ComplexLogicalQA: Benchmarking Knowledge Graph Question Answering Beyond Chain-like Reasoning

Anonymous authors
Paper under double-blind review

Abstract

Knowledge Graph Question Answering (KGQA) benchmarks have substantially advanced the evaluation of multi-hop reasoning. However, our structural audit shows that widely used benchmarks remain heavily skewed toward chain-like queries, with limited coverage of nested logical compositions and no union structures in our sampled analysis. To address this gap, we introduce **ComplexLogicalQA (CLQ)**, a structure-balanced diagnostic benchmark covering nine Existential Positive First-Order Logic (EPFO) query structures. CLQ is constructed through structure-constrained subgraph sampling, followed by LLM-based verbalization and multi-stage verification to preserve executable logical validity while improving linguistic diversity. Experiments across four KGQA paradigms reveal a clear and consistent trend: performance degrades as queries move beyond chain-like projections, with the largest and most stable drops on nested compositions. Union structures are not uniformly hardest under every metric, but they consistently expose weaknesses in complete answer-set recovery. These trends remain visible after controlling for multiple observable non-structural factors. Further diagnostic analyses reveal paradigm-specific bottlenecks in evidence acquisition, graph exploration, and logical-form grounding. CLQ thus serves as a diagnostic testbed for developing more structure-aware KGQA systems.

1 Introduction

Knowledge Graph Question Answering (KGQA) has become a central testbed for studying grounded reasoning over structured knowledge. With the rise of Large Language Models (LLMs), recent KGQA systems have achieved strong performance on popular benchmarks by combining parametric reasoning with external knowledge sources (OpenAI, 2025; Achiam et al., 2023; DeepSeek-AI et al., 2024; Lewis et al., 2020). However, progress in KGQA is increasingly constrained not only by model capability, but also by what current benchmarks actually test. In particular, widely used KGQA benchmarks are heavily skewed toward chain-like multi-hop queries, with much weaker coverage of branching and nested logical compositions.

To make this issue precise, we adopt the lens of Existential Positive First-Order Logic (EPFO), following established formulations from complex query answering (Ren et al., 2020). Under this view, KGQA queries can be grouped into projection structures ($1p, 2p, 3p$), intersection structures ($2i, 3i$), union structures ($2u$), and nested compositions (ip, pi, up). These structure families place different demands on a system: projection-dominant queries mainly test sequential path following, whereas branching and nested structures require models to preserve multiple constraints, combine evidence across branches, and recover complete answer sets under compositional reasoning. Using this lens, we conduct a structural audit of 3,000 randomly sampled questions from three widely used KGQA benchmarks: WebQSP (Yih et al., 2016), CWQ (Talmor & Berant, 2018), and GrailQA (Gu et al., 2021). A broader audit including recent KGQA benchmarks (Peng et al., 2025; Cao et al., 2022; Zhang et al., 2025) is provided in Appendix B. As shown in Table 2, the sampled questions are heavily concentrated in chain-like structures. WebQSP and GrailQA are dominated by linear paths ($1p, 2p$), accounting for 94.0% and 91.1% of the sampled questions, respectively, while CWQ includes some intersection structures but it still provides limited coverage of nested logic. Most strikingly, union structures ($2u, up$) do not appear in our sampled analysis of any of the three benchmarks.

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

Table 1: Summary of the 9 logical query structures in our dataset. The notation P denotes the projection operator, while $\cap$ and $\cup$ denote intersection and union operations, respectively.

Type	Formal Definition	Natural Language Question
1p	$P(\{e\}, r_1)$	Who directed the movie *Inception*?
2p	$P(P(\{e\}, r_1), r_2)$	What is the birth country of the director of *Inception*?
3p	$P(P(P(\{e\}, r_1), r_2), r_3)$	What awards were won by actors who starred in movies directed by *Nolan*?
2i	$P(\{e_1\}, r_1) \cap P(\{e_2\}, r_2)$	Who directed both *Inception* and *Interstellar*?
3i	$P(\{e_1\}, r_1) \cap P(\{e_2\}, r_2) \cap P(\{e_3\}, r_3)$	Who is the director of *Inception*, *Dunkirk*, and *Tenet*?
2u	$P(\{e_1\}, r_1) \cup P(\{e_2\}, r_2)$	Who starred in *Inception* or *Titanic*?
ip	$P(P(\{e_1\}, r_1) \cap P(\{e_2\}, r_2), r_3)$	What is the nationality of the actor who starred in both *Inception* and *Titanic*?
pi	$P(P(\{e_1\}, r_1), r_2) \cap P(\{e_2\}, r_3)$	Who was born in a city located in the *UK* and also starred in *Inception*?
up	$P(P(\{e_1\}, r_1) \cup P(\{e_2\}, r_2), r_3)$	What awards were won by movies directed by *Nolan* or *Cameron*?

These results suggest an important evaluation gap: strong performance on current leaderboards may overestimate a model's ability to generalize beyond path-centric reasoning to branch-sensitive logical structures. The classification criteria and counting protocol used in the structural audit are detailed in Appendix B.

To address this gap, we introduce **ComplexLogicalQA (CLQ)**, a diagnostic benchmark designed to explicitly balance nine EPFO query structures in the mainstream KGQA setting. Our goal is not simply to generate more synthetic KGQA data, but to isolate *logical structure* as a primary evaluation axis. CLQ is constructed through structure-constrained subgraph sampling, followed by LLM-based verbalization and multi-stage verification to preserve executable logical validity while improving linguistic diversity.

Table 2: Distribution of logical structures in 1,000 randomly sampled questions from each benchmark.

Dataset	1p	2p	3p	2i	3i	ip	pi	2u/up
WebQSP	649	291	0	45	0	7	8	0
CWQ	55	504	201	182	8	7	43	0
GrailQA	784	127	22	36	20	9	2	0

Using CLQ, we evaluate representative methods from four KGQA paradigms: direct LLM reasoning (Wei et al., 2022; Wang et al., 2023), retrieval-based methods (Mavromatis et al., 2025), agent-based methods (Sun et al., 2024; Xu et al., 2024), and semantic parsing methods (Huang et al., 2025). We observe a clear and consistent pattern: performance degrades substantially as query structures move beyond chain-like projections, with the strongest drops on nested compositions. Union structures are not uniformly hardest under every metric, but they particularly expose weaknesses in complete answer-set recovery. These trends remain visible even after controlling for multiple observable non-structural factors, supporting the view that current KGQA systems remain highly sensitive to logical structure. Further diagnostic analyses reveal paradigm-specific failure modes, including incomplete branch-level evidence recovery, unstable branch preservation during graph exploration, and imperfect grounding of executable logical forms.

In summary, our contributions are as follows:

- We conduct a structural audit of mainstream KGQA benchmarks and quantify their strong bias toward chain-like query structures, with limited coverage of branching and nested logical forms.
- We introduce **ComplexLogicalQA**, a structure-balanced *diagnostic* benchmark covering nine EPFO logical structures, constructed through structure-constrained sampling, LLM-based verbalization, and multi-stage verification.
- We provide a structure-conditioned empirical study across four KGQA paradigms, showing that performance degrades most consistently on nested compositions and that union structures particularly stress complete answer-set recovery.
- We further use CLQ as a diagnostic testbed to reveal paradigm-specific failure modes in evidence acquisition, graph exploration, and logical-form grounding.

2 Related Work

2.1 KGQA Methods

Existing KGQA methods can be broadly grouped into four paradigms. Direct LLM reasoning methods leverage the parametric knowledge of models such as GPT-5 and DeepSeek-V3 (OpenAI, 2025; DeepSeek-AI et al., 2024), often with prompting strategies such as Chain-of-Thought (CoT) (Wei et al., 2022) and Self-Consistency (Wang et al., 2023). However, without explicit grounding in an external knowledge graph, these methods remain vulnerable to hallucination.

Retrieval-based methods mitigate this issue by retrieving subgraph evidence, ranging from sparse and dense retrieval to graph-aware frameworks such as G-Retrieval (He et al., 2024) and BYOKG-RAG (Mavromatis et al., 2025). Yet retrieval remains difficult when logically relevant evidence is distributed across multiple branches, as in union and nested queries. Agent-based methods, such as Think-on-Graph (Sun et al., 2024), Generate-on-Graph (Xu et al., 2024), and Plan-on-Graph (Chen et al., 2024), ground reasoning through graph exploration, but may suffer from pruning errors that discard valid low-probability branches. Semantic parsing methods such as ChatKBQA (Luo et al., 2024) and TARGA (Huang et al., 2025) instead translate questions into executable logical forms, offering precise execution when parsing succeeds, but becoming less reliable as compositional structure becomes more complex.

2.2 KGQA Benchmarks and Complex Query Answering

KGQA benchmarks have evolved from early single-relation settings such as WebQuestions (Berant et al., 2013) to multi-hop datasets such as WebQSP (Yih et al., 2016) and ComplexWebQuestions (Talmor & Berant, 2018), with later benchmarks including GrailQA (Gu et al., 2021), MarkQA (Huang et al., 2023), and M3GQA (Peng et al., 2025). These datasets have substantially advanced evaluation for path-based reasoning, but branching and nested logical structures remain limited in mainstream benchmarks.

