Co-Evolving Structured Knowledge and Reasoning in Language Models

Ryan Thomas Noonan* **Linxi Zhao*,†** **Menghan Xu*** **Akanksha Sarkar**

Mihir Mishra **Dongyoung Go** **Kilian Q. Weinberger** **Yoav Artzi** **Jennifer J. Sun**

Cornell University
{rtn27,lz586,mx253,as2637,mrm367,dg793,kilian,yoavartzi,jennifer.sun}@cornell.edu

Abstract

Retrieval-augmented methods improve factual accuracy by grounding language models in external knowledge, but retrieving over unstructured text often introduces irrelevant context and offers limited control over the retrieved information. Structured knowledge bases offer a more controllable alternative, yet they are expensive to construct and often brittle to reason over. To address these limitations, we propose KBEVO: a *co-evolving* framework that jointly learns to construct a structured knowledge base and reason over it for knowledge-intensive question answering. By optimizing both components end-to-end with QA outcome rewards, our method enables reasoning success to directly improve the quality of the constructed knowledge base. This leads to larger, better-connected knowledge structures with higher answer reachability, while also improving compositional factual reasoning and controllability compared to standard retrieval baselines.

1 Introduction

While language models have demonstrated strong reasoning and generation capabilities, their ability to reliably store and recall parametric factual knowledge remains limited. When knowledge is memorized during pretraining, the resulting representations are inherently lossy: facts may be only partially memorized, conflated with related but distinct information (Meng et al., 2022; Bommasani et al., 2021). More fundamentally, parametric knowledge cannot be easily inspected, corrected, or updated without expensive retraining, making it ill-suited for knowledge-intensive applications where accuracy and controllability are essential.

Retrieval-augmented generation (RAG) and search agents have emerged as prominent approaches to address these limitations by grounding model outputs in external text corpora at inference time. By retrieving relevant passages and conditioning generation on them, these systems can access up-to-date information and reduce hallucination (Lewis et al., 2020; Izacard & Grave, 2020; Li et al., 2025; Jin et al., 2025; Zheng et al., 2025). The dominant paradigm, however, retrieves from unstructured text split into fixed-size chunks, a format optimized for coverage rather than reasoning (Karpukhin et al., 2020; Jin et al., 2024). As a result, retrieved chunks can inject irrelevant context or only partially address the query (Shi et al., 2023; Barnett et al., 2024; Gao et al., 2025; Chang et al., 2026).

Structured knowledge bases (KBs) offer a promising alternative by representing facts as discrete, queryable entities and relations. This representation enables precise access to individual facts and supports compositional reasoning, while making the stored knowledge more interpretable and editable (Vrandečić & Krötzsch, 2014; Saxena et al., 2020; Ho et al.,

* Equal contribution. † Project lead.
‡We open-source our code and models at https://github.com/kilian-group/KBevo.

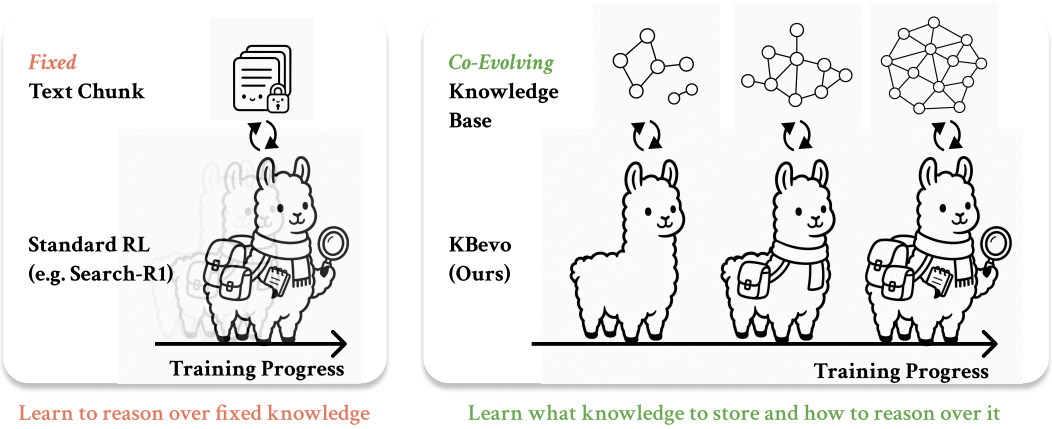


Figure 1: **Co-evolving knowledge and reasoning.** Standard retrieval-based RL improves the reasoning policy while operating over fixed text chunks. In contrast, KBEVO jointly optimizes reasoning and knowledge construction, allowing the model and its structured knowledge base to improve together during training.

2020; Zhong et al., 2023; Kansal & Jha, 2026). Because individual facts are stored explicitly, they can be modified without updating the model parameters. This gives users more direct control over the external knowledge available during reasoning. However, KB coverage is often constrained by the schema specified at construction time, while building high-quality KBs requires substantial human annotation or costly frontier-model inference (Vrandečić & Krötzsch, 2014; Dedhia et al., 2025). This construction cost makes structured KBs difficult to scale and build on demand.

The limitations of existing approaches share a common root: the decoupling of knowledge representation from downstream reasoning. In both paradigms, the knowledge store is constructed independently of the reasoning process, leaving no mechanism for reasoning failures to inform and improve the knowledge base. Addressing this gap requires a framework in which knowledge construction and reasoning are jointly optimized.

To this end, we propose KBEVO, a *co-evolving* framework that jointly learns to construct a structured knowledge base and reason over it, allowing downstream QA outcomes to directly shape what knowledge is constructed. Rather than treating KB construction as a fixed preprocessing step, our method treats it as a learnable component trained end-to-end alongside the reasoning module using QA outcome rewards (Figure 1). Our method consists of two phases: (1) the model reads a set of related documents and constructs a structured KB that can be extracted and indexed offline for reuse across downstream queries; (2) at inference time, the model answers questions by retrieving and reasoning over the constructed KB. Both phases are optimized jointly, allowing reasoning signals to propagate back and guide the KB toward structures better aligned with compositional factual reasoning.

Our contributions are: (1) a co-evolving training framework that jointly optimizes KB construction and reasoning using outcome rewards, offering a scalable path to KB construction beyond the constraints of human or LLM annotation (Sec. 3); (2) competitive performance on knowledge-intensive QA benchmarks while enabling direct, training-free knowledge editing (Sec. 5.1 and Sec. 5.2); and (3) evidence that co-evolution produces more complete and effective KBs and improves grounded reasoning (Sec 5.3).

2 Related Work

Retrieval-Augmented LLMs and Search-Based Knowledge Agents. A growing body of work augments language models with external knowledge to improve factual reasoning. Early retrieval-augmented generation (RAG) methods retrieve relevant passages from an external corpus and condition generation on the retrieved context (Lewis et al., 2020; Izacard & Grave, 2020; Guu et al., 2020; Borgeaud et al., 2021; Ram et al., 2023). More recent work

moves beyond one-shot retrieval and studies *search-based knowledge agents* that interleave reasoning with multi-turn information access. Methods such as ReAct and Toolformer equip models with explicit tool-use capabilities, enabling them to decide when and how to invoke external tools during reasoning (Schick et al., 2023; Yao et al., 2022). Building on this direction, recent reinforcement learning approaches such as Search-R1 train models to issue search queries and interact with retrieval systems during step-by-step reasoning using only outcome supervision, leading to stronger multi-hop and knowledge-intensive reasoning (Jin et al., 2025). Subsequent work further scales this paradigm to more realistic and long-horizon settings, including open-web research environments and broad knowledge-agent benchmarks (Zheng et al., 2025; Chen et al., 2025; Chang et al., 2026). While these methods substantially improve *access* to external knowledge, they primarily operate over *unstructured* information and provide limited control over the form, consistency, and editability of the knowledge being used.

Knowledge Storage, Compression, and Controllability in LLMs. A fundamental question in language modeling is how factual knowledge should be stored and represented. The dominant paradigm in modern LLMs is *parametric* knowledge storage, where factual knowledge is implicitly compressed into model weights through next-token prediction (Devlin et al., 2019; Petroni et al., 2019). While this representation is compact and broadly generalizable, it is also inherently lossy: even large models often struggle to reliably retain long-tail facts and may hallucinate when knowledge is missing or weakly encoded (Allen-Zhu & Li, 2023; Kandpal et al., 2022). To address this, prior work has explored *non-parametric* knowledge, including external corpora, knowledge databases, and learned memory that store knowledge externally (Zhao et al., 2026; Pouransari et al., 2025; Bi et al., 2026).

A second challenge is that knowledge stored in model parameters is often highly entangled. Rather than being cleanly localized, factual information is distributed across shared representations together with linguistic patterns and other facts, a phenomenon often described as *knowledge superposition* (Elhage et al., 2022). This makes knowledge difficult to inspect, edit, update, or remove, creating challenges for continual learning, knowledge editing and unlearning, and interpretability. Our work is motivated by these limitations. We study a setting in which knowledge is represented externally in a structured form that is more controllable and editable, while still being dynamically constructed and used by the model during reasoning.

