Lag-aware rule grounding. This baseline follows prior rule learners with time-dependent confidence (Omran et al., 2019; Xiong et al., 2023; Huang et al., 2024; Gastinger et al., 2026), and differs from them in using a non-parametric delay distribution for each rule, with bodies of up to three atoms. For each rule ρ with head relation r_ρ, we collect its groundings in the training split and record the delay $\Delta = t_{\mathrm{head}} - t_{\mathrm{last\ body}}$. We estimate $p_\rho(\Delta)$ with a Laplace-smoothed histogram, backing off to the relation-level histogram when ρ has fewer than 20 groundings. At query time, each grounding g of a rule with $r_\rho = r$ is scored $s(g) = c_\rho\, p_\rho(\tau - \tau_g)$, where c_ρ is the rule confidence and τ_g is the time of the grounding's last body fact. The context contains the body facts of the highest-scoring

groundings. Candidate entities are scored by noisy-or aggregation over their groundings. The key comparison is against GenTKG-style retrieval with the *same* rule bank, so the two differ only in how groundings are ranked.

BALBUNDLE. Candidates are the top-m entities ($m = 5$ by default) by summed reciprocal rank under the frequency heuristic and RE-Net. Each candidate c receives $\lfloor B/m \rfloor$ facts: the shortest undirected paths from h to c (length at most 3, most recent first), followed by up to two facts attached to c. Budget left unused by one candidate is not given to another. Bundles are presented in random order as "Candidate i (entity c): …", and the model ranks the m candidates. As ablations we consider an unbalanced union of the same facts, balanced but ungrouped facts, balanced bundles that also state each candidate's history count, and bundles without attached facts.

EVRANK. Training examples are the facts within 3 undirected hops of h in the window (at most 500 per query), labeled positive if they belong to a ground-truth antecedent set. Negatives are subsampled 20:1. The features contain no entity or relation identifiers: hop distance from h, whether the fact is incident to h as head or tail, whether it lies on a time-monotone path from h or on a path to one of the top-10 frequency candidates, the lag between the fact and the query time τ, the fact's recency rank among h's facts, the pointwise mutual information between the fact's relation and r within the window, the maximum confidence of a mined rule with head r whose body contains that relation, and the frequency of both relations; and the count of the triple in the window and the degrees of its endpoints. We train LightGBM with early stopping on validation AUC. At query time we keep the top-B facts, swapping in missing connector facts when they score within the top $2B$.

C.3 REACHABILITY ANALYSIS

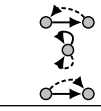
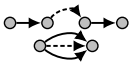
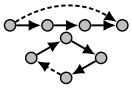
Table 2 assumes each shape's canonical orientation and time order. Our generator also instantiates shapes with other relation orientations and antecedent orderings (Section A.1), and these change which strategies reach the evidence. For example, reversing the direction of an antecedent incident to h changes what head-only retrieval (E) sees, and reordering antecedent times changes which time-monotone direction covers a path. We therefore compute reachability programmatically for every instantiated pattern in each dataset, and additionally measure it empirically per query, where background facts can create extra paths and the budget B can truncate them. Figure 11 compares this symbolic reachability with the empirical full-evidence rate of each retriever by pattern shape. Most methods do well with retrieval of 1-hop shapes, but struggle with the relatively more restricted reflective shape. Bridge-based shapes are hard to retrieve in full, as we saw previously.

C.4 RECOVERING GENERATING PATTERNS WITH RULE MINING

As an additional validity check, we ask whether a temporal rule miner can recover the patterns that generated our datasets, in the spirit of evaluating rule learners against gold-standard rules on synthetic data (Wang et al., 2024a). We run TLogic (Liu et al., 2022) on the training split of each synthetic TKG (rule lengths 1–3, with 800, 3,200, and 12,800 random walks) and match the mined rules against the ground-truth patterns. A pattern is *expressible* if its antecedents, in time order, form a TLogic rule body leading from the consequence head to its tail. Only 49%, 15%, and 3% of our 1-, 2-, and 3-hop patterns are expressible, and TLogic recovers every expressible pattern by 3,200 walks. Mining more walks does not raise recall further, but it floods the rule set. At 12,800 walks it contains about 7,000, 121,000, and 145,000 rules of the corresponding length, and precision against the ground truth falls to at most 1%. Table 8 breaks recall down by shape. The unrecovered shapes are exactly those that Table 2 predicts are not time-monotone paths from the head: bridge, bridging composition, and cyclic entailment. Mining patterns from data is therefore unreliable for multi-hop structure, which is why we generate data from an explicit pattern library calibrated to real statistics instead of from patterns mined from ICEWS14.

D FORMAL RESULTS ON RETRIEVAL

This section states the reachability observations of Section 4.2 formally and adds a bound on how the chance of retrieving the full evidence decays with hop length. Our impossibility results are worst-case

Hops		Shape	# of patterns	Expressible	Recall	Sufficient rule recalled
1		Inverse	38	1.00	1.00	1.00
		Reflexive	91	0.16	0.16	0.16
		Self-entailment	20	1.00	1.00	1.00
2		Bridge	19	0.00	0.00	0.00
		Multi-derivation	2	0.00	0.00	1.00
		Reflexive (any self-loop)	88	0.12	0.12	0.23
		Reinforcement of premise	27	0.00	0.00	1.00
		Transitive closure	14	0.79	0.79	0.79
3		Bridging composition	16	0.00	0.00	0.12
		Cyclic entailment	2	0.00	0.00	0.00
		Multi-step transitive closure	1	1.00	1.00	1.00
		Other 3-hop	9	0.00	0.00	0.33
		Premise re-derivation	38	0.05	0.05	1.00
		Reflexive (any self-loop)	81	0.02	0.02	0.31
		Shortcut entailment	3	0.00	0.00	0.33

Precision of mined rules	800 walks	3,200 walks	12,800 walks
1-hop dataset	0.014	0.011	0.010
2-hop dataset	0.001	<0.001	<0.001
3-hop dataset	<0.001	<0.001	<0.001

Table 8: TLogic pattern recovery on the hop datasets. Top: recovery by shape at 12,800 walks (rule lengths 1–3). Expressible: the pattern's antecedents, in time order, form a TLogic rule body from the consequence head to its tail. Recall: the exact rule was mined. Sufficient rule recalled: a mined rule over an order-preserving subset of the antecedents links head to tail. Bottom: precision of the mined rules whose body length equals the dataset's hop length, by number of random walks.

statements. They hold on any TKG in which background facts do not open alternative paths between the entities of a pattern instance, and in particular on the TKG that consists of the instance alone.