A complementary line of work studies synthetic or semi-synthetic KGQA construction, including GraphQuestions (Su et al., 2016), KQAPro (Cao et al., 2022), and KGQAGen (Zhang et al., 2025). Our work is related in spirit but different in goal: CLQ is not proposed as a larger synthetic dataset or a general-purpose generation framework, but as a diagnostic benchmark that explicitly balances nine EPFO structures in a natural-language KGQA setting. Our work is also closely related to complex query answering (CQA) over knowledge graphs (Ren et al., 2020; Bai et al., 2023). Unlike KGQA, CQA typically provides models with symbolic graph queries directly, rather than natural-language questions, and therefore evaluates logical query answering without testing the language-to-graph mapping. In addition, many CQA queries are generated from graph topology rather than realistic information needs, so their natural-language interpretation is often weak or underspecified. CLQ complements CQA by bringing explicit EPFO structure control into end-to-end natural-language KGQA. Appendix A provides an expanded discussion of related work.

3 Preliminaries

We formalize KGQA in terms of Existential Positive First-Order (EPFO) logic queries.

3.1 Knowledge Graph and Logical Queries

A knowledge graph is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{E})$, where $\mathcal{V}$ is the set of entities, $\mathcal{R}$ is the set of binary relations, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{R} \times \mathcal{V}$ is the set of factual triples. A triple $(s, r, o) \in \mathcal{E}$ indicates that relation r holds between subject entity s and object entity o. A logical query q defines an answer set $[\![q]\!] \subseteq \mathcal{V}$ consisting of entities that satisfy the associated logical constraint.

3.2 Atomic Operators and Logical Structures

Following established formulations (Ren et al., 2020), we express KGQA queries using three atomic operators: **Projection** (P), which retrieves adjacent entities; **Intersection** (I), representing conjunction ($\cap$); and **Union** (U), representing disjunction ($\cup$). We focus on Existential Positive

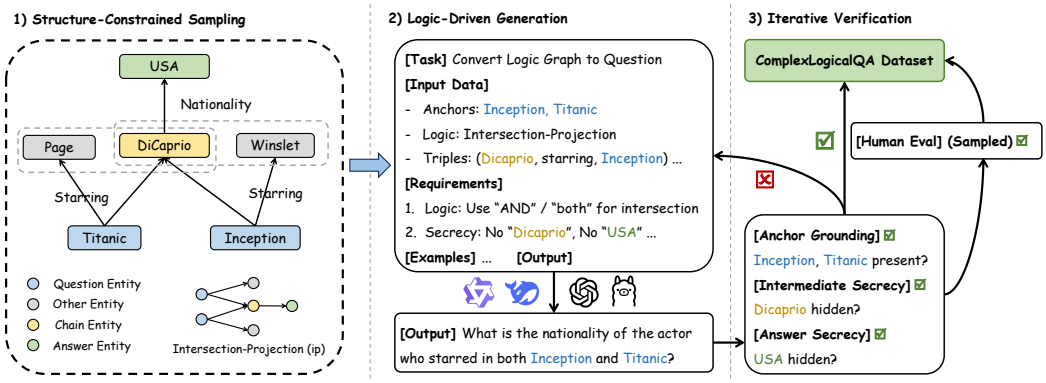


Figure 1: **Data Construction Pipeline.** (1) **Structure-Constrained Sampling:** Subgraphs are extracted based on target logical topologies and validity constraints. (2) **Logic-Driven Generation:** LLMs translate structured triples into natural language questions under strict constraints. (3) **Iterative Verification:** A multi-stage quality control loop combining symbolic checks and automated auditing.

First-Order (EPFO) logic and exclude negation, since under the Open World Assumption of knowledge graphs, incomplete facts make negative gold standards unreliable. By composing these operators, we define nine logical structures (Table 1): chain queries ($1p, 2p, 3p$), intersection queries ($2i, 3i$), union queries ($2u$), and nested queries (ip, pi, up).

3.3 Instance Validity for CLQ

A CLQ instance consists of a logical query, its answer set, a supporting subgraph, and a natural-language question that verbalizes the query. To serve as a valid diagnostic instance, it must satisfy four requirements. **(1) Structure validity:** the sampled subgraph must match the target EPFO topology. **(2) Branch indispensability:** for branching or nested structures, each branch must contribute non-redundantly to the final answer set, so that the instance cannot be reduced to a simpler effective query. **(3) Executable consistency:** the gold answers must be obtained by executing the underlying logical query on the KG. **(4) Surface-form validity:** the natural-language question must preserve anchor grounding while hiding shortcut cues such as intermediate entities or answer names. Section 4 describes how these requirements are enforced during data construction.

4 Data Construction

To systematically evaluate structure-sensitive reasoning in KGQA, we construct **ComplexLogicalQA (CLQ)** through a validity-constrained reverse-generation pipeline. Rather than simply sampling structurally diverse subgraphs, the pipeline is designed to satisfy the instance validity requirements defined in Section 3. As illustrated in Figure 1, it consists of three stages: structure-constrained sampling, logic-to-language verbalization, and multi-stage verification.

4.1 Anchor-Centric Structure Sampling

We collect anchor entities from mainstream KGQA benchmarks, including WebQSP, CWQ, and GrailQA, and use their local neighborhoods as realistic seeds for structure-constrained sampling. For each target query type, we first construct a candidate subgraph that matches the intended EPFO topology, and then enforce additional validity constraints to ensure that the resulting instance is diagnostically meaningful rather than merely structurally diverse.

Intersections ($2i, 3i$). For intersection queries, we adopt target-driven backward traversal. Starting from a candidate answer entity $e_{\text{ans}} \in \mathcal{V}$, we trace incoming relations to recover anchor entities whose projected neighborhoods intersect at e_{ans}. Beyond requiring the branch result sets to be non-identical, we enforce *branch indispensability*: letting A_j denote the result set of branch j and $I = \bigcap_j A_j$ the

Table 3: Detailed statistics of the ComplexLogicalQA Dataset. The dataset features high diversity in query length, answer set size, and the number of reasoning triples required.

Metric	1p	2p	3p	2i	3i	ip	pi	2u	up	Avg/Total
Avg. Q. Tokens	9.6	14.3	18.4	14.2	18.1	19.1	19.2	16.6	19.9	16.6
Avg. Ans. Entities	1.66	3.09	3.10	1.45	1.18	1.76	1.28	5.60	1.49	2.28
Avg. Triples	1.67	6.56	10.70	2.89	3.53	4.79	3.55	7.05	12.74	5.94
Unique Entities	371	858	719	664	795	959	805	1,466	1,614	5,764

final intersection, we require $I \subset A_j$ for every branch j. This ensures that each branch strictly reduces the candidate answer space rather than being redundant.

Unions ($2u$). For union queries, we construct disjunctive branches anchored at semantically related entities and relations, and require the participating entities to remain within a coherent high-level domain. Let A_1 and A_2 denote the answer sets of the two branches. We require $A_1 \setminus A_2 \neq \emptyset$ and $A_2 \setminus A_1 \neq \emptyset$, so that each branch contributes branch-unique answers and the union cannot collapse into a single dominant branch.

Nested Structures (ip, pi, up). Nested structures are constructed by extending valid branching subgraphs. Specifically, for ip queries, we project from the intermediate result set of a valid intersection. For pi queries, we intersect a projected branch with another branch, ensuring the final intersection remains irreducible to either side alone. For up queries, we first form a valid union—where each branch contributes unique predecessors—and then apply a projection to the combined result set. To guarantee structural integrity, we validate all nested instances at both the predecessor-set and final answer-set levels, and explicitly filter out overly generic relations (e.g., `common.topic.description`) to maintain reasoning quality.

4.2 Question Verbalization

We use LLMs to convert each sampled logical query into a natural-language question. Instead of relying on rigid templates, we provide the generator with the target logical structure (e.g., *Intersection-Projection*), the anchor entities, and the supporting triples associated with the executable query instance. The prompt also specifies surface-form constraints, including the requirement to preserve anchor grounding while avoiding shortcut cues such as answer names or intermediate bridge entities. In this way, verbalization is guided not only by local graph facts but also by the intended reasoning pattern of the underlying EPFO structure.

4.3 Multi-Stage Verification and Refinement

Each candidate instance is validated in three stages. **First, we perform symbolic validation.** The sampled subgraph must match the target EPFO topology, the gold answer set must be obtained by executing the underlying logical query on the KG, and the branch-indispensability constraints described in Section 4 must be satisfied. This stage ensures that the instance is structurally valid and cannot be reduced to a simpler effective query.

Second, we enforce surface-form validity. Each generated question must satisfy three constraints: **(1) Anchor Grounding**, which requires the question to explicitly mention the anchor entities (e.g., *Inception*, *Titanic*) to preserve traversability; **(2) Answer Secrecy**, which prohibits canonical names or aliases of final answers to avoid string-matching shortcuts; and **(3) Intermediate Secrecy**, which hides bridge entities in the reasoning chain (e.g., *DiCaprio* in an ip query) so that the answer cannot be recovered without executing the intended reasoning process.

Third, we apply linguistic auditing and refinement. A secondary LLM evaluates whether the question is natural, unambiguous, and grammatically well-formed while remaining faithful to the underlying logical query. If a candidate is judged awkward or ambiguous, it is sent back for regeneration with targeted feedback. This refinement loop continues for up to three attempts, after which failed samples are discarded.

4.4 Dataset Statistics

Table 3 presents detailed statistics of the ComplexLogicalQA dataset (1,000 samples per type; 9,000 in total). Question complexity varies substantially across logical structures. For example, $3p$ and up involve longer reasoning paths, reflected in higher average token counts and a larger number of supporting triples. Union queries ($2u$) also yield substantially larger answer sets (5.60 entities on average) than intersection queries such as $3i$ (1.18), posing distinct challenges for precision and recall. We assess the final dataset quality through expert human evaluation, detailed results are reported in Section 5.5 and Appendix C

5 Experiments

This section evaluates CLQ from three complementary perspectives. **RQ1** asks how representative KGQA methods perform across different logical structures. **RQ2** examines whether the observed performance gaps persist after controlling for correlated non-structural factors. **RQ3** investigates the failure mechanisms behind these gaps across different paradigms and their implications for future KGQA system design.