Method	Construction	Schema	Quality Control	Downstream-Coupled	Persistence
EDC	Extract–Define–Canonicalize	Induced	Canonicalization	✗	Persistent
Wikontic	Multi-stage pipeline	Wikidata ontology	Ontology filtering + dedup	✗	Persistent
AutoSchemaKG	Autonomous pipeline	Induced	Schema-guided filtering	✗	Persistent
Ours	Joint RL training	Emergent (task-driven)	QA reward signal	✓	Persistent + Editable

Table 1: Comparison of knowledge graph construction methods.

Structured Knowledge Base Construction and Reasoning. As shown in Table 1, prior KG construction methods such as EDC (Zhang & Soh, 2024), Wikontic (Chepurova et al., 2026), and AutoSchemaKG (Bai et al., 2026) treat knowledge construction as a standalone pipeline, focusing on extraction, schema induction, canonicalization, or ontology-based filtering. Wikidata (Vrandečić & Krötzsch, 2014) is a large human-curated knowledge base rather than an automated KG construction method. In contrast, our approach is schema-free and directly optimized with downstream QA supervision. As a result, we evaluate not only downstream QA performance but also the correctness, structure, and coverage of the constructed database. Prior work on multi-hop reasoning over knowledge graphs largely assumes a fixed graph and improves inference over it, for example through path construction and pruning (Tan et al., 2025), LLM-based query planning (Chen et al., 2024), improved query representations (Kim et al., 2024), or traversal strategies such as hierarchical reinforcement learning (Wang et al., 2025). In contrast, our method jointly optimizes graph construction and reasoning during training, while keeping the graph fixed at inference time.

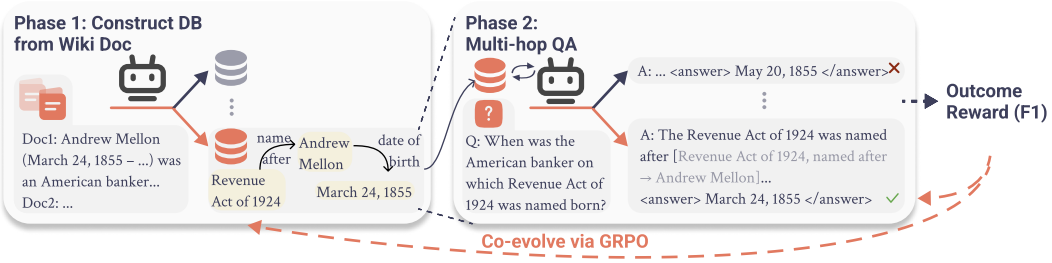


Figure 2: **Overview of our co-evolving framework.** The model first constructs a structured knowledge base from context, then answers multihop questions by retrieving and reasoning over it. Both phases are jointly optimized so that knowledge construction and reasoning improve together.

3 Methodology

3.1 Co-Evolving Framework

We focus on *entity-level atomic factual knowledge* as the basic unit of knowledge representation, where facts are stored as structured `(entity, relation, value)` triples.

An overview of our framework is illustrated in Figure 2. We frame the problem as jointly training a single policy π_θ that operates in two phases: *knowledge base construction* and *question answering retrieving over the knowledge base*. Both phases share the same model parameters θ, allowing improvements in one phase to transfer to the other.

Phase 1: Knowledge Base Construction. Given a supporting passage c for question q, the model extracts a set of factual triplets $\{(\texttt{entity}, \texttt{relation}, \texttt{value})\}$, each representing an atomic fact. The collection of all extracted triplets naturally forms a knowledge base $G = (V, E)$, where the nodes V are entities and values, and the edges E are relations.

Phase 2: Question Answering Retrieving over KB. To answer a factual question q, our model has access to the knowledge base G through targeted retrieval. The model is able to issue tool calls, querying `(entity, relation)` pairs, and retrieving the corresponding value. Here is an example of Phase 2:

```
The football club of Lionel Messi is <|db_entity|> Lionel Messi
<|db_relationship|> Football club <|db_return|> Inter Miami FC <|db_end|>
Inter Miami FC.
<answer> Inter Miami FC </answer>
```

Here, `Inter Miami FC` is injected from the KB into the context at inference time. More details about retrieval in Appendix B.1.

3.2 GRPO Training

We train the model using GRPO (Shao et al., 2024), with the full procedure summarized in Algorithm 1.

Sampling and Reward. For each question, we first sample K candidate knowledge bases in Phase 1 (line 3). For each constructed knowledge base, we then generate M answer rollouts in Phase 2 (line 4), yielding a total of $K \times M$ QA rollouts per question. We use F1 score between the predicted answer and ground truth answer as the Phase 2 reward (line 5). The reward for each Phase 1 knowledge base is defined as the average downstream QA reward over its associated Phase 2 rollouts (line 6).

Optimization. We compute group-relative advantages separately for the two phases (lines 7–8). For Phase 2, we normalize QA rewards within each (question, constructed KB)

Algorithm 1 Co-evolving KB Construction and QA via GRPO

Require: Training set $\mathcal{D}$, batch size B, KB rollouts per sample K, QA rollouts per KB M, KB construction prompt p^{kb}, QA rollout prompt p^{qa}

1: **for** $t = 1$ to T **do**

2: Sample minibatch of B samples $\{(c_b, q_b, a_b^*)\}_{b=1}^{B} \sim \mathcal{D}$

Phase 1: KB Construction

3: For each passage c_b, construct K knowledge bases:

$$\{G_{b,k}\}_{b=1,k=1}^{B,K} \sim \pi_\theta(\cdot \mid p^{\text{kb}}(c_b)) \quad \forall\, b$$

Phase 2: QA Rollouts (conditioned on Phase 1)

4: For each constructed knowledge base $G_{b,k}$, generate M answers:

$$\{\hat{y}_{b,k,m}\}_{m=1}^{M} \sim \pi_\theta(\cdot \mid p^{\text{qa}}(q_b), G_{b,k}) \quad \forall\, b, k$$

Reward and Advantage Computation

5: Compute QA rewards for $\hat{y}_{b,k,m}$: $r_{b,k,m} \leftarrow r(\hat{y}_{b,k,m}, a_b^*) \quad \forall\, b, k, m$

6: Compute KB rewards for $G_{b,k}$: $\bar{r}_{b,k} \leftarrow \frac{1}{M}\sum_{m=1}^{M} r_{b,k,m} \quad \forall\, b, k$

7: Compute QA advantages $\{A_{b,k,m}^{\text{qa}}\}$ by normalizing $\{r_{b,k,m}\}_{m=1}^{M}$ within each b, k

8: Compute KB advantages $\{A_{b,k}^{\text{kb}}\}$ by normalizing $\{\bar{r}_{b,k}\}_{k=1}^{K}$ within each b

Joint GRPO Update

9: Update π_θ with a joint GRPO step over the concatenated Phase 1 and Phase 2 rollout batch, using advantages $\{A_{b,k}^{\text{kb}}\}$ and $\{A_{b,k,m}^{\text{qa}}\}$

10: **end for**

pair across its M answer rollouts. For Phase 1, we average Phase 2 rewards per KB and then normalize across the K candidate KBs for the same question. Unlike standard GRPO, where all rollouts in a group share the same input, our Phase 2 rollouts are conditioned on different constructed knowledge bases. We then optimize π_θ using the standard GRPO clipped surrogate objective over the entire batch of Phase 1 and Phase 2 rollouts (line 9). To balance the gradient contribution of the two phases, we scale each Phase 1 advantage $A_{b,k}^{\text{kb}}$ by M, the number of associated Phase 2 rollouts. The full objective is provided in Appendix A.1. In Phase 2, retrieved result tokens are masked out from the loss.

The two phases co-evolve during training: better knowledge bases enable better answers, and sharper answer-level reward signals in turn drive the construction of more effective knowledge bases. We ablate this coupling in Sec 5.4.

3.3 Inference

Given a collection of input documents, we first construct and index a structured KB using Phase 1. The resulting KB is then fixed and reused across downstream queries, where the model answers questions by querying the indexed KB through targeted retrieval.

4 Experimental Setup

Training Setting. We use QWEN3-1.7B and QWEN3-4B (Yang et al., 2025) and train on 7k HotpotQA (Yang et al., 2018) examples. We generate 6k SFT trajectories from HotpotQA using Gemini-2.5-Flash (Comanici et al., 2025), covering both Phase 1 KB construction and Phase 2 question answering, and fine-tune each model for 3 epochs. We subsequently train with GRPO on 7k HotpotQA examples for 500 steps using an F1-based outcome reward. For retrieval, we use `all-MiniLM-L6-v2` (Reimers & Gurevych, 2019) with a similarity threshold of 0.6 (returning `unknown` otherwise) and top-$k = 4$. Additional details are provided in Appendix B.1.