Setup. A query $q = (h, r, ?, \tau)$ with answer $t^\star$ is generated by a pattern instance with antecedents $A = \{a_1, \ldots, a_k\}$, where $a_i = (x_i, r_i, y_i, \tau_i)$ and $\tau_1 < \cdots < \tau_k < \tau$. The *evidence graph* H_q is the undirected multigraph on the entities of A with one edge per antecedent. A *retriever* maps the history $\mathcal{G}_{<\tau} = \{f \in \mathcal{G} : \tau_f < \tau\}$ and q to a context $C \subseteq \mathcal{G}_{<\tau}$ with $|C| \leq B$. Following Section 4.1, C contains the *full evidence* if $A \subseteq C$, and it contains the answer if $t^\star \in \mathrm{ent}(C)$.

Definition 1 (Sufficient context)**.** A context is *sufficient* for q if it contains the full evidence A and at least one solved instance of the same pattern (its antecedents and its consequence), as in the OICL+ex condition.

Full evidence is thus necessary for sufficiency, and every result below that rules out full evidence also rules out a sufficient context. We consider three families of retrievers, defined by the facts they are able to return:

- *head-anchored*: $C \subseteq \{f \in \mathcal{G}_{<\tau} : \mathrm{head}(f) = h\}$ (entity and pair history),
- *query-anchored*: every fact in C lies in the connected component of h in the undirected graph of $\mathcal{G}_{<\tau}$ restricted to a window $[\tau - W, \tau)$ (breadth-first search to any depth, path enumeration, personalized PageRank, rule grounding from h),
- *time-monotone*: every fact in C lies on a path from h whose timestamps are non-decreasing (TM↑) or non-increasing (TM↓) with distance from h.

Proposition 1 (Head-anchored retrieval)**.** *Let R be head-anchored. If no antecedent of q has head h and tail $t^\star$, then on any TKG without a background fact of the form $(h, \cdot, t^\star, \tau')$ with $\tau' < \tau$, the context of R does not contain $t^\star$, and no model restricted to the candidates in its context can rank $t^\star$ first. For the canonical shapes of Table 5, this is the case for inverse, transitive closure, bridge, multi-step transitive closure, shortcut entailment, bridging composition, and cyclic entailment.*

Proof. Every fact in the context has head h, so an entity other than h enters the context only as the tail of such a fact. By assumption, neither the antecedents nor the background contain a fact with head h and tail $t^\star$, and $t^\star \neq h$ in the listed shapes, so $t^\star \notin \mathrm{ent}(C)$. The candidate set is $\mathrm{ent}(C) \cup \{h\}$, so $t^\star$ is not a candidate. For the shape list, inspect the canonical antecedents (Table 5). In each listed shape, $h = x_c$ is either not the head of any antecedent (inverse, bridge, cyclic entailment) or heads only an antecedent whose tail is not $y_c = t^\star$ (the others). □

Proposition 2 (No single time direction)**.** *On the TKG consisting of a single instance of the canonical transitive-closure shape, no TM↓ retriever returns the full evidence. On the TKG consisting of a single instance of the canonical cyclic-entailment shape, no TM↑ retriever returns the full evidence. Hence neither family returns the full evidence on every shape.*

Proof. Transitive closure: $h = e_1$, $A = \{(e_1, r_1, e_2, \tau_1), (e_2, r_2, e_3, \tau_2)\}$, $\tau_1 < \tau_2$. The only path from h that contains a_2 is $e_1 \to e_2 \to e_3$, whose timestamps $\tau_1 < \tau_2$ increase with distance from h, so it is not non-increasing and a TM↓ retriever cannot return a_2.

Cyclic entailment: $h = e_4$, $A = \{(e_1, r_1, e_2, \tau_1), (e_2, r_2, e_3, \tau_2), (e_3, r_3, e_4, \tau_3)\}$, $\tau_1 < \tau_2 < \tau_3$. The only path from h that contains a_1 is $e_4 \to e_3 \to e_2 \to e_1$, with timestamps $\tau_3 > \tau_2 > \tau_1$ decreasing with distance from h, so a TM↑ retriever cannot return a_1. □

Since the cyclic rule bodies of TLogic are TM↑ paths ($\tau_1 \leq \cdots \leq \tau_l$), rule-grounding retrievers built on them inherit the second half of Proposition 2. This matches the zero recovery of cyclic-entailment and bridging-composition patterns in Table 8.

Proposition 3 (Answer anchoring is necessary for bridges)**.** *Let q be generated by the bridge shape $(e_1, r_1, e_2, \tau_1) \wedge (e_3, r_2, e_4, \tau_2) \to (e_2, r_x, e_3, \tau_3)$, so $h = e_2$ and $t^\star = e_3$. On any TKG in which e_3 is not connected to e_2 within the window, no query-anchored retriever returns $a_2 = (e_3, r_2, e_4, \tau_2)$ or places $t^\star$ in the context, for any search depth, time constraint, or budget.*

Proof. By assumption, e_3 and e_4 lie outside the connected component of $h = e_2$, so neither a_2 nor any fact containing $t^\star = e_3$ lies in that component, and a query-anchored retriever returns only facts from it. □

A retriever can therefore return the full evidence of a bridge only if it also expands from candidate answers, as the candidate-anchored strategies of Section 4.3 do.

A budget bound. Even when the evidence is reachable, a retriever must choose B facts from a neighborhood that grows with the search depth. The following lemma bounds what a retriever achieves when it has no signal that separates evidence from the other facts in its pool.

Lemma 4 (Uninformed retrieval)**.** *Let a retriever return a uniformly random subset C of size B of a pool P with $|P| = N \geq B$, and let $A \subseteq P$ with $|A| = k$. Then*

$$\Pr[A \subseteq C] = \frac{\binom{N-k}{B-k}}{\binom{N}{B}} = \prod_{i=0}^{k-1} \frac{B-i}{N-i} \leq \left(\frac{B}{N}\right)^k .$$

Proof. There are $\binom{N}{B}$ equally likely subsets, of which $\binom{N-k}{B-k}$ contain A, since the remaining $B-k$ elements are chosen from the $N-k$ elements outside A. Expanding the binomial coefficients gives the product, and each factor satisfies $(B-i)/(N-i) \leq B/N$ because $B \leq N$. □

Corollary 5. *To contain the full evidence of a k-hop pattern with probability at least p, an uninformed retriever over a pool of size N_k needs a budget $B \geq p^{1/k} N_k$.*

Proof. By Lemma 4, $p \leq \Pr[A \subseteq C] \leq (B/N_k)^k$. Rearranging gives the bound. □

For a k-hop pattern, the pool must include the facts within distance k of h, whose number N_k typically grows geometrically with k. Without a signal that ranks evidence above other facts, the budget must therefore grow in proportion to N_k, and the probability of full evidence at a fixed budget decays at least as fast as $(B/N_k)^k$. Heuristic signals such as recency, frequency, and relation statistics raise

this probability above the uninformed rate, which is why structural retrievers outperform random selection, but our results show that they fall far short of the ground-truth evidence. Learned signals are the natural next step.

