GR-Agent: Adaptive Graph Reasoning Agent under Incomplete Knowledge

Dongzhuoran Zhou[1,2], Yuqicheng Zhu[2,3], Xiaxia Wang[4], Hongkuan Zhou[2,3], Jiaoyan Chen[6], Steffen Staab[3,7], Yuan He[5,4]*, Evgeny Kharlamov[1,2]*
[1]University of Oslo, [2]Bosch Center for AI, [3]University of Stuttgart, [4]University of Oxford,
[5]Amazon, [6]The University of Manchester, [7]University of Southampton
dongzhuoran.zhou@de.bosch.com

Abstract

Large language models (LLMs) achieve strong results on knowledge graph question answering (KGQA), but most benchmarks assume complete knowledge graphs (KGs) where direct supporting triples exist. This reduces evaluation to shallow retrieval and overlooks the reality of incomplete KGs, where many facts are missing and answers must be inferred from existing facts. We bridge this gap by proposing a methodology for constructing benchmarks under KG incompleteness, which removes direct supporting triples while ensuring that alternative reasoning paths required to infer the answer remain. Experiments on benchmarks constructed using our methodology show that existing methods suffer consistent performance degradation under incompleteness, highlighting their limited reasoning ability. To overcome this limitation, we present the Adaptive Graph Reasoning Agent (GR-Agent). It first constructs an interactive environment from the KG, and then formalizes KGQA as agent environment interaction within this environment. GR-Agent operates over an action space comprising graph reasoning tools and maintains a memory of potential supporting reasoning evidence, including relevant relations and reasoning paths. Extensive experiments demonstrate that GR-Agent outperforms non-training baselines and performs comparably to training-based methods under both complete and incomplete settings.

1 Introduction

Large language models (LLMs) have demonstrated impressive capability across a wide range of knowledge-intensive tasks [Lewis et al., 2020, Khandelwal et al., 2020, Izacard and Grave, 2021, Borgeaud et al., 2022, Ram et al., 2023]. One representative task is knowledge graph question answering (KGQA), which aims to answer natural language questions using the information provided by a knowledge graph (KG) [Yih et al., 2016, Gu et al., 2021, Ye et al., 2021, Yao et al., 2019, Baek et al., 2023]. A growing body of work suggests that LLMs can be effective at retrieving relevant triples and reasoning over them to answer questions [Luo et al., 2024, He et al., 2024, Mavromatis and Karypis, 2024, Chen et al., 2024, Sun et al., 2024, Jiang et al., 2023].

However, existing benchmarks fall short of properly testing this reasoning ability. Specifically, current datasets are constructed under the assumption that the KG is complete, i.e., every question has direct supporting triples in the KG [Yih et al., 2016, Talmor and Berant, 2018, Gu et al., 2021]. This design reduces the task to a retrieval problem: *models can answer correctly simply by locating the explicit facts in the KG*. For instance, consider the question "Who

*Shared supervision.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Scaling Environments for Agents (SEA).

is Justin Biebers uncle?". In a complete KG, this might be directly answered by a triple such as $\langle \texttt{Justin Bieber}, \texttt{hasUncle}, \texttt{Brad Bieber} \rangle$. Such evaluations therefore conflate true reasoning with shallow retrieval and fail to reflect the reality of incomplete KGs, where *many facts are missing and questions must be answered by reasoning over existing facts*. In this case, if $\langle \texttt{Justin Bieber}, \texttt{hasUncle}, \texttt{Brad Bieber} \rangle$ is missing, the answer must be inferred by combining $\langle \texttt{Brad Bieber}, \texttt{hasSon}, \texttt{Jaxon Bieber} \rangle$ with $\langle \texttt{Jaxon Bieber}, \texttt{hasBrother}, \texttt{Justin Bieber} \rangle$.

To enable systematic evaluation under this realistic setting, we first propose a general method for constructing benchmarks datasets in which each question requires genuine reasoning. That is, for every question, the direct supporting triples are removed from the KG, ensuring that the answer can only be inferred via high-confidence logical rules rather than shallow retrieval. We then evaluate six representative baseline methods on the resulting benchmark, and observe consistent performance degradation across all models when moving from the complete to the incomplete setting. This demonstrates that current approaches predominantly rely on shallow retrieval rather than reasoning.

To address this challenge, we propose the Adaptive Graph Reasoning Agent (GR-Agent), a training-free agentic framework specifically designed to reason over KGs. We construct an interactive environment on top of the KG, which consists of: (i) a mutable state represented by a memory that stores explored paths, grounded reasoning paths, and entities; (ii) an action space comprising reasoning tools for relation-path exploration, reasoning-path grounding, and answer synthesis; and (iii) transition and observation mechanisms that update the memory and expose new candidates during interaction. Through this interaction, the agent adaptively expands the search frontier, prioritizes promising paths, and synthesizes a final answer. A comprehensive empirical study substantiates our claims, showing that GR-Agent consistently outperforms non-training baselines and achieves performance comparable to training-based baselines across both complete and incomplete settings.

Our contributions are three-fold: **First**, we propose a general methodology for constructing benchmarks under incompleteness and instantiate it on the Family and FB15k-237 KGs, providing systematic testbeds for evaluating reasoning ability of LLMs. **Second**, we highlight the challenge posed by KG incompleteness and empirically demonstrate its impact on the reasoning ability of existing methods. **Third**, we introduce GR-Agent, a novel training-free agentic framework for KGQA task.

2 Related Works

KGQA The KGQA task aims to answer natural language questions using the KG $\mathcal{G}$, where $\mathcal{G}$ is represented as a set of binary facts $r(s, o)$, with $r \in \mathcal{R}$ denoting a predicate and $s, o \in \mathcal{E}$ denoting entities. The answer to each question is one or more entities in $\mathcal{G}$. Approaches to KGQA fall into three categories: (1) *Semantic parsing-based methods* [Yih et al., 2016, Gu et al., 2021, Ye et al., 2021], which translate questions into formal queries (e.g., SPARQL) and execute them over the KG. They provide high precision and interpretability but struggle with complex language, diverse logical forms, and large parsing search spaces [Lan et al., 2021]. (2) *Embedding-based methods* [Yao et al., 2019, Baek et al., 2023], which encode questions and entities into a shared space and rank candidates by similarity. They are end-to-end and annotation-free but face challenges in multi-hop reasoning [Qiu et al., 2020], interpretability [Biswas et al., 2023], and prediction uncertainty [Zhu et al., 2024, 2025b,a]. (3) *RAG-based methods*, which go beyond previous approaches by coupling retrieval with the generative reasoning capabilities of LLMs [Luo et al., 2024, He et al., 2024, Mavromatis and Karypis, 2024, Chen et al., 2024].

KGQA Benchmarks Benchmarks like WebQSP Yih et al. [2016], CWQ Talmor and Berant [2018], and GrailQA Gu et al. [2021] assume complete KGs by construction, retaining only questions answerable via direct supporting triples. To study incompleteness, recent work deletes triples randomly or along shortest paths Xu et al. [2024], Zhou et al. [2025], but this often leaves questions unanswerable, conflating missing knowledge with model reasoning limits.

KGQA with Agents Building on RAG-based approaches, a new line of work adopts an *agentic perspective*, where the LLM is treated as an autonomous agent that iteratively interacts with the KG. Representative examples include ToG [Sun et al., 2024], StructGPT [Jiang et al., 2023], KG-Agent [Jiang et al., 2024], and GoG [Xu et al., 2024]. While these approaches highlight the promise of LLMs as reasoning agents, they often rely on fixed exploration ranges, assume complete KGs, or defer reasoning to the LLMs internal knowledge. In contrast, our work introduces an agent

that formalizes the KG as an environment and reasons directly at the relation-path level, enabling multi-hop reasoning over both complete and incomplete KGs.

3 Benchmark Construction

This section introduces a general method for constructing benchmarks to evaluate KGQA under varying degrees of knowledge incompleteness. The key objective is to create natural language questions whose answers are not directly stated in the KG but can be logically inferred through reasoning over alternative paths.

To achieve this, we first mine high-confidence logical rules from the KG to identify triples that are inferable via reasoning. We then remove a subset of these triples while preserving the supporting facts required for inference. Natural language questions are generated based on the removed triples, meaning that models must rely on reasoning rather than direct retrieval to answer the questions.

3.1 Rule Mining

To ensure that questions in our benchmark require reasoning rather than direct lookup, we first identify triples that are logically inferable from other facts. We achieve this by mining high-confidence Horn rules [Horn, 1951] from the original KG using the AMIE3 algorithm [Lajus et al., 2020].

