

FBV-KGQA: Forward-Backward Verification for Knowledge Graph Question Answering

Jiaqi Wu¹, Jiakang Li¹, Zhenxuan Ma¹, Li Wang^{2,*},

¹College of Computer Science and Technology, Taiyuan University of Technology,

²College of Artificial Intelligence, Taiyuan University of Technology

{wujiaqi0252, 2024520727, 2024520689}@link.tyut.edu.cn, wangli@tyut.edu.cn

Abstract

Large language models (LLMs) have shown strong reasoning capabilities in knowledge graph question answering (KGQA), yet their predictions are not always supported by explicit and verifiable reasoning paths. Existing methods often retrieve or generate plausible candidate paths, but path plausibility alone does not establish that a prediction actually depends on the retrieved evidence. To address this issue, we propose FBV-KGQA, a forward-backward verification framework for reliable and interpretable KGQA. FBV-KGQA first derives executable reasoning paths through relation-level planning and evaluates their forward support for candidate answers. It then identifies critical entity and relation pivots and applies schema-constrained counterfactual perturbations to assess their influence on model predictions. Reverse questions are subsequently constructed from candidate answers to verify whether they can be consistently traced back to the original question entity. Finally, an adaptive selection strategy integrates forward support, counterfactual effects, and backward consistency. Experiments on WebQSP and CWQ show competitive performance against representative KGQA baselines, while human evaluation indicates improved traceability, clarity, and usefulness of the resulting reasoning paths.

1 Introduction

Large language models (LLMs) have demonstrated remarkable generative capabilities and multi-step reasoning abilities across a wide range of natural language processing tasks (Jiang et al., 2026; Sui et al., 2025). However, despite their strong performance in language understanding and generation, LLMs remain susceptible to factual hallucinations, where the generated answers may be inconsistent with factual knowledge or even entirely fabricated

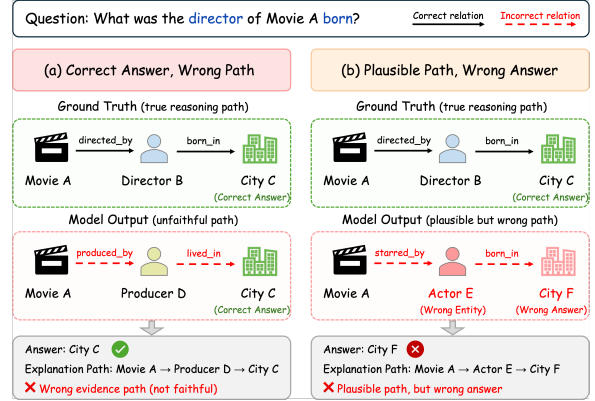


Figure 1: Failure modes of existing KGQA.

(Guan et al., 2024; Sansford et al., 2024). Such hallucinations undermine the reliability of LLM-based question answering, particularly when answers are expected to be supported by explicit factual evidence (González et al., 2025).

Knowledge graphs (KGs) provide structured factual knowledge in the form of explicit entities and relations, making them particularly suitable for grounding LLM-based question answering (Luo et al., 2024). By organizing knowledge as explicit triples, KGs allow candidate reasoning paths between question entities and potential answers to be retrieved and inspected during multi-hop inference (Li et al., 2025). These structured paths provide traceable factual evidence and make it possible to examine the entities and relations involved in multi-hop reasoning (Wu et al., 2025). Moreover, the relational structure of KGs also provides explicit constraints for reasoning over intermediate entities and relations in complex questions (Wang and Yu, 2026).

Existing LLM-based knowledge graph question answering (KGQA) methods mainly retrieve question-relevant graph structures or iteratively explore reasoning paths over KGs. Retrieval-based methods retrieve question-relevant triples or sub-

*Corresponding author.

graphs as external evidence for LLM generation (Peng et al., 2025), but they may miss intermediate entities or relations needed for complex multi-hop reasoning (Sui et al., 2025). Agent-based methods use LLMs to iteratively explore KG paths and select subsequent entities or relations (Shen et al., 2025), producing explicit reasoning traces at the cost of repeated LLM-KG interactions and potential error propagation (Xiao et al., 2026).

Despite these advances, several challenges remain, as illustrated in Figure 1. First, from the perspective of reliability, the generated answer is not necessarily derived from an explicit reasoning path. Existing candidate paths may appear plausible at a surface level, but cannot truly verify the correctness of the answer. Second, from the perspective of interpretability, even when candidate paths are provided, existing methods often struggle to identify which entities or relations play a critical role in the final answer, resulting in a reasoning process that lacks transparency and traceability (Shah et al., 2024).

To address the above challenges, we propose **FBV-KGQA**, short for **F**orward-**B**ackward **V**erification for **K**nowledge **G**raph **Q**uestion **A**nswering. FBV-KGQA is a cognitively inspired KGQA framework that follows a forward-derivation and backward-verification mechanism. Specifically, given a question and its linked starting entity, FBV-KGQA first generates relation-level plans and executes them over the KG to construct candidate reasoning paths. The retained paths are evaluated according to their relevance to the question and their forward support for candidate answers. FBV-KGQA then identifies critical entity and relation pivots and applies schema-constrained counterfactual perturbations to assess how changes to these pivots affect the prediction. Next, reverse questions are constructed from candidate answers to check whether they can be consistently traced back to the original question entity. Finally, forward support, counterfactual effects, and backward consistency are integrated to rank candidate answers and determine the final answer set. The main contributions of this work are summarized as follows:

- We propose FBV-KGQA, a forward-backward verification framework that derives executable KG paths in the forward direction and verifies candidate answers through backward consistency checking.

- We introduce a logical-pivot attribution module that identifies critical entities and relations and uses schema-constrained counterfactual perturbation to assess path reliability.
- Experiments on WebQSP and CWQ demonstrate the effectiveness of FBV-KGQA, with further analyses confirming its reliability and interpretability.

2 Related Work

Existing KGQA methods with LLMs mainly follow semantic parsing, retrieval-augmented reasoning, and agent-based graph exploration paradigms. Semantic parsing methods translate natural language questions into executable logical queries over KGs (Banerjee, 2023; Bai et al., 2023), while retrieval-augmented methods retrieve relevant triples or subgraphs as external evidence for answer generation (Xu et al., 2024; Luo et al., 2024, 2025). Recent LLM-based approaches further construct query-specific subgraphs or iteratively explore reasoning paths over KGs, such as GraphRAG (Touvron et al., 2025), RAS (Jiang et al., 2026), FiDeLiS (Sui et al., 2025), RWT (Shen et al., 2025), and KG-Reasoner (Wang and Yu, 2026). These methods improve structured reasoning, but still face challenges in verifying whether retrieved or generated paths genuinely support the final answer.