E Context Conditions and Prompts

Distractors. The distractor conditions add $k \in \{5, 10, 25, 50, 100\}$ facts to the OICL+ex context, with the ground-truth evidence always present: *random* facts from the history, *near-miss* facts that share the oracle's entities or relations (e.g., the same pattern with wrong bindings), which also raise the answer's frequency in the context, *near-miss, frequency removed* (the same draw without any fact that touches the answer), and *cue-adversarial* facts that touch only the non-answer entities of the context, most recent first, so that both the most frequent and the most recent entity differ from the answer. The near-miss and cue-adversarial constructions use the known answer. They are diagnostic interventions, not realistic retrieval methods.

Label-informed analogies. The analogy conditions AH, AR, and BAR pair the query with a solved instance of its own generating pattern, whose history is retrieved by head entity (AH), by the analogy's antecedent relations (AR), or by those relations with equal counts per relation (BAR). Because they use the generating pattern, which is unavailable in practice, we treat them as references for the value of knowing the pattern (Section 4).

Scoring. Where exact scoring is higher for an open model, we show it alongside the written-list score. Llama-3.1-8B often fails to produce a valid list (for example, answering with Python code), so we mark those cells and rely on its exact scores.

Prompts. Every prompt starts with the same system message: "You are given a chronologically ordered list of events from a temporal knowledge graph. Each event is written as time:[a. head, relation, b. tail], where head, relation and tail are numeric IDs and the numbers before the periods (a., b.) are labels for the entities. The events are generated by recurring temporal patterns: combinations of earlier events trigger later events. Predict the tail entity of the final, incomplete event." The user message lists the context events in time order (events with the same timestamp in random order, so that the ground-truth evidence has no fixed position), ends with the incomplete query event, and closes with an answer instruction. Every entity in the context, heads and tails alike, receives a label from a random permutation drawn per query and condition, so labels carry no frequency or position cue. For exact scoring, the instruction is "Answer with only the label of the tail entity followed by a period, e.g. "7."", and we score each candidate by the log-probability of its label. For written lists, it is "Do not explain. Output only one line: ANSWER: followed by a Python list of the labels of the 10 most likely tail entities, most likely first, e.g. ANSWER: [7, 2, 11, ...]. Use only labels that appear in the events." (reasoning models are instead asked to reason as needed before the final ANSWER line). With real names (Appendix B), identifiers are replaced by ICEWS entity names and event types, and the system message says so. Table 9 and Table 10 show the context conditions as the actual prompts for one query.

F Full Experimental Results

This section reports the complete results behind Section 5.2. Table 11 gives Hits@3 for every method on all ten datasets, Table 12 the random-ranking baseline per condition, and Table 13 the cross-model comparison under one scoring method. Figure 9 shows the full distractor sweep, Figure 10 the three dataset families per model, Figure 11 empirical full-evidence rates by pattern shape, and Figure 12 the effect of the context budget.

F.1 Time Dispersion and Entity/Relation Distributions

Accuracy falls with hop count under every realistic context (GPT-4.1 with entity history: Hits@3 0.57, 0.43, and 0.28), following the drop in evidence recall, whereas time dispersion and entity/relation imbalance change it less (0.37–0.39 across time lags and 0.23–0.37 across distributions). Figure 10 shows Hits@3 along all three dimensions our generator controls, for naive entity context and for the

Condition	Context	Prompt for the example query
N-E	The query head's 25 most recent facts. Here it contains the first antecedent but not the second, so the answer (479) is not a candidate.	*Events:* 42:[1. 514, 11, 0. 236] 96:[1. 514, 55, 2. 294] **171:[1. 514, 77, 3. 865]** 176:[1. 514, 75, **?**
N-P	The query head's facts with the query relation. Here there are none.	*Events:* 176:[0. 514, 75, **?**
OICL	Exactly the antecedents of the generating pattern instance.	*Events:* **171:[0. 514, 77, 1. 865]** **175:[1. 865, 23, 2. 479]** 176:[0. 514, 75, **?**
N-E+ex	N-E plus one solved earlier instance of the same pattern (its antecedents and consequence), merged in time order.	*Events:* 42:[4. 514, 11, 1. 236] 96:[4. 514, 55, 6. 294] 166:[0. 843, 77, 3. 941] 170:[3. 941, 23, 2. 1315] **171:[4. 514, 77, 5. 865]** 174:[0. 843, 75, 2. 1315] 176:[4. 514, 75, **?**
OICL+ex	The antecedents plus the same solved example.	*Events:* 166:[0. 843, 77, 1. 941] 170:[1. 941, 23, 3. 1315] **171:[5. 514, 77, 4. 865]** 174:[0. 843, 75, 3. 1315] **175:[4. 865, 23, 2. 479]** 176:[5. 514, 75, **?**

Table 9: Context conditions (a): naive, oracle, and solved-example contexts, shown as the actual prompts for one 2-hop query from the transitive-closure family, $(514, 75, \textbf{?}, 176)$ with answer 479, generated by the antecedents $(514, 77, 865, 171)$ and $(865, 23, 479, 175)$. Each event is written `time:[a. head, relation, b. tail]`; the numbers before the periods are entity labels, drawn as a fresh random permutation for every prompt, and the model answers with a label. **Bold blue**: ground-truth evidence, gray: the solved example. Every prompt is preceded by the same system message and followed by the answer instruction (Appendix E), long contexts are truncated (. . .).

ground-truth evidence plus one solved example. Time dispersion has little effect once the evidence is given: with the ground-truth evidence and a solved example, GPT-4.1 scores 0.85, 0.80, and 0.81 on lags $\mathcal{U}(1, 3)$, $\mathcal{U}(1, 5)$, and $\mathcal{U}(1, 8)$. On the distribution family, TLogic is the strongest method (0.61–0.67 Hits@3), well above every LLM with naive context, while the recency heuristic is strongest among the heuristics. Skewed entity distributions help the frequency heuristic, as expected. Table 11 gives all ten datasets.