AMIE3 is a widely used rule mining system designed to operate efficiently over large-scale KGs. A logical rule discovered by AMIE3 has the following form:

$$B_1 \wedge B_2 \wedge \cdots \wedge B_n \Rightarrow H\,,$$

where B_i are body atoms and H is the head atom. For example:

$$\texttt{hasParent}(X, Y) \wedge \texttt{hasSibling}(Y, Z) \Rightarrow \texttt{hasUncle}(X, Z)\,,$$

expressing that if Y is a parent of X and Y has a sibling Z, then Z is likely an uncle of X.

A *grounding* of this rule substitutes entities in the KG for variables (e.g., X=`Justin`, Y=`Mary`, Z=`John`). A rule is considered *well supported* if many such groundings exist in the KG, and *confidence* is measured by the proportion for all grounded rule bodies in the KG, their grounded head atom also exists in the KG. Only high-confidence and well-supported rules are retained. Detailed metrics and filtering criteria are provided in Appendix A.

3.2 Dataset Generation

We aim to generate questions that cannot be answered using direct supporting triples, but for which sufficient information is implicitly available in the KG. The core idea is to first remove triples that can be reliably inferred using high-confidence rules mined by AMIE3, and then generate questions based on these removed triples.

- **Triple Removal.** For each mined rule, we select up to 30 groundings where both body and head triples exist. The head triple is then removed while preserving all body triples, ensuring that each removed fact remains inferable and that no body triple required for other groundings is lost.
- **Question Generation.** For each removed triple, we prompt GPT-4 to generate a natural-language question asking for the answer entity based on the predicate and a specified topic entity. To encourage diversity, either the head or tail entity is randomly chosen as the topic, with the other as the answer. The prompt template is given in Appendix E.
- **Answer Set Completion.** Although each question is initially generated based on a single deleted triple, there may exist multiple correct answers in the KG. To ensure rigorous and unbiased evaluation, we construct for each question a complete set of correct answers and mark the answers requiring inference (i.e., without direct supporting triples) as "hard".

3.3 Dataset Overview

KGs. To support a systematic evaluation of reasoning under knowledge incompleteness, we construct benchmark datasets based on two well established KGs: **Family** [Sadeghian et al., 2019]

Rule Type	Family	FB15k-237
Symmetry: $r(x, y) \Rightarrow r(y, x)$	0	27
Inversion: $r_1(x, y) \Rightarrow r_2(y, x)$	6	50
Hierarchy: $r_1(x, y) \Rightarrow r_2(x, y)$	0	76
Composition: $r_1(x, y) \wedge r_2(y, z) \Rightarrow r_3(x, z)$	56	343
Other	83	570
Total	**145**	**1,066**

Table 1: Statistics of mined rules.

Dataset	#Triples	Train	Val	Test	Total Qs
Family-Complete	17,615	1,749	218	198	2,165
Family-Incomplete	15,785	1,749	218	198	2,165
FB15k-237-Complete	204,087	4,374	535	540	5,449
FB15k-237-Incomplete	198,183	4,374	535	540	5,449

Table 2: Dataset statistics.

and **FB15k-237** [Toutanova and Chen, 2015]. These datasets differ in size, structure, and domain coverage, enabling evaluation across both synthetic and real-world settings.

Mined Rules. Table 1 summarizes the number of mined rules for each dataset, categorized by rule type. The listed types (e.g., symmetry, inversion, composition) correspond to common logical patterns, while the *other* category includes more complex or irregular patterns (See Appendix L for details).

Dataset	Example
Family	*Question:* Who is 139's brother? *Topic Entity:* 139 — *Answer:* [205, 138, 2973, 2974] *Direct Supporting Triples:* `brotherOf(139,205)` *Alternative Paths:* `fatherOf(139,14)` $\wedge$ `uncleOf(205,14)` $\Rightarrow$ `brotherOf(139,205)`
FB15k-237	*Question:* What is the currency of the estimated budget for 5297 (Annie Hall)? *Topic Entity:* 5297 (Annie Hall) — *Answer:* [1109 (United States Dollar)] *Direct Supporting Triples:* `filmEstimatedBudgetCurrency(5297, 1109)` *Alternative Paths:* `filmCountry(5297 (Annie Hall), 2896 (United States of America))` $\wedge$ `locationContains(2896 (United States of America), 9397 (New York))` $\wedge$ `statisticalRegionGdpNominalCurrency(9397 (New York), 1109 (United States Dollar))` $\Rightarrow$ `filmEstimatedBudgetCurrency(5297 (Annie Hall), 1109 (United States Dollar))`

Table 3: Examples from our benchmark datasets. Each instance includes a *natural-language question*, a *topic entity*, and the full *set of correct answers*. The red-highlighted answer denotes the *hard answer*, i.e., the one whose supporting triple has been removed in the incomplete KG setting. We also show the corresponding *direct supporting triples* (the deleted triple) and an *alternative reasoning path* derived from a mined rule that enables inference of the answer.

Datasets. Each dataset instance consists of (1) a natural-language question, (2) a topic entity referenced in the question, and (3) a complete set of correct answer entities derived from the original KG. Table 3 presents representative examples from each dataset. The final question set is randomly partitioned into training, validation, and test sets using an 8:1:1 ratio. This split is applied uniformly across both datasets to ensure consistency.

We provide two retrieval sources per dataset: (1) **Complete KG**: the original KG containing all triples. (2) **Incomplete KG**: a modified version where selected triples, deemed logically inferable via AMIE3-mined rules, are removed (Section 3.2).

Table 2 summarizes the number of KG triples and generated questions in each split for both datasets, under complete and incomplete KG settings.

3.4 Evaluation Protocol

Evaluation Setup. Given a natural language question $q \in \mathcal{Q}$, access to a KG $\mathcal{G}$, and a topic entity, the model is designed to return a set of predicted answer entities $\mathcal{P}_q$. Since LLMs typically produce raw text sequences as output, we extract the final prediction set $\mathcal{P}_q$ by applying string partitioning and normalizing, following Luo et al. [2024]. Details of this postprocessing step are provided in Appendix J. Without specific justification, all entities are represented by randomly assigned indices without textual labels (e.g., `Barack Obama` becomes `39`) to ensure that models rely solely on knowledge from the KG rather than memorized surface forms.

Evaluation Metrics. Given a set of questions $\mathcal{Q}$, we denote the predicted answer set $\mathcal{P}_q$ and the gold answer set $\mathcal{A}_q$, respectively, for each question $q \in \mathcal{Q}$. The evaluation metrics are defined as follows:

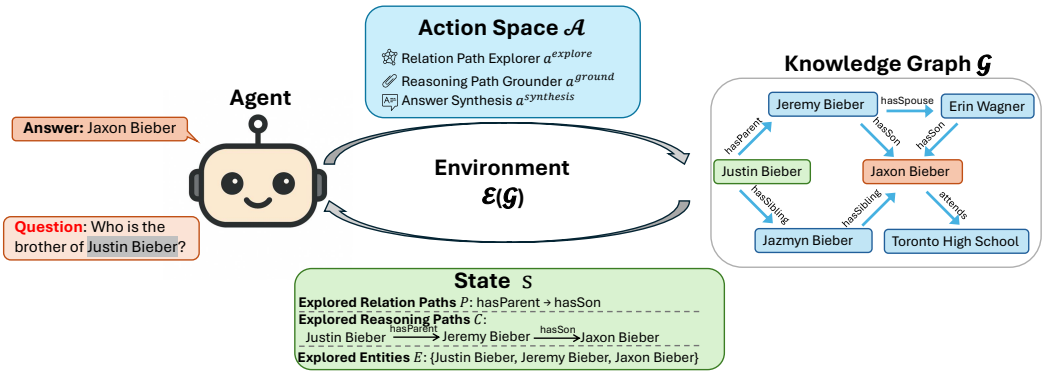


Figure 1: Overview of our GR-Agent. The agent interacts with the environment $\mathcal{E}(\mathcal{G})$, which responds to actions by exposing relation paths and entities. The agent selects actions from the action space $\mathcal{A}$, and each action updates the current state s by updating newly discovered relation paths, grounded reasoning paths, or entities. The action space consists of relation-path exploration, path grounding, and answer synthesis.