Baselines. We compare KBEVO against several representative baselines:

- *Direct*: direct answer generation without external retrieval.

Method	HotpotQA*	MuSiQue	2Wiki	PopQA	Avg
QWEN3-1.7B					
Direct	12.1	0.7	21.7	16.0	12.6
RAG	25.4	6.3	21.8	57.8	27.8
IRCoT	23.7	12.2	44.0	60.7	35.2
Search-R1†	44.1	14.3	45.9	65.3	42.4
KBEVO-SFT	29.5	13.3	44.9	57.0	36.2
KBEVO-GRPO	37.3	18.6	47.8	61.4	41.3
QWEN3-4B					
Direct	12.3	1.5	18.5	13.7	11.5
RAG	32.4	6.7	20.3	63.1	30.6
IRCoT	33.0	19.7	27.1	65.0	36.2
Search-R1†	51.1	22.6	50.4	71.8	49.0
KBEVO-SFT	35.6	16.7	40.7	54.3	36.8
KBEVO-GRPO	46.1	26.4	51.4	62.5	46.6

Table 2: Evaluation on multi-hop QA benchmarks and PopQA (EM). * marks the in-domain benchmark. † Reimplemented on QWEN3 backbone. See Section 4.

- *RAG* (Lewis et al., 2020): retrieval-augmented generation over text chunks.
- *IRCoT* (Trivedi et al., 2023): interleaves chain-of-thought reasoning with iterative BM25 retrieval. We run IRCoT over the same benchmark-specific corpus used by the other retrieval methods.
- *Search-R1* (Jin et al., 2025): trains a model to interleave multi-turn search with reasoning over unstructured text. Since Search-R1 was not released for QWEN3, we reimplement it with matched training data, training steps, and overall training budget. In our QWEN3 reimplementation, the original prompt often reverted to long-form reasoning without search; we therefore provide a single in-context example illustrating targeted search and the expected tool-call format. Full implementation and prompt details are provided in Appendix B.2.

We refer to the model after supervised fine-tuning as KBEVO-SFT, and to the final model after GRPO post-training as KBEVO-GRPO.

Benchmarks and Metrics. We evaluate on three multi-hop question answering benchmarks: HotpotQA (7,405 samples) (Yang et al., 2018), MuSiQue (2,417 samples) (Trivedi et al., 2021), and 2WikiMultiHopQA (12,576 samples) (Ho et al., 2020). We additionally evaluate on PopQA (1,399 samples) for single-hop factual QA. Since training uses HotpotQA, we treat HotpotQA as in-domain and the remaining benchmarks as out-of-domain evaluation. We report exact match (EM) as the primary metric.

Retrieval and Knowledge Base Setup. For each benchmark, all retrieval-based methods use the same source documents. Text-based baselines index document chunks for retrieval, while KBEVO constructs a structured KB from the same documents and retrieves over the extracted triplets. We aggregate all chunks or triplets within each benchmark into a single retrieval datastore.

5 Results

5.1 Knowledge Intensive QA

We report results on three multi-hop QA benchmarks and PopQA in Table 2. Across both model scales and all benchmarks, KBEVO-GRPO consistently improves over KBEVO-SFT, increasing average EM by 5.1 points at 1.7B and 9.8 points at 4B. This consistent gain shows

that downstream QA reward provides an effective learning signal for jointly improving knowledge construction and multi-hop reasoning.

SEARCH-R1 provides our closest comparison to the full method: both approaches use outcome-based RL to learn retrieval and reasoning, but SEARCH-R1 retrieves directly from unstructured text, whereas KBEVO learns to construct and reason over a structured KB. Under our matched Qwen3 training setup, KBEVO-GRPO achieves comparable overall performance to SEARCH-R1 (41.3 vs. 42.4 average EM at 1.7B and 46.6 vs. 49.0 at 4B), with complementary strengths across benchmarks. These results suggest that replacing unstructured text search with a structured knowledge interface does not substantially sacrifice the effectiveness of RL-trained retrieval and reasoning.

The two approaches differ, however, in what they provide beyond QA accuracy. Structured knowledge represents individual facts explicitly, allowing the knowledge available to the model to be inspected, edited, and controlled. As shown in Table 3, this structure also enables targeted knowledge updates that the model can effectively incorporate during reasoning. Thus, KBEVO achieves competitive RL-trained reasoning performance while retaining the controllability and reusability that motivate structured external knowledge.

5.2 Reasoning with Edited External Knowledge

Structured knowledge allows individual facts to be directly edited. We evaluate whether models can effectively use such updates for multi-hop reasoning on ConFiQA-MR (Bi et al., 2025), which introduces counterfactual facts that conflict with a model's parametric knowledge. We consider three settings: MR-ORIG uses the original facts, while MR-CF-100 and MR-CF-356 replace 100 and 356 examples with conflict-free counterfactual versions and update the corresponding knowledge stores.

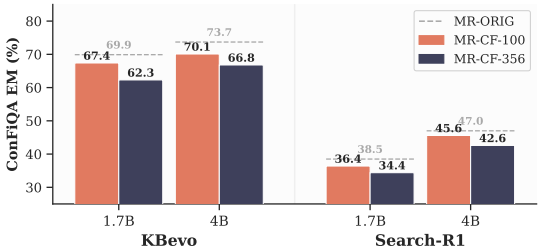


Figure 3: **Reasoning with edited external knowledge on ConFiQA-MR (EM).** KBEVO maintains strong performance under counterfactual knowledge edits, showing that its structured KB can be directly updated and effectively used for reasoning without model retraining.

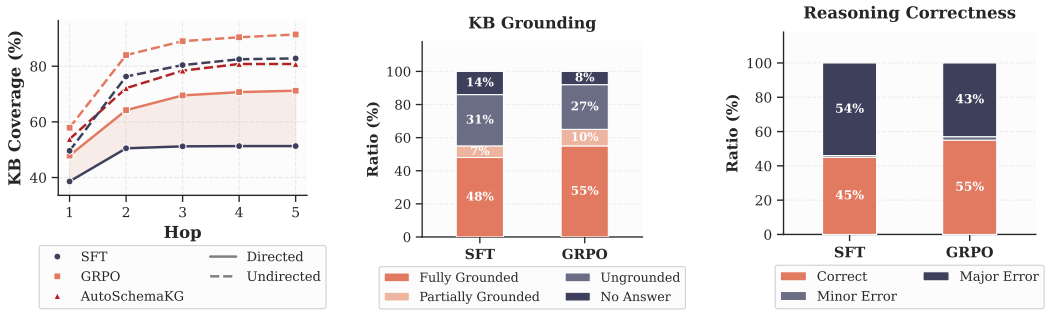
As shown in Figure 3, KBEVO remains substantially stronger than Search-R1 after these knowledge edits across both model scales. These results show that its structured KB can be directly updated and effectively used for reasoning without retraining.

5.3 Analysis: Knowledge Base Coverage and Grounded Reasoning

One of the main outcomes of our method is that it enables an affordable small model to *construct* its own knowledge base rather than relying on a fixed external one. A natural concern, however, is that GRPO provides supervision only through downstream QA outcomes, meaning that only a subset of the constructed knowledge may receive direct learning signal. We therefore conduct a comprehensive analysis of both the *quality of the constructed knowledge base* and the model's *ability to reason over it*.

Co-evolution improves the constructed knowledge base. Figure 4a measures whether the answer is reachable from question entities within a bounded number of hops in the constructed KB. GRPO consistently improves reachability over SFT across hop depths, suggesting that outcome supervision encourages the construction of knowledge paths that better support multi-hop reasoning.

Co-evolution improves reasoning over the knowledge base. The gains are not limited to KB construction. Using Gemini-2.5-Flash as a judge, we evaluate whether the model's answers are supported by retrieved knowledge base entries and whether its reasoning chains are logically correct. As shown in Figures 4b and 4c, KBEVO increases the fraction of answers fully grounded in retrieved knowledge from 48% to 55%, while reasoning



(a) Knowledge Base Coverage. (b) Knowledge Grounding. (c) Reasoning composition.

Figure 4: Co-evolution improves KB coverage and grounded reasoning. (a) GRPO improves answer reachability across hop depths and outperforms static KB construction methods. (b–c) GRPO also improves answer grounding and reasoning correctness over the constructed KB. The analysis is conducted on QWEN3-4B using 1K HotpotQA examples.