Condition	Context	Prompt for the example query
+ random distractors	OICL+ex plus k random history facts ($k = 5$ shown).	*Events:* 60:[1. 1025, 5, 5. 1121] 104:[9. 53, 63, 3. 225] . . . 166:[0. 843, 77, 10. 941] 170:[10. 941, 23, 2. 1315] **171:[7. 514, 77, 12. 865]** 174:[0. 843, 75, 2. 1315] . . . **175:[12. 865, 23, 8. 479]** 176:[7. 514, 75, **?**
+ cue-adversarial distractors	OICL+ex plus k recent facts that touch only non-answer entities, so the most frequent and most recent entities differ from the answer ($k = 5$ shown). Near-miss distractors (not shown) share the oracle's entities or relations.	*Events:* 98:[3. 1286, 17, 6. 514] 159:[5. 865, 25, 5. 865] 164:[8. 132, 21, 4. 941] 166:[2. 843, 77, 4. 941] 170:[4. 941, 23, 1. 1315] **171:[6. 514, 77, 5. 865]** 172:[9. 1740, 22, 2. 843] 174:[2. 843, 75, 1. 1315] **175:[5. 865, 23, 7. 479]** 175:[1. 1315, 70, 0. 1052] 176:[6. 514, 75, **?**
MOTIFCASE	Label-free retriever: time-ordered paths from the query head to candidate answers, plus the past case of the query relation whose anonymized path motif matches best, shown as a solved example. Most other retrievers return a plain event list like N-E.	*Example 1 (a solved event with its history):* 8:[12. 1470, 77, 13. 569] 9:[13. 569, 77, 3. 1095] 11:[11. 1507, 77, 8. 205] . . . 15:[11. 1507, 75, 13. 569] *Events for the new query:* 171:[7. 479, 48, 1. 865] **171:[0. 514, 77, 1. 865]** 175:[9. 294, 44, 2. 783] . . . **175:[1. 865, 23, 7. 479]** . . . 176:[0. 514, 75, **?**
BALBUNDLE	Label-free retriever: candidates proposed by frequency and RE-Net, each with an equal, labeled bundle of evidence (its shortest paths to the query head plus facts attached to it). The answer is candidate 6.	*Evidence grouped by candidate answer:* *Candidate 3. 294:* 175:[3. 294, 77, 2. 872] 175:[3. 294, 6, 0. 1118] *Candidate 5. 865:* **171:[7. 514, 77, 5. 865]** 171:[6. 479, 48, 5. 865] **175:[5. 865, 23, 6. 479]** *Candidate 4. 1286:* **171:[7. 514, 77, 5. 865]** 171:[6. 479, 48, 5. 865] 175:[4. 1286, 57, 6. 479] **175:[5. 865, 23, 6. 479]** 175:[6. 479, 56, 4. 1286] *Candidate 1. 236:* *(no evidence)* *Candidate 6. 479:* **171:[7. 514, 77, 5. 865]** 171:[6. 479, 48, 5. 865] 175:[6. 479, 72, 0. 1118] 175:[6. 479, 11, 6. 479] **175:[5. 865, 23, 6. 479]** 176:[7. 514, 75, **?**
AR (label-informed)	A solved instance of the query's own generating pattern with facts retrieved by that pattern's antecedent relations. This uses the generating pattern, so it serves only as a reference.	*Example 1 (a solved event with its history):* 171:[2. 1315, 23, 2. 1315] 171:[33. 1216, 77, 0. 183] . . . **171:[7. 514, 77, 11. 865]** . . . 174:[3. 843, 75, 2. 1315] *Events for the new query:* 172:[27. 1520, 23, 5. 1518] 173:[30. 1467, 77, 28. 1286] . . . **175:[11. 865, 23, 32. 479]** . . . 176:[7. 514, 75, **?**

Table 10: Context conditions (b): distractors and retrievers, for the same query as Table 9. Each event is written `time:[a. head, relation, b. tail]`; the numbers before the periods are entity labels, drawn as a fresh random permutation for every prompt, and the model answers with a label. **Bold blue**: ground-truth evidence, gray: the solved example. Every prompt is preceded by the same system message and followed by the answer instruction (Appendix E), long contexts are truncated (. . .).

		Heuristics				Graph			Qwen3-8B				Llama-3.1-8B				Qwen3-32B				GPT-4.1				GPT-5-mini				GPT-5.6-terra			
Variation	Dataset	F-E	F-P	R-E	R-P	CyGNet	RE-Net	TLogic	N-E	N-P	OICL	OICL+ex	N-E	N-P	OICL	OICL+ex	N-E	N-P	OICL	OICL+ex	N-E	N-P	OICL	OICL+ex	N-E	N-P	OICL	OICL+ex	N-E	N-P	OICL	OICL+ex
Hops	Only 1-hop	0.665	0.618	0.630	0.609	0.504	0.547	0.367	0.374	0.488	1.000	0.808	0.292	0.400	1.000	0.784	0.550	0.576	1.000	0.906	0.568	0.600	1.000	0.978	0.606	0.583	1.000	0.950	0.601	0.585	1.000	0.990
	Only 2-hop	0.486	0.438	0.428	0.415	0.371	0.427	0.304	0.250	0.344	0.972	0.668	0.188	0.304	0.964	0.680	0.360	0.402	0.982	0.778	0.426	0.420	0.985	0.798	0.506	0.448	0.980	0.750	0.480	0.460	0.995	0.850
	Only 3-hop	0.384	0.323	0.340	0.301	0.315	0.368	0.221	0.194	0.244	0.876	0.516	0.178	0.220	0.854	0.518	0.294	0.296	0.854	0.678	0.279	0.290	0.834	0.808	0.321	0.271	0.877	0.645	0.306	0.290	0.895	0.885
Time lag	U(1, 3)	0.478	0.427	0.407	0.411	0.384	0.406	0.235	0.212	0.322	0.966	0.630	0.216	0.282	0.978	0.616	0.342	0.404	0.976	0.786	0.388	0.413	0.979	0.853	–	–	–	–	–	–	–	–
	U(1, 5)	0.487	0.443	0.432	0.430	0.364	0.414	0.290	0.220	0.346	0.970	0.638	0.182	0.332	0.974	0.586	0.324	0.378	0.978	0.810	0.376	0.420	0.981	0.801	–	–	–	–	–	–	–	–
	U(1, 8)	0.480	0.427	0.402	0.422	0.364	0.404	0.240	0.202	0.330	0.978	0.656	0.160	0.292	0.984	0.640	0.352	0.394	0.988	0.758	0.375	0.407	0.978	0.812	–	–	–	–	–	–	–	–
E/R dist	Ent-Γ Rel-Γ	0.413	0.396	0.487	0.397	0.253	0.333	0.654	0.164	0.396	0.954	0.630	0.166	0.388	0.974	0.574	0.346	0.390	0.978	0.744	0.304	0.396	0.961	0.812	–	–	–	–	–	–	–	–
	Ent-Γ Rel-U	0.419	0.391	0.492	0.391	0.272	0.263	0.609	0.249	0.389	0.936	0.621	0.177	0.387	0.957	0.623	0.356	0.389	0.965	0.815	0.371	0.407	0.970	0.803	–	–	–	–	–	–	–	–
	Ent-U Rel-Γ	0.302	0.276	0.452	0.276	0.169	0.226	0.665	0.210	0.288	0.976	0.582	0.148	0.290	0.974	0.572	0.280	0.290	0.994	0.816	0.257	0.303	0.986	0.899	–	–	–	–	–	–	–	–
	Ent-U Rel-U	0.323	0.294	0.448	0.294	0.161	0.258	0.668	0.206	0.288	0.950	0.622	0.160	0.288	0.952	0.598	0.306	0.288	0.972	0.790	0.234	0.287	0.972	0.811	–	–	–	–	–	–	–	–