- **Hits@Any.** Hits@Any measures the proportion of questions for which the predicted answer set overlaps with the gold answer set, i.e., at least one correct answer is predicted:

$$\text{Hits@Any} = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \mathbb{1}[\mathcal{P}_q \cap \mathcal{A}_q \neq \varnothing] \,.$$

- **Precision** and **Recall.** Precision measures the fraction of predicted answers that are correct, while recall measures the fraction of gold answers that are predicted:

$$\text{Precision} = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \frac{|\mathcal{P}_q \cap \mathcal{A}_q|}{|\mathcal{P}_q|} \,, \quad \text{Recall} = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \frac{|\mathcal{P}_q \cap \mathcal{A}_q|}{|\mathcal{A}_q|} \,.$$

- **F1-score.** The F1-score is the harmonic mean of precision and recall, computed per question and averaged across all questions:

$$\text{F1} = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \frac{2 \cdot |\mathcal{P}_q \cap \mathcal{A}_q|}{|\mathcal{P}_q| + |\mathcal{A}_q|} \,.$$

- **Hard Hits Rate.** For each question q, the *hard answer* is defined as the selected entity $a_q \in \mathcal{A}_q$ whose supporting triple was intentionally removed from the KG. *Hard Hits Rate* (HHR) measures the fraction of correctly answered questions (i.e., Hits@Any) that include the hard answer in predictions:

$$\text{HHR} = \frac{\text{Hits@Hard}}{\text{Hits@Any}} \,, \quad \text{where} \quad \text{Hits@Hard} = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \mathbb{1}[a_q \in \mathcal{P}_q] \,.$$

4 GR-Agent

In this section, we formalize the KGQA task as an agent-environment interaction problem, where is the environment is constructed from a KG $\mathcal{G}$. Following Sutton et al. [1999], we define the environment induced by the KG $\mathcal{G}$ as a tuple $\mathcal{E}(\mathcal{G}) = (\mathcal{S}, \mathcal{A}, T)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, and T is the state transition function.

4.1 Action Space

The agent operates over an action space $\mathcal{A}$, which consists of three reasoning tools that enable interaction with the knowledge graph and facilitate question answering.

Basically, the reasoning tools help the agent explore plausible combinations of predicates that can form relation paths, ground these relation paths into reasoning paths, and synthesize the gathered reasoning paths to answer the question.

Relation Path Exploration. For an entity $e \in \mathcal{E}$ and a hop limit $H \in \mathbb{N}$, the relation path exploration action a^{explore} generates a set of relation paths of length up to H:

$$a^{\text{explore}}(e, H) \subseteq \mathcal{R}^{\leq H}, \tag{1}$$

where each relation path $(r_1, \ldots, r_k) \in \mathcal{R}^k$ with $k \in [1, H]$ represents a sequence of predicates in $\mathcal{G}$ starting from e.

Concretely, these relation paths are obtained by breadth-first exploration (BFS) Korf [1985] over the graph starting from the entity e. Intuitively, this process collects potential rule bodies (cf. Section 3.1) that can serve as reasoning skeletons toward the answer.

With this tool, the agent can flexibly decide the exploration range by choosing the hop limit H. If the current range is insufficient to reach relevant evidence, the agent can iteratively expand it (e.g., from $H = 1$ to $H = 3$) to cover longer relation paths.

For example, applying the exploration action with an entity `Justin Bieber` and hop limit $H = 2$ on the KG in Figure 1 yields the following set of relation paths:

$$a^{\text{explore}}(\texttt{Justin Bieber}, 2) = \tag{2}$$

$$\{(\texttt{hasParent}), (\texttt{hasSibling}), (\texttt{hasParent}, \texttt{hasSon}), (\texttt{hasSibling}, \texttt{hasSibling}), (\texttt{hasParent}, \texttt{hasSpouse})\}. \tag{3}$$

The agent executes this action using the dedicated tool prompt provided in Appendix F.

Reasoning Path Grounding. Given an entity $e \in \mathcal{E}$, we denote by $P_e(H)$ the set of relation paths returned by the relation path exploration action with hop limit H. The reasoning path grounding action a^{ground} instantiates each relation path by fixing e as the subject of the first predicate and substituting the remaining variables with entities from the KG, as described in Section 3.1. This process produces $a^{\text{ground}}(e, P_e(H)) = (C^{ground}, E^{ground})$, where C^{ground} is the set of grounded triple sequences, which we refer to as *reasoning paths* following the terminology in Luo et al. [2024], and E^{ground} is the set of corresponding entities appearing in these paths, which serve as the *frontier* for further exploration.

With this tool, the agent can instantiate relation paths with concrete entities, turning the relation paths into reasoning paths that directly support reasoning, via the tool prompt described in Appendix G.

For example, grounding the relation paths of topic entity `Justin Bieber` in Equation (2) yields the following set of reasoning paths and frontier entities:

$$a^{\text{ground}}\big(\texttt{Justin Bieber}, P_{\texttt{Justin Bieber}}(2)\big) = (C^{ground}, E^{ground}), \tag{4}$$

$$\begin{aligned} C^{ground} = \{&(\langle \texttt{Justin Bieber}, \texttt{hasParent}, \texttt{Jeremy Bieber}\rangle), (\langle \texttt{Justin Bieber}, \texttt{hasSibling}, \texttt{Jazmyn Bieber}\rangle), \\ &(\langle \texttt{Justin Bieber}, \texttt{hasParent}, \texttt{Jeremy Bieber}\rangle, \langle \texttt{Jeremy Bieber}, \texttt{hasSon}, \texttt{Jaxon Bieber}\rangle), \\ &(\langle \texttt{Justin Bieber}, \texttt{hasSibling}, \texttt{Jazmyn Bieber}\rangle, \langle \texttt{Jazmyn Bieber}, \texttt{hasSibling}, \texttt{Jaxon Bieber}\rangle), \\ &(\langle \texttt{Justin Bieber}, \texttt{hasParent}, \texttt{Jeremy Bieber}\rangle, \langle \texttt{Jeremy Bieber}, \texttt{hasSpouse}, \texttt{Erin Wagner}\rangle)\}, \end{aligned}$$

$$E^{ground} = \{\texttt{Jeremy Bieber}, \texttt{Jaxon Bieber}, \texttt{Jazmyn Bieber}, \texttt{Erin Wagner}\}.$$

Answer Synthesis. The answer synthesis action $a^{\text{synthesis}}$ takes as input a set of relevant reasoning paths C', together with the natural language question q, and infers the final answers based on this information:

$$\hat{y} = a^{\text{synthesis}}(C', q). \tag{5}$$

The complete set of reasoning paths generated by the executions of the reasoning path grounding action is maintained in the state s and denoted by C (see Section 4.2). Within this tool, the agent selects the most relevant subset $C' \subseteq C$ to support answer inference, using the tool prompt provided in Appendix H.

4.2 State Space

Let $\mathcal{P} = \bigcup_{k=1}^{\infty} \mathcal{R}^k$ denote the set of all possible relation paths, $\mathcal{C} = \bigcup_{k=1}^{\infty} \mathcal{T}^k$ denote the set of all possible reasoning paths, where $\mathcal{T} = \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is the space of all possible triples. Let $\mathcal{E}$ be the set of all possible entities. The state space is then defined as

$$\mathcal{S} = \mathcal{P} \times \mathcal{C} \times \mathcal{E} \tag{6}$$

where each state $s = (P, C, E) \in \mathcal{S}$ represent a specific state.

4.3 State Transition

The state transition function T defines how the agent's state evolves as it interacts with the environment. Formally,

$$T : \mathcal{S} \times \mathcal{A} \to \mathcal{S}, \tag{7}$$

where $\mathcal{S} = \mathcal{P} \times \mathcal{C} \times \mathcal{E}$ is the state space and $\mathcal{A} = \{a^{\text{explore}}, a^{\text{ground}}, a^{\text{synthesis}}\}$ is the action space.

At each step, the agent is in a state s, which can be viewed as agent's memory, and selects an action $a \in \mathcal{A}$. Executing a updates the state as follows:

- if $a = a^{\text{explore}}$, new relation paths are added to P.
- if $a = a^{\text{ground}}$, new reasoning paths are added to C and new entities appearing in these paths are added to E (serving as the frontier).
- if $a = a^{\text{synthesis}}$, the agent selects a relevant subset $C' \subseteq C$ to infer the final answers and then terminates the interaction.

The state transition function T is realized by the agent through a carefully designed system prompt (shown in Appendix I). The system prompt lists the available actions $\{a^{\text{explore}}, a^{\text{ground}}, a^{\text{synthesis}}\}$ and instructs the model to update the state s until the question can be answered.