5.1 Experimental Setup

Knowledge Base. Following prior KGQA benchmarks and methods, we use Freebase (Bollacker et al., 2008) as the underlying knowledge base for CLQ, ensuring consistency with existing datasets such as WebQSP, CWQ, and GrailQA.

Data Split. ComplexLogicalQA contains 9,000 queries, balanced across the nine logical structures with 1,000 instances per type. To evaluate generalization to unseen entities, we adopt an entity-disjoint split: anchor entities appearing in the validation or test sets do not appear as anchors in the training set. The dataset is partitioned into training, validation, and test sets with a 7:1:2 ratio.

Baselines. We evaluate representative methods from four paradigms: (1) Direct LLM reasoning: GPT-3.5 (Achiam et al., 2023), DeepSeek-V3 (DeepSeek-AI et al., 2024), and GPT-5 (OpenAI, 2025), augmented with CoT (Wei et al., 2022) and Self-Consistency (SC) (Wang et al., 2023); (2) Retrieval-based: Sparse (BM25), Dense Retrieval, and BYOKG-RAG (Mavromatis et al., 2025); (3) Agent-based: ToG (Sun et al., 2024), GoG (Xu et al., 2024), PoG (Chen et al., 2024), and GCR (Luo et al., 2025); and (4) Semantic Parsing: TARGA (Huang et al., 2025) and ChatKBQA (Luo et al., 2024).

We make two benchmark-specific adaptations. For supervised methods (e.g., GCR and ChatKBQA), we train on the CLQ training split and select checkpoints on the validation split. For methods that use few-shot demonstrations or in-context exemplars (e.g., CoT, SC, ToG, GoG, and PoG), we replace the original examples with CLQ training examples while preserving the original core reasoning procedure. Detailed implementation settings and benchmark-specific adaptations for all baselines are provided in Appendix D.

Metrics. We report **Hit@1 (H@1)** and **F1** to assess single-answer correctness and complete answer set retrieval, respectively.

Implementation Details. For data construction, we use `gpt-5-2025-08-07` as the primary generator and `gemini-2.5-pro` as an auditor for logical consistency and linguistic fluency. In the evaluation, `gpt-3.5-turbo` serves as the default backbone for prompting-based baselines (including CoT and SC) unless otherwise specified. For dense retrieval, we use `all-MiniLM-L12-v2` as the embedding model. We further examine the robustness of CLQ to different verbalization models and baseline models in Appendix E

5.2 Main Results

Table 4 reports performance on CLQ across nine logical structures and four KGQA paradigms. Because these paradigms differ substantially in training and inference protocol, we use this comparison

Table 4: Performance on ComplexLogicalQA. **Bold** and underlined values denote the best and second-best results, respectively. Double vertical lines separate structure-specific scores from the overall average.

Method	1p		2p		3p		2i		3i		ip		pi		2u		up		Avg.	
	H@1	F1	H@1	F1	H@1	F1	H@1	F1	H@1	F1	H@1	F1	H@1	F1	H@1	F1	H@1	F1	H@1	F1
LLM-based Reasoning																				
GPT-3.5-Turbo	0.36	0.29	0.30	0.18	0.18	0.13	0.28	0.21	0.26	0.22	0.19	0.14	0.26	0.22	0.37	0.14	0.18	0.15	0.26	0.18
DeepSeek-V3	0.37	0.29	0.29	0.17	0.19	0.13	0.25	0.19	0.21	0.17	0.20	0.14	0.27	0.22	0.40	0.16	0.20	0.16	0.26	0.18
GPT-5	0.40	0.32	0.35	0.20	0.21	0.17	0.31	0.28	0.29	0.27	0.25	0.17	0.33	0.30	0.49	0.21	0.22	0.17	0.31	0.23
CoT	0.40	0.32	0.36	0.21	0.20	0.15	0.34	0.30	0.33	0.30	0.29	0.20	0.37	0.33	0.50	0.19	0.22	0.18	0.33	0.24
SC	0.41	0.33	0.36	0.21	0.21	0.18	0.31	0.29	0.30	0.27	0.26	0.17	0.35	0.31	0.50	0.20	0.21	0.18	0.32	0.23
Retrieval-based																				
Sparse (BM25)	0.67	0.63	0.53	0.47	0.31	0.28	0.47	0.46	0.44	0.42	0.38	0.35	0.37	0.34	0.78	0.40	0.36	0.31	0.47	0.40
Dense (MiniLM)	0.70	0.65	0.57	0.52	0.33	0.31	0.48	0.46	0.46	0.45	0.40	0.37	0.36	0.33	0.77	0.46	0.38	0.36	0.49	0.43
BYOKG-RAG	**0.77**	**0.72**	0.61	0.57	0.35	0.30	0.56	0.53	0.51	0.49	0.47	0.42	0.40	0.37	**0.84**	0.53	0.46	0.40	0.55	0.47
Agent-based																				
ToG	0.75	0.69	0.65	0.60	0.43	0.42	0.57	0.57	0.55	0.54	0.43	0.40	0.38	0.34	0.79	0.43	0.44	0.41	0.55	0.48
GoG	0.71	0.68	0.63	0.57	0.41	0.38	0.57	0.56	0.56	0.55	0.44	0.40	0.40	0.36	0.80	0.45	0.50	0.46	0.55	0.49
PoG	0.74	0.70	0.66	0.61	0.45	0.43	0.59	0.58	0.57	0.56	0.46	0.42	0.42	0.38	0.81	0.47	0.52	**0.48**	0.58	**0.51**
GCR	0.76	0.71	**0.67**	0.62	**0.46**	**0.44**	0.60	0.59	0.58	0.57	**0.47**	**0.43**	**0.43**	**0.39**	0.82	0.49	**0.53**	0.49	**0.59**	**0.52**
Semantic Parsing (SP)																				
TARGA	0.69	0.68	0.64	0.63	0.40	0.39	0.62	0.61	0.60	0.60	0.35	0.31	0.34	0.32	0.75	0.66	0.49	0.47	0.54	0.51
ChatKBQA	0.72	0.70	0.66	**0.65**	0.42	0.41	**0.64**	**0.63**	**0.62**	**0.62**	0.39	0.35	0.37	0.35	0.77	**0.68**	0.52	**0.50**	0.57	0.54

primarily to examine *structure-sensitive trends* rather than to establish a unified ranking. A clear pattern emerges: performance drops as queries move beyond chain-like projections, with nested structures remaining the most consistently challenging across paradigms. This trend is visible in both H@1 and F1, suggesting that multi-branch compositional reasoning remains difficult for current KGQA systems.

A second pattern is that union structures particularly stress complete answer-set recovery. Several methods achieve relatively strong H@1 on *2u*, yet their F1 scores remain much lower, indicating difficulty in recovering all correct answers. Semantic parsing methods, especially ChatKBQA, are comparatively more robust on union queries, whereas nested structures remain challenging across paradigms. Overall, CLQ reveals two complementary weaknesses: nested queries mainly challenge compositional structural reasoning, while union queries more strongly expose incomplete answer-set recovery.

5.3 Controlling for Confounding Factors

A natural alternative explanation for the performance drop on CLQ is that it may be driven not by logical structure itself, but by correlated non-structural difficulty factors. To test this possibility, we perform a stricter matched analysis over three pairwise comparisons: Chain vs. Nested, Chain vs. Union, and Intersection vs. Nested. We control for three factors in total: question length, answer-set size, supporting triple count. We then evaluate F1 on the resulting matched subsets.

To construct the matched subsets, we form strata using these three attributes, retain only strata with support from both sides of each comparison, and downsample the larger side when needed. The final matched subsets contain 198, 161, and 214 paired instances per side for Chain vs. Nested, Chain vs. Union, and Intersection vs. Nested, respectively. In Table 5, *Simpler F1* and *Complex F1* denote the average F1 on the matched subsets from the simpler and more complex side of each comparison, respectively, and *Gap* is defined as their difference. Although the matched subsets are fixed after construction, we estimate 95% confidence intervals for the gap via bootstrap resampling to quantify the stability of the observed difference.

The matched analysis reveals a consistent pattern. Chain remains stronger than Nested across all representative methods, and Intersection also outperforms Nested, suggesting that the difficulty of Nested structures is not reducible to branching alone. Union remains weaker than Chain for most methods, although the gap is smaller and semantic parsing is comparatively robust on Union. Taken together, these results reduce several plausible non-structural explanations and support the view that CLQ captures a robust structure-sensitive difficulty trend.

Table 5: Matched analysis under stronger controls. We compare matched subsets from three pairwise structure families.