Table 11: Hits@3 across synthetic datasets. Protocol: Shared query set of test-split pattern consequences, heuristics and graph baselines on all queries, LLMs on seeded subsamples of 500 (open models), 300 (GPT-4.1) or 200 (closed-source reasoning models, on hop datasets only). OICL+ex: oracle evidence plus one solved example of the same pattern. Random-ranking Hits@3 over the same candidate sets (N-E / N-P / OICL / OICL+ex), averaged over datasets: 0.174 / 0.324 / 0.962 / 0.585, per dataset in Table 12.

Dataset	N-E	N-P	OICL	OICL+ex
Only 1-hop	0.258	0.431	1.0	0.806
Only 2-hop	0.181	0.321	0.973	0.596
Only 3-hop	0.16	0.212	0.871	0.481
Lag $\mathcal{U}(1,3)$	0.173	0.297	0.969	0.581
Lag $\mathcal{U}(1,5)$	0.178	0.328	0.969	0.568
Lag $\mathcal{U}(1,8)$	0.163	0.28	0.972	0.582
Ent-Γ Rel-Γ	0.141	0.393	0.958	0.544
Ent-Γ Rel-$\mathcal{U}$	0.186	0.394	0.964	0.584
Ent-$\mathcal{U}$ Rel-Γ	0.147	0.294	0.978	0.55
Ent-$\mathcal{U}$ Rel-$\mathcal{U}$	0.157	0.288	0.965	0.561

Table 12: Hits@3 of a random ranking over each condition's candidate set, per dataset.

Hops	Condition	gpt-4.1	gpt-oss-20b	qwen3-32b	qwen3-8b	llama3.1-8b
1	N-E	0.57	0.58	0.42 (0.57)	0.59	$0.16^{\dagger}$ (0.29)
	N-P	0.60	0.57	0.59	0.60	$0.10^{\dagger}$ (0.42)
	N1	0.61	0.63	0.40 (0.65)	0.53	$0.00^{\dagger}$ (0.45)
	N3	0.53	0.44	0.37 (0.41)	0.52	0.42
	OICL	1.00	1.00	1.00	1.00	$0.00^{\dagger}$ (1.00)
	OICL+ex	0.98	0.93	0.90 (0.90)	0.81 (0.81)	$0.11^{\dagger}$ (0.80)
	+50 rand	0.67	0.71	0.43 (0.69)	0.20 (0.23)	0.17 (0.32)
	+50 hard-na	0.44	0.33	0.24 (0.60)	0.01 (0.42)	0.04 (0.27)
	+50 adv	0.24	0.35	0.12 (0.34)	0.05 (0.30)	0.03 (0.18)
2	N-E	0.43	0.44	0.34 (0.37)	0.46	$0.13^{\dagger}$ (0.21)
	N-P	0.42	0.41	0.42	0.43	$0.08^{\dagger}$ (0.31)
	N1	0.44	0.46	0.31 (0.46)	0.37	$0.01^{\dagger}$ (0.25)
	N3	0.38	0.36	0.31 (0.36)	0.44	0.38
	OICL	0.98	0.97	0.98	0.96 (0.97)	$0.00^{\dagger}$ (0.97)
	OICL+ex	0.80	0.80	0.69 (0.79)	0.60 (0.68)	$0.01^{\dagger}$ (0.67)
	+50 rand	0.54	0.55	0.45 (0.50)	0.21	0.20 (0.30)
	+50 hard-na	0.34	0.26	0.19 (0.38)	0.03 (0.24)	0.03 (0.20)
	+50 adv	0.21	0.31	0.10 (0.23)	0.06 (0.18)	0.04 (0.12)
3	N-E	0.28	0.30	0.24 (0.27)	0.32	$0.14^{\dagger}$ (0.18)
	N-P	0.29	0.28	0.29 (0.29)	0.31	$0.05^{\dagger}$ (0.22)
	N1	0.35	0.36	0.27 (0.39)	0.34	$0.01^{\dagger}$ (0.22)
	N3	0.29	0.27	0.29 (0.30)	0.33	0.28
	OICL	0.83	0.85	0.86	0.85 (0.88)	$0.10^{\dagger}$ (0.85)
	OICL+ex	0.81	0.72	0.63 (0.67)	0.48 (0.51)	$0.01^{\dagger}$ (0.51)
	+50 rand	0.42	0.43	0.41 (0.44)	0.19	0.21 (0.24)
	+50 hard-na	0.21	0.21	0.13 (0.27)	0.02 (0.11)	0.03 (0.11)
	+50 adv	0.09	0.24	0.08 (0.18)	0.02 (0.11)	0.03 (0.09)

Table 13: Cross-model Hits@3 under one scoring method: every model writes a ranked list (GPT-4.1 and open models without reasoning, gpt-oss-20b with reasoning) on the same 300 seeded queries. In parentheses: exact log-probability scoring of the open models where it is higher (computed on the queries shared with the list runs). †: fewer than 90% of list answers contained a valid label (e.g. Llama-3.1-8B answers the short oracle prompt with Python code).