5 Experiments

5.1 Experimental Setup

Datasets. We evaluate models on the benchmark datasets introduced in Section 3, which instantiate our proposed methodology on Family and FB15k-237. For each dataset, we construct both complete and incomplete versions of the KG, where direct supporting triples are removed but sufficient alternative paths are retained to ensure answerability. All datasets are split into train/validation/test with an 8:1:1 ratio.

Evaluation Metrics. Following prior work, we report Hits@Any, precision, recall, and F1. In addition, we use the *Hard Hits Rate* (HHR), which specifically measures the proportion of questions for which the model recovers the answer entity that was deliberately removed from the incomplete KG. HHR directly reflects the models ability to perform multi-hop reasoning under incompleteness.

Baselines. We evaluate seven representative KG+LLM methods on these benchmarks. The *training-based* methods include RoG [Luo et al., 2024], G-Retriever [He et al., 2024], and GNN-RAG [Mavromatis and Karypis, 2024]. The *training-free* methods include PoG [Chen et al., 2024], StructGPT [Jiang et al., 2023], and ToG [Sun et al., 2023]. Finally, we compare all baselines against our proposed **GR-Agent**. Training-based methods use the train/validation splits, while training-free methods are evaluated zero-shot; all results are reported on the test split. For training-free models as well as our GR-Agent, we use *GPT-4o-mini* as the underlying LLM.

5.2 Overall Performance

Figure 2 reports Hits@Any and F1-scores across all methods and datasets. In most cases, **both metrics drop noticeably when moving from the complete to the incomplete KG setting**, highlighting the challenge posed by missing direct supporting triples. Precision and recall follow a similar trend (see Appendix D). Overall, training-based methods (e.g., RoG and GNN-RAG) achieve stronger performance than most non-trained approaches, while G-Retriever remains consistently low due to its reliance on noisy textual similarity retrieval rather than reasoning.

Our GR-Agent achieves the best results among training-free methods on both datasets and exhibits a smaller performance drop compared to existing non-trained baselines, demonstrating stronger robustness to incompleteness. On FB15k-237, GR-Agent further surpasses training-based methods such as RoG and GNN-RAG, achieving the best overall balance between Hits@Any and F1-score. These results confirm the effectiveness of path-centric reasoning for improving robustness under KG incompleteness.

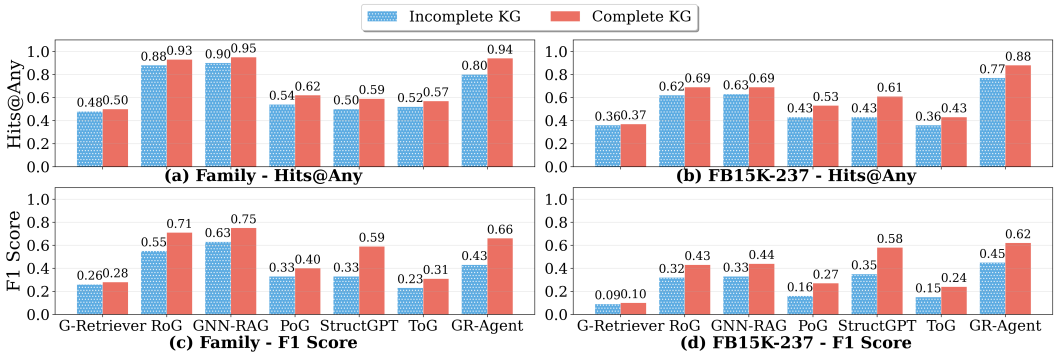
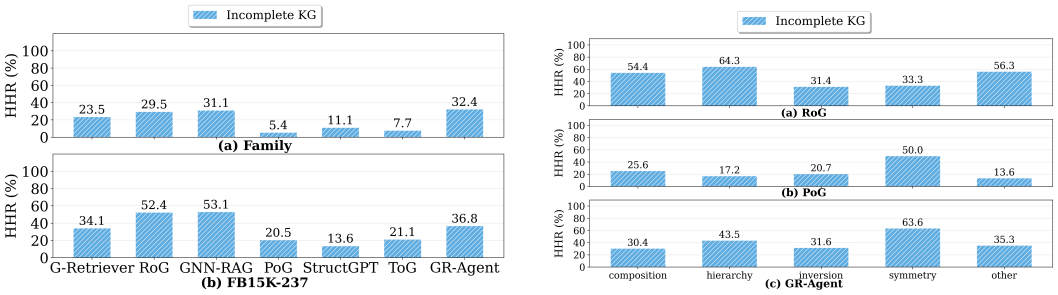


Figure 2: Performance comparison of KG+LLM models under incomplete (blue) and complete (red) KG settings, measured by **Hits@Any** (top) and **F1-Score** (bottom).



(a) HHR under different KG settings.

(b) HHR across rule types on FB15k-237.

Figure 3: Comparative analysis: (a) performance under different KG settings; (b) performance across rule types.

5.3 Impact of Removing Supporting Triples

To examine the impact of removing direct supporting triples, we report the Hard Hits Rate (HHR) under the *incomplete KG* setting(Figure 3a). Recall that HHR is defined as the fraction of correctly answered questions (Hits@Any) in which the hard answer whose direct supporting triples were deliberately removed is also recovered. It thus provides a meaningful indicator of reasoning ability when direct supporting triples are absent.

Across both datasets, the HHR remains modest, indicating that even when models successfully answer a question, they often fail to recover the missing hard answer. This underscores the difficulty of reconstructing facts through alternative reasoning paths. Training-based methods (e.g., RoG and GNN-RAG) generally achieve higher HHR than non-trained methods (e.g., PoG and ToG), suggesting that exposure to missing-evidence scenarios during training improves generalization over multi-hop reasoning paths.

In contrast, among non-trained methods, our GR-Agent achieves the highest HHR under incomplete KGs, indicating substantially stronger robustness to missing facts. While other training-free approaches degrade sharply once direct support triples are absent, GR-Agent is able to identify and leverage alternative reasoning paths. This demonstrates that **GR-Agent achieves the strongest and most generalizable reasoning ability under incomplete KGs among non-trained approaches, while remaining comparable to training-based methods**.

5.4 Fine-Grained Analysis by Rule Type

To gain deeper insights into reasoning behavior, we break down HHR by rule type on FB15k-237, comparing RoG, PoG, and our GR-Agent (Figure 3b).

Overall, RoG exhibits the highest robustness across most rule types, reflecting the benefit of training-based adaptation to missing evidence. PoG, in contrast, struggles on all categories except symmetry, where it achieves 50.0% HHR. This is consistent with the observation that symmetric relations (e.g.,

sibling) are easier for an LLM to handle, since reversing arguments does not require complex multi-hop reasoning. However, PoGs poor performance on hierarchy and "other" rules highlights its strong reliance on shallow retrieval.

Our proposed GR-Agent consistently outperforms PoG and narrows the gap with RoG. Notably, it reaches 63.6% HHR on symmetry and 43.5% on hierarchy rules, showing robustness across both simple and structurally complex patterns. While still below RoG on composition-heavy cases, GR-Agent surpasses PoG by large margins across all rule types, demonstrating that adaptive path exploration enables stronger generalization even without training.

5.5 Case Study

To better understand the limitations of current KG+LLM methods, we analyze representative failure cases from our benchmarks. Table 4 shows two illustrative examples, each highlighting a distinct failure pattern: (1) insufficient multi-hop retrieval and (2) reasoning failure despite correct retrieval.

(1) Example 1: Insufficient Multi-hop Retrieval. Answering "What is the country of the administrative division Calvados?" requires a three-hop path. Existing retrievers instead return shorter partial paths, which stall at intermediate nodes. Lacking the final hop, the generator hallucinates answers like Spain. By contrast, the GR-Agent Trace successfully explores the full relation path, grounds it, and synthesizes the correct answer.

(2) Example 2: Reasoning Failure. Here, the retriever does return the correct supporting triple `spouse(Ian Holm, Penelope Wilton)`, which should enable inferring the reverse direction. Yet the generator outputs Marriage, distracted by unrelated `typeOfUnion` paths. This shows that even with correct retrieval, models may fail to prioritize relevant evidence during generation. The GR-Agent Trace instead focuses on the relevant relation path, grounds it correctly, and outputs the right answer.