Method	HotpotQA*	MuSiQue	2Wiki	Avg
(a) Swap KB at inference (reasoning policy fixed)				
KBEVO-1.7B GRPO†	40.3	19.3	47.3	35.6
– EDC DB (Mistral-7B)	30.2	8.3	40.1	26.2
– AutoSchemaKG DB (Llama-3.1-8B)	29.0	16.1	28.8	24.6
– Gemini DB	38.1	19.0	54.5	37.2
(b) Train with a fixed KB (w/o Co-evolution)				
KBEVO-1.7B GRPO†	40.3	19.3	47.3	35.6
– AutoSchemaKG DB (Llama-3.1-8B)	34.1	13.5	38.0	28.5
– Gemini DB	38.6	15.1	54.5	36.1

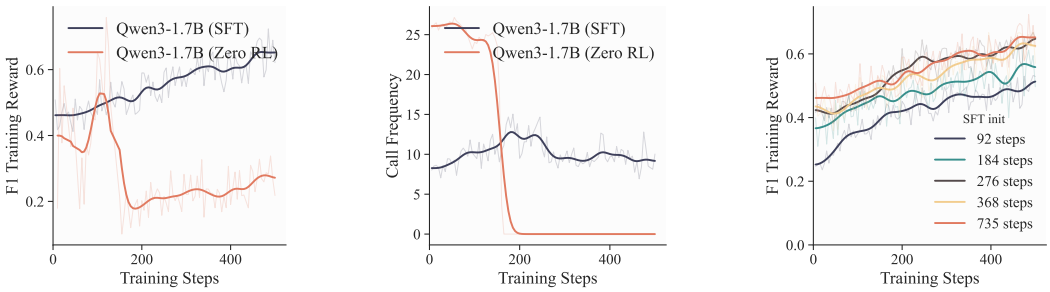
Table 3: Ablating co-evolution. Our learned KB substantially outperforms KBs from static construction methods, while training with a fixed KB generally underperforms joint co-evolution. Results are EM on 1K examples per dataset due to the cost of Gemini-based KB construction. † KBEVO results are evaluated on this same 1K subset and therefore differ slightly from the full-set results in Table 2. * marks the in-domain benchmark.

correctness increases from 45% to 55%. These results together indicate that co-evolution improves both the knowledge available to the model and its ability to use that knowledge during reasoning. Additional analyses of KB structure, lookup behavior, and faithfulness are provided in Appendix C.1.

5.4 What Does Co-Evolution Buy Us?

Our framework jointly optimizes KB construction and reasoning, making their contributions intertwined. We therefore ablate this coupling in two complementary ways: (a) swapping the KB at inference while keeping the reasoning policy fixed, which isolates the quality of the constructed KB; and (b) fixing the KB throughout training, which removes co-evolution between knowledge construction and reasoning.

(a) Swap KB at inference. Holding the KBEVO-1.7B GRPO reasoning policy fixed, our learned KB substantially outperforms KBs constructed by existing static pipelines: 35.6 average EM compared with 26.2 for EDC (Zhang & Soh, 2024) and 24.6 for AutoSchemaKG (Bai et al., 2026). This is despite both baselines using substantially larger extraction models. Replacing our KB with a Gemini-constructed KB yields 37.2 average EM, only 1.6 points higher overall, suggesting that our learned KB captures much of the downstream utility of the stronger teacher KB.



(a) Zero-RL reward collapse. (b) Lookup usage drops. (c) Early SFT is sufficient.

Figure 5: Effect of SFT warmup before RL. Starting RL directly from the base model leads to collapse in both reward and lookup usage, while even relatively early SFT checkpoints are sufficient to enter the structured reasoning regime and achieve similar improvement during RL.

(b) Without co-evolution. We next keep an externally constructed KB fixed throughout training and optimize only the reasoning policy. With AutoSchemaKG, performance drops from 35.6 to 28.5 average EM, showing that reasoning optimization alone cannot compensate for a weaker fixed KB. A fixed Gemini KB reaches 36.1 average EM, comparable to the full co-evolving model, providing a strong reference point for fixed-KB training with a frontier-model teacher.

5.5 Is Supervised Fine-Tuning Necessary Before Reinforcement Learning?

We study whether the model can be trained directly with RL from the base model, optionally with format rewards, following Zeng et al. (2025). In practice, we find that this is insufficient: even with dedicated prompt and format rewards, the model fails to reliably perform database construction and reasoning over it. In Figure 5b, the model struggles to produce valid structured traces and collapses to relying on internal parametric knowledge instead of external lookup. This suggests that structured reasoning with dblookup is substantially out-of-distribution for the base model, even with dedicated prompt tuning, making SFT a crucial initialization step. This aligns with prior work showing that RL requires an established capability foundation to be effective (Gandhi et al., 2025; Yue et al., 2025).

We further study the role of SFT by initializing RL from checkpoints at different stages of SFT (Figure 5c). Earlier SFT checkpoints start with lower performance but quickly improve during RL and eventually reach comparable final performance. This suggests that even a relatively early SFT stage is sufficient to bootstrap the model into the structured reasoning regime, after which RL can effectively refine both database construction and downstream reasoning. A more thorough investigation of this trade-off is left for future work.

6 Limitation

Our framework highlights a few challenges for future work. First, it relies on SFT for warm-starting, although scaling to larger base models with stronger prompting may reduce this dependence. Second, because QA rewards directly supervise only the subset of triplets retrieved during training, they do not guarantee faithfulness across the full constructed KB. While our analyses suggest that unsupported triplets primarily introduce retrieval noise rather than drive the observed gains, broader reward coverage and explicit verification remain important directions for future work. Third, although our analysis in Sec 5.3 suggests reward hacking is not the dominant behavior, the framework may still be vulnerable to it. Finally, a promising direction for future work is to introduce self-play or synthetic question generation with stronger verification, which may further strengthen the co-evolution of knowledge construction and reasoning.

Figure 6: A representative example. The model constructs a knowledge base from source documents and then reasons over it to answer a multi-hop question.

7 Conclusion

We presented a co-evolving framework for knowledge-intensive question answering that jointly optimizes KB construction and reasoning via outcome-based reward signals. Rather than treating KB construction as a fixed preprocessing step, our method learns to construct structured knowledge representations end-to-end alongside the reasoning module, allowing reasoning failures to directly inform and improve the KB. At inference time, offline KB indexing decouples document processing from query time, offering a scalable path to KB construction beyond the constraints of human or LLM annotation. Across the main QA benchmarks, KBEVO-SFT performs comparably to IRCoT, while GRPO substantially improves over SFT, achieving comparable performance with Search-R1. KBEVO-GRPO achieves the best results on MuSiQue and 2Wiki at both model scales, demonstrating that co-evolution can provide competitive reasoning performance while retaining a structured, inspectable, and reusable knowledge base. Our work builds the foundation for this co-evolution, and leveraging these learned KBs to improve model capabilities beyond QA is a promising direction for future research.

8 Acknowledgments

This material is based on work supported by the AI Research Institutes program supported by the NSF and Intel Corporation under NSF award DMR-2433348. This research was also supported by the NSF under awards IIS-2505098 and OAC-2311521; a gift to the LinkedIn–Cornell Bowers Strategic Partnership; Gemini credits grant from Google. DG is supported by an Empire AI Postdoctoral Fellowship. Opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. We thank the members of PIs' labs for helpful discussions.

References

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.2, knowledge manipulation. *International Conference on Learning Representations*, 2023. doi: 10.2139/ssrn.5250621.

Jiaxin Bai, Wei Fan, Qi Hu, Qing Zong, Chunyang Li, Hong Ting Tsang, Hongyu Luo, Yauwai Yim, Haoyu Huang, Xiao Zhou, et al. Autoschemakg: Autonomous knowledge graph construction through dynamic schema induction from web-scale corpora. *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 20557–20584, 2026. doi: 10.18653/v1/2026.acl-long.942.

Scott Barnett, Stefanus Kurniawan, Srikanth Thudumu, Zach Brannelly, and Mohamed Abdelrazek. Seven failure points when engineering a retrieval augmented generation system. In *2024 IEEE/ACM 3rd International Conference on AI Engineering – Software Engineering for AI (CAIN)*, pp. 194–199. ACM, 2024. doi: 10.1145/3644815.3644945.

Baolong Bi, Shaohan Huang, Yiwei Wang, Tianchi Yang, Zihan Zhang, Haizhen Huang, Lingrui Mei, Junfeng Fang, Zehao Li, Furu Wei, et al. Context-dpo: Aligning language models for context-faithfulness. *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 10280–10300, 2025. doi: 10.18653/v1/2025.findings-acl.536.

Baolong Bi, Shenghua Liu, Yiwei Wang, Yilong Xu, Junfeng Fang, Lingrui Mei, and Xueqi Cheng. Parameters vs. context: Fine-grained control of knowledge reliance in language models. In *International Conference on Learning Representations*, volume 2026, pp. 106786–106808, 2026.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pp. 2206–2240. PMLR, 2021.

Jonathan D. Chang, Andrew Drozdov, Shubham Toshniwal, Owen Oertell, Alex Trott, J. Portes, Abhay Gupta, Pallavi Koppol, Ashutosh Baheti, Sean Kulinski, et al. Karl: Knowledge agents via reinforcement learning. *arXiv.org*, 2026. doi: 10.48550/arXiv.2603.05218.

Mingyang Chen, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z. Pan, Wen Zhang, Hua-zeng Chen, Fan Yang, et al. Learning to reason with search for llms via reinforcement learning. *Advances in Neural Information Processing Systems 38*, pp. 94974–94994, 2025. doi: 10.52202/085713-2858.