Another related line focuses on LLM interpretability and attribution reasoning. Prompting methods such as CoT (Wei et al., 2022) and ToT (Yao et al., 2023) elicit intermediate reasoning steps, but these steps are not necessarily grounded or verified. Recent work incorporates structured evidence for factual verification and attribution, including StructGPT (Jiang et al., 2023a), GraphEval (Sansford et al., 2024), Think&Cite (Li and Ng, 2025), and atomic fact decomposition methods (Yan et al., 2025). In contrast to these studies, FBV-KGQA verifies candidate KG paths from both forward and backward directions, and further uses logical-pivot counterfactual attribution to assess whether critical entities and relations affect the predicted answer. A more detailed discussion of related work is provided in Appendix E.

3 Method

Existing KGQA methods typically follow a path retrieval and answer generation paradigm, but the retrieved paths may not faithfully support the generated answer and can instead serve as superficial

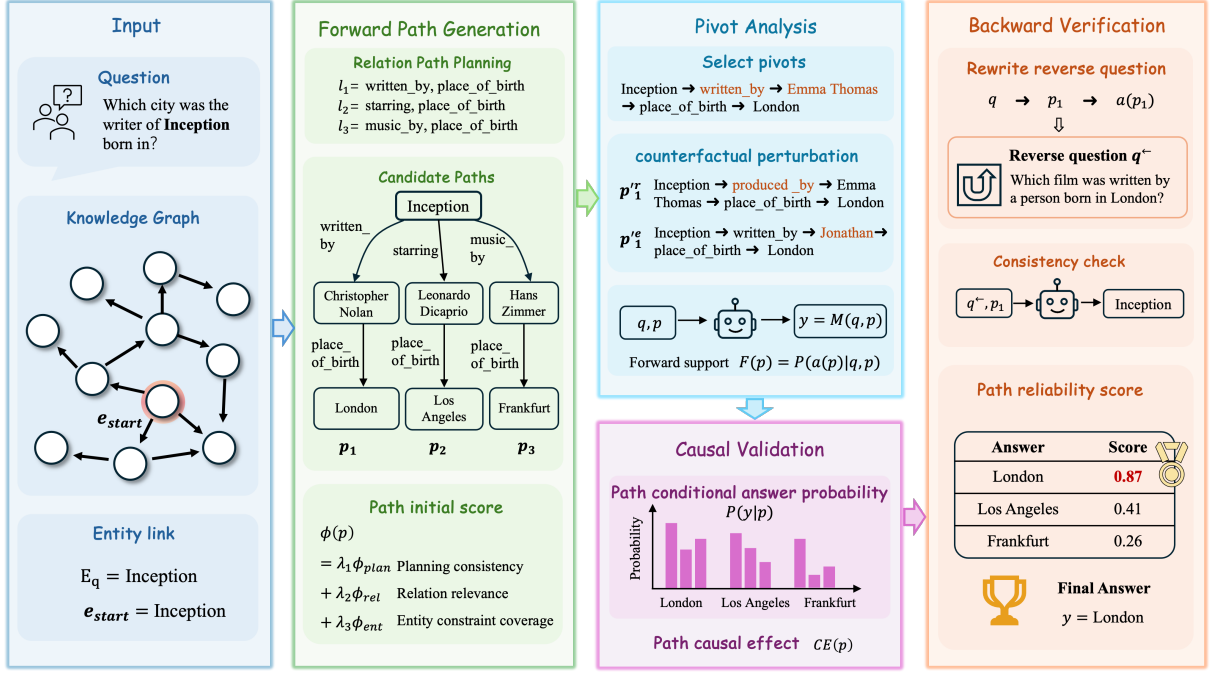


Figure 2: Overview of the FBV-KGQA framework.

post-hoc explanations. To address this issue, we propose a Forward-Backward Verification framework for Knowledge Graph Question Answering, as shown in Figure 2.

FBV-KGQA follows a forward derivation and backward verification mechanism. It first performs relation path planning and executable path querying from the question entity to obtain candidate paths and answers. It then identifies critical relations and intermediate entities in each path, and applies schema-constrained counterfactual perturbations to test whether these logical pivots affect the LLM output. Finally, FBV-KGQA rewrites each candidate answer into a reverse question and requires the LLM to select an answer from the entities in the current path. A path passes backward verification if the selected entity returns to the original main entity e_{start} .

3.1 Forward Path Querying

3.1.1 Relation Path Planning

The goal of relation path planning is to convert the question semantics into relation-level query plans, which guide subsequent executable path retrieval over the KG.

Given a question q , the entity linking module identifies the question entity set $\mathcal{E}_q$, where the main entity is denoted as $e_{start} \in \mathcal{E}_q$. The entity e_{start} is used as the query starting point, while the remaining entities serve as constraints for path filtering.

Given q and e_{start} , the LLM generates a set of relation path plans $\mathcal{L} = \{\ell_1, \ell_2, \dots, \ell_N\}$, where N is the number of generated plans, and each ℓ_i is a relation sequence. For example, given the question *Which city was the director of film A born in?*, the LLM may generate the plan *director* $\rightarrow$ *place of birth*, indicating that the system should first query the director of the film and then query the director’s birthplace.

3.1.2 Plan-Guided Path Querying

Since LLM-generated relation descriptions may differ from canonical KG relation names, we first align each planned relation with KG relations by selecting semantically similar relation labels. After alignment, executable paths are retrieved by starting from the main entity e_{start} and following the planned relation order in the KG. Paths that can be successfully executed are retained as candidate reasoning paths:

$$\tilde{\mathcal{P}} = \bigcup_{\ell_i \in \mathcal{L}} \text{Retrieve}(e_{start}, \ell_i, \mathcal{G}), \quad (1)$$

where $\text{Retrieve}(\cdot)$ denotes KG path retrieval guided by a relation path plan.

For example, given the plan *director* $\rightarrow$ *place of birth*, the system first retrieves relations semantically matching *director* from the film entity, and then retrieves relations matching *place of birth* from the resulting director entity. If the resulting path is executable, it is added to $\tilde{\mathcal{P}}$.