Example 1: *Question:* What is the country of the administrative division **Calvados**? *Answer:* [France]
Alternative Path: `capital(Calvados, Caen) ∧ capitalOf(Caen, Calvados) ∧ contains(France, Calvados) ⇒ country(Calvados, France)`
—
Prediction: [Spain]
Retrieved Paths: `administrativeParent(Calvados, LowerNormandy) ∧ contains(Calvados, Caen)`
—
GR-Agent Trace:
(1) Relation Path Exploration: `capital → capitalOf, contains`
(2) Reasoning Path Grounding: `capital(Calvados, Caen) ∧ capitalOf(Caen, Calvados); contains(France, Calvados)` *(3) Answer Synthesis:* [France]

Example 2: *Question:* Who is **Ian Holm**'s spouse? *Answer:* [Penelope Wilton]
Alternative Paths: `spouse(Ian Holm, Penelope Wilton) ⇒ spouse(Penelope Wilton, Ian Holm)`
—
Prediction: [Marriage]
Retrieved Paths: `spouse(Ian Holm, Penelope Wilton)`
`awardNominee(Ian Holm, Cate Blanchett) ∧ typeOfUnion(Cate Blanchett, Marriage)`
`awardNominee(Ian Holm, Kate Beckinsale) ∧ typeOfUnion(Kate Beckinsale, Domestic Partnership)`
—
GR-Agent Trace:
(1) Relation Path Exploration: `spouse` *(2) Reasoning Path Grounding:* `spouse(Ian Holm, Penelope Wilton)` *(3) Answer Synthesis:* [Penelope Wilton]

Table 4: Case studies comparing baseline failures with GR-Agent. Unlike KG+LLM models that retrieve incomplete multi-hop paths or follow noisy evidence, GR-Agent performs multiple actions to derive the correct answer. (green = expected paths, red = incorrect predictions).

6 Discussion and Conclusion

In this work, we studied the challenge of reasoning under incomplete KGs, where direct supporting triples are missing and answers must be inferred through alternative paths. To address this, we proposed **GR-Agent**, a training-free adaptive graph reasoning agent that explores relation paths, grounds them into reasoning paths, and synthesizes answers with the help of a lightweight planner and evolving memory. Alongside the method, we introduce a general construction methodology for simulating incomplete KGs while ensuring answerability: direct supporting triples are removed, but sufficient alternative paths are preserved. We instantiate this paradigm on Family and FB15k-237, providing two benchmarks for systematic assessment of reasoning ability under incompleteness.

Experiments show that GR-Agent consistently outperforms other training-free methods and achieves performance comparable to training-based systems. These results suggest that path-centric exploration is an effective way to recover missing knowledge in real-world KGs. This work is ongoing. In the future, we plan to extend GR-Agent with reinforcement learning to optimize tool usage and further improve reasoning efficiency.

7 Acknowledgements

The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Yuqicheng Zhu and Hongkuan Zhou. The work was partially supported by EU Projects Graph Massivizer (GA 101093202), enRichMyData (GA 101070284), SMARTY (GA 101140087), and the EPSRC project OntoEm (EP/Y017706/1).

References

Jinheon Baek, Alham Fikri Aji, Jens Lehmann, and Sung Ju Hwang. Direct fact retrieval from knowledge graphs without entity linking. *arXiv preprint arXiv:2305.12416*, 2023.

Russa Biswas, Lucie-Aimée Kaffee, Michael Cochez, Stefania Dumbrava, Theis E Jendal, Matteo Lissandrini, Vanessa Lopez, Eneldo Loza Mencía, Heiko Paulheim, Harald Sack, et al. Knowledge graph embeddings: open challenges and opportunities. *Transactions on Graph Data and Knowledge*, 1(1):4–1, 2023.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pages 2206–2240. PMLR, 2022.

Lihu Chen, Simon Razniewski, and Gerhard Weikum. Knowledge base completion for long-tail entities. In *Proceedings of the First Workshop on Matching From Unstructured and Structured Data (MATCHING 2023)*, pages 99–108, 2023.

Liyi Chen, Panrong Tong, Zhongming Jin, Ying Sun, Jieping Ye, and Hui Xiong. Plan-on-graph: Self-correcting adaptive planning of large language model on knowledge graphs. In *Proceedings of the 38th Conference on Neural Information Processing Systems*, 2024.

Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M Suchanek. Fast rule mining in ontological knowledge bases with amie+. *The VLDB Journal*, 24(6):707–730, 2015.

Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. Beyond iid: three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference 2021*, pages 3477–3488. ACM, 2021.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *Advances in Neural Information Processing Systems*, 37:132876–132907, 2024.

Alfred Horn. On sentences which are true of direct unions of algebras1. *The Journal of Symbolic Logic*, 16(1):14–21, 1951.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering. In *EACL*, pages 874–880. Association for Computational Linguistics, 2021.

Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen. Structgpt: A general framework for large language model to reason over structured data. *arXiv preprint arXiv:2305.09645*, 2023.

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

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. In *ICLR*. OpenReview.net, 2020.

Richard E Korf. Depth-first iterative-deepening: An optimal admissible tree search. *Artificial intelligence*, 27(1):97–109, 1985.

Jonathan Lajus, Luis Galárraga, and Fabian Suchanek. Fast and exact rule mining with amie 3. In *The Semantic Web: 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31–June 4, 2020, Proceedings 17*, pages 36–52. Springer, 2020.

Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. A survey on complex knowledge base question answering: Methods, challenges and solutions. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 4483–4491. International Joint Conferences on Artificial Intelligence Organization, 2021.

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

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *ICLR*. OpenReview.net, 2024.

Costas Mavromatis and George Karypis. Gnn-rag: Graph neural retrieval for large language model reasoning. *arXiv preprint arXiv:2405.20139*, 2024.

Aisha Mohamed, Shameem Parambath, Zoi Kaoudi, and Ashraf Aboulnaga. Popularity agnostic evaluation of knowledge graph embeddings. In *Conference on Uncertainty in Artificial Intelligence*, pages 1059–1068. PMLR, 2020.

OpenAI. Chatgpt(3.5)[large language model]. `https://chat.openai.com`, 2024.

Nico Potyka, Yuqicheng Zhu, Yunjie He, Evgeny Kharlamov, and Steffen Staab. Robust knowledge extraction from large language models using social choice theory. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, pages 1593–1601, 2024.

Yunqi Qiu, Yuanzhuo Wang, Xiaolong Jin, and Kun Zhang. Stepwise reasoning for multi-relation question answering over knowledge graph with weak supervision. In *Proceedings of the 13th international conference on web search and data mining*, pages 474–482, 2020.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. In-context retrieval-augmented language models. *Trans. Assoc. Comput. Linguistics*, 11:1316–1331, 2023.

Ali Sadeghian, Mohammadreza Armandpour, Patrick Ding, and Daisy Zhe Wang. Drum: End-to-end differentiable rule mining on knowledge graphs. *Advances in neural information processing systems*, 32, 2019.

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. *arXiv preprint arXiv:2307.07697*, 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 *ICLR*. OpenReview.net, 2024.

Richard S Sutton, Andrew G Barto, et al. Reinforcement learning. *Journal of Cognitive Neuroscience*, 11(1):126–134, 1999.

Alon Talmor and Jonathan Berant. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*, 2018.

Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd workshop on continuous vector space models and their compositionality*, pages 57–66, 2015.

Yao Xu, Shizhu He, Jiabei Chen, Zihao Wang, Yangqiu Song, Hanghang Tong, Guang Liu, Kang Liu, and Jun Zhao. Generate-on-graph: Treat llm as both agent and kg in incomplete knowledge graph question answering. *arXiv preprint arXiv:2404.14741*, 2024.

Liang Yao, Chengsheng Mao, and Yuan Luo. Kg-bert: Bert for knowledge graph completion. *arXiv preprint arXiv:1909.03193*, 2019.

Xi Ye, Semih Yavuz, Kazuma Hashimoto, Yingbo Zhou, and Caiming Xiong. Rng-kbqa: Generation augmented iterative ranking for knowledge base question answering. *arXiv preprint arXiv:2109.08678*, 2021.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. The value of semantic parse labeling for knowledge base question answering. In *ACL (2)*. The Association for Computer Linguistics, 2016.

Dongzhuoran Zhou, Yuqicheng Zhu, Yuan He, Jiaoyan Chen, Evgeny Kharlamov, and Steffen Staab. Evaluating knowledge graph based retrieval augmented generation methods under knowledge incompleteness. *arXiv preprint arXiv:2504.05163*, 2025.