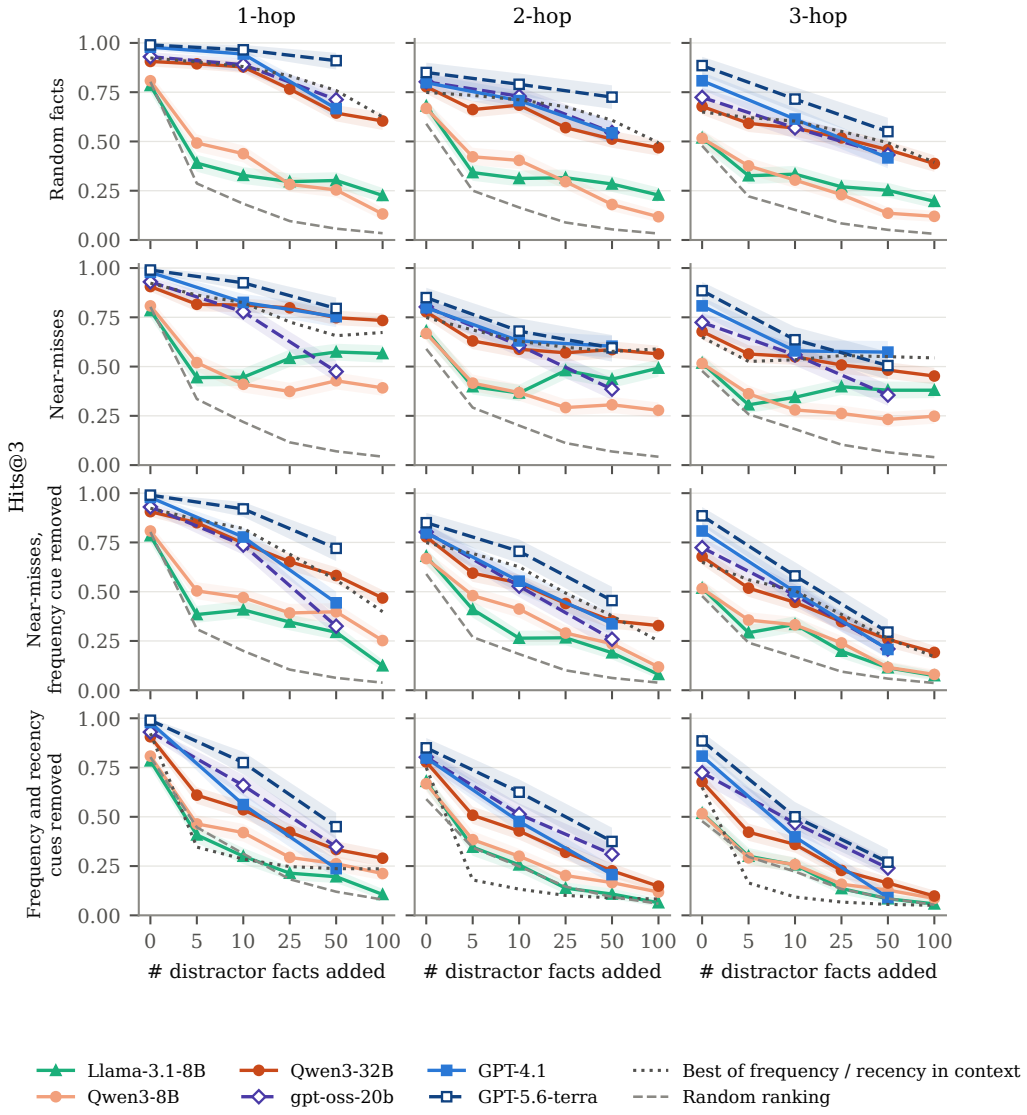


Figure 9: Full distractor sweep on the oracle evidence plus one solved example: Hits@3 as k random, near-miss, near-miss without the frequency cue, or cue-adversarial facts are added (rows), for 1-, 2-, and 3-hop patterns (columns). Models are colored by family (lighter = smaller); dashed lines with open markers are reasoning models; shaded bands are 95% confidence intervals. Dotted: the better of the frequency and recency rankers on the same contexts; dashed gray: random ranking.

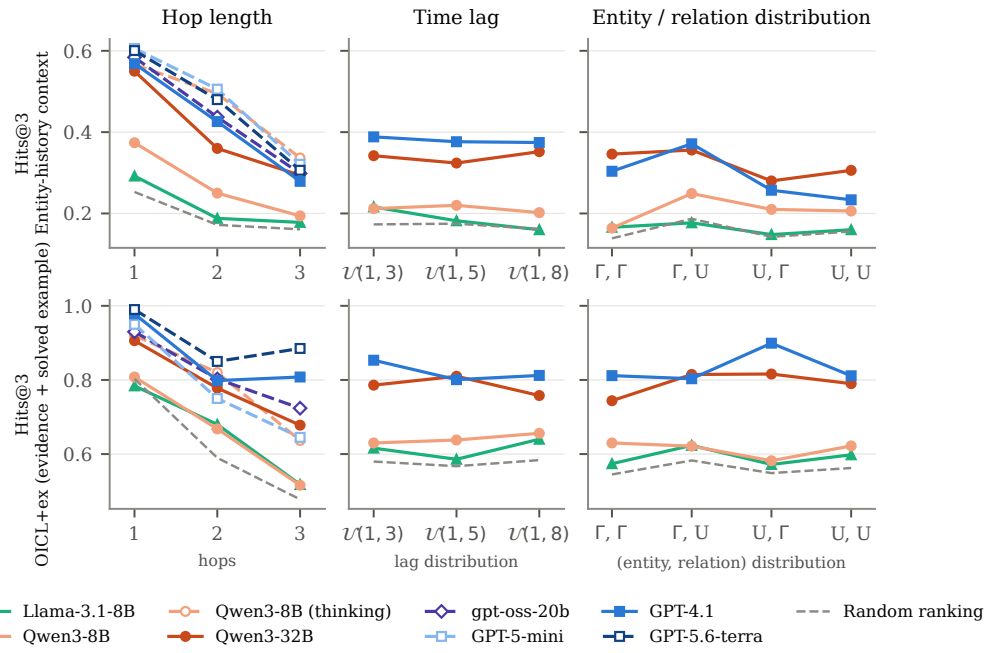


Figure 10: Hits@3 across the three dataset families (hops, time lag, entity/relation distribution) with naive entity context (top) and oracle evidence plus one solved example (bottom), per model (colors as in Figure 9). Dashed gray: random ranking.

