Explore-over-Graph: Hybrid Embedding–LLM Reasoning for Knowledge Graph Question Answering under Incompleteness

Ola El Khatib **Djellel Difallah**
New York University Abu Dhabi, UAE
{oge208,djellel}@nyu.edu

Abstract

Large language models (LLMs) are increasingly combined with knowledge graphs (KGs) to ground reasoning in structured evidence. However, most LLM-based KGQA methods rely on traversing existing graph edges and become unreliable when reasoning paths are broken by missing facts. Alternatives that ask LLMs to generate missing knowledge risk introducing hallucinated evidence. We introduce XoG (eXplore-over-Graph), a framework for multi-hop question answering over incomplete KGs that recovers missing reasoning paths from learned graph structure rather than LLM parametric knowledge. XoG combines type-level entity–relation statistics to identify candidate relations with KG embeddings to retrieve plausible missing entities, using the LLM as a semantic selector and reasoner. These mechanisms are integrated into an iterative planning–exploration–reasoning process. Experiments on WebQSP, CWQ, and the Wikidata-based BRINK benchmark show that XoG remains competitive on complete KGs and consistently outperforms comparable methods without task-specific KGQA training under KG incompleteness. These gains persist across multiple LLM backbones, indicating that stronger LLMs alone do not resolve missing graph evidence. XoG also reduces LLM token consumption by up to 33% compared with a closely related planning-based approach.

1 Introduction

LLMs have demonstrated remarkable success across a wide range of natural language processing tasks, including question answering and complex reasoning (Wei et al., 2022; Yao et al., 2023b), but remain prone to hallucinations and out-of-date knowledge due to static parametric memory, which poses fundamental limitations for knowledge-intensive and multi-hop reasoning tasks

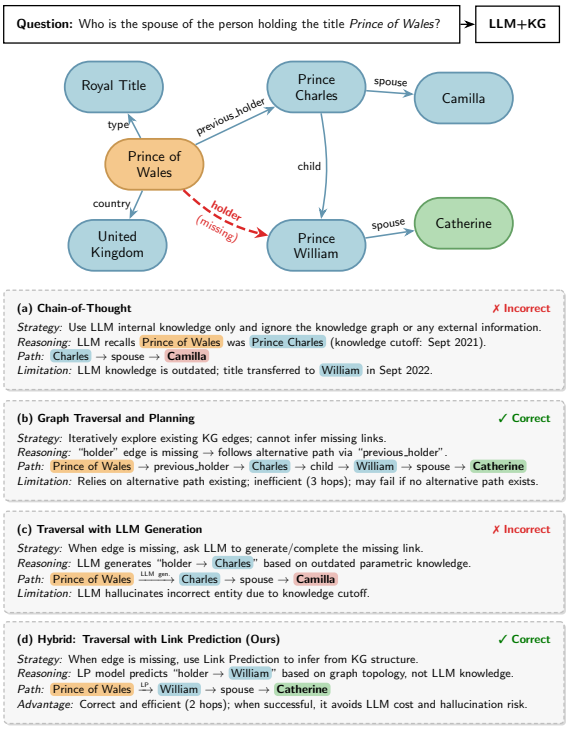


Figure 1: Comparison of LLM+KG reasoning paradigms.

(Huang et al., 2025). To mitigate these limitations, recent research integrates LLMs with KGs (Ji et al., 2022), combining the structured and verifiable knowledge provided by KGs with the language understanding and reasoning capabilities of LLMs. However, real-world KGs are inherently incomplete due to their scale, dynamic nature, and long-tail relation distributions, with missing entities and relations that hinder multi-hop reasoning. We refer to this setting as KG incompleteness, where one or more triples required to form a correct multi-hop reasoning path are missing from the graph. Such incompleteness poses a fundamental challenge for knowledge graph question answering (KGQA).

Existing work on LLM-based KGQA uses LLMs to guide interactive graph exploration (Sun et al.,

Findings of the Association for Computational Linguistics: IJCNLP-AACL 2026, pages 1962–1975
November 6-10, 2026 ©2026 Association for Computational Linguistics

2024b), but their performance can degrade substantially when required facts are missing. Generation-based methods address this limitation by generating missing triples during exploration, but may introduce incorrect information (Xu et al., 2024). Fig. 1 illustrates these limitations for question answering over an incomplete KG: pure chain-of-thought (CoT) reasoning relies on the LLM's internal knowledge (Fig. 1(a)); planning-based approaches may compensate for broken paths by traversing longer routes at the cost of increased exploration depth (Fig. 1(b)); and generation-based methods generate missing triples directly but may introduce outdated or incorrect facts (Fig. 1(c)). These limitations reveal a trade-off between *efficiency* and *reliability* under KG incompleteness (Fig. 1(d)).

Motivated by these challenges, we introduce XoG, a multi-hop KGQA framework that addresses KG incompleteness through KG-grounded recovery. XoG reconnects broken reasoning paths while constraining the LLM to select from curated candidate sets. Specifically, XoG uses (a) co-occurrence statistics between entity types and relations to prune the large relation search space, and (b) embedding-based link prediction to retrieve likely target entities, with language models acting as a semantic selector over noisy candidates. These are integrated into an iterative planning–exploration–reasoning strategy that combines query decomposition, graph exploration, and final answer reasoning. Experiments on benchmark KGQA datasets show that XoG improves answer accuracy and token efficiency under KG incompleteness while remaining competitive in complete-KG settings.

In summary, our main contributions are:

- ⋆ We introduce XoG, a multi-hop KGQA framework that addresses KG incompleteness through two complementary KG-grounded recovery mechanisms: type-level entity–relation co-occurrence statistics to identify likely relations given context, and embedding-based link prediction to retrieve the tail entities that ground LLM reasoning.
- ⋆ To control the candidate-expansion noise introduced by recovery, we propose a hierarchical bucketing strategy that scales LLM relation pruning to KGs with thousands of relations and use DistilBERT-based semantic filtering for entity candidates, all within an iterative *planning–exploration–reasoning* loop rather than as one-shot graph completion.
- ⋆ We conduct extensive experiments on WebQSP and CWQ under varying degrees of KG incompleteness, and validate generalization on the Wikidata-based BRINK benchmark, demonstrating that XoG is competitive with state-of-the-art methods on complete graphs and consistently strongest among methods without task-specific KGQA training under incompleteness, while improving token efficiency.

2 Related Work

KGQA under incomplete KGs. Early KGQA approaches relied on semantic parsing to convert questions into executable logical forms, requiring complete and well-curated knowledge graphs. To mitigate KG incompleteness, embedding-based methods recover missing entities or relations via similarity scores in learned vector spaces (e.g., EmbedKGQA (Saxena et al., 2020), Query2Box (Ren et al., 2020), LEGO (Ren et al., 2021), BeamQA (Atif et al., 2023)). While effective, these approaches depend heavily on training data quality and often introduce noise in long-tail settings. Other works address incompleteness through relation prediction (Zhao et al., 2022) or multi-level knowledge generation (Li et al., 2024), while Var2Vec (Wang et al., 2023a) embeds logical variables alongside link prediction for efficient query answering.