Ruirui Chen, Weifeng Jiang, Chengwei Qin, Ishaan Singh Rawal, Cheston Tan, Dongkyu Choi, Bo Xiong, and Bo Ai. Llm-based multi-hop question answering with knowledge graph integration in evolving environments. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 14438–14451. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-emnlp.844.

Alla Chepurova, Aydar Bulatov, Mikhail Burtsev, and Yurii Kuratov. Wikontic: Constructing wikidata-aligned, ontology-aware knowledge graphs with large language models. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8304–8319. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.eacl-long.388.

Gheorghe Comanici, E. Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit S. Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv.org*, 2025.

Bhishma Dedhia, Yuval Kansal, and Niraj K Jha. Bottom-up domain-specific superintelligence: A reliable knowledge graph is what we need. *arXiv.org*, 2025. doi: 10.48550/arXiv.2507.13966.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, pp. 4171–4186, 2019. doi: 10.18653/v1/N19-1423.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, T. Henighan, S. Kravec, Zac Hatfield-Dodds, R. Lasenby, Dawn Drain, Carol Chen, et al. Toy models of superposition. *arXiv.org*, 2022. doi: 10.48550/arXiv.2209.10652.

Kanishk Gandhi, Ayush Chakravarthy, Anikait Singh, Nathan Lile, and Noah D Goodman. Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars. *arXiv.org*, 2025. doi: 10.48550/arXiv.2503.01307.

Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu, Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu. Beyond ten turns: Unlocking long-horizon agentic search with large-scale asynchronous rl. *arXiv.org*, 2025. doi: 10.48550/arXiv.2508.07976.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *International conference on machine learning*, pp. 3929–3938. PMLR, 2020.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *International Conference on Computational Linguistics*, pp. 6609–6625. International Committee on Computational Linguistics, 2020. doi: 10.18653/V1/2020.COLING-MAIN.580.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *Conference of the European Chapter of the Association for Computational Linguistics*, pp. 874–880. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2021.eacl-main.74.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Jiajie Jin, Yutao Zhu, Xinyu Yang, Chenghao Zhang, and Zhicheng Dou. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. In *The Web Conference*, pp. 737–740. ACM, 2024. doi: 10.1145/3701716.3715313.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. Large language models struggle to learn long-tail knowledge. In *International conference on machine learning*, pp. 15696–15707. PMLR, 2022.

Yuval Kansal and Niraj K Jha. Knowledge graphs are implicit reward models: Path-derived signals enable compositional reasoning. *arXiv.org*, 2026. doi: 10.48550/arXiv.2601.15160.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Conference on Empirical Methods in Natural Language Processing*, pp. 6769–6781. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.550.

Jeonghoon Kim, Heesoo Jung, Hyeju Jang, and Hogun Park. Improving multi-hop logical reasoning in knowledge graphs with context-aware query representation learning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 15978–15991. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.946.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, F. Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, M. Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Neural Information Processing Systems*, 33:9459–9474, 2020.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 5420–5438. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.276.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Neural Information Processing Systems*, 35:17359–17372, 2022. doi: 10.52202/068431-1262.

F. Petroni, Tim Rocktäschel, Patrick Lewis, A. Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. Language models as knowledge bases? In *Conference on Empirical Methods in Natural Language Processing*, pp. 2463–2473. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1250.

Hadi Pouransari, David Grangier, C Thomas, Michael Kirchhof, and Oncel Tuzel. Pretraining with hierarchical memories: separating long-tail and common knowledge. *arXiv.org*, 2025. doi: 10.48550/arXiv.2510.02375.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331, 2023. doi: 10.1162/tacl_a_00605.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Conference on Empirical Methods in Natural Language Processing*, pp. 3980–3990. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1410.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Annual Meeting of the Association for Computational Linguistics*, pp. 4498–4507. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.412.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, R. Raileanu, M. Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems 36*, 36:68539–68551, 2023. doi: 10.52202/075280-2997.

Zhihong Shao, Peiyi Wang, Qihao Zhu, R. Xu, Jun-Mei Song, Mingchuan Zhang, Y. K. Li, Yu Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv.org*, 2024. doi: 10.48550/arXiv.2402.03300.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Scharli, and Denny Zhou. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pp. 31210–31227. PMLR, 2023. doi: 10.48550/arXiv.2302.00093.

Xingyu Tan, Xiaoyang Wang, Qing Liu, Xiwei Xu, Xin Yuan, and Wenjie Zhang. Paths-over-graph: Knowledge graph empowered large language model reasoning. In *Proceedings of the ACM on Web Conference 2025*, pp. 3505–3522. ACM, 2025. doi: 10.1145/3696410.3714892.

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, 2021. doi: 10.1162/tacl_a_00475.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 10014–10037. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.acl-long.557.

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

Zijian Wang, Bin Wang, Haifeng Jing, Huayu Li, and Hongbo Dou. Walk wisely on graph: Knowledge graph reasoning with dual agents via efficient guidance-exploration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 12818–12826. Association for the Advancement of Artificial Intelligence (AAAI), 2025. doi: 10.1609/aaai.v39i12.33398.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv*, 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 *Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380. Association for Computational Linguistics, 2018. doi: 10.18653/v1/D18-1259.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2022.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *Advances in Neural Information Processing Systems 38*, pp. 64304–64339, 2025. doi: 10.52202/085713-1933.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv.org*, 2025. doi: 10.48550/arXiv.2503.18892.

Bowen Zhang and Harold Soh. Extract, define, canonicalize: An LLM-based framework for knowledge graph construction. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 9820–9836, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.548. URL https://aclanthology.org/2024.emnlp-main.548/.

Linxi Zhao, Sofian Zalouk, Christian Belardi, Justin Lovelace, Jin Zhou, Ryan Noonan, Dongyoung Go, Kilian Weinberger, Yoav Artzi, and Jennifer Sun. Pre-training limited memory language models with internal and external knowledge. In *International Conference on Learning Representations*, volume 2026, pp. 60117–60152, 2026.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 414–431. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.22.

Zexuan Zhong, Zhengxuan Wu, Christopher D Manning, Christopher Potts, and Danqi Chen. Mquake: Assessing knowledge editing in language models via multi-hop questions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 15686–15702. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.971.

A Method Details

A.1 GRPO objective function

The joint GRPO objective is:

$$\mathcal{J}(\theta) = \lambda\, \mathcal{J}^{\text{kb}}(\theta) + \mathcal{J}^{\text{qa}}(\theta) \tag{1}$$

where λ balances the contribution of the two phases. Since each question produces K KB trajectories but $K \times M$ QA trajectories, we set $\lambda = M$ to match the total weight of both phases.

For Phase 1 (KB construction):

$$\mathcal{J}^{\text{kb}}(\theta) = \mathbb{E}_{\substack{(c,q,a^*)\sim\mathcal{D},\\ \{G_k\}_{k=1}^{K}\sim\pi_{\theta_{\text{old}}}(\cdot|p^{\text{kb}}(c))}} \Bigg[\sum_{k=1}^{K} \frac{1}{|G_k|} \sum_{t=1}^{|G_k|} \min\Big(\rho^{\text{kb}}_{k,t} A^{\text{kb}}_k,\ \text{clip}\big(\rho^{\text{kb}}_{k,t},\, 1-\epsilon,\, 1+\epsilon\big) A^{\text{kb}}_k\Big)\Bigg]. \tag{2}$$

For Phase 2 (QA), conditioned on the constructed KB G_k:

$$\mathcal{J}^{\text{qa}}(\theta) = \mathbb{E}_{\substack{(c,q,a^*)\sim\mathcal{D},\ G_k\sim\pi_{\theta_{\text{old}}}(\cdot|p^{\text{kb}}(c))\\ \{\hat{y}_{k,m}\}_{m=1}^{M}\sim\pi_{\theta_{\text{old}}}(\cdot|p^{\text{qa}}(q,G_k))}} \Bigg[\sum_{k=1}^{K} \sum_{m=1}^{M} \frac{1}{|\hat{y}_{k,m}|} \sum_{t=1}^{|\hat{y}_{k,m}|} \min\Big(\rho^{\text{qa}}_{k,m,t} A^{\text{qa}}_{k,m},\ \text{clip}\big(\rho^{\text{qa}}_{k,m,t},\, 1-\epsilon,\, 1+\epsilon\big) A^{\text{qa}}_{k,m}\Big)\Bigg]. \tag{3}$$

where the importance ratios are:

$$\rho^{\text{kb}}_{k,t} = \frac{\pi_\theta(G_{k,t} \mid p^{\text{kb}}(c),\, G_{k,<t})}{\pi_{\theta_{\text{old}}}(G_{k,t} \mid p^{\text{kb}}(c),\, G_{k,<t})} \tag{4}$$

$$\rho^{\text{qa}}_{k,m,t} = \frac{\pi_\theta(\hat{y}_{k,m,t} \mid p^{\text{qa}}(q, G_k),\, \hat{y}_{k,m,<t})}{\pi_{\theta_{\text{old}}}(\hat{y}_{k,m,t} \mid p^{\text{qa}}(q, G_k),\, \hat{y}_{k,m,<t})} \tag{5}$$