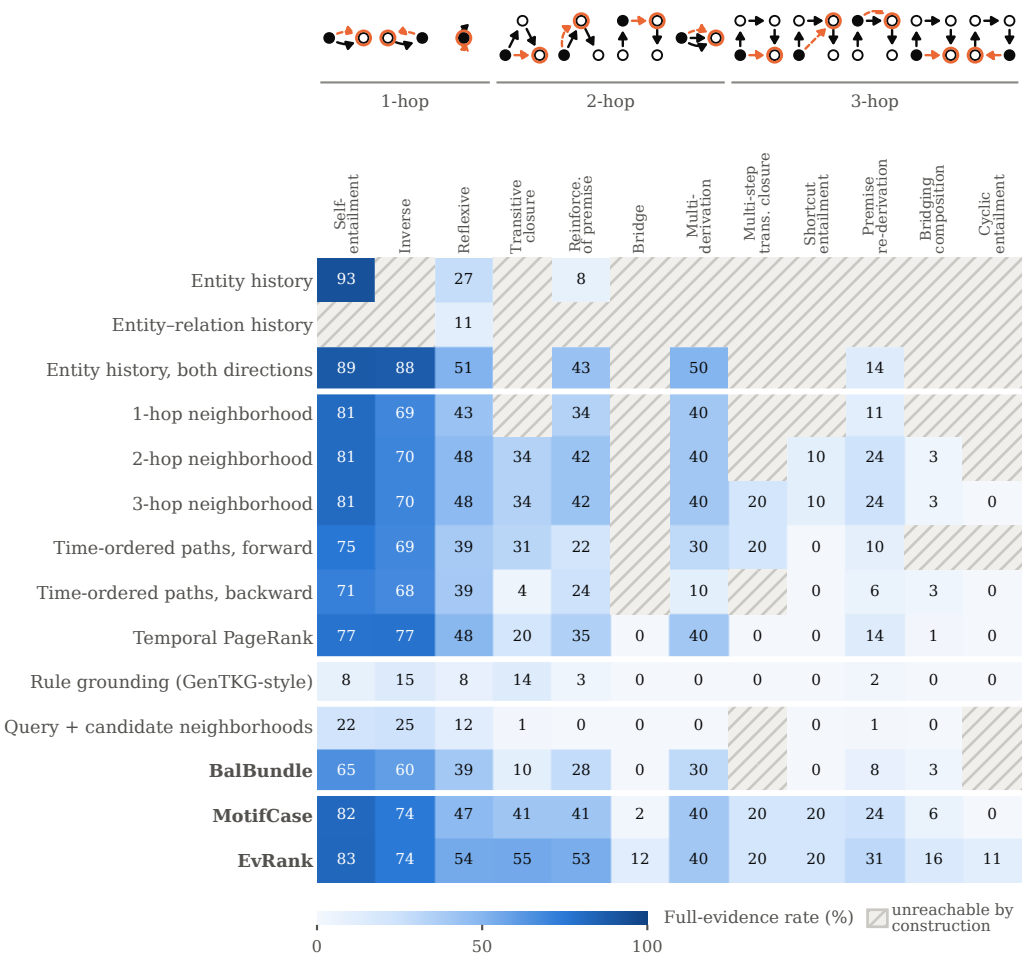


Figure 11: Empirical full-evidence rate (%) by generating pattern shape (columns) and retriever (rows, with our methods in bold), hop family, all test queries, $B = 25$. Icons: antecedents are black arrows, the consequence is the dashed orange arrow, the filled node is the query head, and the ringed node is the answer. Hatched cells are unreachable by construction for that retriever's family (Table 2). Empirically, 45 of these 49 cells are exactly zero, and the other four (bridge patterns, 1–3%) arise when background facts happen to connect the query head to the evidence.

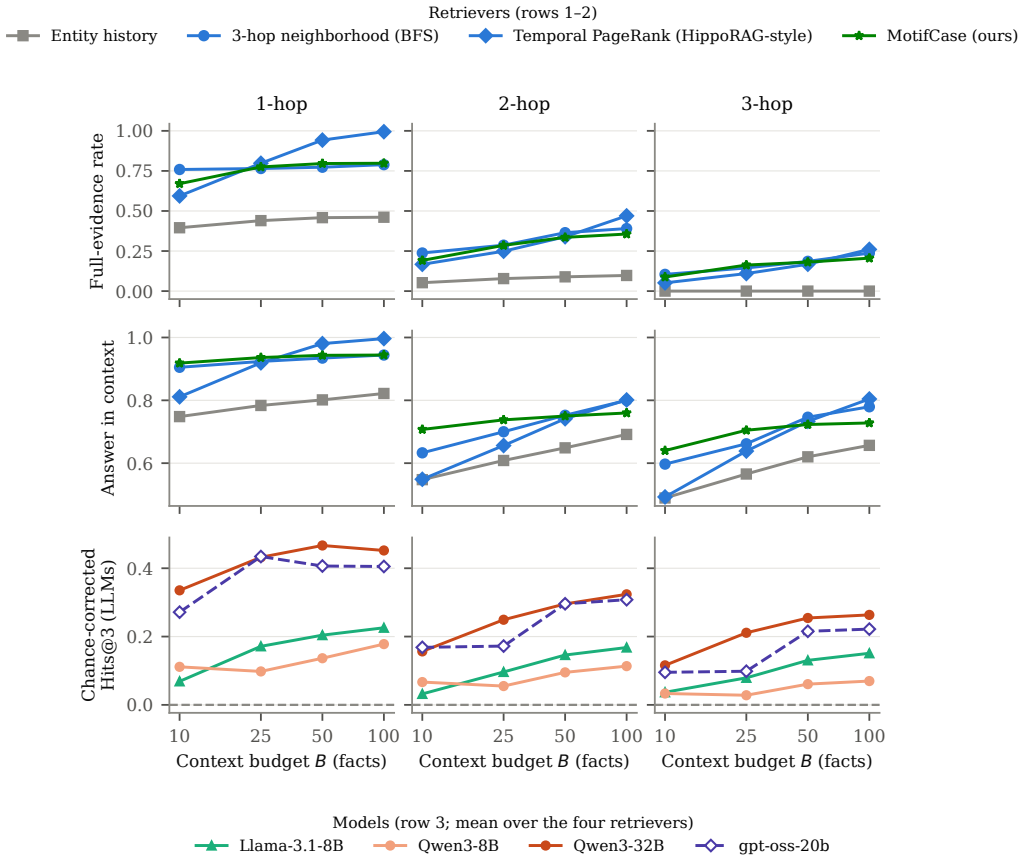


Figure 12: Effect of the context budget $B \in \{10, 25, 50, 100\}$ on evidence recall and downstream Hits@3. Raw Hits@3 can fall as B grows because larger contexts contain more candidates. Chance-corrected Hits@3 (bottom row) rises with B for all four models and saturates between $B = 50$ and $B = 100$.

G Extended Related Work

This section extends Section 2.