Yuqicheng Zhu, Nico Potyka, Mojtaba Nayyeri, Bo Xiong, Yunjie He, Evgeny Kharlamov, and Steffen Staab. Predictive multiplicity of knowledge graph embeddings in link prediction. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 334–354, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

Yuqicheng Zhu, Daniel Hernández, Yuan He, Zifeng Ding, Bo Xiong, Evgeny Kharlamov, and Steffen Staab. Predicate-conditional conformalized answer sets for knowledge graph embeddings. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 4145–4167, Vienna, Austria, July 2025a. Association for Computational Linguistics.

Yuqicheng Zhu, Nico Potyka, Jiarong Pan, Bo Xiong, Yunjie He, Evgeny Kharlamov, and Steffen Staab. Conformalized answer set prediction for knowledge graph embedding. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 731–750, 2025b.

A Details of Rule Mining

AMIE3 is a widely used rule mining system designed to operate efficiently over large-scale KGs. A logical rule discovered by AMIE3 has the following form (*Horn rules* Horn [1951]):

$$B_1 \wedge B_2 \wedge \cdots \wedge B_n \Rightarrow H\,,$$

where each item is called an *atom*, a binary relation of the form $r(X, Y)$, in which r is a predicate and X, Y are variables. The left-hand side of the rule is a conjunction of *body atoms*, denoted as $\mathbf{B} = B_1 \wedge \cdots \wedge B_n$, and the right-hand side is the *head atom* H. Intuitively, a rule expresses that if the body $\mathbf{B}$ holds, then the head H is likely to hold as well.

A *substitution* σ maps every variable occurring in an atom to a entity that exists in $\mathcal{G}$. For example applying $\sigma = \{X \mapsto \texttt{Justin}, Y \mapsto \texttt{Jaxon}\}$ to the atom `hasSibling(X,Y)` yields the grounded fact `hasSibling(Justin,Jaxon)`. A *grounding* of a rule $\mathbf{B} \Rightarrow H$ is

$$\sigma(B_1) \wedge \ldots \sigma(B_n) \Rightarrow \sigma(H)\,.$$

Quality Measure. AMIE3 uses the following metrics to measure the quality of a rule:

- **Support.** The *support* of a rule is defined as the number its groundings for which all grounded facts are observed in the KG:

$$support(\mathbf{B} \Rightarrow H) = \\ |\{\sigma(H) \mid \forall i, \sigma(B_i) \in \mathcal{G} \wedge \sigma(H) \in \mathcal{G}\}|\,.$$

- **Head coverage.** *Head coverage (hc)* measures the proportion of observed head groundings in the KG that are successfully explained by the rule. It is defined as the ratio of the rules support to the number of head groundings in the KG:

$$hc(\mathbf{B} \Rightarrow H) = \frac{support(\mathbf{B} \Rightarrow H)}{|\{\sigma \mid \sigma(H) \in \mathcal{G}\}|}\,.$$

- **Confidence.** *Confidence* measures the proportion of body groundings that also lead to the head being observed in the KG. It is defined as the ratio of the rule's support to the number body groundings in the KG:

$$confidence(\mathbf{B} \Rightarrow H) = \frac{support(\mathbf{B} \Rightarrow H)}{|\{\sigma \mid \sigma(\mathbf{B}) \in \mathcal{G}\}|}\,.$$

We retain only rules with high confidence and sufficient support, filtering out noisy or spurious patterns. Specifically, we run AMIE3 with a confidence threshold of 0.3, a head coverage threshold of 0.1, and a maximum rule length of 4. AMIE3 incrementally generates candidate rules via breadth-first refinement Lajus et al. [2020] and evaluates them using confidence and head coverage; only those meeting the specified thresholds are retained. Additional details on the rule generation and filtering process are provided in Appendix A.

A.1 AMIE3 Candidate Rule Refinement

Refinement is carried out using a set of operators that generate new candidate rules:

- Dangling atoms, which introduce a new variable connected to an existing one;
- Closing atoms, which connect two existing variables;
- Instantiated atoms, which introduce a constant and connect it to an existing variable.

AMIE3 generate candidate rules by a refinement process using a classical breadth-first search Lajus et al. [2020]. It begins with rules that contain only a head atom (e.g. $\Rightarrow \texttt{hasSibling}(X, Y)$) and refines them by adding atoms to the body. For example, it may generate the refined rule:

$$\texttt{hasParent}(X, Z) \wedge \texttt{hasChild}(Z, Y) \\ \Rightarrow \texttt{hasSibling}(X, Y)\,.$$

This refinement step connects existing variables and introduces new ones, gradually building meaningful patterns.

A.2 AMIE3 Hyperparameter Settings

We use AMIE3 with a confidence threshold of 0.3 and a PCA confidence threshold θ_{PCA} of 0.4 for both datasets. The maximum rule length is set to 3 for **Family** to avoid overly complex patterns, and 4 for **FB15k-237** to allow richer rules. See Appendix B.1 for the definition of PCA confidence.

B Properties of Horn Rules Mined by AMIE3

AMIE3 mines logical rules from knowledge graphs in the form of (Horn) rules:

$$B_1 \wedge B_2 \wedge \cdots \wedge B_n \implies H$$

where B_i and H are atoms of the form $r(X, Y)$. To ensure interpretability and practical utility, AMIE3 imposes the following structural properties on all mined rules:

- **Connectedness:** All atoms in the rule are transitively connected via shared variables or entities. This prevents rules with independent, unrelated facts (e.g., $\texttt{diedIn}(x, y) \implies \texttt{wasBornIn}(w, z)$). Two atoms are connected if they share a variable or entity; a rule is connected if every atom is connected transitively to every other atom.
- **Closedness:** Every variable in the rule appears at least twice (i.e., in at least two atoms). This avoids rules that merely predict the existence of some fact without specifying how it relates to the body, such as $\texttt{diedIn}(x, y) \implies \exists z : \texttt{wasBornIn}(x, z)$.
- **Safety:** All variables in the head atom also appear in at least one body atom. This ensures that the rule's predictions are grounded by the body atoms and avoids uninstantiated variables in the conclusion.

These restrictions are widely adopted in KG rule mining [Galárraga et al., 2015, Lajus et al., 2020] to guarantee that discovered rules are logically well-formed and meaningful for downstream reasoning tasks.

B.1 PCA Confidence

To understand the concept of rule mining better for the reader we simplified notation of confidence in main body. Note AMIE3 also supports a more optimistic confidence metric known as *PCA confidence*, which adjusts standard confidence to account for incompleteness in the KG.

Motivation. Standard confidence for a rule is defined as the proportion of its correct predictions among all possible predictions suggested by the rule. However, this metric is known to be pessimistic for knowledge graphs, which are typically incomplete: many missing triples may be true but unobserved, unfairly penalizing a rule's apparent reliability.

Definition. To address this, AMIE3 introduces *PCA confidence* (Partial Completeness Assumption confidence) [Galárraga et al., 2015], an optimistic variant that partially compensates for KG incompleteness. Given a rule of the form

$$B_1 \wedge \cdots \wedge B_n \implies r(x, y)$$

the **standard confidence** is

$$\text{conf}(R) = \frac{|\{(x, y) : B_1 \wedge \cdots \wedge B_n \wedge r(x, y)\}|}{|\{(x, y) : B_1 \wedge \cdots \wedge B_n\}|}$$

where the denominator counts all predictions the rule could possibly make, and the numerator counts those that are actually present in the KG.

PCA confidence modifies the denominator to include only those (x, y) pairs for which at least one $r(x, y')$ triple is known for the subject x. That is, the rule is only penalized for predictions about entities for which we have observed at least some information about the target relation. Formally,

$$\text{conf}_{\text{PCA}}(R) = \frac{|\{(x, y) : B_1 \wedge \cdots \wedge B_n \wedge r(x, y)\}|}{|\{(x, y) : B_1 \wedge \cdots \wedge B_n \wedge \exists y' : r(x, y')\}|}$$

Here, the denominator sums only over those x for which some y' exists such that $r(x, y')$ is observed in the KG.

Intuition. This approach assumes that, for any entity x for which at least one fact $r(x, y')$ is known, the KG is "locally complete" with respect to r for x, so if the rule predicts other $r(x, y)$ facts for x, and they are missing, we treat them as truly missing (i.e., as counterexamples to the rule). But for entities where no $r(x, y)$ fact is observed at all, the rule is not penalized for predicting additional facts.