Advantage calculation. For Phase 1 (KB construction), we first aggregate QA rewards into a KB-level reward $\bar{r}_{b,k} = \frac{1}{M}\sum_{m=1}^{M} r_{b,k,m}$, then normalize across the K KBs within each question b:

$$A^{\text{kb}}_{b,k} = \frac{\bar{r}_{b,k} - \mu^{\text{kb}}_b}{\sigma^{\text{kb}}_b}, \quad \mu^{\text{kb}}_b = \frac{1}{K}\sum_{k=1}^{K} \bar{r}_{b,k} \tag{6}$$

For Phase 2 (QA), we normalize rewards within each (b, k) group across the M rollouts:

$$A^{\text{qa}}_{b,k,m} = \frac{r_{b,k,m} - \mu^{\text{qa}}_{b,k}}{\sigma^{\text{qa}}_{b,k}}, \quad \mu^{\text{qa}}_{b,k} = \frac{1}{M}\sum_{m=1}^{M} r_{b,k,m} \tag{7}$$

Both baselines are derived solely from the QA reward signal, but at different aggregation scopes: Phase-1 baselines average across the K candidate KBs for a question, while Phase-2 baselines average across the M rollouts conditioned on a single KB.

A.2 Training and Inference Cost

The SFT warm-start is light: 6k traces, with a 0.4-epoch checkpoint already sufficient (Figure 5c). GRPO uses more rollouts per question, but each is short, a lookup injects

roughly ten tokens, whereas document-based search appends a full passage every turn and grows the context across turns (Jin et al., 2025), which is why Table 2 matches rollout budget rather than rollout count. At inference the knowledge base is built offline once per document and reused, so the per-query cost is one Phase-2 generation plus embedding lookups, with no $K \times M$ sampling, no teacher model, and no schema induction stage.

Phase	Cost	Recurs at inference?
Training (one-time)		
SFT warm-start	6k HotpotQA traces (Gemini 2.5 Flash), 3 epochs	No
GRPO	36 rollouts/question (4 KB + 4×8 QA), 500 steps, $\beta = 0$	No
Inference (per query)		
KB construction	Built offline once per document, reused across queries	Amortized
Retrieval	top-$k = 4$ MiniLM lookup, no LLM call	Per lookup
QA	One short Phase-2 generation	Per query

Table 4: Cost breakdown of KBEVO. The additional expense is confined to a one-time SFT warm-start and the $K + K \times M$ rollouts during GRPO. At inference the knowledge base is constructed offline once per document and reused across queries, so per-query cost is a single Phase-2 generation plus embedding lookups; no $K \times M$ sampling and no teacher model recur at inference.

B Experimental Setup

B.1 Training Details

B.1.1 SFT Data Generation

We generate synthetic two-phase supervision using Gemini 2.5 Flash. For each training example, the teacher model first produces a Phase 1 knowledge base constructed from the supporting context, and then generates a Phase 2 reasoning trace that answers the question by issuing database lookups over the constructed knowledge base. These trajectories are used as supervised fine-tuning data.

Extract triplets from the following context. Each triplet must be of the form (entity, relationship, value).
Rules:

- An entity is the subject of a sentence. Do not use pronouns (e.g., "he" or "it") to describe an entity; use the full descriptive name.
- Relationships refer to any characteristic of an entity. Even if a relationship is not explicitly mentioned, you must include it. Relationships can describe things the entity was involved in or any descriptive characteristic of that entity. If an entity is described with multiple characteristics at once, create separate triplet entries for each characteristic.

Example: From the text "Albert Einstein was a German theoretical physicist best known for developing the theory of relativity," the triplets are (Albert Einstein, nationality, Germany), (Albert Einstein, occupation, theoretical physicist), and (Albert Einstein, best known for, developing the theory of relativity).
For example, from the text "In 1796, Napoleon commanded a military campaign against the Austrians and their Italian allies in the War of the First Coalition," you would extract: (Napoleon, commanded a military campaign in 1796 against, the Austrians and their Italian allies in the War of the First Coalition), (Napoleon, commanded a military campaign against the Austrians and their Italian allies in the War of the First Coalition, in 1796), and (Napoleon, commanded a military campaign against Austrians and Italians, in the War of the First Coalition in 1796).
Given these examples, generate explicit and implicit triplets from the following context.
Rules:

- The relationships should be as precise as possible while not requiring external information to look them up.
- Do NOT include multiple (entity, relation) pairs with different values. In such cases, the relationship or entity is not precise enough, and you must add more precision.

Here is the context:
{context}

B.1.2 Database and Retrieval Implementation

The database is a collection of `(entity, relationship, value)` triplets. Each `(entity, relationship)` pair is embedded using `sentence-transformers/all-MiniLM-L6-v2`, and during retrieval, the `(entity, relationship)` query is matched to the nearest pairs via cosine similarity of the embeddings, retrieving the top k=4 results above a 0.6 cosine similarity threshold. We found that this combination of threshold and top-k yielded the highest performance after SFT training. Furthermore, we augment the database with *reverse-index entries*: for each `(entity, relationship, value)` triplet, we include `(value, relationship, entity)` in the database. We find that this improves initial SFT performance, the learning ability of the model during reinforcement learning, and the final performance of the model.

Category	SFT	GRPO
Model & Data		
Base model	Qwen3-1.7B / Qwen3-4B	SFT checkpoint (step 368 for 1.7B, step 735 for 4B)
Dataset	Gemini 2-phase trajectories (HotpotQA, 6k)	HotpotQA
Train size / Eval size	6k / –	7000 / 100
Use special DB lookup tokens	Yes	Yes
Optimization		
Training objective	Supervised fine-tuning	GRPO
Learning rate	5×10^{-5}	5×10^{-6}
Epochs / Max steps	3 epochs	500 steps
Scheduler	Cosine	Cosine
Warmup ratio	0.1	0.1
Weight decay	0.01	–
Max grad norm	–	1.0
β	–	0.0
Batching		
Per-device train batch size	24 / 8	16 / 8
Per-device eval batch size	24 / 8	32
Gradient accumulation	2 / 6	8 / 16
Effective batch size	48	512
Num GPUs	1	4 (B200)
Sequence / Generation		
Max sequence length	2048 / 1024	4096 (vLLM), 1024 completion
Top-p / Temperature / Top-k	–	0.95 / 1.0 / 4
Num generations (train / eval)	–	$36 = 4 + 4 \times 8$ (4 KB + 32 QA)
Two-phase & Retrieval		
Two-phase training	Implicit in supervision data	Enabled
Num DB rollouts	–	4
QA rollouts per DB	–	8
Reward function	–	F1
Phase-1 prompt type	–	SFT
Phase-1 DB weight mode	–	Count
Retrieval threshold / top-k	–	0.6 / 4
Use inverses	–	Enabled
Adaptive k	–	Disabled
System		
Precision	bf16	bf16 / vLLM
Gradient checkpointing	–	Enabled
vLLM (colocate)	–	Enabled
vLLM memory utilization	–	0.4 / 0.15
Logging & Checkpointing		
Logging steps	10	5
Eval strategy	Epoch	Steps (every 100)
Save strategy	Steps	Steps
Save steps	0.125 epoch	25
Save total limit	8	5

Table 5: Training configurations for KBEVO on multi-hop QA. We first perform supervised fine-tuning (SFT) on two-phase KB-construction and QA trajectories, then continue joint optimization with GRPO.

B.1.3 Training hyperparameters

We train KBEVO in two stages. In the first stage, we perform supervised fine-tuning (SFT) from QWEN3-1.7B and QWEN3-4B on Gemini-generated two-phase reasoning traces constructed from HotpotQA. We train for 3 epochs with learning rate 5×10^{-5}, per-device batch size 24, gradient accumulation 2, cosine learning rate decay, warmup ratio 0.1, weight decay 0.01, and maximum sequence length 2048.

In the second stage, we initialize GRPO from the SFT checkpoint and continue training on HotpotQA for 500 steps with learning rate 5×10^{-6}, effective batch size 512, cosine learning rate decay, warmup ratio 0.1, and maximum gradient norm 1.0. For each question, we sample K=4 database (KB) rollouts and N=32 QA rollouts (M=8 QA rollouts per KB candidate, 36 total rollouts), jointly optimizing database construction and reasoning with an outcome-based F1 reward. We use top-p sampling ($p = 0.95$), temperature 1.0, top-$k = 4$, retrieval threshold 0.6, maximum completion length 1024, and vLLM colocation with gradient checkpointing for efficient batched inference.