KGQA with LLM Reasoning. Advances in LLM reasoning through prompting strategies, including Chain-of-Thought (Wei et al., 2022) and its variants (Zhang et al., 2023; Fu et al., 2023; Wang et al., 2023b; Kojima et al., 2022; Sun et al., 2024a; Yao et al., 2023a; Besta et al., 2024), decomposition-based methods (Khot et al., 2023), and ReAct (Yao et al., 2023b), have enabled integration of LLMs into KGQA pipelines. ToG (Sun et al., 2024b) and PoG (Chen et al., 2024) guide LLMs to iteratively explore KG evidence, but assume complete graphs. GoG (Xu et al., 2024) relaxes this by allowing LLMs to infer missing links from parametric knowledge, at the cost of increased hallucination risk, while RoG (Luo et al., 2024b) relies on fine-tuning. Other graph-enhanced LLM approaches, including G-Retriever (He et al., 2024) and GNN-RAG (Mavromatis and Karypis, 2025), integrate trainable graph retrieval or graph neural modules with LLM reasoning, requiring additional training

or adaptation components. Overall, existing approaches either depend on complete KGs, require additional task-specific training, or remain vulnerable to hallucination under KG incompleteness.

3 Methodology

3.1 Problem Statement

A *knowledge graph* $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$ is a directed labeled graph, where each triple $(h, r, t) \in \mathcal{T} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ represents a relation r from head entity h to tail entity t.

Knowledge graph embedding (KGE) models represent entities and relations in a low dimensional vector space and measure the likelihood of a triple (h, r, t) via a scoring function $\phi : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \to \mathbb{R}$. In this work, the scoring function is instantiated using the ComplEx model (Trouillon et al., 2016), which embeds entities and relations in a complex vector space $\mathbb{C}^d$ and is capable of modeling asymmetric relations.

Knowledge graph question answering (KGQA) aims to answer a query Q given a topic entity T by retrieving answer entities $W \subset \mathcal{E}$. Answers are typically reached through multi-hop reasoning chains such as: $e_0 \xrightarrow{r_1} e_1 \xrightarrow{r_2} \cdots \xrightarrow{r_n} e_n$.

We consider *KGQA under incompleteness*, where the knowledge graph $\mathcal{G}$ may miss facts required for correct reasoning. In particular, some triples along such a chain may be missing, i.e., for some indices i, $(e_{i-1}, r_i, e_i) \notin \mathcal{T}$. Our goal is therefore to recover missing triples using the observed graph structure and learned embeddings.

3.2 Framework Overview

We propose a modular framework for KGQA designed to be robust under KG incompleteness. As summarized in Fig. 2, XoG follows an iterative *planning-exploration-reasoning* paradigm. In the **planning stage**, the input question is decomposed into sub-questions whose completion status is explicitly tracked throughout the process. The **exploration stage** consists of two components: (i) *relation exploration*, which proposes relations relevant to the current reasoning state, and (ii) *entity exploration*, which retrieves plausible next-hop entities, including those inferred via embeddings. In the **reasoning stage**, an LLM evaluates whether the explored paths are sufficient to answer the question. If not, XoG iterates back to the exploration stage to further expand the paths; otherwise, the answer is extracted directly from the explored evidence.

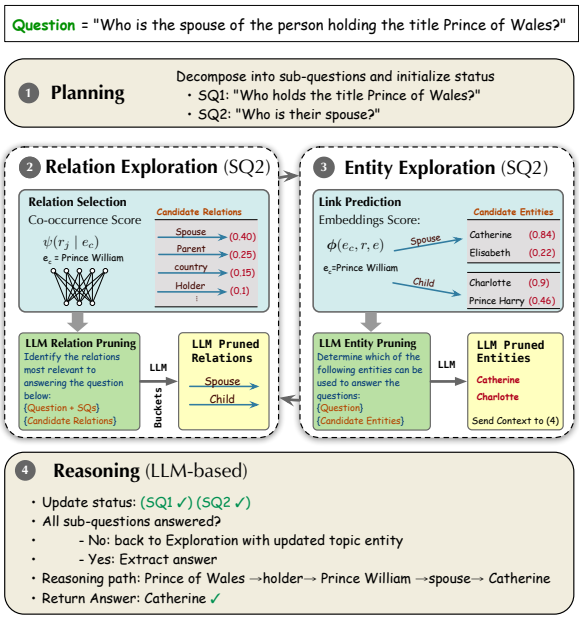


Figure 2: The iterative exploration-reasoning process in XoG for answering a KG-based question.

To handle incompleteness, XoG incorporates recovery mechanisms: relation exploration leverages a curated candidate set derived from corpus-level statistics, while entity exploration uses KG embeddings and link prediction to infer missing entities. Throughout this process, the LLM acts as a guiding agent, pruning noisy candidates and steering traversal based on the reasoning context.

3.3 Planning

To guide adaptive exploration and reasoning, XoG decomposes the input question into a set of sub-questions that capture its underlying question semantics. This decomposition is produced by LLM prompting to generate intermediate questions. Given a question Q, the resulting sub-questions $\{SQ_i\}$ form an explicit execution plan that directs subsequent exploration and reasoning steps.

Following Chen et al. (2024), XoG maintains a *sub-question status* for each SQ_i, indicating whether it has been resolved based on the reasoning paths explored so far. All sub-questions are initially marked as unanswered, and their status is updated iteratively as new paths are explored.

3.4 Relation and Entity Exploration

The exploration phase expands the search over the KG by retrieving relevant facts across multiple hops from the topic entity. Unlike purely traversal-based approaches, XoG augments exploration by inferring missing relations and entities from graph

structure and learned embeddings, enabling recovery of incomplete reasoning paths.

Exploration is initialized from a set of topic entities identified in the question, which serve as the starting points for all reasoning paths. We denote the initial topic entity set as $\mathcal{E}^0 = \{e_1^0, e_2^0, \ldots, e_{N_0}^0\}$, where N_0 is the number of topic entities in the question. Starting from $\mathcal{E}^0$, XoG incrementally expands reasoning paths relevant to the question. A reasoning path $p_n \in \mathcal{P}$ at iteration D is defined as $p_n = \{(e_{s,n}^d, r_n^d, e_{o,n}^d)\}_{d=1}^{D_{pn}}$. Let $\mathcal{E}^{D-1}$ denote the set of entities reached at iteration $D-1$: $\mathcal{E}^{D-1} = \{e_1^{D-1}, e_2^{D-1}, \ldots, e_{N_{D-1}}^{D-1}\}$. This set defines the starting point for the next exploration step, from which new candidate triples are retrieved or inferred.

The exploration phase consists of two complementary components: **relation exploration** and **entity exploration**. In each component, XoG retrieves candidate relations or entities, which are then filtered by an LLM based on their relevance to the question and the current reasoning context.