3.1.3 Candidate Path Filtering

To retain high-quality candidate paths, we compute an initial score for each retrieved path by considering its consistency with the LLM-generated relation plan, semantic relevance to the question, and coverage of question entities. For a candidate path $p = (e_0, r_1, e_1, \dots, r_m, e_m)$ and its corresponding relation plan $\ell = (\hat{r}_1, \hat{r}_2, \dots, \hat{r}_m)$, the planning consistency is defined as

$$\phi_{\text{plan}}(p) = \frac{1}{m} \sum_{j=1}^m \cos(f_{\text{enc}}(\hat{r}_j), f_{\text{enc}}(r_j)), \quad (2)$$

where f_{enc} denotes the text encoder. This term measures the semantic consistency between planned relations and executable KG relations.

The question-path relation relevance is computed as

$$\phi_{\text{rel}}(p) = \frac{1}{m} \sum_{j=1}^m \cos(f_{\text{enc}}(q), f_{\text{enc}}(r_j)). \quad (3)$$

Let $\mathcal{E}(p) = \{e_0, e_1, \dots, e_m\}$ denote the entity set of path p . The entity constraint coverage is defined as

$$\phi_{\text{ent}}(p) = \frac{1}{|\mathcal{E}_q|} \sum_{e \in \mathcal{E}_q} \mathbf{1}[e \in \mathcal{E}(p)]. \quad (4)$$

The overall initial path score is then given by

$$\phi(p) = \lambda_1 \phi_{\text{plan}}(p) + \lambda_2 \phi_{\text{rel}}(p) + \lambda_3 \phi_{\text{ent}}(p), \quad (5)$$

where λ_1, λ_2 , and λ_3 are nonnegative weights. Candidate paths with higher $\phi(p)$ are retained for subsequent attribution verification and backward verification, ensuring that the selected paths are consistent with both the LLM plan and the semantic constraints of the question.

3.2 Logical-Pivot Attribution

3.2.1 Logical Pivot Identification

To make candidate paths explainable, we identify logical pivots that play critical roles in determining the reasoning direction or the final answer. Logical pivots are divided into two types: critical relation pivots and critical entity pivots.

For each candidate path $p = (e_0, r_1, e_1, \dots, r_m, e_m)$, we define $a(p)$ as the path-conditioned answer entity predicted by the LLM given the question q and the path p . Each relation r_j is scored by considering its relevance to the question, proximity to the candidate answer, and relation information content:

$$S_r(r_j; p, q) = \alpha_1 s_r(q, r_j) + \alpha_2 \rho(r_j, a(p)) + \alpha_3 I(r_j), \quad (6)$$

where α_1, α_2 , and α_3 are nonnegative weights. The definitions of $s_r(q, r_j)$, $\rho(r_j, a(p))$, and $I(r_j)$ are provided in Appendix A. The relation with the highest score is selected as the critical relation pivot:

$$r^* = \arg \max_{r_j \in p} S_r(r_j; p, q). \quad (7)$$

For entity pivots, we exclude the starting entity and the path-conditioned candidate answer, since replacing either of them would conflate path attribution with changes to the question entity or answer entity. The candidate entity pivot set is defined as

$$\mathcal{E}_{\text{pivot}}(p) = \mathcal{E}(p) \setminus \{e_0, a(p)\}. \quad (8)$$

Each candidate entity $e_j \in \mathcal{E}_{\text{pivot}}(p)$ is scored by combining its semantic relevance to the question, its coverage of question constraints, and its connection to the candidate answer:

$$S_e(e_j; p, q) = \beta_1 s_e(q, e_j) + \beta_2 C(e_j) + \beta_3 A(e_j), \quad (9)$$

where β_1, β_2 , and β_3 are nonnegative weights. The definitions of $s_e(q, e_j)$, $C(e_j)$, and $A(e_j)$ are given in Appendix A. The entity with the highest score is selected as the critical entity pivot:

$$e^* = \arg \max_{e_j \in \mathcal{E}_{\text{pivot}}(p)} S_e(e_j; p, q). \quad (10)$$

If $\mathcal{E}_{\text{pivot}}(p)$ is empty, only the critical relation pivot is perturbed.

3.2.2 Schema-Constrained Counterfactual Perturbation

Given the critical relation r^* and critical entity e^* , we construct counterfactual paths by perturbing these logical pivots under KG schema constraints. The goal is to test whether changes to the selected pivots lead to changes in the model prediction, rather than introducing arbitrary or invalid path modifications. To avoid spurious attribution, the perturbation is restricted to type-compatible and locally executable alternatives.

For relation perturbation, r^* is replaced with a semantically different relation whose domain and range are compatible with those of the original relation. For entity perturbation, e^* is replaced with a type-compatible entity, while preserving the executability of its adjacent relations whenever possible. If direct replacement does not yield valid triples, we perform local reconnection within the type-consistent candidate space.

For each candidate path p , the above operations produce a counterfactual path set $\mathcal{P}^{\text{cf}}(p)$,

which contains paths derived from relation-level and entity-level perturbations. The output difference between the original path p and its counterfactual variants is then used to estimate whether the logical pivots have a causal influence on the predicted answer. A larger change indicates stronger path dependence, while a smaller change suggests that the path may provide only superficial support.

3.2.3 Path-Level Counterfactual Effect

To quantify whether a candidate path truly affects model prediction, we compare the answer distributions under the original path and its counterfactual variants. For the filtered candidate path set $\tilde{\mathcal{P}}$, each path produces a path-conditioned candidate answer $a(p)$. The candidate answer set is defined as

$$\mathcal{Y} = \{a(p) \mid p \in \tilde{\mathcal{P}}\}. \quad (11)$$

Given a path p and its counterfactual path set $\mathcal{P}^{\text{cf}}(p)$, the path-level counterfactual effect is computed as

$$CE(p) = \frac{1}{|\mathcal{P}^{\text{cf}}(p)|} \sum_{p' \in \mathcal{P}^{\text{cf}}(p)} D_{\text{KL}}(P_M(\cdot \mid q, p) \parallel P_M(\cdot \mid q, p')). \quad (12)$$

where $P_M(\cdot \mid q, p)$ denotes the probability distribution over $\mathcal{Y}$ produced by the LLM M given question q and path p , and D_{KL} denotes KL divergence. A smoothing term is used to avoid zero probabilities. A larger $CE(p)$ indicates that perturbing the critical pivots in p substantially changes the answer distribution, suggesting stronger causal influence on the model prediction. A smaller $CE(p)$ indicates that the model output is insensitive to the pivots, and the path may be only superficially relevant. The counterfactual effect evaluates whether changing a logical pivot alters the prediction distribution, thereby measuring functional contribution rather than surface association.