B.2 Search-R1 Reimplementation

Training and retrieval setup. Since Search-R1 (Jin et al., 2025) does not release checkpoints for QWEN3, we reimplement it on QWEN3-1.7B and QWEN3-4B using `verl` with multi-turn tool interaction. We initialize directly from the base models, without SFT, following the original Search-R1 training paradigm. To enable a controlled comparison with KBEVO, we

use the same 7K HotpotQA training examples, train for 500 update steps, and use the same token-level F1 outcome reward. Each training step samples 16 prompts with 5 rollouts per prompt. We use temperature 1.0, top-$p = 0.95$, and top-$k = 4$, with at most five assistant turns and a maximum response length of 2,048 tokens.

Search-R1 retrieves over unstructured text using an E5-base-v2 dense retriever with a FAISS index, returning the top three passages for each search query. Retrieved tool responses are truncated to 1,024 characters. The search tool is exposed through the native QWEN3 tool-calling interface as a function taking a list of search queries.

Prompt adaptation for QWEN3. In preliminary runs, directly applying the original Search-R1 prompt to QWEN3 frequently resulted in long-form chain-of-thought generation without invoking the search tool. We therefore make two prompt-level adaptations while leaving the training and retrieval setup unchanged. First, we use `<thinking>...</thinking>` rather than `<think>...</think>` to avoid interfering with QWEN3's native thinking interface. Second, we provide a single three-hop in-context demonstration illustrating concise decomposition, iterative search, and the exact JSON tool-call format expected by the QWEN3 tool interface. No supervised trajectories are used for Search-R1.

System: You are a helpful and harmless assistant.
User: Answer the given question. You must conduct reasoning inside `<thinking>` and `</thinking>` first every time you get new information. Each `<thinking>` should be ONE BRIEF SENTENCE stating what you need next — not detailed analysis. After reasoning, if you lack knowledge, call the search tool. When you have enough information, provide the answer inside `<answer>` and `</answer>` without detailed illustrations.
Here is an example showing the expected format and brevity:
Question: What is the population of the capital city of the country where the inventor of the World Wide Web was born?
`<thinking>`I need to identify the inventor of the World Wide Web and where he was born.`</thinking>`
`<tool_call>`{"name": "search", "arguments": {"query_list": ["inventor of the World Wide Web birthplace"]}}`</tool_call>`
`<tool_response>`Doc 1 (Title: "Tim Berners-Lee"): Sir Timothy John Berners-Lee, also known as TimBL, is an English computer scientist best known as the inventor of the World Wide Web. He was born in London, England.`</tool_response>`
`<thinking>`Tim Berners-Lee was born in England, so I need the capital of the United Kingdom.`</thinking>`
`<tool_call>`{"name": "search", "arguments": {"query_list": ["capital of the United Kingdom"]}}`</tool_call>`
`<tool_response>`Doc 1 (Title: "London"): London is the capital and largest city of England and the United Kingdom.`</tool_response>`
`<thinking>`The capital is London, so I need its population.`</thinking>`
`<tool_call>`{"name": "search", "arguments": {"query_list": ["population of London"]}}`</tool_call>`
`<tool_response>`Doc 1 (Title: "London"): London has a population of approximately 9 million people.`</tool_response>`
`<thinking>`The population is about 9 million.`</thinking>`
`<answer>`9 million`</answer>`
Now answer this question: {QUESTION}

B.3 ConFiQA Evaluation Details

Data and counterfactual conditions. We use the ConFiQA-MR evaluation set released with Context-DPO (Bi et al., 2025), which provides paired original and counterfactual contexts, labeled reasoning paths represented as `(entity, relation, value)` triplets, answers, and aliases. We first shuffle the dataset once with seed 42 and then retain the first 1,000 examples. We use the same 1,000 source examples and ordering in every condition.

We construct three evaluation sets. MR-ORIG uses the original context, labeled reasoning path, answer, and aliases for every example. To construct the counterfactual conditions, we define a forward knowledge key as `(entity, relation)` and require that the aggregate labeled paths contain at most one distinct value for every such key. We solve this selection problem with a deterministic binary program and use lexicographic tie-breaking that prefers counterfactual substitutions at earlier positions in the fixed ordering. MR-CF-100 uses an conflict-free selection of 100 counterfactual examples. MR-CF-356 uses 356 counterfactual examples, the maximum feasible number under this forward-key definition. All remaining examples retain their original versions.

Database and corpus construction. For KBEVO, we aggregate the labeled triplets from the reasoning paths using the selected version of each example into a single database and use that database for all questions in the corresponding setting. The model does not receive the gold path for the current question directly; it retrieves from the shared database constructed from all 1,000 examples.

For Search-R1 (Jin et al., 2025), we instead build the retrieval corpus and E5-base-v2 index from the corresponding 1,000 ConFiQA contexts directly. Each passage is the full natural-language context supplied by ConFiQA, which embeds the labeled reasoning path in longer descriptive text. Search-R1 retrieves the top three passages from this closed corpus. We evaluate using the ICL three-hop tool-call format, with temperature 1.0, top-p 0.95, sampling top-k 4, at most five turns, and a maximum response length of 2,048 tokens. Following the checkpoints' training-time configuration, each combined retrieval response is truncated to its first 1,024 characters.

Metric. We report normalized exact match against the condition-specific answer and its aliases. Before comparison, predictions and reference answers are lowercased, stripped of punctuation and the articles *a*, *an*, and *the*, and normalized for whitespace. KBEVO uses the checkpoints and inference configuration described in Appendix B.1.

B.4 LLM as a judge prompt

LLM as a judge prompt for evaluating knowledge base faithfulness and soundness.

[System]
You are an expert evaluator for knowledge graph databases. Your task is to evaluate the quality of a knowledge database (a set of triplets) that was generated from a given source context (a wiki document).
You will evaluate the database on two dimensions: **Faithfulness** and **Soundness**.

Dimension 1: Faithfulness (Is the triplet grounded in the context?)
For each triplet (entity, relation, value), determine whether the information it expresses can be found in or directly inferred from the provided context.

1. **Faithful**: The triplet's meaning is supported by the context. It does not need to be a verbatim match — reasonable paraphrasing and direct inference are acceptable. Inference must be strictly logical based ONLY on the provided text, without incorporating external world knowledge.
2. **Hallucinated**: The triplet contains information that does NOT appear in and cannot be directly inferred from the context. This includes fabricated entities, invented relations, values not mentioned or implied in the source, or facts that are only true based on external knowledge not present in the context.

Dimension 2: Soundness (Is the triplet well-formed and useful?)
For each triplet that is NOT hallucinated, check for the following soundness issues. A single triplet may have multiple issues.
Issue types:

1. **Ambiguous Entity/Value**: The entity or value uses unclear references (e.g., pronouns like "he", "it", "the company") instead of explicit names, or is ambiguous without additional context.
   - Bad: (He, founded, the company)
   - Good: (Elon Musk, founded, SpaceX)
2. **Trivial / Meaningless**: The triplet expresses an obvious, near-tautological, or extremely low-information fact that adds no meaningful knowledge.
   - Bad: (Einstein, is, a person) — states the obvious, no useful information
   - Good: (Einstein, developed, Theory of Relativity)
3. **Non-specific**: Any field (entity, relation, or value) is vague or imprecise where the context provides more specific information.
   - Bad: (Company X, revenue, very high) — when the context states an exact figure
   - Good: (Company X, revenue, 574.8B)
4. **Malformed / Span Error**: The triplet has severe structural problems. Only flag this when the relation or entity field contains a full clause or sentence (with multiple verbs or subordinate clauses), when entity fields pack multiple distinct entities, or when the triplet is incomplete / unparseable. Short prepositional phrases in the relation field are acceptable.
   - Bad: (Apple Inc., released the first iPhone which revolutionized the market in, 2007) — relation is an entire clause
   - Good: (Apple Inc., released, the first iPhone)
5. **Reversed / Misplaced Roles**: The entity and value are swapped, or the directionality of the relation is incorrect, causing the triplet to express the opposite or a nonsensical meaning.
   - Bad: (Theory of Relativity, developed, Einstein) — subject and object are reversed
   - Good: (Einstein, developed, Theory of Relativity)

Output Format

1. 'reasoning': Brief explanation covering why the triplet is faithful/hallucinated, and if faithful, why specific soundness issues were flagged (if any). Keep it concise.
2. For hallucinated triplets, skip soundness evaluation — set soundness_issues to [].
3. Evaluate every triplet in the database. Do not skip any.

[User]
Context (Source Wiki Document): {context}
Knowledge Database to Evaluate: {database_triplets}
Evaluate each triplet in this database according to the rubric. Return your evaluation as JSON.

LLM as a judge prompt for evaluating knowledge base grounding and reasoning correctness.

[System]
You are an expert evaluator for knowledge-graph-augmented question answering. You will be given: 1. A question 2. A knowledge database (a set of triplets) 3. A model's completion (its thinking process and DB lookups)
You will evaluate the completion on two dimensions: **DB Grounding** and **Reasoning Correctness**.