Comparison	Method	Simpler F1	Complex F1	Gap	95% CI
Chain vs. Nested	GPT-5	0.26	0.19	0.07	[0.04, 0.10]
	BYOKG-RAG	0.56	0.41	0.15	[0.11, 0.19]
	ToG	0.58	0.39	0.19	[0.15, 0.23]
	GCR	0.61	0.45	0.16	[0.12, 0.20]
	ChatKBQA	0.60	0.42	0.18	[0.14, 0.22]
Chain vs. Union	GPT-5	0.25	0.21	0.04	[0.01, 0.07]
	BYOKG-RAG	0.55	0.49	0.06	[0.02, 0.10]
	ToG	0.57	0.44	0.13	[0.09, 0.17]
	GCR	0.60	0.50	0.10	[0.06, 0.14]
	ChatKBQA	0.59	0.61	-0.02	[-0.05, 0.01]
Intersection vs. Nested	GPT-5	0.28	0.20	0.08	[0.05, 0.11]
	BYOKG-RAG	0.52	0.41	0.11	[0.07, 0.15]
	ToG	0.56	0.39	0.17	[0.13, 0.21]
	GCR	0.58	0.45	0.13	[0.09, 0.17]
	ChatKBQA	0.63	0.42	0.21	[0.17, 0.25]

5.4 Failure Mechanisms and Design Implications

While Table 4 shows that all paradigms degrade on structurally complex queries, answer-level metrics alone do not explain why these gaps arise. We therefore use CLQ as a diagnostic testbed to study failure mechanisms across paradigms, focusing on evidence acquisition, graph exploration, and logical-form grounding. We analyze Dense Retrieval and BYOKG-RAG as retrieval-based methods, ToG as an agent-based method, and TARGA as a semantic parsing method. For Dense Retrieval, BYOKG-RAG, and ToG, we measure triple-level recall and precision against the gold supporting evidence; for ToG, we additionally quantify intermediate reasoning errors and erroneous pruning; and for TARGA, we manually inspect 50 failed predictions per structure family to assess whether the predicted logical form preserves the correct high-level topology.

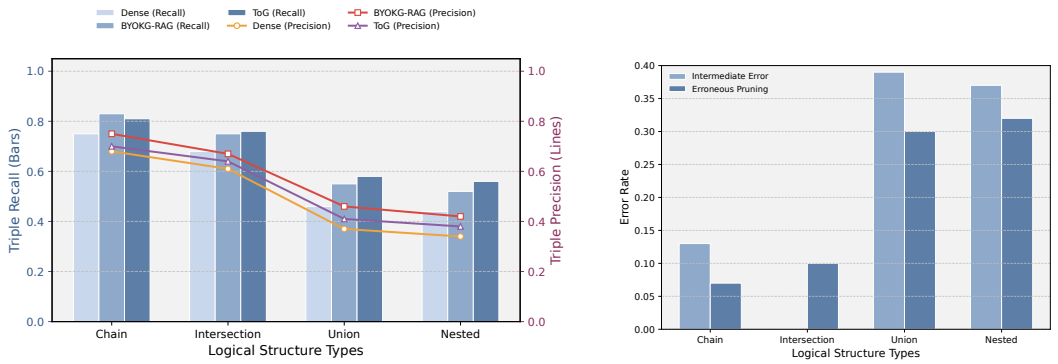
Figure 2a shows that evidence quality declines as logical structures become more compositional. Across Dense Retrieval, BYOKG-RAG, and ToG, both triple recall and precision are lower on branching and nested structures than on chain-like queries, suggesting increasing difficulty in recovering complete relevant evidence while filtering irrelevant branches. Compared with pure retrieval methods, ToG often achieves higher recall through active graph exploration, but at the cost of lower precision, indicating a trade-off between broader search coverage and distractor accumulation.

A more fine-grained analysis reveals distinct failure patterns across paradigms. For ToG, Figure 2b shows that both intermediate reasoning errors and erroneous pruning become more frequent on union and nested structures, suggesting difficulty in preserving valid branches under constrained search. For TARGA, Table 6 shows that many failed cases still preserve the correct high-level topology, especially for chain and intersection queries, whereas union and nested failures more often involve topology violations or errors in relation prediction, entity grounding, or execution. This suggests that semantic parsing often captures coarse structural intent but struggles to instantiate the correct grounded logical form under more compositional structures.

Table 6: Manual diagnosis of TARGA failures by structure family.

Family	Topology Preserved	Topology Violated
Chain	0.62	0.38
Intersection	0.54	0.46
Union	0.41	0.59
Nested	0.36	0.64

Overall, CLQ reveals different structure-sensitive bottlenecks across paradigms. Retrieval-based methods mainly struggle with branch-complete evidence acquisition, agent-based methods with branch-preserving exploration, and semantic parsing methods with reliable grounding and execution.



(a) Supporting evidence quality.

(b) ToG error analysis.

Figure 2: Diagnostic analysis on CLQ. We report supporting-evidence quality for Dense Retrieval, BYOKG-RAG, and ToG, together with ToG's intermediate reasoning errors and erroneous pruning rates. The results suggest distinct structure-sensitive bottlenecks across paradigms.

These findings motivate future work on branch-aware retrieval, more stable graph search, and stronger separation between structural prediction and grounding.

5.5 Dataset Validation

Finally, we validate CLQ from three perspectives: human evaluation, surface-level comparability to existing human-annotated benchmarks, and the filtering profile of the verification pipeline.

Human Evaluation. We evaluate 1,800 stratified instances (200 per logical structure) using three expert annotators and two criteria: Linguistic Fluency and Logical Consistency. The evaluation achieves 98.0% inter-annotator agreement, and 96.0% of sampled instances are judged acceptable overall. The overall pass rate remains high across structure families: 98.0% for Chain, 96.3% for Intersection, 94.5% for Nested, and 93.8% for Union. Detailed results are provided in Appendix Table 9.

Table 7: Surface-level comparison between CLQ and human-annotated KGQA benchmarks.

Dataset	Avg. Len.	Ent. Mentions	Dominant WH
WebQSP	10.8	1.18	what / who
CWQ	15.9	1.64	what / which
GrailQA	16.8	1.71	what / which
CLQ	16.6	1.69	what / which

Surface-Level Comparability. As shown in Table 7, CLQ is close to CWQ and GrailQA in average question length, entity mention count, and dominant WH-word usage, suggesting that it remains broadly comparable in surface form to existing human-annotated KGQA benchmarks.

Filtering Profile. Starting from 15,803 candidate samples, the pipeline removes 1,580 samples that violate anchor grounding, 1,195 that fail answer secrecy, and 861 that breach intermediate secrecy. The final LLM-based auditor filters out another 3,167 ambiguous or logically mismatched questions, leaving 9,000 retained instances. This indicates that CLQ is produced through successive structural and linguistic verification rather than unconstrained generation.

6 Conclusion

We introduce **ComplexLogicalQA** (CLQ), a structure-balanced KGQA benchmark covering nine EPFO query structures. CLQ enables systematic evaluation of branching and nested reasoning beyond chain-like queries. Across four KGQA paradigms, we observe consistent structure-sensitive performance degradation, with nested and union queries revealing key reasoning limitations.

AI USE STATEMENT

In this work, we used generative AI tools for assisting the design and refinement of research methodology, experimental analysis, data generation and verification procedures, and the preparation and editing of the manuscript. Specifically, generative AI models were used as part of the question verbalization pipeline for constructing natural-language questions from structured logical queries. We also used generative AI tools to assist with brainstorming, literature exploration, and improving the readability and organization of the manuscript.

We have not used generative AI tools to generate mathematical proofs, formulate unsupported scientific claims, or replace human evaluation and verification. All AI-assisted outputs, including generated questions, experimental analyses, and manuscript edits, were reviewed by the authors. Generated data samples were validated through the proposed verification pipeline, and experimental results and conclusions were independently checked by the authors. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses publicly available knowledge graph resources and existing KGQA datasets for benchmark construction, following their respective usage policies and licenses. The natural-language questions generated in CLQ are produced with the assistance of generative AI models and validated through automatic verification and quality control procedures. This work does not involve human subjects, private or personally identifiable information, or sensitive data. We acknowledge that knowledge graphs and language models may contain biases or incomplete information, and we take responsibility for the released benchmark, including its documentation, data processing procedures, and potential limitations.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the CLQ construction pipeline, including structure-constrained sampling, question verbalization, and multi-stage verification. The logical structure definitions, annotation criteria, and benchmark analysis procedures are documented in the appendix. Experimental settings, baseline implementations, evaluation protocols, and additional robustness analyses are also provided in the supplementary materials. We will release the benchmark data, source code, and necessary scripts to facilitate independent verification and future research.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Yushi Bai, Xin Lv, Juanzi Li, and Lei Hou. Answering complex logical queries on knowledge graphs via query computation tree optimization. In *International Conference on Machine Learning*, pp. 1472–1491. PMLR, 2023.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1533–1544, 2013.

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

Shulin Cao, Jiaxin Shi, Liangming Pan, Lunyiu Nie, Yutong Xiang, Lei Hou, Juanzi Li, Bin He, and Hanwang Zhang. KQA pro: A dataset with explicit compositional programs for complex question answering over knowledge base. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6101–6119, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.422. URL https://aclanthology.org/2022.acl-long.422/.

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 *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=CwCUEr6wO5.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. Deepseek-v3 technical report, 2024. URL https://arxiv.org/abs/2412.19437.

Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. Beyond i.i.d.: Three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference 2021*, WWW '21, pp. 3477–3488, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383127. doi: 10.1145/3442381.3449992. URL https://doi.org/10.1145/3442381.3449992.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 132876–132907. Curran Associates, Inc., 2024. doi: 10.52202/079017-4224. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/efaf1c9726648c8ba363a5c927440529-Paper-Conference.pdf.

Xiang Huang, Sitao Cheng, Yuheng Bao, Shanshan Huang, and Yuzhong Qu. MarkQA: A large scale KBQA dataset with numerical reasoning. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 10241–10259, Singapore, December 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.emnlp-main.633/.

Xiang Huang, Jiayu Shen, Shanshan Huang, Sitao Cheng, Xiaxia Wang, and Yuzhong Qu. TARGA: Targeted synthetic data generation for practical reasoning over structured data. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2704–2726, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.137. URL https://aclanthology.org/2025.acl-long.137/.