We also compute the forward support score of path p for its candidate answer:

$$F(p) = P_M(a(p) \mid q, p), \quad (13)$$

where $F(p)$ measures whether the path directly supports the candidate answer, while $CE(p)$ measures whether the prediction depends on the path's critical pivots. The two scores are complementary for assessing path reliability.

3.3 Reverse Verification

3.3.1 Reverse Question Construction

Although a candidate path may support an answer after forward generation and counterfactual attribution, this does not guarantee that the path is logically self-consistent. To verify whether the candidate answer can be traced back to the original question entity, we introduce a reverse question rewriting mechanism.

Given the original question q and the path-conditioned candidate answer $a(p)$, the LLM generates a reverse question $q^{\leftarrow}$ by treating $a(p)$ as a known condition and taking the main entity e_{start} as the target answer. This process is not a simple reversal of the path. Instead, the reverse question is constructed according to the semantics of q and the current candidate path. Therefore, backward verification serves as a consistency check rather than an additional retrieval process.

For example, given the question *Which city was the director of film A born in?* and the forward path $A \xrightarrow{\text{director}} B \xrightarrow{\text{place of birth}} C$, where C is the candidate answer, the reverse question can be rewritten as *Which film has a director who was born in C?* The expected answer is the original main entity A .

To reduce rewriting hallucinations, the LLM is constrained to use only the original question, the candidate answer, and the entities and relations in the given path. The reverse question is therefore used not to introduce new knowledge, but to test whether the candidate answer forms a consistent backward inference relation with e_{start} .

3.3.2 In-Path Consistency Check

To avoid ambiguity caused by open-ended entity generation, we formulate backward verification as an in-path candidate selection task. Given the reverse question $q^{\leftarrow}$, the model is required to select an answer only from the entity set of the current path, rather than freely generating an entity name. In implementation, candidate entities are randomly shuffled, and the model outputs only the selected index to reduce position and formatting biases.

Let $\hat{e}^{\leftarrow}$ denote the entity selected by the model under the reverse question $q^{\leftarrow}$. The path passes backward verification if the selected entity matches the original main entity e_{start} :

$$B_{\text{rev}}(p) = \mathbf{1}[\hat{e}^{\leftarrow} = e_{\text{start}}]. \quad (14)$$

When $B_{\text{rev}}(p) = 1$, the candidate answer can be traced back to the question entity through the current path. Otherwise, the path fails the backward

consistency check and is not treated as a valid explanatory path.

3.4 Adaptive Answer Selection

3.4.1 Path Reliability Scoring

To integrate different sources of evidence, we compute a final reliability score for each candidate path. This score simultaneously considers forward support, initial relevance, and counterfactual effect:

$$S(p) = w_1 F(p) + w_2 \phi(p) + w_3 CE(p), \quad (15)$$

where $F(p)$ measures the degree to which the path positively supports the candidate answer, $\phi(p)$ measures the initial relevance between the path and the question, and $CE(p)$ measures the counterfactual sensitivity of the model output to the critical pivots in the path. w_1 , w_2 , and w_3 are weight hyperparameters.

We regard paths that satisfy the forward support, counterfactual effect, and backward verification conditions as valid explanatory paths:

$$\mathcal{P}_{\text{valid}} = \left\{ p \in \tilde{\mathcal{P}} \mid F(p) \geq \tau_{\text{fwd}}, CE(p) \geq \tau_{\text{ce}}, B_{\text{rev}}(p) = 1 \right\}. \quad (16)$$

Here, τ_{fwd} denotes the forward support threshold, and τ_{ce} denotes the counterfactual effect threshold. Only paths included in $\mathcal{P}_{\text{valid}}$ are considered capable of providing reliable attribution explanations for the final answer.

3.4.2 Adaptive Answer Selection

For each candidate answer $y \in \mathcal{Y}$, we perform weighted voting over all valid paths:

$$\text{Vote}(y) = \sum_{p \in \mathcal{P}_{\text{valid}}} S(p) \mathbf{1}[a(p) = y], \quad (17)$$

where $a(p)$ denotes the path conditioned candidate answer corresponding to path p . If multiple valid paths support the same candidate answer, the voting score of that answer increases accordingly.

To eliminate the effect of different numbers of candidate paths across questions, we further normalize the voting score of each candidate answer:

$$\text{Conf}(y) = \frac{\text{Vote}(y)}{\sum_{y' \in \mathcal{Y}} \text{Vote}(y') + \epsilon}, \quad (18)$$

where ϵ is a smoothing term. $\text{Conf}(y)$ denotes the relative support strength of candidate answer y among all valid explanatory paths for the current question.

Instead of applying an additional fixed threshold to the final answer confidence, we adopt a ranking-gap-based strategy to determine the output answer

set. First, candidate answers are sorted in descending order according to their confidence scores:

$$\text{Conf}(y_{(1)}) \geq \text{Conf}(y_{(2)}) \geq \dots \geq \text{Conf}(y_{(n)}). \quad (19)$$

Then, the relative confidence drop between adjacent candidate answers is computed as

$$\Delta_i = \frac{\text{Conf}(y_{(i)}) - \text{Conf}(y_{(i+1)})}{\text{Conf}(y_{(i)}) + \epsilon}, \quad 1 \leq i < n. \quad (20)$$

We treat the position with the most significant confidence drop as the adaptive boundary of the answer set:

$$k^* = \arg \max_{1 \leq i < n} \Delta_i. \quad (21)$$

The final answer set is defined as

$$\hat{\mathcal{Y}} = \{y_{(1)}, y_{(2)}, \dots, y_{(k^*)}\}. \quad (22)$$

This gap-based strategy avoids using a fixed answer threshold and determines the number of output answers according to the support distribution of the current question.

4 Experiment

4.1 Experimental details

To evaluate the effectiveness of the proposed method on knowledge graph question answering tasks, we conduct experiments on two widely used benchmark datasets, including WebQuestionsSP (WebQSP) (Yih et al., 2016) and Complex WebQuestions (CWQ) (Talmor and Berant, 2018). We further compare our method with 24 representative baselines from six categories to comprehensively assess its performance under different KGQA modeling paradigms. Detailed dataset descriptions, baseline settings, and evaluation metrics are provided in the Appendix C.