Dimension 1: DB Grounding (Is the final answer based on DB lookups?)

Determine whether the model's final answer is supported by the information retrieved from DB lookups.

- **fully_grounded**: The final answer is entirely based on information retrieved from DB lookups. Every key claim in the answer can be traced back to a DB return value.
- **partially_grounded**: The final answer is partly based on DB lookups and partly based on the model's own internal knowledge. Some claims are supported by DB returns, but others are not.
- **ungrounded**: The final answer has no basis in the DB lookups. The model either used entirely internal knowledge, or fabricated an answer that contradicts or ignores what the DB returned.
- **no_answer**: The model failed to produce a final answer (e.g., got stuck in a loop, gave up, or the completion was cut off before answering).

Dimension 2: Reasoning Correctness (Is the multi-hop reasoning chain correct?)
Evaluate whether the model's reasoning process — how it chains together information from multiple DB lookups to arrive at a conclusion — is logically sound.

- **correct**: The reasoning chain is logically valid. Each step follows from the previous one, and the conclusion is properly supported by the retrieved information.
- **minor_error**: The overall reasoning direction is correct, but there are small issues — e.g., a skipped step that doesn't affect the final answer, or a slightly imprecise inference.
- **major_error**: The reasoning chain has significant logical errors — e.g., drawing wrong conclusions from the retrieved data, confusing entities, or making invalid logical jumps.

Note: If the grounding is "no_answer", set reasoning to "major_error" since no valid reasoning was completed.

Output Instructions

- 'grounding_reasoning': Brief explanation of why the grounding label was chosen.
- 'reasoning_reasoning': Brief explanation of why the reasoning correctness label was chosen.
- Evaluate based on the FINAL answer, not intermediate thoughts.
- The DB lookup format in completions is:`<|db_entity|>ENTITY <|db_relationship|>RELATIONSHIP<|db_return|>VALUE<|db_end|>`

[User]
Question: {question}
Knowledge Database: {database_triplets}
Model Completion: {completion}
Evaluate this completion according to the rubric.

C Detailed Analysis

Checkpoint note. Analyses in Section 5.3 and Appendix C use an earlier checkpoint trained with retrieval top-$k = 1$, while the main results use the final $k = 4$ configuration. Absolute QA results in these analyses are therefore not directly comparable to Table 2.

C.1 Intrinsic Quality of the Constructed Knowledge Base

We further analyze how co-evolution changes the constructed knowledge base, independently of downstream QA.

Co-evolution improves KB structure and coverage. As shown in Table 6, GRPO produces larger, better-connected KBs and reduces failed or redundant lookups. Compared with static KG construction methods, KBEVO also achieves higher triplet quality and substantially better multi-hop answer reachability (Table 7). These results suggest that outcome supervision encourages knowledge structures that are better suited for downstream reasoning.

	Qwen3-1.7B		Qwen3-4B	
Metric	SFT	GRPO	SFT	GRPO
KB Structure				
$\|E\|$ (unique entities)	156.6	169.4	156.8	186.6
$\|R\|$ (unique relations)	111.3	135.9	109.7	175.3
#Triplets	166.2	225.9	165.7	299.1
Avg Degree	2.15	2.71	2.14	3.30
#Components ↓	13.1	10.3	13.0	7.9
Giant Comp. (%) ↑	57.2	65.1	57.8	73.8
Lookup Efficiency				
Redundancy Rate (%) ↓	6.3	2.3	7.2	6.3
Unknown Rate (%) ↓	6.6	2.3	7.1	2.0

Table 6: Knowledge base structure and lookup efficiency on HotpotQA†, averaged per example. *Redundancy Rate* is the fraction of lookups repeating an entity–relation pair already queried; *Unknown Rate* is the fraction matching no entry. Co-evolution yields larger, better-connected knowledge bases that the reasoning policy also queries more successfully. †These diagnostics were computed using an earlier set of model checkpoints and are retained for reference.

	Triplet Quality			Graph Structure			Answer Reachability	
KB Source	Prec.↑	Rec.↑	F1↑	$\|R\|$	#Comp.↓	Giant %↑	≤2 hop↑	≤4 hop↑
EDC (Mistral-7B)	0.864	0.477	0.600	–	–	–	63.7%	69.5%
AutoSchemaKG (Llama-3.1-8B)	0.888	0.838	0.860	58.8	18.8	51.0%	72.1%	80.8%
KBEVO-1.7B GRPO	0.925	0.923	0.924	135.9	10.3	65.1%	81.3%	88.4%
KBEVO-4B GRPO	0.938	0.952	0.945	175.3	7.9	73.8%	84.0%	90.4%

Table 7: Intrinsic knowledge base quality on HotpotQA, independent of downstream QA. KBEVO produces more accurate and better-connected KBs than static construction pipelines, with substantially higher multi-hop answer reachability.

Improved coverage does not imply uniformly higher faithfulness. GRPO increases coverage and connectivity, but also increases hallucination and soundness-issue rates relative to SFT (Table 8). We therefore examine below whether these unfaithful triplets contribute to the downstream gains.

Category	AutoSchemaKG	Qwen3-1.7B		Qwen3-4B	
	(Llama-3.1-8B)	SFT	GRPO	SFT	GRPO
Valid Triplets ↑	35.6%	92.3%	85.0%	96.0%	84.3%
Hallucinated ↓	11.0%	3.3%	7.5%	1.5%	6.2%
Soundness Issues ↓	53.4%	4.4%	7.4%	2.6%	9.5%

Table 8: Triplet-level correctness on HotpotQA (100 samples), judged with the same LLM-as-judge protocol across all knowledge bases (Appendix B.4). *Valid Triplets* are free of hallucination and soundness issues. GRPO increases both hallucination and soundness-issue rates relative to SFT; we examine whether this explains the downstream gains in Appendix C.2.

C.2 Are the Gains Driven by Faithful Knowledge?

Because QA reward directly supervises only retrieved triplets, we examine whether unretrieved knowledge degrades and whether the QA gains rely on hallucinated content.

Unretrieved knowledge remains faithful. Although only about 5% of constructed triplets are retrieved during reasoning, unused triplets are nearly as faithful as used ones (Table 9). This suggests that sparse reward exposure does not substantially degrade the unretrieved portion of the KB.

Hallucinated knowledge does not drive the gains. Removing unfaithful triplets slightly improves EM (Table 9), indicating that they primarily act as retrieval noise. Moreover, GRPO substantially improves both EM and grounding even when all retrieved triplets are faithful (Table 10). Thus, the gains are better explained by improved use of faithful knowledge than by exploiting hallucinated content.

	Post-hoc Filtering (EM)		**Faithfulness by Reward Exposure**			
Model	No Filter	Filter Unfaithful	Used Faithful%	Used Cov.%	Unused Faithful%	Unused Cov.%
KBEVO-1.7B SFT	–	–	97.6	5.3	96.8	94.7
KBEVO-1.7B GRPO	36.2	38.4	94.3	5.3	92.7	94.7
KBEVO-4B SFT	–	–	98.6	5.4	98.2	94.6
KBEVO-4B GRPO	41.2	43.6	94.1	5.1	93.2	94.9

Table 9: Faithfulness analysis of the constructed knowledge base on HotpotQA (1K examples). *Left:* removing triplets judged unfaithful slightly improves EM, indicating that hallucinated triplets act primarily as retrieval noise rather than an exploitable shortcut. *Right:* triplets retrieved during reasoning (*Used*) and never retrieved (*Unused*) show similar faithfulness under the same evaluation protocol (Appendix B.4), suggesting that sparse reward exposure does not substantially degrade the unretrieved portion of the knowledge base.

Model	Retrieval Faithfulness	Share%	EM	Fully Grounded	Partially Grounded	Ungrounded
KBEVO-1.7B SFT	All faithful	84.0	33.0	49.6	10.6	30.2
KBEVO-1.7B SFT	Has hallucinated	16.0	17.5	28.1	9.4	39.4
KBEVO-1.7B GRPO	All faithful	60.7	40.3	59.7	11.1	27.1
KBEVO-1.7B GRPO	Has hallucinated	39.3	24.2	41.6	10.2	39.5
KBEVO-4B SFT	All faithful	90.7	35.0	50.1	8.8	27.2
KBEVO-4B SFT	Has hallucinated	9.3	24.7	35.5	7.5	35.5
KBEVO-4B GRPO	All faithful	58.0	48.8	68.2	9.7	16.8
KBEVO-4B GRPO	Has hallucinated	42.0	30.5	42.2	10.0	27.7

Table 10: Faithfulness-conditioned grounding and QA performance on HotpotQA (1K examples). Reasoning chains are partitioned by whether all retrieved triplets are judged faithful or at least one is hallucinated. GRPO substantially improves both EM and grounding even when reasoning relies entirely on faithful triplets, indicating that the gains are not driven by hallucinated content. The fraction of chains retrieving at least one hallucinated triplet nevertheless increases under GRPO.