Comparison. PCA confidence thus provides a more optimistic and fairer assessment of a rule's precision in the presence of incomplete data. It is widely adopted in KG rule mining, and is the default metric for filtering and ranking rules in AMIE3.

For further details, see [Galárraga et al., 2015].

B.2 Rule Mining Procedure

Algorithm 1 AMIE3

Require: Knowledge graph $\mathcal{G}$, maximum rule length l, PCA confidence threshold θ_{PCA}, and head coverage threshold θ_{hc}.
Ensure: Set of mined rules $\mathcal{R}$.

```
1:  q ← all rules of the form ⊤ ⇒ r(X, Y)
2:  R ← ∅
3:  while q is not empty do
4:      R ← q.dequeue()
5:      if SatisfiesRuleCriteria(R) then
6:          R ← R ∪ {R}
7:      if len(R) < l and θ_PCA(R) < 1.0 then
8:          for all R_c ∈ refine(R) do
9:              if hc(R_c) ≥ θ_hc and R_c ∉ q then
10:                 q.enqueue(R_c)
11: return R
```

AMIE3 generate candidate rules by a refinement process using a classical breadth-first search Lajus et al. [2020]. Algorithm 1 summarizes the rule mining process of AMIE3. The algorithm starts with an initial set of rules that contain only a head atom (i.e. $\top \Rightarrow r(X, Y)$, where $\top$ denotes an empty body) and maintains a queue of rule candidates (Line 1). At each step, AMIE3 dequeues a rule R from the queue and evaluates whether it satisfies three criteria (Line 5):

- the rule is *closed* (i.e., all variables in at least two atoms),
- its PCA confidence is higher than θ_{PCA},
- its PCA confidence is higher than the confidence of all previously mined rules with the same head atom as R and a subset of its body atoms.

If these conditions are met, the rule is added to the final output set $\mathcal{R}$.

If R has fewer than l atoms and its confidence can still be improved (Line 8), AMIE3 applies a `refine` operator (Line 9) that generates new candidate rules by adding a body atom (details in Appendix A). Refined rules are added to the queue only if they have sufficient head coverage (Line 11) and have not already been explored. This process continues until the queue is empty, at which point all high-quality rules satisfying the specified constraints have been discovered.

C Details of Dataset Generation

KGs typically exhibit a "long-tail" distribution, where a small number of entities participate in a disproportionately large number of triples, while the majority appear only infrequently [Mohamed et al., 2020, Chen et al., 2023]. This imbalance can cause many generated questions to share the same answer entity, leading to biased evaluation.

To reduce answer distribution bias, we apply frequency-based downsampling to the generated questions $\mathcal{Q}$, yielding a more balanced subset $\mathcal{Q}' \subseteq \mathcal{Q}$. The procedure is summarized in Algorithm 2, which is provided in Appendix C. Specifically, for each answer entity a, we retain at most $\tau \cdot |\mathcal{Q}|$ questions if a exceeds the frequency threshold τ; otherwise, all associated questions are kept.

Algorithm 2 Downsampling Procedure

Require: Question set $\mathcal{Q}$; threshold $\tau \in (0, 1]$
Ensure: Balanced subset $\mathcal{Q}' \subseteq \mathcal{Q}$

```
Let A ← set of unique answer entities in Q
Q' ← ∅
for all a ∈ A do
    Q_a ← {q ∈ Q | answer(q) = a}
    if |Q_a| > τ · |Q| then
        Randomly sample S_a ⊂ Q_a
        of size ⌊τ · |Q|⌋
    else
        S_a ← Q_a
    Q' ← Q' ∪ S_a
return Q'
```

C.1 Benchmark Construction Code and Data

We release the source code for benchmark construction, along with the Family and FB15k-237 benchmark datasets, at `https://anonymous.4open.science/r/INCK-EA16`.

D Additional Results of the experiment

Figure 4 presents the recall and precision of all evaluated KG-RAG models on the constructed benchmarks.

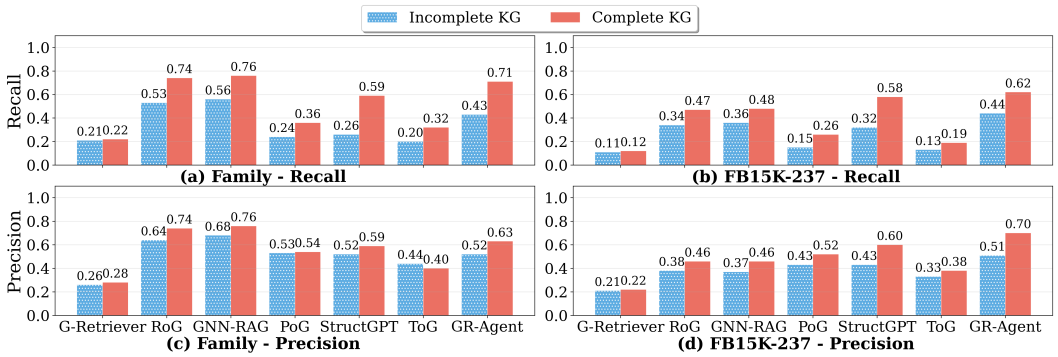


Figure 4: Performance comparison of KG-RAG models under incomplete (blue) and complete (red) KG settings, measured by **Recall** (top) and **Precision** (bottom).

E Prompt Template

Prompt for generating questions from triples:

```
You are an expert in knowledge graph question generation.

Given:
Removed Triple: ({entity_h}, {predicate_T}, {entity_t})
Question Entity: {topic_entity}
Answer Entity: {answer_entity}
```

```
Write a clear, natural-language question that asks for the Answer Entity, using the
    given predicate and Topic Entity.

Requirements:
- Express the predicate {predicate_T} naturally (paraphrasing allowed, but preserve core
     meaning; e.g., "wife_of" -> "wife").
- Mention the Topic Entity {topic_entity}.
- The answer should be the Answer Entity {answer_entity}.
- Do not mention the Answer Entity {answer_entity} in the question.
- Do not ask a yes/no question.
- Output only the question as plain text.

Example:
Removed Triple: ("Alice", "wife_of", "Carol")
Question Entity: Carol
Answer Entity: Alice

Output:
Who is Carol's wife?

Now, generate the question for:
Removed Triple: ({entity_h}, {predicate_T}, {entity_t})
Question Entity: {topic_entity}
Answer Entity: {answer_entity}
```

To ensure reproducibility and mitigate randomness in LLM outputs Potyka et al. [2024], we set the generation temperature to 0 in all experiments.

F Tool Prompt for Relation Path Exploration

```
Mine relation paths from an entity to discover reasoning patterns.


This tool returns ALL paths from 1-hop up to max_hops combined in one list.

CRITICAL USAGE:
- Select the starting entity strategically.
- Use small hop limits for efficient exploration, and increase gradually
   if evidence is insufficient.
- Collected relation paths will serve as potential reasoning skeletons
  for grounding and synthesis.

Args:
    entity: Starting entity for exploration
    max_hops: Maximum path length


Returns:
    Combined relation paths represented as strings, e.g.,
    ['rel1', 'rel1 -> rel4', 'rel2 -> rel1']
```

G Tool Prompt for Reasoning Path Grounding

```
Ground relation paths to find concrete entity sequences that answer the question.


This tool finds actual entity sequences that follow the selected patterns, providing
concrete evidence for reasoning.
```

```
Args:
    entity: Starting entity for grounding
    relation_paths: Selected relation path strings from relation_path_match


Returns:
    Grounded path descriptions with entity sequences and evidence triples
```

H Tool Prompt for Answer Synthesis

```
Complete the knowledge graph exploration when reasoning paths are sufficient
to answer the question.

The agent should return the final answer based on reasoning over the discovered
reasoning paths to terminate the exploration.

CRITICAL QUESTION UNDERSTANDING:
- Carefully analyze what the question is asking for.
- Extract answer entities that directly answer the question, not related but
  irrelevant entities
- Select only reasoning paths that lead to the correct entity type being asked for

The explored_reasoning_paths are formatted as strings containing the grounded
path evidence:
     Evidence: <supporting_reasoning_paths>


The agent should focus on answering the original question using reasoning over
these paths. Use the reasoning paths to infer the correct answer entities
through pattern matching and evidence analysis.



Reasoning strategies:
- Direct matches: triples that directly answer the query
- Fuzzy matches: similar relation/entity names that approximately match the target
- Inverse relationships: if you find "A relation B", consider "B inverse_relation A"
- Chain reasoning: use patterns like "A rel1 B" + "B rel2 C" to infer "A answers C"
- Evidence stacking: multiple consistent triples together provide sufficient evidence

CRITICAL: Among the explored reasoning paths, only a subset can actually answer
the question. The agent should carefully select the most reasonable and reliable
subset as supporting reasoning paths. Note that entities appearing in reasoning
paths as intermediate steps may not be answer entities. Use them to infer the
answer while excluding false evidence.

Args:
    explored_reasoning_paths (list[str]): Grounded reasoning paths that support the
     final answer
    answer_entities (list[str]): Final answer entity IDs only

Returns:
    dict[str, Any]: Final results with answers and reasoning path evidence
```