4.2 Results and Analysis

4.2.1 Overall Performance

Table 1 compares our method with different KGQA baselines on WebQSP and CWQ. Our method achieves 85.6% Hits@1 and 78.8% F1 on WebQSP, and 72.3% Hits@1 and 64.2% F1 on CWQ, outperforming traditional representation learning, retrieval-augmented, and semantic parsing based methods. These results indicate that relying only on entity-relation representations, local subgraph retrieval, or executable query generation is insufficient for handling multi-hop constraints and path reliability in complex KGQA, especially on the more compositional CWQ dataset.

Category	Methods	WebQSP		CWQ	
		Hits@1 (%)	F1 (%)	Hits@1 (%)	F1 (%)
Embedding	KV-Mem (Miller et al., 2016)	46.7	34.5	18.4	15.7
	EmbedKGQA (Saxena et al., 2020)	66.6	-	45.9	-
	NSM (He et al., 2021)	68.7	62.8	47.6	42.4
	TransferNet (Shi et al., 2021)	71.4	-	48.6	-
	KGT5 (Saxena et al., 2022)	56.1	-	36.5	-
Retrieval	GraftNet (Sun et al., 2018)	66.4	60.4	36.8	32.7
	PullNet (Sun et al., 2019)	68.1	-	45.9	-
	SR+NSM (Zhang et al., 2022)	68.9	64.1	50.2	47.1
	SR+NSM+E2E (Zhang et al., 2022)	69.5	64.1	49.3	46.3
Semantic Parsing	SPARQA (Sun et al., 2020)	-	-	31.6	-
	QGG (Lan and Jiang, 2020)	73.0	73.8	36.9	37.4
	ArcaneQA (Gu and Su, 2022)	-	75.3	-	-
	RnG-KBQA (Ye et al., 2022)	-	76.2	-	-
LLMs	Flan-T5-xl (Chung et al., 2024)	31.0	-	14.7	-
	Alpaca-7B (Taori et al., 2023)	51.8	-	27.4	-
	LLaMA2-Chat-7B (Touvron et al., 2023)	64.4	-	34.6	-
	ChatGPT	66.8	-	39.9	-
	ChatGPT+CoT	75.6	-	48.9	-
Finetuning LLMs+KGs	KD-CoT (Wang et al., 2023)	68.6	52.5	55.7	-
	UniKGQA (Jiang et al., 2023b)	77.2	72.2	51.2	49.1
	DECAF (DPR+FiD-3B) (Yu et al., 2023)	82.1	-	70.4	-
	RoG (Luo et al., 2024)	83.2	69.8	61.4	56.2
Prompting LLMs+KGs	ToG (Sun et al., 2024)	81.8	76.0	68.5	60.2
	FiDeLiS (Sui et al., 2025)	84.4	78.3	71.5	64.3
	Our method	85.6	78.8	72.3	64.2

Table 1: Comparison of FBV-KGQA with baseline methods on WebQSP and CWQ datasets. Prompting-based LLM+KG methods are evaluated with gpt-4-turbo.

Compared with pure LLM-based methods, our method also shows clear gains. ChatGPT+CoT obtains 75.6% and 48.9% Hits@1 on WebQSP and CWQ, while our method improves them to 85.6% and 72.3%, respectively. This suggests that CoT prompting can elicit explicit reasoning steps, but lacks structured KG constraints and may still produce unverifiable answers. By combining LLM-based relation planning, executable KG path querying, counterfactual perturbation of logical pivots, and reverse question verification, our method converts open-ended generation into path-constrained verification-oriented reasoning.

Against recent LLM+KG baselines, our method remains competitive. Compared with RoG, it improves Hits@1 and F1 by 2.4% and 9.0% on WebQSP, and by 10.9% and 8.0% on CWQ. Compared with ToG, it improves Hits@1 by 3.8% on both datasets. Compared with FiDeLiS, it improves WebQSP Hits@1 and F1 by 1.2% and 0.5%, and CWQ Hits@1 by 0.8%, while achieving compara-

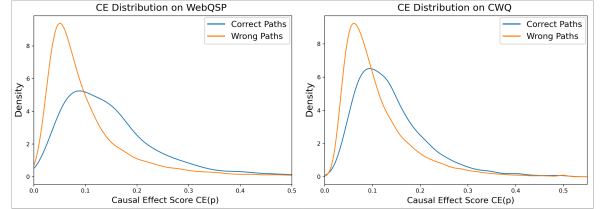


Figure 3: Causal effect distributions of correct and wrong reasoning paths.

ble F1 on CWQ. These improvements show that the effectiveness of our method comes not only from KG path retrieval, but also from verifying whether candidate paths genuinely support the answer. By testing the causal influence of critical entities and relations and checking backward consistency with the original question entity, our method filters out superficially relevant paths and improves both answer accuracy and interpretability.

Variant	WebQSP		CWQ	
	Hits@1 (%)	F1 (%)	Hits@1 (%)	F1 (%)
w/o Path Planning	72.1	65.3	56.6	50.2
w/o Initial Path Scoring	73.2	67.3	57.2	53.1
w/o Forward Path Support	69.6	60.1	51.2	44.8
w/o Causal Attribution	72.4	66.5	55.4	49.7
w/o Reverse Verification	75.8	68.4	60.1	55.5
Full method	78.3	75.8	64.9	59.9

Table 2: Ablation study of FBV-KGQA using gpt-3.5-turbo as the backend LLM.

4.2.2 Analysis of the Causal Effect Module

Figure 3 compares the path-level causal effect distributions of correct and incorrect paths on WebQSP and CWQ. The CE scores are smoothed by kernel density estimation, where the horizontal axis denotes the CE score and the vertical axis denotes probability density. The blue and orange curves correspond to correct and incorrect paths, respectively.

Overall, correct paths exhibit a right-shifted CE distribution with higher scores than incorrect paths. This indicates that perturbing critical entities or relations in correct paths causes larger changes in the model’s answer distribution, suggesting stronger causal influence on answer prediction. In contrast, incorrect paths are mostly concentrated in the low-score region, implying limited influence on the output and weaker explanatory value.