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.

Haoran Luo, E Haihong, Zichen Tang, Shiyao Peng, Yikai Guo, Wentai Zhang, Chenghao Ma, Guanting Dong, Meina Song, Wei Lin, et al. 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*, pp. 2039–2056, 2024.

Linhao Luo, Zicheng Zhao, Gholamreza Haffari, Yuan-Fang Li, Chen Gong, and Shirui Pan. Graph-constrained reasoning: Faithful reasoning on knowledge graphs with large language models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Fr7kH2SFq7.

Costas Mavromatis, Soji Adeshina, Vassilis N. Ioannidis, Zhen Han, Qi Zhu, Ian Robinson, Bryan Thompson, Huzefa Rangwala, and George Karypis. BYOKG-RAG: Multi-strategy graph retrieval for knowledge graph question answering. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 27869–27886, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1417. URL https://aclanthology.org/2025.emnlp-main.1417/.

OpenAI. GPT-5 system card. System card, OpenAI, August 2025. URL https://cdn.openai.com/gpt-5-system-card.pdf.

Boci Peng, Yongchao Liu, Xiaohe Bo, Jiaxin Guo, Yun Zhu, Xuanbo Fan, Chuntao Hong, and Yan Zhang. M^{3}GQA: A multi-entity multi-hop multi-setting graph question answering benchmark. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 30594–30620, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1478. URL https://aclanthology.org/2025.acl-long.1478/.

Hongyu Ren, Weihua Hu, and Jure Leskovec. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJgr4kSFDS.

Yu Su, Huan Sun, Brian Sadler, Mudhakar Srivatsa, Izzeddin Gür, Zenghui Yan, and Xifeng Yan. On generating characteristic-rich question sets for QA evaluation. In Jian Su, Kevin Duh, and Xavier Carreras (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 562–572, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1054. URL https://aclanthology.org/D16-1054/.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=nnVO1PvbTv.

Alon Talmor and Jonathan Berant. The web as a knowledge-base for answering complex questions. In Marilyn Walker, Heng Ji, and Amanda Stent (eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 641–651, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1059. URL https://aclanthology.org/N18-1059/.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

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

Yao Xu, Shizhu He, Jiabei Chen, Zihao Wang, Yangqiu Song, Hanghang Tong, Guang Liu, Jun Zhao, and Kang Liu. Generate-on-graph: Treat LLM as both agent and KG for incomplete knowledge graph question answering. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 18410–18430, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1023. URL https://aclanthology.org/2024.emnlp-main.1023/.

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 *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 201–206, 2016.

Liangliang Zhang, Zhuorui Jiang, Hongliang Chi, Haoyang Chen, Mohammed ElKoumy, Fali Wang, Qiong Wu, Zhengyi Zhou, Shirui Pan, Suhang Wang, and Yao Ma. Diagnosing and addressing pitfalls in KG-RAG datasets: Toward more reliable benchmarking. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025. URL https://openreview.net/forum?id=Vd5JXiX073.

A Extended Related Work

A.1 KGQA Benchmarks and Structural Coverage

Knowledge graph question answering (KGQA) benchmarks have progressed from early single-relation settings such as WebQuestions Berant et al. (2013) to multi-hop datasets such as WebQSP Yih et al. (2016) and ComplexWebQuestions Talmor & Berant (2018). More recent benchmarks, including GrailQA Gu et al. (2021), MarkQA Huang et al. (2023), and M3GQA Peng et al. (2025), further broaden the evaluation landscape by emphasizing compositional generalization, schema variation, numerical reasoning, or more diverse multi-entity settings. Collectively, these datasets have substantially improved the evaluation of end-to-end KGQA.

At the same time, most mainstream KGQA benchmarks were not designed to explicitly balance logical structure families under an EPFO view. As a result, their effective coverage is often concentrated on projection-dominant or chain-like reasoning, while branching and especially nested compositions receive much weaker representation. This does not diminish their value as realistic QA benchmarks; rather, it highlights a difference in evaluation objective. Existing benchmarks primarily measure overall end-to-end KGQA performance under natural question distributions, whereas CLQ is designed as a diagnostic benchmark that controls logical structure as a first-class variable.

A.2 Synthetic and LLM-Assisted KGQA Data Construction

A parallel line of work studies how to construct KGQA benchmarks with broader compositional coverage. Earlier efforts such as GraphQuestions Su et al. (2016) derive questions from graph structure through reverse or graph-guided construction, while later datasets such as KQAPro Cao et al. (2022) introduce explicit compositional programs for more controlled supervision. More recent work, such as KGQAGen Zhang et al. (2025), further explores automatic or semi-automatic generation pipelines for diagnosing model behavior under richer compositional settings.

Across these efforts, a central challenge is balancing structural controllability with natural-language quality. Instances derived directly from graph structure or executable programs are easier to verify formally, but they are also harder to keep natural, semantically coherent, and comparable to human-authored KGQA questions. Recent LLM-based pipelines help address this issue by improving question fluency, paraphrasing, filtering, and auditing, but they also introduce risks such as semantic drift, answer leakage, and mismatch between surface form and executable logic. For this reason, effective benchmark construction increasingly combines LLM generation with symbolic constraints and executable verification.

CLQ is most closely related to this line of work. However, unlike prior synthetic or semi-synthetic KGQA datasets, our goal is not to maximize scale or provide general-purpose program supervision, but to build a diagnostic benchmark with explicitly balanced EPFO structures in a natural-language KGQA setting. This requires structure-constrained sampling together with verbalization and verification procedures that preserve anchor grounding, executable consistency, and shortcut resistance.

A.3 Complex Query Answering and Its Relation to KGQA

Our work is also closely related to complex query answering (CQA) over knowledge graphs Ren et al. (2020); Bai et al. (2023). In current practice, widely used CQA benchmarks are typically built on knowledge graphs such as FB15k, FB15k-237, and NELL995, and evaluate query families such as 1p, 2p, 3p, 2i, 3i, ip, pi, 2u, and up. In these benchmarks, queries are instantiated automatically from predefined logical structures over the graph, rather than written as natural-language questions.

Consequently, the input in CQA is typically a symbolic logical query. For example, Query2Box illustrates CQA with a conjunctive query corresponding to "Where did Canadian citizens with Turing Award graduate?", while Query2Particles presents examples directly in predicate-logic form, such as

$$q_1 = V?.\exists V : \mathrm{Win}(\mathrm{TuringAward}, V) \wedge \neg \mathrm{Citizen}(\mathrm{Canada}, V) \wedge \mathrm{Graduate}(V, V?),$$

with the gloss "Find where the non-Canadian Turing award laureates graduated from." Importantly, such natural-language sentences are explanatory glosses for readers; the benchmark input itself remains the logical form or query graph, where anchor entities, relations, and logical operators are already explicit.

This differs fundamentally from natural-language KGQA. In KGQA, the system must recover the underlying formalization from the question itself: it must identify the anchor entities, infer the intended relations, determine the logical composition, and then execute the corresponding query. CQA therefore isolates logical query answering once the formal structure is given, whereas KGQA evaluates the full language-to-graph reasoning pipeline.

CQA also differs from KGQA in semantic grounding. Because many CQA benchmarks are generated directly from graph topology and predefined query structures, they primarily guarantee structural validity and executable semantics on the graph, rather than realistic human question intent. In other words, many CQA instances are meaningful as graph queries, but not necessarily as naturally occurring user questions. CLQ is intended to bridge this gap: like CQA, it explicitly controls logical structure, but like mainstream KGQA, it evaluates end-to-end reasoning from natural-language questions rather than from symbolic queries alone.

B Structural Analysis of Existing Benchmarks

To quantify structural bias, we analyzed 3,000 sampled questions using a dual-track protocol. For WebQSP and CWQ (SPARQL), three expert annotators manually categorized the logical structures. They cross-verified the results, achieving a 95% inter-annotator agreement, ensuring high reliability. For GrailQA (S-expressions), we used a deterministic parsing script to map function calls to our topological definitions. As shown in Table 2, WebQSP and GrailQA are dominated by linear paths ($> 90\%$), while CWQ lacks nested logic. Crucially, Union operations ($2u, up$) are completely absent (0%) in all baselines. In contrast, our ComplexLogicalQA strictly balances all nine logical types. This appendix details the classification criteria and mapping rules used for this analysis.

B.1 WebQSP and CWQ (Human Annotation)

Since WebQSP and CWQ provide ground truth in SPARQL format, structural identification requires analyzing the graph topology implied by the query. We instructed annotators to visualize the SPARQL query as a computation graph and classify it according to the following taxonomy:

- **Chain Structures ($1p, 2p, 3p$):** Defined by a sequence of triple patterns where the object of one triple serves as the subject of the next (e.g., $?y \leftarrow (e, r_1, ?x) \wedge (?ans \leftarrow (?x, r_2, ?y))$ is classified as $2p$).
- **Intersection ($2i, 3i$):** Identified by multiple triple patterns sharing the same variable as the object, representing a conjunction of constraints (e.g., $?ans \leftarrow (?x, r_1, ?ans) \wedge (?y, r_2, ?ans)$).
- **Nested Logic (ip, pi, up):** Defined as a chain structure where one node is further constrained by an intersection (e.g., $?ans$ is the object of a chain, and the intermediate node is an intersection of two other entities).
- **Union ($2u$):** Identified by the presence of the `UNION` keyword in SPARQL, connecting disjoint graph patterns that map to the same answer variable.