3.4.1 Relation Exploration

Starting from a topic entity, exploration entails choosing a fitting relation. Instead of considering the full relation vocabulary, which would be prohibitively large, we restrict to a smaller set of relevant relations. Our approach relies on the observation that entity types exhibit characteristic patterns of associated relations (Balaraman et al., 2018). To capture these patterns, we aggregate relation statistics at the entity-type level. For example, relations commonly associated with humans, such as `educated_at` or `spouse`, differ from those associated with universities, such as `has_chancellor`. Entity type information is obtained from the KG ontology when available (e.g., `P31:instance_of` in Wikidata). For Freebase, we infer coarse entity types from relation namespace prefixes, e.g., entities participating in `film.film.directed_by` are associated with the type `film.film`. Relation frequencies are then aggregated across entities sharing the same type, enabling sparse entities to benefit from type-level co-occurrence patterns.

Relation Selection. XoG first retrieves relations directly connected to e_c in the KG. To mitigate graph incompleteness, this set is augmented with curated relations derived from corpus-level statistics. Specifically, we estimate the relevance of a relation $r_j \in \mathcal{R}$ to e_c using a **type-level entity–relation co-occurrence matrix** F, where $F[t, j]$ denotes the frequency with which entities of type t co-occur with relation r_j in the KG. Relations are ranked using the score $\psi(r_j \mid e_c)$, where t_c denotes the type of the current entity e_c:

$$\psi(r_j \mid e_c) = \frac{F[t_c, j]}{\sum_{k \in \mathcal{R}} F[t_c, k]}, \tag{1}$$

LLM Relation Pruning. We form a candidate relation set $\mathcal{R}_{\text{cand}}(e_c)$ by combining relations observed in the KG with curated relations: $\mathcal{R}_{\text{cand}} = \mathcal{R}_{\text{KG}} \cup \mathcal{R}_{\text{cur}}$, where $\mathcal{R}_{\text{KG}}$ corresponds to the entity's neighboring relations and $\mathcal{R}_{\text{cur}}$ to its type-level co-occurrence relations, so that candidates remain structurally and semantically connected through the current entity and its associated entity type. If $\mathcal{R}_{\text{cand}}(e_c)$ exceeds a threshold (bucket size), using a single LLM prompt for pruning can become ineffective; hence, we introduce a novel **hierarchical bucketing prompt strategy** that partitions $\mathcal{R}_{\text{cand}}(e_c)$ into smaller buckets, prunes each independently with the LLM, and merges the retained relations across stages for a final refinement step that yields $\mathcal{R}(e_c)$. Fig. 3 illustrates this approach.

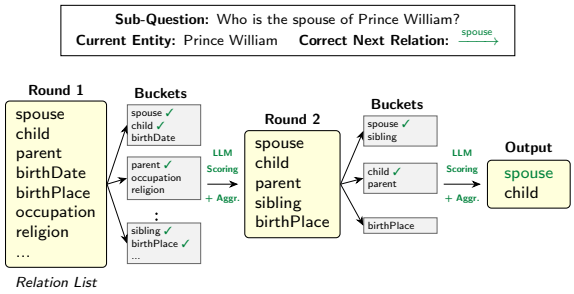


Figure 3: Hierarchical bucketing prompt strategy for relation exploration.

Relations are assigned to buckets at random; this is sufficient because the candidate pool is already restricted to structurally and semantically connected relations, and multi-retention with cross-stage merging recovers useful relations even when their bucketmates are unrelated. In each round, each bucket is independently evaluated by the LLM conditioned on the question Q, sub-questions $\{SQ_i\}$, and current topic entity e_c, which together provide contextual and structural information about the current reasoning step. Multiple highly relevant relations may be retained per bucket. This multi-retention is critical for bridge relations whose value only becomes apparent under longer compositions, avoiding premature commitment to a single locally

dominant choice. The refined set $\mathcal{R}(e_c)$ is then used for subsequent entity exploration.

3.4.2 Entity Exploration

Given the refined relation set from relation exploration, entity exploration identifies plausible next-hop entities that extend the current reasoning paths. At each step, for the current entity e_c and a selected relation $r \in \mathcal{R}(e_c)$, candidate entities are explored using link prediction.

Link Prediction. For each (e_c, r) pair, candidate entities are generated by scoring entities in the KG embedding space using the KGE scoring function ϕ. Depending on relation direction, entities are ranked according to either $\phi(e_c, r, e)$ or $\phi(e, r, e_c)$. The top-K ranked entities (**Entity Context K**) form the candidate set $\mathcal{E}_{\text{cand}}(e_c, r)$. This step allows XoG to recover plausible entities not explicitly connected in the incomplete KG. When $\mathcal{E}_{\text{cand}}(e_c, r)$ becomes large, we use a lightweight pre-trained DistilBERT model (Sanh et al., 2019) to score semantic similarity between the question Q and candidate entities, filtering out irrelevant ones.

LLM Entity Pruning. Since embedding-based inference may introduce noise, XoG employs an LLM to prune candidate entities. The LLM is provided with Q, e_c, r, and $\mathcal{E}_{\text{cand}}(e_c, r)$, and outputs a refined entity set $\mathcal{E}(e_c, r)$. Each retained entity $e \in \mathcal{E}(e_c, r)$ extends the current reasoning path, yielding new candidate paths for subsequent exploration.

3.5 Reasoning Phase

Once candidate reasoning paths have been constructed, XoG enters the reasoning phase to decide whether to continue exploration or produce an answer. In this phase, the LLM reasons using the question, the current reasoning paths, and the maintained sub-question states.

Planning Status Update. Following PoG (Chen et al., 2024), the LLM is prompted with the question Q and the currently explored reasoning paths $\mathcal{P}$ to update the status of each sub-question. Specifically, the LLM summarizes which sub-questions SQ_i are answered by the current paths and updates their corresponding status, which guides subsequent exploration and reasoning iterations.

Evaluation. Based on the current reasoning paths $\mathcal{P}$ and sub-question status $\mathcal{S}$, the LLM evaluates whether sufficient evidence has been accumulated to infer an answer. If so, the LLM integrates the reasoning paths and sub-question status to identify the answer entity. Otherwise, if the maximum search depth has not been reached, the terminal entities of the current paths become new current entities e_c, and XoG returns to the exploration phase to gather additional evidence. This ensures exploration is guided by unresolved sub-questions and answers are grounded in the collected evidence.

4 Experiments

Our main evaluation examines XoG's performance, robustness, and cost under a standard KG incompleteness protocol. We then focus our analysis on component contributions and sensitivity to key design choices. We conclude by assessing generalization to a different KG and incompleteness setting.

4.1 Experimental Setup

Datasets. We evaluate XoG on three multi-hop KGQA benchmarks: WebQSP (Yih et al., 2016), CWQ (Talmor and Berant, 2018), and BRINK (Zhou et al., 2026). WebQSP and CWQ require multi-hop reasoning over the Freebase knowledge graph (Bollacker et al., 2008), while BRINK evaluates KGQA over Wikidata5m (Wang et al., 2021) under both complete and incomplete graph settings.