The separation is more evident on WebQSP, showing that CE can effectively distinguish supportive paths from noisy ones in relatively simple questions. On CWQ, the two distributions partially overlap because some incorrect paths may still contain question-relevant intermediate entities or locally correct relations in complex multi-hop settings. These results show that CE provides a useful signal for path reliability estimation and supports the design of FBV-KGQA, which combines forward support, causal attribution, and backward verification.

4.2.3 Ablation Study

Table 2 reports the ablation results of FBV-KGQA on WebQSP and CWQ. Removing any core module consistently degrades performance, indicating that path planning, initial path scoring, forward path support, causal attribution, and backward verification all contribute to the final results. Among them, forward path support is the most critical component: removing it reduces Hits@1 and F1 by 8.7%

Method	Faithful	Traceable	Clarity	Useful
FiDeLiS	70.0%	72.0%	76.0%	74.0%
FBV-KGQA	72.0%	78.0%	80.0%	80.0%

Table 3: Human evaluation of reasoning-path interpretability on 50 sampled WebQSP questions.

and 15.7% on WebQSP, and by 13.7% and 15.1% on CWQ, respectively. This shows that direct path support for candidate answers is essential for reliable answer selection.

Path planning and initial path scoring also bring clear gains by reducing irrelevant path expansion and filtering weakly related paths. The larger drops on CWQ suggest that complex multi-hop questions are more sensitive to candidate path quality. In addition, removing causal attribution decreases F1 by 9.3% on WebQSP and 10.2% on CWQ, confirming that path relevance and forward support alone cannot reliably distinguish truly supportive paths from superficial ones. Backward verification further improves robustness by filtering paths that score highly in the forward direction but cannot be traced back to the original question entity. Overall, the gains of FBV-KGQA come from the synergy of these modules, with forward path support and causal attribution contributing most to answer accuracy and explanation reliability.

4.2.4 Interpretability Evaluation

To further evaluate the interpretability of the reasoning paths, we conduct a human evaluation on 50 sampled questions from WebQSP and compare FBV-KGQA with FiDeLiS. As shown in Table 3, the reasoning paths are evaluated from four complementary dimensions. *Faithful* measures whether a reasoning path provides valid support for the predicted answer. *Traceable* evaluates whether the reasoning process can be followed through explicit KG entities and relations. *Clarity* reflects how clearly the reasoning path presents the inference

Cost item	WebQSP	CWQ
Retained paths/question	3	3
Avg. LLM calls/path	7.5	7.8
Avg. LLM calls/question	23.4	24.4
Avg. runtime/evaluated question	25.4 s	29.7s
Estimated tokens/question	5.5K	7.0K

Table 4: Computational cost of FBV-KGQA with gpt-3.5-turbo as the backend LLM.

process. *Useful* assesses whether the path provides informative evidence for understanding the final prediction.

FBV-KGQA achieves higher scores than FiDeLiS across all four dimensions. In particular, traceability improves from 72.0% to 78.0%, while usefulness increases from 74.0% to 80.0%. FBV-KGQA also obtains gains in faithfulness and clarity, reaching 72.0% and 80.0%, respectively. These results suggest that explicitly verifying candidate paths and identifying influential logical pivots can improve not only answer reliability but also the transparency of the reasoning process, making the resulting paths easier to trace and more informative for interpreting the final prediction.

4.2.5 Computational Cost Analysis

Table 4 reports the computational cost of FBV-KGQA on WebQSP and CWQ. The additional overhead mainly comes from the explicit verification process. For each retained path, FBV-KGQA performs forward path-conditioned answer scoring, counterfactual scoring for the selected logical pivots, and backward verification. Relation alignment is implemented using embedding-based cosine similarity and therefore does not introduce additional LLM calls. With three candidate paths retained for each question, FBV-KGQA requires an average of 23.4 and 24.4 LLM calls per question on WebQSP and CWQ, respectively.

The average runtime is 25.4 seconds per question on WebQSP and 29.7 seconds on CWQ, with approximately 5.5K and 7.0K tokens consumed per question. The slightly higher cost on CWQ is mainly due to its longer and more compositional reasoning paths, which require more counterfactual evaluations. These results indicate that FBV-KGQA introduces additional computational overhead compared with methods relying only on path retrieval or forward reasoning. This overhead represents a trade-off for explicitly assessing whether candidate paths support the predicted answers through counterfactual attribution and back-

ward consistency verification.

4.2.6 Error Analysis

To better understand the remaining errors of FBV-KGQA, we analyze representative failure cases and identify two major error types: path scoring failure and path retrieval failure. Table 10 in Appendix F presents a path scoring failure for the question “Who did Cam Newton sign with?” The correct path leading to *Carolina Panthers* is successfully retrieved as Path 3. However, its final score is 0.630, which is lower than the score of 0.669 assigned to the incorrect Path 2 leading to *Auburn Tigers football*. Consequently, the model selects an incorrect answer despite the presence of the correct reasoning path. This case indicates that the current reliability scoring mechanism may still have difficulty distinguishing semantically plausible but incorrect paths when multiple candidates share similar relation structures.

Table 11 in Appendix F illustrates a path retrieval failure. For the question “What countries are part of the UK?”, none of the retained candidate paths reaches the correct answer *Northern Ireland*. Instead, the retrieved paths follow the relation *form_of_government* and lead to countries such as Canada, the Netherlands, and Antigua and Barbuda. Since the correct constituent-country relation is absent from the candidate set, subsequent counterfactual attribution and backward verification cannot recover the correct answer. This example highlights the dependence of FBV-KGQA on the quality of relation planning and candidate path retrieval. Overall, these cases suggest that future improvements should focus on more accurate candidate path construction and more discriminative reliability scoring.

5 Conclusion

In this paper, we proposed FBV-KGQA, a forward-backward verification framework for knowledge graph question answering. FBV-KGQA derives executable reasoning paths in the forward direction, verifies candidate answers through reverse question rewriting, and uses logical-pivot-based counterfactual attribution to assess path reliability. Experiments on WebQSP and CWQ show that FBV-KGQA achieves competitive performance over representative KGQA baselines, demonstrating the effectiveness of combining forward derivation with backward verification for reliable and interpretable KGQA.

Limitations

FBV-KGQA still has several limitations. It depends on accurate entity linking and sufficient KG coverage, since missing entities or incomplete reasoning paths may affect both forward derivation and backward verification. In addition, the use of relation planning, counterfactual attribution, and reverse verification requires multiple LLM calls, leading to higher computational cost. Finally, our evaluation mainly focuses on WebQSP and CWQ, and further experiments on larger and domain-specific KGs are needed to assess its broader generalizability.