I System Prompt

Default system prompt for knowledge graph reasoning:

```
You are a helpful assistant that answers queries by exploring a knowledge
graph using advanced path-based reasoning.
```

```
Available tools:
- relation_path_mining: Discover all possible relation paths around an entity to find
    reasoning patterns.
- path_grounding: Instantiate selected relation paths with concrete entities
  to find actual reasoning chains.
- complete_task: Finalize the answer using reasoning paths as evidence.

The toolkit maintains an evidence store of discovered entities and triples.
Use the tools iterativelydiscover, select, extractthen finalize when ready
with concrete entity answers.

Important context:
- Real-world knowledge graphs are always incomplete. Do NOT expect to always
  find a direct triple that answers the question.
- Instead, you must rely on indirect evidence, combining multiple facts and
  relation paths. If no direct edge exists, reason over intermediate nodes and
  multi-hop chains to imply the answer.
- Avoid finalizing prematurely if only partial evidence is present; keep
  exploring relation paths to assemble a reasoning chain.

Key principles:
- Focus on RELATION PATHS as reasoning patterns, not individual triples
- Multi-hop reasoning is essential  explore 1-hop, 2-hop, and 3-hop patterns
- Select paths strategically based on semantic relevance to the question
- Ground selected paths to get concrete evidence chains
- A reasoning path can connect topic entity and answer entity through
  intermediate entities
- Look for both direct relations and inverse relations
- Use path grounding results as structured evidence for your final answer
```

J Detailed Evaluation Settings

All evaluated models are required to produce their predictions as a *list of answers*, but in practice, the model output is often a raw string P_{str} (e.g., `"Paris, London"` or `"Paris London"`). To obtain a set-valued prediction suitable for evaluation, we first apply a splitting function $\text{split}(P_{\text{str}})$, which splits the raw string into a list of answer strings $P = [p_1, p_2, \ldots, p_n]$ using delimiters such as commas, spaces, or newlines as appropriate.

We then define a normalization function $\text{norm}(\cdot)$, which converts each answer string to lowercase, removes articles (`a`, `an`, `the`), punctuation, and extra whitespace, and eliminates the special token `<pad>` if present. The final prediction set is then defined as $\mathcal{P} = \{\text{norm}(p) \mid p \in P\}$, i.e., the set of unique normalized predictions. The same normalization is applied to each gold answer in the list A to obtain the set $\mathcal{A}$.

All evaluation metrics are computed based on the resulting sets of normalized predictions $\mathcal{P}$ and gold answers $\mathcal{A}$.

Algorithm 3 Output Processing

Require: Model output string P_{str}, gold answer list A
Ensure: Normalized prediction set $\mathcal{P}$, normalized gold set $\mathcal{A}$
1: $P \leftarrow \text{split}(P_{\text{str}})$
2: $\mathcal{P} \leftarrow \{\, \text{norm}(p) \mid p \in P \,\}$
3: $\mathcal{A} \leftarrow \{\, \text{norm}(a) \mid a \in A \,\}$
4: **return** $\mathcal{P}, \mathcal{A}$

K Baseline Details

Unless otherwise specified, for all methods we use the LLM backbone and hyperparameters as described in the original papers.

RoG, G-Retriever, and GNN-RAG are each trained and evaluated separately on the 8:1:1 training split of each dataset (Family and FB15k-237) using a single NVIDIA H200 GPU, as described in Section 3.3. For RoG, we use LLaMA2-Chat-7B as the LLM backbone, instruction-finetuned on the training split of Family or FB15K-237 for 3 epochs. The batch size is set to 4, the learning rate to 2×10^{-5}, and a cosine learning rate scheduler with a warmup ratio of 0.03 is adopted [Luo et al., 2024]. For G-Retriever, the GNN backbone is a Graph Transformer (4 layers, 4 attention heads per layer, hidden size 1024) with LLaMA2-7B as the LLM. Retrieval hyperparameters and optimization follow He et al. [2024]. For GNN-RAG [Mavromatis and Karypis, 2024], we use the recommended ReaRev backbone and sBERT encoder; the GNN component is trained for 200 epochs with 80 epochs of warmup and a patience of 5 for early stopping. All random seeds are fixed for reproducibility. For PoG [Chen et al., 2024], StructGPT [Jiang et al., 2023], and ToG [Sun et al., 2024], we use GPT-4o-mini as the underlying LLM, and the original prompt and generation settings from each method. For our proposed GR-Agent, we also adopt GPT-4o-mini as the backbone LLM to ensure fairness of comparison.

L Detailed Analysis of Other Rule Types

The *Other* category in Table 1 encompasses a broad range of logical rules that do not fall into standard symmetry, inversion, hierarchy, or composition classes. Below we summarize the main patterns observed, provide representative examples, and discuss their impact on model performance.

Longer Compositional Chains. Rules involving three,

$$r_1(x, y) \wedge r_2(y, z) \wedge r_3(z, w) \Rightarrow r_4(x, w)$$

Triangle Patterns. Rules connecting three entities in a triangle motif,

$$r_1(x, y) \wedge r_2(x, z) \Rightarrow r_3(y, z)$$

Intersection Rules. Rules where multiple body atoms share the same argument,

$$r_1(x, y) \wedge r_2(x, y) \Rightarrow r_3(x, y)$$

Other Patterns. Some rules do not exhibit simple interpretable motifs, involving unusual variable binding or rare predicate combinations. Like recursive rules (check AMIE3 Lajus et al. [2020] for more details)

M Personal Identification Issue in FB15k-237

While FB15k-237 contains information about individuals, it typically focuses on well-known public figures such as celebrities, politicians, and historical figures. Since this information is already widely available online and in various public sources, its inclusion in Freebase doesn't significantly compromise individual privacy compared to datasets containing sensitive personal information.

N AI Assistants In Writing

We use ChatGPT OpenAI [2024] to enhance our writing skills, abstaining from its use in research and coding endeavors.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes] .

   Justification: The abstract and introduction clearly state the benchmark-construction methodology under KG incompleteness, the evaluation of existing methods, and the introduction of the GR-Agent framework.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes] .

   Justification: The paper acknowledges limitations in the Conclusion, including evaluation scope (two KGs), robustness under incompleteness, and future extensions such as RL-based optimization.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper is empirical and methodological; it formalizes the agent decision process but does not introduce theorems requiring formal proofs.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We describe dataset construction, splits, metrics, baseline configurations, and prompt settings, and provide an anonymized repository to enable reproducibility.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We release anonymized code and processed datasets with detailed instructions for reproduction in the supplementary material. This includes data preparation, hyperparameters, and commands to reproduce all reported results.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so No is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies dataset splits, hyperparameters, optimizer, learning rate schedules, and early stopping criteria.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report results over multiple random seeds and include standard deviations as error bars in tables and figures, capturing variability due to initialization and train/validation splits.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the hardware (NVIDIA H200 GPUs, 80GB memory) and total compute time used for each experiment, along with estimated total compute across all runs, as detailed in Appendix **??**.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms to the NeurIPS Code of Ethics. All datasets are public benchmarks, no private or sensitive data is used, and anonymity is preserved for the review process.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: We discuss both potential benefits (e.g., more robust KG-based reasoning systems for information retrieval and question answering).

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The work does not involve releasing high-risk models or datasets such as pre-trained LLMs or large-scale scraped corpora. All datasets are standard public benchmarks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All datasets (FB15k-237, Family) and baselines are properly cited.

Guidelines:

- The answer NA means that the paper does not use existing assets.

- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We introduce new benchmark variants. Documentation, data splits, and instructions for usage are provided with the released assets. Data is anonymized and follows licensing requirements of the original sources.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing or human-subject studies.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

Justification: The research does not involve human subjects and therefore does not require IRB approval.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: LLMs are used as part of the agent design for executing tool-based reasoning actions (exploration, grounding, synthesis). Their role in the methodology is fully described.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.