Evaluation Metrics. Following prior work (Chen et al., 2024; Sun et al., 2024b), we report exact match accuracy (Hits@1) on WebQSP and CWQ, where a prediction is correct if the predicted entity matches any gold answer entity. For BRINK, we report F1 following its original evaluation protocol (Zhou et al., 2026).

KG Incompleteness Protocol. To evaluate robustness under incomplete KGs, for WebQSP and CWQ, we follow the protocol in GoG (Xu et al., 2024) where the complete KG is denoted **CKG**, and four incomplete KG versions are created: **IKG-20%**, **IKG-40%**, **IKG-60%**, and **IKG-80%**, by randomly removing 20%, 40%, 60%, and 80% of the *crucial triples* required to answer each question. Crucial triples lie along the gold reasoning path from the topic entity to the correct answer. All relations between the same entity pairs are also removed. For BRINK, we use its provided incomplete split. For each incomplete-KG setting, we

train the KG embeddings on the corresponding pruned graph, excluding all removed triples.

Baselines. We compare XoG against representative KGQA baselines spanning supervised and no task-specific KGQA training settings. As an LLM-only reference, we evaluate Chain-of-Thought (CoT) (Wei et al., 2022) without access to the knowledge graph for each LLM backbone. *LLM-driven semantic parsing methods* translate questions into executable KG queries; we compare against KB-BINDER (Li et al., 2023) and ChatKBQA (Luo et al., 2024a). *Supervised KGQA methods* use task-specific supervision for KG-based question answering; we include EmbedKGQA (Saxena et al., 2020). *LLM-based KG reasoning methods* integrate LLMs with structured retrieval and graph reasoning; we compare against StructGPT (Jiang et al., 2023), RoG (Luo et al., 2024b), ToG (Sun et al., 2024b), PoG (Chen et al., 2024), and GoG (Xu et al., 2024).

Training Regime. We use the term *No Task-Specific KGQA Training* to describe the family of methods that do not train or fine-tune on KGQA supervision, including the use of question-answer pairs or annotated reasoning chains. Nonetheless, this designation permits the use of pretrained components, such as KG embeddings and small language models, as well as other auxiliary tools.

Default LLM. Following prior work, we use GPT-3.5-Turbo as the default model. Alternatives are evaluated in Section 4.2 and Appendix C.

Reproducibility. Details on code, datasets, prompts, relevant parameters, and experimental setup are available on the project repository: https://github.com/colab-nyuad/XoG

4.2 Overall Performance Comparison

Table 1 reports results for WebQSP and CWQ under complete (CKG) and incomplete (IKG-40%) settings. On complete graphs, XoG remains competitive with or outperforms comparable methods that do not use task-specific KGQA training. Fine-tuned models achieve strong performance but require task-specific training data. The performance gap between XoG and ToG highlights the limitations of traversal-based reasoning. In contrast, the relatively close performance of XoG and PoG suggests that XoG's information recovery provides comparatively limited additional benefit when the graph is complete, though, as shown below, this benefit grows substantially once the graph becomes incomplete.

Method	WebQSP		CWQ	
	CKG	IKG-40%	CKG	IKG-40%
CoT (Without Knowledge Graph)				
GPT-3.5-Turbo	76.0		54.1	
GPT-4	76.4		55.5	
GPT-5.5	79.1		74.0	
Claude Opus 4.8	80.7		62.8	
Supervised KGQA				
EmbedKGQA	66.6	42.5	44.7	NA
RoG*	88.6	78.2	66.1	54.2
ChatKBQA*	78.1	49.5	76.5	39.3
No Task-Specific KGQA Training (GPT-3.5-Turbo)				
KB-BINDER*	50.7	38.4	–	–
StructGPT*	76.4	60.1	–	–
ToG*	76.9	63.4	47.2	37.9
GoG*	78.7	66.6	55.7	44.3
PoG±	82.0	69.7	63.2	53.7
XoG	**83.7**	**77.6**	**64.4**	**55.7**
No Task-Specific KGQA Training (GPT-4)				
ToG*	80.3	71.8	71.0	56.1
GoG*	84.4	80.3	**75.2**	60.4
PoG±	85.2	71.6	71.9	58.0
XoG	**85.7**	**83.5**	72.0	64.7
No Task-Specific KGQA Training (GPT-5.5)				
ToG	86.2	78.6	78.0	74.1
XoG	**87.3**	**83.8**	80.5 ▲	75.9 ▲
No Task-Specific KGQA Training (Claude Opus 4.8)				
ToG	85.8	78.6	72.6	68.2
XoG	89.2 ▲	85.4 ▲	**77.6**	**68.8**

* Results are taken from GoG (Xu et al., 2024).
± Results obtained using PoG (Chen et al., 2024) code on the incompleteness datasets.

Table 1: Hits@1 scores on WebQSP and CWQ under complete (CKG) and incomplete (IKG-40%) knowledge graph settings. CoT (No KG) is evaluated without KG access and serves as a backbone-matched LLM-only baseline. **Bold** indicates the best result and underlined the second best within each *"No Task-Specific KGQA Training"* paradigm. ▲ marks the best overall result in each column.

Under incomplete graphs, all KG-based methods experience performance degradation due to missing triples, though the size of this degradation varies by method and dataset. Traversal-based approaches such as ToG are particularly vulnerable to this setting, as broken reasoning paths can terminate exploration prematurely; PoG is designed to partially mitigate this through adaptive exploration and self-correction, which enables the discovery of alternative reasoning paths when the initial path is blocked. Compared to GoG, which relies on LLM-generated facts to fill in missing information, XoG instead adopts a graph-grounded recovery strategy, which we find is associated with consistently stronger performance under KG incompleteness.

We further evaluate XoG across multiple LLM backbones. For each backbone, we evaluate *CoT* as a backbone-matched LLM-only baseline, while

ToG represents iterative LLM-guided KG exploration. Across all evaluated backbones, XoG consistently outperforms both *CoT* and ToG on WebQSP and CWQ under CKG, and continues to outperform ToG under IKG-40% as backbones grow stronger, indicating that improved LLM reasoning alone does not eliminate the challenges introduced by missing graph information.

To assess whether KGQA performance is driven by LLM parametric memorization, we examine the *CoT* setting, where the model must rely solely on its parametric knowledge. CoT accuracy alone cannot confirm or rule out memorization. However, *CoT* falls well below XoG under both CKG and IKG-40%, so parametric knowledge alone cannot reproduce the performance achieved with KG access, and XoG's gains are not primarily due to memorized Freebase facts during pretraining. This trend holds across additional LLM backbones (Appendix C).