Acknowledgements

This work was supported by the Regional Innovation and Development Joint Fund of NSFC (No. U22A20167) and the Special project for guiding the transformation of scientific and technological achievements in Shanxi Province (202404021301033).

References

- Jiaxin Bai, Xin Liu, Weiqi Wang, Chen Luo, and Yangqiu Song. 2023. Complex query answering on eventuality knowledge graph with implicit logical constraints. *Advances in Neural Information Processing Systems*, 36:30534–30553.
- Debayan Banerjee. 2023. Semantic parsing for knowledge graph question answering with large language models. In *European Semantic Web Conference*, pages 234–243. Springer.
- Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.
- Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, and 16 others. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.
- Ana Valeria González, Sidsel Boldsen, and Roland Hangelbroek. 2025. Triplecheck: Transparent post-hoc verification of biomedical claims in ai-generated answers. In *Proceedings of the Fourth Workshop on Bridging Human-Computer Interaction and Natural Language Processing (HCI+ NLP)*, pages 33–47.
- Yu Gu and Yu Su. 2022. Arcaneqa: Dynamic program induction and contextualized encoding for knowledge base question answering. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1718–1731.
- Xinyan Guan, Yanjiang Liu, Hongyu Lin, Yaojie Lu, Ben He, Xianpei Han, and Le Sun. 2024. Mitigating large language model hallucinations via autonomous knowledge graph-based retrofitting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18126–18134.
- Gaole He, Yunshi Lan, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Improving multi-hop knowledge base question answering by learning intermediate supervision signals. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 553–561.
- Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Xin Zhao, and Ji-Rong Wen. 2023a. 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*, pages 9237–9251.
- Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, and Ji-Rong Wen. 2023b. Unikgqa: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph. In *Proceedings of the 11th International Conference on Learning Representations (ICLR 2023)*.
- Pengcheng Jiang, Lang Cao, Ruikun Zhu, Minhao Jiang, Yunyi Zhang, Jiaming Shen, Jimeng Sun, and Jiawei Han. 2026. Ras: Retrieval-and-structuring for knowledge-intensive llm generation. In *The Fourteenth International Conference on Learning Representations (ICLR 2026)*.
- Yunshi Lan and Jing Jiang. 2020. Query graph generation for answering multi-hop complex questions from knowledge bases. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 969–974.
- Junyi Li and Hwee Tou Ng. 2025. Think&cite: Improving attributed text generation with self-guided tree search and progress reward modeling. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9928–9942.
- Xiangci Li, Sihao Chen, Rajvi Kapadia, Jessica Ouyang, and Fan Zhang. 2025. Minimal evidence group identification for claim verification. In *Proceedings of the 5th Workshop on Trustworthy NLP (TrustNLP 2025)*, pages 103–111.
- Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. 2024. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *The Twelfth International Conference on Learning Representations (ICLR 2024)*. ICLR.

- Linhao Luo, Zicheng Zhao, Gholamreza Haffari, Yuanfang Li, Chen Gong, and Shirui Pan. 2025. Graph-constrained reasoning: Faithful reasoning on knowledge graphs with large language models. In *ICML 2025*.
- Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 1400–1409.
- Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. 2025. Graph retrieval-augmented generation: A survey. *ACM Transactions on Information Systems*, 44(2):1–52.
- Hannah Sansford, Nicholas Richardson, Hermina Peteric Maretic, and Juba Nait Saada. 2024. Grapheval: A knowledge-graph based llm hallucination evaluation framework. In *Proceedings of the Workshop on Knowledge-Infused Learning (KiL) co-located with the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 20–31. CEUR-WS.org.
- Apoorv Saxena, Adrian Kochsiek, and Rainer Gemulla. 2022. Sequence-to-sequence knowledge graph completion and question answering. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 2814–2828.
- Apoorv Saxena, Aditay Tripathi, and Partha 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*, pages 4498–4507.
- Mili Shah, Joyce Cahoon, Mirco Milletari, Jing Tian, Fotis Psallidas, Andreas Mueller, and Nick Litombe. 2024. Improving llm-based kgqa for multi-hop question answering with implicit reasoning in few-shot examples. In *Proceedings of the 1st Workshop on Knowledge Graphs and Large Language Models (KaLLM 2024)*, pages 125–135.
- Tiesunlong Shen, Jin Wang, Xuejie Zhang, and Erik Cambria. 2025. Reasoning with trees: faithful question answering over knowledge graph. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 3138–3157.
- Jiaxin Shi, Shulin Cao, Lei Hou, Juanzi Li, and Hanwang Zhang. 2021. Transfernet: An effective and transparent framework for multi-hop question answering over relation graph. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4149–4158.
- Yuan Sui, Yufei He, Nian Liu, Xiaoxin He, Kun Wang, and Bryan Hooi. 2025. Fidelis: Faithful reasoning in large language models for knowledge graph question answering. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 8315–8330.
- Haitian Sun, Tania Bedrax-Weiss, and William Cohen. 2019. PullNet: Open domain question answering with iterative retrieval on knowledge bases and text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2380–2390, Hong Kong, China. Association for Computational Linguistics.
- Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 4231–4242.
- Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. 2024. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *International Conference on Learning Representations*, volume 2024, pages 3868–3898.
- Yawei Sun, Lingling Zhang, Gong Cheng, and Yuzhong Qu. 2020. Sparqa: skeleton-based semantic parsing for complex questions over knowledge bases. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8952–8959.
- 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, Volume 1 (Long Papers)*, pages 641–651.
- Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiohu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2025. Retrieval-augmented generation with graphs (graphrag). *arXiv preprint arXiv:2501.00309*.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.
- Keheng Wang, Feiyu Duan, Sirui Wang, Peiguang Li, Yunsen Xian, Chuantao Yin, Wenge Rong, and Zhang Xiong. 2023. Knowledge-driven cot: Exploring faithful reasoning in llms for knowledge-intensive question answering. *arXiv preprint arXiv:2308.13259*.