Handling Ambiguity and Topological definitions. To ensure structural precision, we enforced three rigorous classification protocols during the manual annotation of WebQSP and CWQ:

- **CVT Node Expansion:** We strictly adhere to the physical graph topology regarding Freebase's Compound Value Types (CVT). A traversal passing through a CVT node is counted as two distinct hops (e.g., Entity $\xrightarrow{r_1}$ CVT $\xrightarrow{r_2}$ Value is classified as $2p$), rather than semantically collapsing it into a single relation. This prevents underestimating the reasoning depth required by the model.
- **Reasoning Path vs. Shortest Path:** We explicitly reject the common heuristic of categorizing questions based solely on the shortest path distance between anchor and answer entities. This metric is imprecise for two reasons: (1) **Path Mismatch:** The intended semantic logic often follows a specific, longer path (e.g., "spouse's co-star") which may differ from a coincidental direct link found by shortest-path algorithms; and (2) **Extended Verification:** Some questions require reasoning steps that extend *beyond* the candidate answer to verify a

Table 8: Structural distribution of recent KGQA benchmarks. We manually annotate 500 randomly sampled questions from each dataset. "Other" includes higher-order EPFO structures and non-EPFO operations (e.g., counting, comparison, and numerical reasoning).

Dataset	1p	2p	3p	2i	3i	2u	ip	pi	up	Other
KGQAGen	48	7	0	41	93	0	0	0	0	311
M3GQA	17	0	6	18	89	0	37	21	0	312
KQAPro	50	66	7	6	0	0	21	0	0	350

property (e.g., "What actor played in X and [actor] won award Y?"), effectively adding a hop that shortest-path metrics ignore.

- **Intersection vs. Linear Qualifiers:** A common ambiguity in CWQ arises between "Co-occurrence" and "Intersection". We established a strict rule: a query is classified as Intersection ($2i/3i$) only if the constraints explicitly narrow down the candidate set from multiple *independent* branches. Linear qualifiers (e.g., limiting a movie by its release date) are classified as part of the chain ($2p$) rather than a topological intersection.

B.2 GrailQA (Script-Based Parsing)

GrailQA uses S-expressions, which allows for deterministic mapping. We developed a parser that traverses the S-expression tree and counts the nesting depth and operator types. The mapping rules are as follows:

- **JOIN**: Treated as a projection step. A single `JOIN` represents $1p$. Nested `JOIN`s (e.g., `(JOIN r2 (JOIN r1 e))`) increase the path length ($2p, 3p$).
- **AND**: Treated as an Intersection node.
  - If `AND` operates on two 1-hop paths originating from different entities, it is mapped to $2i$.
  - If `AND` is nested within a `JOIN`, it is mapped to nested logic (ip or pi).
- **OR**: Although GrailQA's schema supports logical disjunction theoretically, our script scanned all 1,000 sampled S-expressions and found zero instances of the `OR` operator used for merging distinct reasoning paths (Union), confirming the 0% statistic reported in Table 2.
- **Comparatives (e.g., GT, LT)**: These were excluded from the topological count as they function as filters rather than structural hops.

B.3 Recent KGQA Benchmarks

Recent KGQA benchmarks and generation frameworks have expanded evaluation toward diverse capabilities beyond conventional multi-hop reasoning. To examine whether these benchmarks provide complementary structural coverage, we extend our audit to four recently introduced datasets: KGQAGen Zhang et al. (2025), M3GQA Peng et al. (2025), and KQAPro Cao et al. (2022)

Since these datasets are designed with different objectives, not all questions can be directly mapped to the nine EPFO structures considered in CLQ. Therefore, we manually annotate 500 randomly sampled questions from each dataset using the same structural taxonomy and classification criteria described in Section B.1. Questions involving higher-order EPFO queries (e.g., 4i/5i) or non-EPFO operations, such as counting, comparison, and numerical reasoning, are categorized as "Other".

The audit reveals that recent KGQA datasets expand evaluation along different dimensions, but they still do not provide controlled and balanced coverage of the logical structures studied in CLQ. KGQAGen and M3GQA contain a limited number of intersection and nested queries, but union structures remain absent in the sampled subsets. KQAPro contain a substantial proportion of questions outside the nine EPFO structures, reflecting their emphasis on functional operations and numerical reasoning. Therefore, while these benchmarks provide valuable evaluations of specific capabilities, they do not directly isolate logical structure as a controlled evaluation axis.

C Human Evaluation Protocols

To ensure the high quality of ComplexLogicalQA, we conducted a rigorous human evaluation. This section details the annotator profiles, evaluation metrics, and the specific guidelines provided to the annotators.

C.1 Annotator Profiles

We recruited three graduate students as annotators. All annotators met the following qualifications:

- Advanced proficiency in English.
- Specific research experience in Knowledge Graph Question Answering (KGQA) ensuring familiarity with logical forms such as Intersection and Union.
- Prior experience with the Freebase schema.

C.2 Evaluation Criteria

Annotators were presented with the generated Natural Language Question (NLQ), the underlying Logical Form and Triples, and the Answer Set. They were asked to rate each sample on a Pass/Fail basis according to the following two dimensions:

Criterion 1: Linguistic Fluency. A question is marked as **Pass** if:

- It is grammatically correct and free of typos.
- It reads naturally like a question a human would ask, rather than a robotic translation of a database query.
- It avoids redundancy (e.g., "What is the movie that implies the film...").

Criterion 2: Logical Consistency. A question is marked as **Pass** if and only if:

- **Accuracy:** The question accurately reflects all constraints in the logical form (e.g., if the logic specifies "Intersection," the question may imply "both" or "and").
- **No Hallucination:** The question does not introduce extra constraints not present in the logic (e.g., adding "famous actor" when the logic only specifies "actor").
- **No Ambiguity:** The question has a unique interpretation corresponding to the provided logical structure.

C.3 Evaluation Results

Table 9 presents the detailed breakdown of the human evaluation results. The overall pass rate (samples satisfying both criteria) is 96.0%.

Table 9: Detailed human evaluation results. Complex structures pose higher challenges but maintain a robust pass rate ($> 93\%$).

Logical Structure	Fluency (%)	Consistency (%)	Overall Pass (%)
Chain (1p, 2p, 3p)	99.3	98.7	98.0
Intersection (2i, 3i)	98.6	97.0	96.3
Nested (ip, pi, up)	97.5	95.2	94.5
Union (2u)	96.8	94.5	93.8
Total Average	**98.1**	**96.4**	**96.0**

D Baseline Methodologies and Implementation

To ensure a comprehensive evaluation, we benchmarked methods across four paradigms. For foundational baselines (LLM reasoning and pure retrieval), we implemented customized pipelines optimized for the ComplexLogicalQA setting. For complex architectures (Agents, Semantic Parsing, and Advanced RAG), we reproduced results using their official open-source implementations to ensure fairness and reproducibility.

D.1 Implemented Baselines

Chain-of-Thought (CoT) and Self-Consistency (SC) We utilized gpt-3.5-turbo as the backbone model for these reasoning baselines. For **Chain-of-Thought (CoT)**, we implemented a few-shot prompting strategy. The prompt includes three diverse demonstrations (one linear, one intersection, and one union query) to guide the model in breaking down complex logic step-by-step before generating the final answer. The temperature was set to `0.0` to maximize deterministic output. For **Self-Consistency (SC)**, we sampled $k = 5$ distinct reasoning paths for each query by increasing the temperature to `0.7`. The final answer was determined via majority voting over the generated entity sets, aiming to mitigate stochastic hallucinations common in single-pass generation.

Sparse and Dense Retrieval These methods treat KGQA as a document retrieval task, fetching relevant triples to augment the LLM. **Sparse Retrieval (BM25)**: We linearized the Knowledge Graph triples into text documents and used the BM25 algorithm to rank them based on lexical overlap with the query. We set the parameters $k_1 = 1.2$ and $b = 0.75$, retrieving the top-10 triples to ensure high recall. **Dense Retrieval**: We employed a bi-encoder architecture based on `all-MiniLM-L12-v2`. During inference, we encoded the question and graph triples into dense vectors and retrieved the top-10 semantic matches based on cosine similarity.

D.2 Reproduced Baselines

For the following advanced methods, we reproduced results using their official open-source implementations whenever available, and kept the original inference or training pipelines as unchanged as possible. To ensure a unified evaluation protocol on CLQ, we made only two benchmark-specific adaptations: (1) for methods that require supervised training, we trained them on the **CLQ training split**; and (2) for methods whose official prompting protocol relies on few-shot demonstrations or in-context exemplars, we replaced the original examples with examples sampled from the **CLQ training split**, while preserving the original prompt format, the number of demonstrations, and the core reasoning procedure.

BYOKG-RAG (Retrieval-based). BYOKG-RAG (Mavromatis et al., 2025) is an advanced retrieval framework that combines dense retrieval with graph-aware exploration to fetch relevant subgraphs. When exemplar-based prompting was required, we replaced the original examples with CLQ training examples while preserving the original prompt structure.

Think-on-Graph (ToG), Generate-on-Graph (GoG), and Plan-on-Graph (PoG) (Agent-based). These methods model KGQA as an agentic reasoning and graph exploration process. Think-on-Graph (ToG) (Sun et al., 2024) iteratively searches for evidence paths over the KG with beam search; Generate-on-Graph (GoG) (Xu et al., 2024) uses the LLM to propose and validate reasoning paths grounded in the KG; and Plan-on-Graph (PoG) (Chen et al., 2024) performs explicit planning before graph traversal. For these methods, we followed the official implementations and default hyperparameter settings. When the original prompting protocol required few-shot demonstrations or in-context exemplars, we replaced the original examples with examples drawn from the **CLQ training split**, while keeping the same prompt structure and the same number of demonstrations as in the original setup.