4.3 Robustness to KG Sparsity

Next, we analyze robustness under progressively increasing KG sparsity. We focus on WebQSP and CWQ, whose controlled incompleteness protocol enables evaluation across multiple sparsity levels (IKG-20%–80%) as shown in Fig. 4. We observe that planning-based methods such as PoG and XoG achieve similar performance when the KG is complete. As more relations are removed, the performance of all methods gradually degrades. Notably, traversal-based approaches such as ToG exhibit the sharpest accuracy drop because they depend on access to the full KG structure. PoG is more robust due to its reflection module, but suffers noticeable degradation under severe incompleteness. GoG partially compensates for missing links using LLM-generated knowledge, though its gains diminish under extreme sparsity, particularly on CWQ. In contrast, XoG consistently achieves the strongest performance across all sparsity levels on both datasets, suggesting that its augmented exploration strategy supports more robust reasoning as graph incompleteness increases.

4.4 Efficiency Analysis

We study the efficiency of XoG by comparing it with PoG, the most closely related baseline in pipeline structure and performance. We measure the average number of LLM calls and the token consumption required to answer a question (Table 2). Across all settings, XoG consistently requires fewer LLM calls and substantially fewer tokens than PoG. On complete graphs, both methods exhibit similar LLM usage, with XoG showing modest reductions (2–7%). However, as sparsity increases, PoG's iterative reflection mechanism leads to increasing LLM calls and token consumption, while XoG remains stable. Under incomplete settings, XoG reduces total token consumption by up to 33% compared to PoG. This efficiency gain stems from XoG's integration of selection tools such as relation selection and embedding-based entity retrieval. By recovering relevant evidence early, XoG avoids repeated exploration and reflection cycles and reserves LLM usage primarily for pruning and final reasoning. As a result, XoG achieves a superior accuracy–cost trade-off and scales more gracefully as KG incompleteness increases. End-to-end inference takes approximately 12 seconds per question, with LLM inference accounting for roughly 85% of this time; KGE scoring adds only about 0.3 seconds, with the remainder spent on SPARQL querying, relation-expansion lookups, and DistilBERT filtering. This indicates that LLM usage remains the dominant inference cost, while the added embedding-based recovery introduces relatively little overhead.

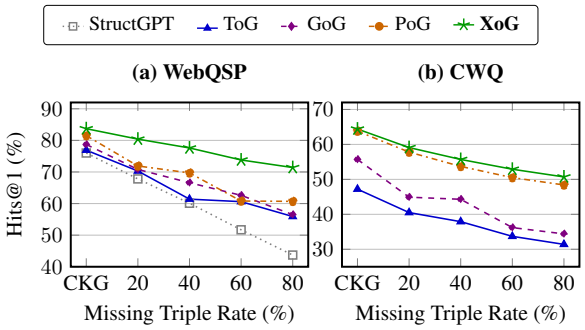


Figure 4: Hits@1 results on complete (CKG) and incomplete (IKG-%) knowledge graphs.

4.5 Sensitivity Analysis

Next, we analyze the sensitivity of XoG to key design choices: hierarchical relation selection and the quality of embedding-based entity retrieval. We use WebQSP for these analyses because its linear reasoning paths provide a clear sequence of relation and entity selections, making the behavior of the individual modules easier to interpret.

The Effect of Hierarchical Relation Selection and Context Size. To evaluate the hierarchical relation selection strategy in isolation, we measure

Dataset	Sparsity	LLM Calls			Input Tokens			Output Tokens			Total Tokens		
		PoG	XoG	Δ	PoG	XoG	Δ	PoG	XoG	Δ	PoG	XoG	Δ
CWQ	CKG	14.67	**13.59**	−7%	8,852	**8,192**	−7%	390	**361**	−7%	9,243	**8,553**	−7%
	IKG-40%	17.68	**14.00**	−21%	10,726	**8,397**	−22%	464	**373**	−20%	11,191	**8,770**	−22%
	IKG-80%	19.50	**14.30**	−27%	11,886	**8,638**	−27%	506	**386**	−24%	12,393	**9,024**	−27%
WebQSP	CKG	9.11	**8.90**	−2%	5,391	**5,160**	−4%	286	**271**	−5%	5,677	**5,432**	−4%
	IKG-40%	13.70	**9.20**	−33%	8,033	**5,376**	−33%	376	**258**	−31%	8,409	**5,634**	−33%
	IKG-80%	13.22	**9.40**	−29%	7,812	**5,447**	−30%	368	**255**	−31%	8,180	**5,703**	−30%

Table 2: LLM usage comparison between PoG and XoG across datasets and KG sparsity versions.

first-hop relation prediction accuracy under varying KG sparsity levels and bucket sizes. Table 3 shows that hierarchical relation selection improves performance under both complete and incomplete KG settings. A bucket size of 30 achieves the best results across all sparsity levels while maintaining stable performance under increasing KG sparsity. The candidate relation set contains approximately 98 relations per query, including both neighboring KG relations and additional relations introduced through type-level co-occurrence statistics. As bucket size increases, performance gradually declines and drops further when hierarchy is removed entirely. These results suggest that hierarchical bucketing benefits relation exploration by enabling progressive refinement over manageable subsets of the candidate space rather than requiring the LLM to reason over the full relation set simultaneously.

Bucket Size	KG Sparsity (%)				
	CKG	20	40	60	80
30	**80.40**	**80.62**	**80.60**	**80.12**	**80.22**
40	80.32	80.12	80.32	80.02	79.82
50	79.31	78.60	79.10	78.50	78.80
70	77.99	78.29	77.89	78.19	76.98
No Hierarchy	77.38	77.28	77.08	77.28	76.97

Table 3: Effect of bucket size on relation selection hit rate on WebQSP under different KG sparsity levels. Columns are CKG (complete knowledge graph), and missing-triples rates (IKG-%).

The Effect of Varying Retrieval Quality. To characterize XoG's dependence on link-prediction quality, we conduct a synthetic study of how retrieval quality affects downstream entity selection. We simulate retrieval quality levels by varying the Hits@100 of the candidate entity set during first-hop entity selection on WebQSP and evaluate the LLM-based selection module under each setting.

Figure 5 shows that LLM entity selection accuracy closely follows the retrieval quality. As Hits@100 decreases, the LLM accuracy also declines, suggesting that the LLM is generally effective at identifying the correct entity when it is present within the retrieved candidate pool. Thus, entity exploration is primarily constrained by embedding-based retrieval quality rather than LLM selection. This is consistent with the error analysis in Appendix B, where retrieval gaps dominate XoG's errors and increase with KG sparsity.

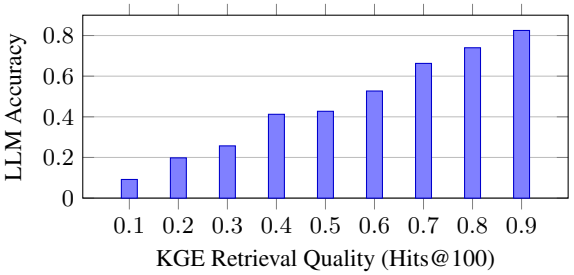


Figure 5: Effect of retrieval quality (Hits@100) on LLM entity selection accuracy.