Rule-mining forecasters. Rule-mining approaches extract interpretable temporal rules that are also useful for forecasting. TLogic (Liu et al., 2022) mines rules from cyclic temporal random walks. TR-Rules (Li et al., 2023) allow acyclic rules and reduces temporal redundancy using a windowed confidence metric, and TRKG (Kiran et al., 2023) relaxes the time order of body atoms that occur close together. Several methods make rule confidence depend on the time elapsed since the body. TLogic uses a single exponential decay (Liu et al., 2022). StreamLearner keeps separate rules for different body–head distances (Omran et al., 2019). TempValid and CountTRuCoLa learn a separate decay or confidence function for each rule (Huang et al., 2024; Gastinger et al., 2026); and TILP models the time gap between pairs of relations with Gaussian or exponential distributions (Xiong et al., 2023).

GNN-based forecasters. History-aware neural models leverage TKG structure with recurrent or temporal modules to model evolution over time (Jin et al., 2020; Li et al., 2022a). TeMP (Wu et al., 2020) performs temporal message passing closely related to the patterns we study. CyGNet (Zhu et al., 2021) learns separate distributions for high-repetition entities and global entities, exploiting repeated simple patterns in benchmarks. CENET (Xu et al., 2023b) uses historical contrastive learning to separate historical from non-historical dependencies. These works implicitly distinguish between *simple* and *complex* temporal patterns, mirroring our axes of investigation.

LLM-based forecasters. Because LLMs operate on text, TKGs must be textualized before inference (Xu et al., 2023a). Once in text form, standard LLM techniques such as in-context learning (Lee et al., 2023), chain-of-thought prompting (Xia et al., 2024), and retrieval-augmented generation (Liao et al., 2023) can be applied. Lee et al. (2023) find that LLMs can match task-specific GNNs on TKG forecasting, even when semantic information is obfuscated. Chain-of-History (Xia et al., 2024) uses higher-order histories as intermediate reasoning steps, while GenTKG (Liao et al., 2023) mines temporal rules and uses them in a RAG framework. We similarly focus on how the provided context interacts with task complexity.

TKG and reasoning datasets. TKG reasoning is typically evaluated on event datasets such as GDELT (Leetaru & Schrodt, 2013), Wikidata (Vrandečić & Krötzsch, 2014), and ICEWS (Boschee et al., 2015). These corpora are realistic but noisy and do not provide guarantees about the necessity or sufficiency of mined patterns, hence motivating synthetic datasets with known generative structure.

For temporal rule languages, iTemporal (Bellomarini et al., 2022) generates DatalogMTL programs and datasets for benchmarking reasoners, and Wang et al. (2024a) use such synthetic data with gold-standard rules to test whether temporal rule learners recover them. Our generator instead targets TKG forecasting: its patterns are calibrated to the statistics of real event data, and we use it to evaluate LLM- and GNN-based forecasters and the context they are given. For static KGs, controllable benchmarks such as FRUNI and FTREE (Martin et al., 2023) vary size and noise under fixed schemas, and synthetic KG QA testbeds have been built to probe robustness of supervised fine-tuning and in-context learning for multi-hop QA (Shah et al., 2025). In contrast, we focus on temporal KGs and directly control the underlying temporal patterns rather than only the surface statistics.

Most closely related, Fatemi et al. (2025) build synthetic temporal-reasoning graphs to avoid pre-training leakage and anonymization artifacts, and study how graph structure and fact order affect LLM accuracy on question answering; we instead target TKG *forecasting*, generate data from explicit multi-hop temporal patterns, and calibrate to real event statistics. Cattaneo et al. (2026) construct synthetic KGQA data with ground-truth answer subgraphs and use them to train and evaluate retrievers for static KGs. We use ground-truth evidence for temporal patterns to diagnose retrieval versus reasoning failures.

Multi-hop reasoning has also been studied on non-KG datasets, particularly multi-hop QA over documents (Yang et al., 2018; Geva et al., 2021). 2WikiMultiHopQA (Ho et al., 2020) defines explicit evidence chains, while MuSiQue (Trivedi et al., 2022) composes single-hop questions to reduce shortcut opportunities. Our setting abstracts away from natural language and evaluates multi-hop reasoning directly on the underlying TKG.

Multi-hop reasoning in LLMs. Recent work shows that LLM performance degrades with hop count and often relies on shortcuts or flawed reasoning chains. Biran et al. (2024) find that 2-hop queries are solved via early identification of *bridge* entities followed by later resolution, leaving relevant information vulnerable to being overwritten. Counterfactual and unprecedented variants of QA benchmarks reveal a strong dependence on spurious correlations (Wu et al., 2024a;b). Although LLMs exhibit some compositional reasoning ability (Yang et al., 2025; Khodadad et al., 2025), it is unreliable. Grounding LLM chain-of-thought rationales in KGs also shows that generated paths are frequently invalid (Nguyen et al., 2024). Our synthetic TKGs isolate the multi-hop reasoning component while controlling for other sources of ambiguity.

Controllable synthetic benchmarks. Template- and generator-based benchmarks vary one difficulty factor at a time to separate reasoning from memorization and surface cues: GSM-Symbolic perturbs math templates and adds inconsequential clauses (Mirzadeh et al., 2025), while CogniLoad and seqBench tune the length, intrinsic difficulty, and distractor density of logic and sequential-reasoning tasks (Kaiser et al., 2025; Ramezanali et al., 2025). Irrelevant context alone degrades accuracy (Shi et al., 2023), as do the position of evidence (Liu et al., 2024) and the length of the context even under perfect retrieval (Du et al., 2025). We bring this controlled methodology to temporal knowledge graphs, where the controllable factors are the structure, timing, and frequency of multi-hop patterns.

Temporal motifs and case-based reasoning. Temporal motifs are the small, time-ordered subgraphs that recur in temporal networks (Kovanen et al., 2011; Paranjape et al., 2017). Temporal link predictors encode them as causal anonymous walks (Wang et al., 2021) or as transitions between open motif instances (Altun & Sarıyüce, 2026), and LLMs have recently been benchmarked on recognizing them (Hao et al., 2026). G2S (Bai et al., 2025) anonymizes entities, relations, and timestamps into "anonymous temporal structures" so that an LLM can learn patterns shared across TKGs, and HiSMatch (Li et al., 2022b) matches the historical structures of the query and candidate entities with neural encoders. Case-based reasoning answers a query by retrieving similar solved cases, both for static KG completion (Das et al., 2020; 2021) and for predicting the effects of events in Wikidata (Shirai et al., 2023). Our MOTIFCASE retriever (Section 4) brings case-based retrieval to TKG forecasting: it uses anonymous, time-ordered, lag-binned motif signatures as retrieval keys to select past cases as in-context demonstrations.