GCR (Agent-based). GCR (Luo et al., 2025) augments graph-constrained reasoning with trainable components for evidence selection and answer prediction. Since GCR requires task-specific training, we trained it on the **CLQ training split** and selected checkpoints based on validation performance on

Table 10: Pilot evaluation of CLQ portability on Wikidata. Results are reported on 10 validated instances per structure and are intended to assess portability rather than establish performance rankings.

Method	1p	2p	3p	2i	3i	2u	ip	pi	up	Avg.
ToG on Wikidata	.67	.59	.44	.60	.55	.43	.42	.47	.41	.51

the CLQ validation split. All other architectural choices and hyperparameters followed the official implementation.

TARGA and ChatKBQA (Semantic Parsing). Semantic parsing methods translate natural-language questions into executable logical forms over the KG. TARGA (Huang et al., 2025) is a few-shot semantic parsing framework that uses targeted synthetic demonstrations to improve robustness to structural variation. We evaluated it using the official implementation and preserved its original prompting and decoding procedure. Following our unified benchmark protocol, we replaced the original in-context examples with examples drawn from the **CLQ training split**, while keeping the prompt format and number of demonstrations unchanged. ChatKBQA (Luo et al., 2024) is a supervised semantic parsing framework for executable KBQA. Since it requires task-specific training, we trained ChatKBQA on the **CLQ training split** and selected checkpoints on the CLQ validation split. Except for adapting the training data to CLQ, we kept the model architecture, decoding procedure, and other hyperparameters the same as in the released implementation.

E Additional Robustness Analysis

E.1 Portability to Wikidata

We use Freebase as the primary knowledge base for CLQ to ensure direct comparability with existing KGQA benchmarks and systems. To assess the portability of our pipeline, we conduct a pilot study on Wikidata by adapting only the knowledge-base-specific components, including relation access and entity mapping. The logical structures, sampling constraints, verbalization procedure, and verification criteria remain unchanged.

After mapping anchors to Wikidata entities, we apply the same construction pipeline and retain 10 validated questions for each of the nine logical structures, resulting in a 90-question pilot set. We evaluate this set with ToG Sun et al. (2024) as a representative KGQA baseline, with results shown in Table 10. This pilot demonstrates that CLQ can be applied to Wikidata with limited schema adaptation, while larger-scale cross-knowledge-base evaluation is left for future work.

E.2 Robustness to Verbalization Models

We evaluate whether the observed CLQ results are sensitive to the choice of verbalization model. Starting from the same 450 logical queries, we generate two matched question sets using GPT-5 and DeepSeek-V4-Pro, respectively. We then evaluate Direct, CoT, Self-Consistency, and ToG using GPT-5 as a common reasoning backbone under the same evaluation protocol.

Across the two verbalization settings, the average F1 difference is only 0.00–0.02, while the relative difficulty pattern across the nine logical structures remains nearly unchanged. These results indicate that the observed structure-sensitive performance trends are robust to the choice of verbalization model.

E.3 Robustness across Reasoning Backbones

To examine whether the observed structure-sensitive patterns depend on a specific reasoning backbone, we further evaluate CLQ using different base models. We compare Direct, Chain-of-Thought (CoT), Self-Consistency (SC), and ToG with GPT-5 and DeepSeek-V4-Pro as the reasoning backbones under the same evaluation protocol.

Table 11: Robustness analysis under different verbalization models. All methods use GPT-5 as the reasoning backbone and are evaluated on matched logical queries generated by different verbalization models.

Generator	Method	1p	2p	3p	2i	3i	ip	pi	2u	up	Avg.
GPT-5	Direct	.36	.28	.20	.31	.29	.22	.25	.20	.19	.26
	CoT	.38	.29	.23	.35	.33	.25	.24	.22	.23	.28
	Self-Consistency	.40	.32	.24	.36	.34	.28	.27	.25	.26	.30
	ToG	.70	.62	.54	.62	.58	.46	.43	.48	.43	.54
DeepSeek-V4-Pro	Direct	.35	.27	.21	.30	.28	.23	.24	.21	.20	.25
	CoT	.37	.28	.22	.34	.32	.26	.25	.24	.24	.28
	Self-Consistency	.39	.31	.23	.37	.35	.27	.26	.23	.25	.30
	ToG	.67	.60	.50	.61	.57	.45	.42	.46	.42	.52

Table 12: Robustness analysis across different reasoning backbones. All methods are evaluated on the same CLQ instances under the same protocol.

Backbone	Method	1p	2p	3p	2i	3i	2u	ip	pi	up	Avg.
GPT-5	Direct	.32	.20	.17	.28	.27	.21	.17	.30	.17	.23
	CoT	.37	.30	.24	.34	.33	.20	.26	.27	.24	.28
	SC	.41	.33	.22	.37	.35	.26	.26	.33	.25	.31
	ToG	.69	.63	.55	.60	.59	.45	.44	.46	.41	.54
DeepSeek-V4-Pro	Direct	.31	.28	.19	.32	.30	.20	.20	.26	.17	.25
	CoT	.38	.31	.25	.35	.34	.21	.26	.23	.20	.28
	SC	.39	.34	.21	.38	.32	.24	.24	.28	.27	.30
	ToG	.71	.60	.54	.61	.55	.47	.45	.41	.46	.53

Across different reasoning backbones, the average F1 difference between GPT-5 and DeepSeek-V4-Pro remains small, while the relative difficulty pattern across the nine logical structures is largely consistent. In particular, both backbones show substantially lower performance on branching and nested structures compared with simple projection queries. These results indicate that the structural performance trends revealed by CLQ are not specific to a particular reasoning backbone.

F Additional Analysis for Confounding Factors

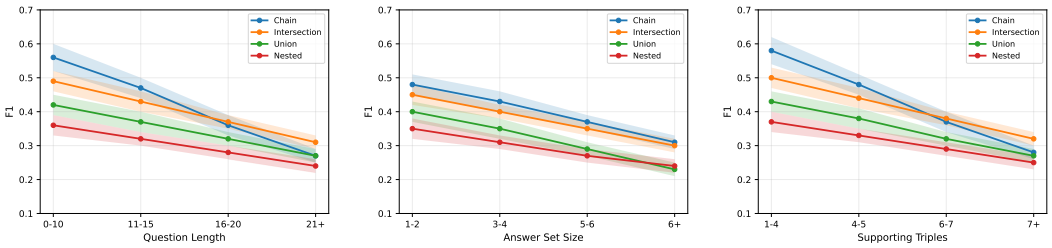
F.1 Coarse-Grained Surface Controls

Before the stricter matched analysis in Section 5.3, we first examine three observable surface-level factors: question length, answer-set size, and supporting triple count. We group the query types into four structure families—*Chain* (1p/2p/3p), *Intersection* (2i/3i), *Union* (2u), and *Nested* (ip/pi/up)—and report the average F1 after binning each factor separately. As shown in Figure 3, the relative gap between simpler and more complex structure families remains broadly stable across all three controls, motivating the stricter matched analysis reported in the main text.

F.2 Matching Protocol

For each test instance, we consider three observable difficulty attributes, following the main-text analysis in Section 5.3: question length, answer-set size, and supporting triple count. To avoid overly sparse strata, we discretize each variable into coarse bins using quantiles computed from the empirical distribution of the CLQ test split.

For each pairwise comparison in Section 5.3, we construct matched strata using these three binned attributes. We retain only strata that contain instances from both sides of the comparison. When the two sides are imbalanced within a stratum, we downsample the larger side to the size of the smaller side, yielding a fixed matched subset for each comparison. We then report average F1 on the matched subsets and estimate 95% confidence intervals for the gap via bootstrap resampling.



(a) Control question length. (b) Control answer set size. (c) Control supporting triple count.

Figure 3: Coarse-grained control analysis over three surface-level factors: question length, answer-set size, and supporting triple count. Performance differences between simpler and more complex logical structures remain visible across all three controls.

In the main text, we report three pairwise comparisons: *Chain vs. Nested*, *Chain vs. Union*, and *Intersection vs. Nested*. This procedure asks whether the observed performance gap between simpler and more complex structure families remains after aligning instances on several observable non-structural difficulty factors. Across all three comparisons, the main qualitative trend remains stable: *Nested* queries remain consistently harder than both *Chain* and *Intersection* queries, while *Union* queries show a smaller but still visible gap relative to *Chain* for most methods.

F.3 Protocol-Separated Variants

Because the methods in the main benchmark differ in training and inference protocol, we additionally examine the stricter matched analysis separately for three groups: (i) supervised methods trained on the CLQ training split, (ii) prompting- or exemplar-based methods using CLQ demonstrations, and (iii) methods evaluated without task-specific supervised retraining.

The first group includes methods such as GCR and ChatKBQA, which require task-specific training and were therefore trained on the CLQ training split with model selection performed on the CLQ validation split. The second group includes methods such as CoT, Self-Consistency, ToG, GoG, PoG, and TARGA when their protocol relies on few-shot demonstrations or in-context exemplars. For these methods, we replaced the original exemplars with CLQ training examples while preserving the original prompt structure and the number of demonstrations. The third group includes methods evaluated without task-specific supervised retraining, such as direct GPT-5 prompting and the retrieval-based baselines.

The same qualitative pattern remains visible across the three groups: *Nested* queries remain the most challenging, while *Union* queries continue to show weaker complete answer-set recovery than *Chain* queries. This suggests that the observed structure effect is not an artifact of a particular evaluation protocol. At the same time, protocol differences can affect absolute performance levels and fine-grained method ranking. We therefore interpret the main benchmark primarily as evidence of *structure-sensitive performance trends*, rather than as a definitive ranking of methods under a perfectly unified training regime.