4.6 Ablation Study

We complement the module-level analysis with end-to-end ablations to assess component contributions to KGQA performance. Table 4 reports Hits@1 on WebQSP under complete and incomplete KG settings. The ablations test two axes: recovery mechanisms (co-occurrence statistics and link prediction) and LLM-guided selection (relation and entity pruning). A detailed standalone evaluation of the relation and entity selection modules is provided in Appendix D.

Recovery mechanisms. The *w/o Relation Selection* version removes the curated relation dictionary and relies only on adjacent relations, while *w/o Link Prediction* disables embedding-based link prediction, restricting entity discovery to existing graph neighbors. Removing relation selec-

Method	CKG	IKG-40%	IKG-80%
XoG (full)	**83.7**	**77.6**	**72.3**
(a) Recovery mechanisms			
w/o Relation Selection	82.9	73.8	68.9
w/o Link Prediction	76.2	62.0	67.0
(b) LLM-guided selection			
w/o LLM Entity Prune (Embed top1)	77.9	69.8	65.7
w/o LLM Entity Prune (Embed top3)	78.6	73.2	66.9
w/o LLM Relation Prune (DistilBERT)	78.2	74.6	70.7

Table 4: Ablation results of XoG under different sparsity levels on WebQSP.

tion has minimal effect when the graph is complete, but causes a noticeable decline on sparser graphs, highlighting its importance when neighborhood information is incomplete. Removing link prediction yields a larger drop across all settings, including CKG. We hypothesize that the benefit under CKG occurs because link prediction can also help disambiguate among multiple candidate entities for many-to-many relations (e.g., siblings and co-actors), where graph neighbors alone may be insufficient for fine-grained selection. The non-monotonicity between IKG-40% and IKG-80% may arise because performance depends not only on the amount of missing information, but also on which triples are removed and which alternative reasoning paths remain available.

LLM-guided selection. During exploration, XoG uses the LLM to prune candidate relations and entities based on the current reasoning context. To assess whether its use improves upon simpler selection strategies, Table 4 compares the full model with two types of ablation: *w/o LLM Entity Prune*, where we retain the top-1 or top-3 embedding-scored entity candidates, and *w/o LLM Relation Prune*, where we select relations using a DistilBERT-based similarity baseline. All three selection alternatives consistently degrade performance, indicating that the LLM's selections improve XoG accuracy over these alternatives.

4.7 Generalization to Wikidata-based KGQA

Finally, to assess whether XoG's effectiveness extends beyond Freebase-based benchmarks, we evaluate it on BRINK: a Wikidata5m-based benchmark. This provides a complementary test under a different incompleteness protocol, which removes rule-mined triples while ensuring that the questions can still be answered. Table 5 shows that under incompleteness, XoG achieves the highest F1 among methods that do not use task-specific KGQA training. Although supervised approaches such as RoG and GNN-RAG remain more robust, XoG substantially narrows this gap without task-specific supervision. On complete graphs, StructGPT performs best among the non-task-trained methods, whereas XoG performs similarly to PoG. This mirrors the Freebase results: XoG offers limited gains on complete graphs but consistently outperforms comparable baselines under KG incompleteness.

Method	**Complete**	**Incomplete**
Supervised KGQA		
G-Retriever	0.32	0.30
RoG	0.78	0.62
GNN-RAG	0.79	0.68 ▲
No Task-Specific KGQA Training (GPT-3.5-Turbo)		
PoG	<u>0.71</u>	0.34
ToG	0.64	0.32
StructGPT	0.81 ▲	<u>0.40</u>
XoG	<u>0.71</u>	**0.52**

Table 5: F1 on the BRINK Wikidata benchmark using entity text labels, under complete and incomplete KG settings. Baseline numbers are taken from Zhou et al. (2026). **Bold** indicates the best result and <u>underlined</u> the second best within the *"No Task-Specific KGQA Training"* paradigm. ▲ marks the best overall result in each column.

5 Conclusion

We presented XoG, a framework for LLM-based question answering over incomplete knowledge graphs. By integrating graph-grounded recovery into an iterative *planning–exploration–reasoning* paradigm, XoG recovers missing reasoning paths while grounding LLM reasoning in graph evidence. Extensive experiments demonstrate that this design improves both answer accuracy and token efficiency, consistently outperforming methods without task-specific KGQA training under KG incompleteness while requiring fewer LLM calls than comparable approaches. These gains hold across six LLM backbones and extend beyond Freebase-based evaluation to Wikidata, showing that the benefits of graph-grounded recovery persist across different backbone strengths and KG settings. The advantage grows with KG sparsity, and our error analysis shows that the remaining failures are dominated by retrieval gaps rather than LLM selection errors, highlighting improved retrieval and KG completion as key directions for further gains.

Limitations

Our framework relies on a knowledge graph embedding (KGE) module for link prediction. In this work, we employ ComplEx as a strong, widely implemented, and computationally efficient baseline, which allows us to focus on analyzing the behavior of the proposed method. Nonetheless, our approach is model-agnostic by design and can readily incorporate more expressive or higher-accuracy knowledge graph completion models. To characterize the dependence of XoG on the underlying KGE component, we conduct a synthetic link-prediction sensitivity study in Section 4.5.

We evaluate under synthetic incompleteness using the benchmark introduced by GoG (Xu et al., 2024), which removes crucial triples at incremental rates. This setup enables a fair comparison with existing baselines without task-specific KGQA training, but may not fully capture real-world incompleteness patterns, such as uneven coverage across entity classes (Luggen et al., 2019) or gaps in coverage of long-tail entities (Tonon et al., 2016). We additionally validate on BRINK, a Wikidata-based benchmark; however, its incomplete split is also synthetically constructed through rule-mined triple removal. Establishing broader incomplete-KGQA benchmarks that better reflect naturally occurring incompleteness and account for KG evolution (Difallah, 2025) remains an important direction for future work.

Our primary evaluation uses WebQSP and CWQ, which are grounded in Freebase and enable direct comparison with prior incomplete-KGQA baselines. We additionally evaluate on BRINK over Wikidata5m, providing evidence that XoG generalizes beyond Freebase. Nevertheless, broader generalization to other KGs and domains remains to be established. Applying XoG to a new KG requires access to entity-type information for constructing type-level relation co-occurrence statistics. Performance on KGs without accessible type structure therefore remains a limitation.

References

Anthropic. 2026. Introducing Claude Opus 4.8. https://www.anthropic.com/news/claude-opus-4-8. Accessed: 2026-09-18.

Farah Atif, Ola El Khatib, and Djellel Eddine Difallah. 2023. BeamQA: Multi-hop knowledge graph question answering with sequence-to-sequence prediction and beam search. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023*, pages 781–790. ACM.

Vevake Balaraman, Simon Razniewski, and Werner Nutt. 2018. Recoin: Relative completeness in Wikidata. In *Companion Proceedings of the The Web Conference 2018, WWW 2018*, pages 1787–1792. ACM.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 17682–17690. AAAI Press.

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

Samuel Broscheit, Daniel Ruffinelli, Adrian Kochsiek, Patrick Betz, and Rainer Gemulla. 2020. LibKGE - a knowledge graph embedding library for reproducible research. In *EMNLP (Demos)*, pages 165–174. Association for Computational Linguistics.

Liyi Chen, Panrong Tong, Zhongming Jin, Ying Sun, Jieping Ye, and Hui Xiong. 2024. Plan-on-graph: Self-correcting adaptive planning of large language model on knowledge graphs. In *Advances in Neural Information Processing Systems*, volume 37, pages 37665–37691. Curran Associates, Inc.

Djellel Eddine Difallah. 2025. WikiRAG: Revisiting Wikidata KGC datasets with community updates and retrieval-augmented generation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining, V.2, KDD 2025*, pages 5391–5401. ACM.

Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2023. Complexity-based prompting for multi-step reasoning. In *The Eleventh International Conference on Learning Representations, ICLR 2023*. OpenReview.net.

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

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and

open questions. *ACM Trans. Inf. Syst.*, 43(2):42:1–42:55.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Networks Learn. Syst.*, 33(2):494–514.

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

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2023. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations, ICLR 2023*. OpenReview.net.

Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213. Curran Associates, Inc.

Qian Li, Zhuo Chen, Cheng Ji, Shiqi Jiang, and Jianxin Li. 2024. LLM-based multi-level knowledge generation for few-shot knowledge graph completion. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 2135–2143. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Tianle Li, Xueguang Ma, Alex Zhuang, Yu Gu, Yu Su, and Wenhu Chen. 2023. Few-shot in-context learning on knowledge base question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023*, pages 6966–6980. Association for Computational Linguistics.

Llama Team. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Michael Luggen, Djellel Eddine Difallah, Cristina Sarasua, Gianluca Demartini, and Philippe Cudré-Mauroux. 2019. Non-parametric class completeness estimators for collaborative knowledge graphs - the case of Wikidata. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference*, volume 11778 of *Lecture Notes in Computer Science*, pages 453–469. Springer.

Haoran Luo, Haihong E, Zichen Tang, Shiyao Peng, Yikai Guo, Wentai Zhang, Chenghao Ma, Guanting Dong, Meina Song, Wei Lin, Yifan Zhu, and Anh Tuan Luu. 2024a. ChatKBQA: A generate-then-retrieve framework for knowledge base question answering with fine-tuned large language models. In *Findings of the Association for Computational Linguistics, ACL 2024*, volume ACL 2024 of *Findings of ACL*, pages 2039–2056. Association for Computational Linguistics.

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

Costas Mavromatis and George Karypis. 2025. GNN-RAG: graph neural retrieval for efficient large language model reasoning on knowledge graphs. In *Findings of the Association for Computational Linguistics, ACL 2025*, volume ACL 2025 of *Findings of ACL*, pages 16682–16699. Association for Computational Linguistics.

Qwen Team. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Hongyu Ren, Hanjun Dai, Bo Dai, Xinyun Chen, Michihiro Yasunaga, Haitian Sun, Dale Schuurmans, Jure Leskovec, and Denny Zhou. 2021. LEGO: latent execution-guided reasoning for multi-hop question answering on knowledge graphs. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021*, volume 139 of *Proceedings of Machine Learning Research*, pages 8959–8970. PMLR.

Hongyu Ren, Weihua Hu, and Jure Leskovec. 2020. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In *8th International Conference on Learning Representations, ICLR 2020*. OpenReview.net.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Apoorv Saxena, Aditay Tripathi, and Partha P. Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*, pages 4498–4507. Association for Computational Linguistics.

Jiashuo Sun, Yi Luo, Yeyun Gong, Chen Lin, Yelong Shen, Jian Guo, and Nan Duan. 2024a. Enhancing chain-of-thoughts prompting with iterative bootstrapping in large language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, volume NAACL 2024 of *Findings of ACL*, pages 4074–4101. Association for Computational Linguistics.

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

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018*, pages 641–651. Association for Computational Linguistics.

Alberto Tonon, Victor Felder, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. 2016. VoldemortKG: Mapping schema.org and web entities to linked open data. In *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference*, volume 9982 of *Lecture Notes in Computer Science*, pages 220–228.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2071–2080, New York, New York, USA. PMLR.

Dingmin Wang, Yeyuan Chen, and Bernardo Cuenca Grau. 2023a. Efficient embeddings of logical variables for query answering over incomplete knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4652–4659. AAAI Press.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Yao Xu, Shizhu He, Jiabei Chen, Zihao Wang, Yangqiu Song, Hanghang Tong, Guang Liu, Jun Zhao, and Kang Liu. 2024. Generate-on-graph: Treat LLM as both agent and KG for incomplete knowledge graph question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024*, pages 18410–18430. Association for Computational Linguistics.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023a. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, pages 11809–11822. Curran Associates, Inc.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023b. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023*. OpenReview.net.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206, Berlin, Germany. Association for Computational Linguistics.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2023. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023*. OpenReview.net.

Fen Zhao, Yinguo Li, Jie Hou, and Ling Bai. 2022. Improving question answering over incomplete knowledge graphs with relation prediction. *Neural Comput. Appl.*, 34(8):6331–6348.

Dongzhuoran Zhou, Yuqicheng Zhu, Xiaxia Wang, Hongkuan Zhou, Yuan He, Jiaoyan Chen, Steffen Staab, and Evgeny Kharlamov. 2026. What breaks knowledge graph based RAG? Benchmarking and empirical insights into reasoning under incomplete knowledge. In *EACL (Volume 1: Long Papers)*, pages 2522–2538. Association for Computational Linguistics.

A Overall Efficiency Analysis

XoG relies on two offline preprocessing components that are computed once per KG setting, and therefore can be amortized over all inference instances. First, we train ComplEx KG embeddings using LibKGE (Broscheit et al., 2020); on Freebase, this step takes approximately 5 hours on a single NVIDIA A100 GPU. Second, XoG constructs the type-level entity–relation co-occurrence matrix F used for relation expansion in Eq. 1. This requires a single pass over the training triples to accumulate counts between entity types and relations, followed by normalization over relations for each entity type; both steps are linear in their inputs. At inference time, XoG combines LLM calls with lightweight non-LLM components, with overall efficiency primarily determined by LLM usage, prompt length, and the number of exploration iterations.

B Error Analysis

B.1 Retrieval and Reasoning Error Decomposition

We split XoG's wrong predictions into two categories to attribute failures to the corresponding component:

- **Selection Error**: the share of wrong predictions where the correct answer was present in the retrieved chains, but the LLM failed to select it. This isolates reasoner-side failures.
- **Retrieval Gap**: the share of wrong predictions where the correct answer was absent from the retrieved chains altogether. This isolates retriever-side failures caused by missing or unretrievable graph evidence.