G Limitations

Our goal in CLQ is not to cover the full space of KGQA phenomena, but to isolate logical structure as a controlled diagnostic axis. Under this goal, our benchmark still has several limitations.

First, CLQ does not include negation. We make this choice intentionally rather than incidentally. Under the Open World Assumption adopted by most knowledge graphs, missing facts do not imply falsehood, which makes reliable gold standards for negation difficult to define. As a result, excluding negation allows us to keep the benchmark executable and diagnostically clean when studying branching, union, and nested compositional reasoning. Therefore, this limitation does not weaken our main contribution, which is to provide a structure-balanced testbed for EPFO reasoning patterns that can be robustly validated on existing KGs.

Second, CLQ is built on Freebase and uses anchor entities collected from mainstream KGQA benchmarks such as WebQSP, CWQ, and GrailQA. This may inherit some bias from the underlying knowledge graph and source datasets, such as domain concentration, entity popularity, or schema conventions. At the same time, this design also improves comparability with prior KGQA work and keeps the evaluation grounded in the mainstream benchmark setting rather than an isolated synthetic environment. Since our claims are about structure-sensitive performance trends across representative KGQA paradigms, rather than universal coverage of all possible KGs or domains, we believe this design is appropriate for the paper's diagnostic objective.

Third, the construction pipeline relies on LLM-based verbalization and auditing, which may introduce residual artifacts such as occasional unnatural phrasing or subtle distributional cues. However, CLQ is not produced by unconstrained generation alone. We combine structure-constrained sampling, symbolic execution checks, anchor-grounding and secrecy constraints, iterative refinement, and sampled human evaluation to reduce logical mismatches and shortcut leakage. Moreover, our main empirical conclusions are based on consistent structure-conditioned trends observed across multiple paradigms and remain visible after controlling for several observable non-structural factors. For this reason, we view possible generation artifacts as an important caveat, but not one that overturns the benchmark's main diagnostic value.

More broadly, CLQ should be viewed as a controlled diagnostic benchmark rather than a complete replacement for existing realistic KGQA datasets. We hope future work can extend this setting to additional logical operators, other knowledge graphs, and more naturally occurring question distributions.

H Broader Impact

This work is a benchmark and evaluation study rather than a deployable end-user system. Its main positive impact is to improve KGQA evaluation by exposing structure-sensitive weaknesses that are under-tested in existing benchmarks, especially for branching, union, and nested reasoning, which may support the development of more reliable and transparent KGQA methods. Potential negative impacts are mostly indirect: stronger KGQA systems may later be deployed in high-stakes settings despite incomplete or biased knowledge graphs, and CLQ itself may inherit representational biases from Freebase and benchmark-derived anchors. We believe these risks are limited because CLQ is intended as a controlled diagnostic benchmark rather than a decision-making system, and we therefore position it as a complement to existing realistic KGQA benchmarks rather than a replacement for them.

I Dataset Samples

Table 13 provides concrete examples for each of the nine logical structures in ComplexLogicalQA. These samples illustrate the linguistic diversity and the structural complexity of the questions generated by our pipeline.

J Prompt Templates

We provide the detailed prompt templates used in our logic-driven reverse-construction pipeline.

Table 13: Samples of generated questions across nine logical structures in ComplexLogicalQA. To enhance readability, reasoning triples are listed sequentially.

Type	Question	Answer	Triples
1p	Which country used the *East German mark* as its former currency?	[East Germany]	(East German mark, finance.currency.countries_formerly_used, East Germany)
2p	What are the colors of the team owned by *Tom Hicks*?	[White, Red, Blue]	(Tom Hicks, sports.sports_team_owner.teams_owned, Texas Rangers) (Texas Rangers, sports.sports_team.colors, Red) (Texas Rangers, sports.sports_team.colors, White) (Texas Rangers, sports.sports_team.colors, Blue)
3p	What is the hex triplet of the color associated with the team that *Lou Seal* is the mascot for?	[000000]	(Lou Seal, sports.mascot.team, San Francisco Giants) (San Francisco Giants, sports.sports_team.colors, Black) (Black, base.schemastaging.visual_color_extra.hex_triplet, 000000)
2i	Which cat breeds are both *Gentle* and *Quiet*?	[Birman, American Shorthair]	(Gentle, base.petbreeds.cat_temperament.cat_breeds, Birman) (Gentle, base.petbreeds.cat_temperament.cat_breeds, American Shorthair) (Quiet, base.petbreeds.cat_temperament.cat_breeds, Birman) (Quiet, base.petbreeds.cat_temperament.cat_breeds, American Shorthair)
3i	Which film was directed by *Clemente Fracassi* and produced by *Gregor Rabinovitch* and *Federico Teti*?	[Aida]	(Clemente Fracassi, film.director.film, Aida) (Gregor Rabinovitch, film.producer.film, Aida) (Federico Teti, film.producer.film, Aida)
ip	When did the mother of *Richard, 1st Earl of Cornwall* and *Aymer de Valence* die?	[1246-05-31]	(Richard 1st Earl of Cornwall, people.person.parents, Isabella of Angoulême) (Aymer de Valence, people.person.parents, Isabella of Angoulême) (Isabella of Angoulême, people.deceased_person.date_of_death, 1246-05-31)
pi	Which battle in *Artois* was led by a commander who also fought in the *First Battle of the Marne*?	[Third Battle of Artois]	(First Battle of the Marne, base.culturalevent.event.entity_involved, Fernand de Langle de Cary) (Fernand de Langle de Cary, military.military_person.participated_in_conflicts, Third Battle of Artois) (Artois, location.location.events, Third Battle of Artois)
2u	Which projects were designed by either *Jacobs Engineering Group* or *HDR, Inc.*?	[Cleveland Clinic Abu Dhabi, Bandra–Worli Sea Link, Mike O'Callaghan–Pat Tillman Memorial Bridge]	(Jacobs Engineering Group, architecture.architecture_firm.projects, Mike O'Callaghan–Pat Tillman Memorial Bridge) (Jacobs Engineering Group, architecture.architecture_firm.projects, Bandra–Worli Sea Link) (HDR, Inc., architecture.architecture_firm.projects, Mike O'Callaghan–Pat Tillman Memorial Bridge) (HDR, Inc., architecture.architecture_firm.projects, Cleveland Clinic Abu Dhabi)
up	What awards were won by movies directed by *Christopher Nolan* or *James Cameron*?	[Academy Award for Best Visual Effects, Academy Award for Best Picture]	(Christopher Nolan, film.director.film, Inception) (James Cameron, film.director.film, Titanic) (Inception, film.film.awards_won, Academy Award for Best Visual Effects) (Titanic, film.film.awards_won, Academy Award for Best Picture) (Titanic, film.film.awards_won, Academy Award for Best Visual Effects)

Generator Prompt

Role: You are an expert Knowledge Graph Question Answering (KGQA) dataset creator. Your task is to verbalize a logical subgraph into a fluent, natural language question.
Input Data (JSON):
- `structure`: The EPFO structure type (e.g., Intersection-Projection (ip), Union (2u)).
- `triples`: A list of reasoning triples forming the path.
- `anchors`: The starting entities that MUST appear in the question.
- `answers`: The final answer entities that MUST be hidden.
- `intermediates`: The bridging entities that MUST be hidden.

Strict Constraints:
1. **Anchor Grounding:** You must explicitly mention the entities listed in `anchors`.
2. **Secrecy:** Do NOT mention `answers` or `intermediates` by name. Use generalized types (e.g., "the director", "the country") instead.
3. **Logical Fidelity:**
   - For **Intersection** (2i, ip), use conjunctions like "and", "both".
   - For **Union** (2u, up), use disjunctions like "or", "either".
4. **Complexity:** The question must require reasoning over ALL provided triples.

Input Example: `{ "structure": "ip", "anchors": ["Inception", "Titanic"], "intermediates": ["Leonardo DiCaprio"], "answers": ["USA"], "triples": [ ["DiCaprio", "starred_in", "Inception"], ["DiCaprio", "starred_in", "Titanic"], ["DiCaprio", "nationality", "USA"] ] }`
Output Example: "What is the nationality of the actor who starred in both Inception and Titanic?"

Auditor Prompt (Refinement)

Role: You are a strict Linguistic and Logical Auditor. Your task is to review the generated question against the ground truth graph data to ensure it is fluent, natural, and logically precise.
Input Data (JSON):
- `structure`: The EPFO structure type (e.g., Union 2u).
- `triples`: The ground truth reasoning triples.
- `anchors`: The starting entities.
- `answers`: The expected answer entities.
- `intermediates`: The bridging entities.
- `question`: The generated natural language question to be audited.

Evaluation Criteria:
1. **Linguistic Fluency:**
   - Is the grammar correct?
   - Is the phrasing natural, human-like, and free of ambiguity?
2. **Logical Consistency:**
   - Does the question accurately reflect the intended `structure`?
   - **Critical for Union:** Does it clearly use disjunctive phrasing (e.g., "or", "either")?
   - **Critical for Intersection:** Does it clearly imply mandatory conjunction (e.g., "both", "and")?
3. **Refinement:** If the question is robotic or logically mismatched, provide a corrected version.

Output Example: `{ "status": "FAIL", "reason": "The question uses 'and' but the structure is Union.", "refined_question": "Who directed either Inception or Interstellar?" }`

Figure 4: Prompt templates used in our pipeline. **Top:** The Generator Prompt, showing the input constraints. **Bottom:** The Auditor Prompt, focusing on linguistic and logical verification.