As shown in Table 6, retrieval gaps dominate the error distribution across both datasets and grow further as sparsity increases, while selection errors remain comparatively low. This indicates that performance degradation under KG incompleteness is driven primarily by the quality of the embedding space used for retrieval, rather than by the LLM misusing the chains it does retrieve.

Dataset	Metric	CKG	IKG-40%	IKG-80%
WebQSP	Accuracy	83.7	77.6	72.3
	Selection Error	15.9	10.3	6.6
	Retrieval Gap	84.1	89.7	93.4
CWQ	Accuracy	64.4	55.7	48.7
	Selection Error	13.9	10.7	6.5
	Retrieval Gap	86.1	89.3	93.5

Table 6: Error Analysis of XoG under increasing KG incompleteness (all values in %).

B.2 Robustness and Efficiency vs. PoG

We next compare XoG against PoG on two axes: how often each system's answer is backed by a retrieved chain (i.e., a *graph-grounded answer*), rather than produced from the LLM memory; and the *average chain length*, which reflects how much exploration each system needs.

Fig. 6(a) reports the portion of *graph-grounded answers* as the KG gets sparsified. For CKG the two methods behave similarly. As more triples are removed, PoG's dependence on direct traversal causes its grounded rate to drop, while XoG degrades more gracefully. This shows that embedding-based recovery lets XoG keep grounding its answers in the graph when paths are broken.

Fig. 6(b) reports the average chain length among graph-grounded answers. XoG keeps its reasoning chains short across all sparsity levels, whereas PoG's chains become longer under incompleteness as it explores laterally to compensate for the missing links.

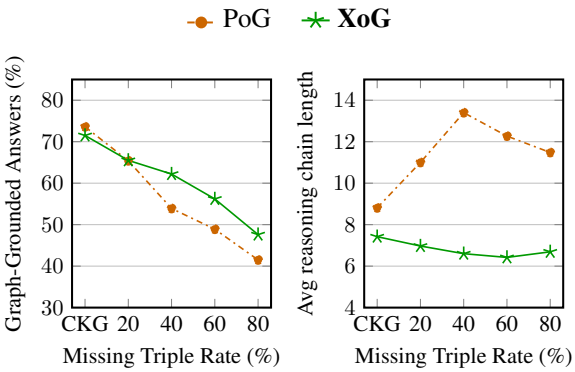


Figure 6: PoG vs. XoG under varying KG sparsity on WebQSP. **(a)** Graph-grounded answer rate. **(b)** Average reasoning chain length.

C Extended LLM Backbone Comparison

Table 7 reports results for XoG with six backbone LLMs drawn from several model families and generations: GPT-3.5-Turbo, GPT-4, GPT-5.5, Qwen3-32B (Qwen Team, 2025), LLaMA-3.3-70B (Llama Team, 2024), and Claude Opus 4.8 (Anthropic, 2026). The choice of backbone has a large effect on performance under both complete and incomplete KGs, and no single model is best on both datasets. Claude Opus 4.8 is the strongest on WebQSP, while GPT-5.5 leads on CWQ. The gap between backbones is wider on CWQ. This may stem from the complex compositional nature of CWQ questions, which rely more heavily on question decomposition and sub-question planning; an area where stronger backbones can have an edge.

	Setting	GPT-3.5	Qwen3	Llama-3.3	GPT-4	GPT-5.5	Opus 4.8
WebQSP	CKG	83.7	77.9	84.9	85.7	87.3	89.2
	IKG-40%	77.6	71.8	81.6	83.5	83.8	85.4
	CoT	76.0	62.7	76.9	76.4	79.1	80.7
CWQ	CKG	64.4	60.6	66.1	72.0	80.5	77.6
	IKG-40%	55.7	53.0	61.3	64.7	75.9	68.8
	CoT	54.1	48.0	57.5	55.5	74.0	62.8

Table 7: Hits@1 (%) of XoG with different backbone LLMs across complete (CKG), incomplete (IKG-40%), and CoT (No KG) settings.

We also find that strong reasoning from parametric knowledge alone does not necessarily lead to strong reasoning over a KG. For example, in the CoT(No KG) setting, LLaMA-3.3 slightly outperforms GPT-4 on both WebQSP and CWQ. With

KG access, however, GPT-4 comes out ahead on both. The benefit of KG access also varies across backbones, which ssuggests that some models use external knowledge more effectively than others. Finally, XoG remains effective across all six backbones under KG incompleteness. Although performance decreases from the complete KG to IKG-40% for every model, the IKG-40% results remain well above the corresponding CoT baselines. The robustness of XoG to missing facts is therefore not limited to a particular LLM family.

D Relation and Entity Selection Analysis

We evaluate the relation and entity selection modules individually. Since both modules rely on LLM pruning, we report *hit rate*, defined as the proportion of queries for which the correct relation or entity is present in the LLM-returned candidate set. All experiments are conducted on WebQSP versions, as the questions' linear reasoning paths enable controlled analysis of the selection modules.

Method	CKG	IKG-40%	IKG-80%
All Relations	6.20	6.20	6.20
Neighbors only	**80.20**	35.00	10.54
Relation Selection	77.99	**77.89**	**76.98**

Table 8: Relation selection hit rate on WebQSP.

Relation Selection Module. To carry out this isolated evaluation, we apply relation selection to the question's topic entity to retrieve $K = 70$ candidate relations, followed by hierarchical LLM pruning using the question and topic entity as context. Table 8 shows that on the complete graph (CKG), the Neighbors baseline achieves the highest hit rate by leveraging existing entity connections. However, as sparsity increases, this approach's performance degrades since relevant relations are proportionally missing from the neighborhood. In contrast, our relation selection module maintains stable hit rate across all sparsity levels, demonstrating effectiveness in recommending correct relations despite the graph incompleteness.

Entity Selection Module. We apply link prediction to each question's (topic entity, ground-truth first-hop relation) pair to retrieve the top-K candidate entities, followed by LLM pruning using the question as context. Fig. 7 shows that on the complete graph (CKG), where the correct entity ranks near the top (average answer rank, AAR=2), a small context ($K = 10$) achieves the best hit rate; increasing K only introduces noise. As sparsity increases, link prediction quality degrades and average answer rank increases to 15 for IKG-40% and 51 for IKG-80%, requiring larger context sizes to ensure coverage of the correct entity. Ultimately, expanding K introduces noise with larger context sizes.

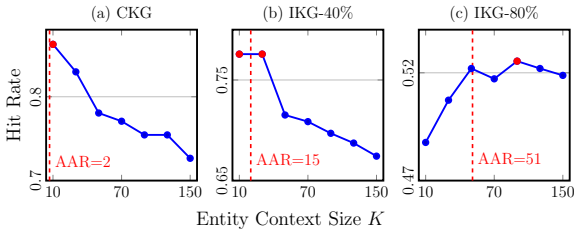


Figure 7: Effect of entity context size K on entity selection hit rate. Average answer rank (AAR) is indicated.