- Shuai Wang and Yinan Yu. 2026. Kg-reasoner: A re-inforced model for end-to-end multi-hop knowledge graph reasoning. *arXiv preprint arXiv:2604.12487*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.
- Jiaqi Wu, Jiakang Li, Zefei Ning, and Li Wang. 2026. From relevance to utility: Faith-rank for utility-driven evidence re-ranking and reliable answering in retrieval-augmented generation. *Machine Learning*, 115(8):191.
- Rong Wu, Pinlong Cai, Jianbiao Mei, Licheng Wen, Tao Hu, Xuemeng Yang, Daocheng Fu, and Botian Shi. 2025. Kg-traces: Enhancing large language models with knowledge graph-constrained trajectory reasoning and attribution supervision. *arXiv preprint arXiv:2506.00783*.
- Yilin Xiao, Chuang Zhou, Qinggang Zhang, Bo Li, Qing Li, and Xiao Huang. 2026. Reliable reasoning path: Distilling effective guidance for llm reasoning with knowledge graphs. *IEEE Transactions on Knowledge and Data Engineering*.
- Zhentao Xu, Mark Jerome Cruz, Matthew Guevara, Tie Wang, Manasi Deshpande, Xiaofeng Wang, and Zheng Li. 2024. Retrieval-augmented generation with knowledge graphs for customer service question answering. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pages 2905–2909.
- Zhichao Yan, Jiapu Wang, Jiaoyan Chen, Xiaoli Li, Jiye Liang, Ru Li, and Jeff Z Pan. 2025. Atomic fact decomposition helps attributed question answering. *IEEE Transactions on Knowledge and Data Engineering*.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.
- Xi Ye, Semih Yavuz, Kazuma Hashimoto, Yingbo Zhou, and Caiming Xiong. 2022. Rng-kbqa: Generation augmented iterative ranking for knowledge base question answering. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 6032–6043.
- Wen-tau Yih, Matthew Richardson, Christopher 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.
- Donghan Yu, Sheng Zhang, Patrick Ng, Henghui Zhu, Alexander Hanbo Li, Jun Wang, Yiqun Hu, William Wang, Zhiguo Wang, and Bing Xiang. 2023. Decaf: Joint decoding of answers and logical forms for question answering over knowledge bases. In *Proceedings of the 11th International Conference on Learning Representations (ICLR 2023)*.
- Jing Zhang, Xiaokang Zhang, Jifan Yu, Jian Tang, Jie Tang, Cuiping Li, and Hong Chen. 2022. Subgraph retrieval enhanced model for multi-hop knowledge base question answering. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 5773–5784.

A Details of Logical Pivot Scoring

This section provides the detailed computation of relation and entity pivot scores used in Section 3.2.1. Given a candidate path

$$p = (e_0, r_1, e_1, \dots, r_m, e_m), \quad (23)$$

the goal of logical pivot scoring is to identify the relation and entity that contribute most critically to the reasoning process and the path-conditioned candidate answer $a(p)$.

For each relation r_j in path p , the relation pivot score is computed as

$$S_r(r_j; p, q) = \alpha_1 s_r(q, r_j) + \alpha_2 \rho(r_j, a(p)) + \alpha_3 I(r_j), \quad (24)$$

where $s_r(q, r_j)$ measures the semantic relevance between the question and the relation, $\rho(r_j, a(p))$ measures the structural proximity between the relation and the candidate answer, and $I(r_j)$ measures the information content of the relation. The weights α_1 , α_2 , and α_3 control the relative importance of these three factors.

Specifically, the question-relation relevance is defined as

$$s_r(q, r_j) = \cos(f_{\text{enc}}(q), f_{\text{enc}}(r_j)), \quad (25)$$

where f_{enc} denotes the text encoder. This term assigns higher scores to relations whose textual semantics are more consistent with the question.

The proximity between relation r_j and the candidate answer $a(p)$ is defined as

$$\rho(r_j, a(p)) = \frac{1}{1 + \min(\text{dist}_p(e_{j-1}, a(p)), \text{dist}_p(e_j, a(p)))}, \quad (26)$$

where $\text{dist}_p(\cdot, \cdot)$ denotes the distance between two entities within path p . Since relation r_j connects e_{j-1} and e_j , this term measures how close the relation is to the candidate answer through its two endpoint entities. A relation closer to $a(p)$ usually imposes a more direct constraint on the answer.

The information content of relation r is computed as

$$I(r) = -\log \frac{\text{freq}(r) + \epsilon}{\sum_{r' \in \mathcal{R}} \text{freq}(r') + \epsilon |\mathcal{R}|}, \quad (27)$$

where $\text{freq}(r)$ denotes the frequency of relation r in the knowledge graph, and ϵ is a smoothing term. Less frequent relations usually have stronger discriminative power and therefore receive higher information content scores.

For entity pivots, the starting entity and the path-conditioned candidate answer are excluded from the candidate set:

$$\mathcal{E}_{\text{pivot}}(p) = \mathcal{E}(p) \setminus \{e_0, a(p)\}. \quad (28)$$

This avoids conflating path attribution with direct replacement of the question entity or the predicted answer entity.

For each candidate entity $e_j \in \mathcal{E}_{\text{pivot}}(p)$, the entity pivot score is computed as

$$S_e(e_j; p, q) = \beta_1 s_e(q, e_j) + \beta_2 C(e_j) + \beta_3 A(e_j), \quad (29)$$

where $s_e(q, e_j)$ measures the semantic relevance between the entity and the question, $C(e_j)$ measures the coverage of question-related entity constraints, and $A(e_j)$ measures the structural connection between the entity and the candidate answer. The weights β_1 , β_2 , and β_3 control the contribution of these three factors.

The semantic relevance between the question and entity e_j is defined as

$$s_e(q, e_j) = \cos(f_{\text{enc}}(q), f_{\text{enc}}(e_j)). \quad (30)$$

This term favors entities whose textual semantics are more relevant to the question.

The entity constraint coverage score is defined as

$$C(e_j) = \frac{1}{|\mathcal{E}_q|} \sum_{e \in \mathcal{E}_q} \mathbf{1}[\text{dist}_G(e_j, e) \leq \delta], \quad (31)$$

where $\mathcal{E}_q$ denotes the set of entities mentioned in the question, $\text{dist}_G(\cdot, \cdot)$ denotes the shortest-path distance in the knowledge graph, and δ is a distance threshold. This term measures whether the candidate entity is structurally close to the constraint entities in the question.

The answer connection score is defined as

$$A(e_j) = \frac{1}{1 + \text{dist}_p(e_j, a(p))}. \quad (32)$$

This term assigns higher scores to entities that are closer to the candidate answer within the current path, since such entities usually provide more direct structural support for answer determination.

Based on these scores, the critical relation pivot and critical entity pivot are selected as

$$r^* = \arg \max_{r_j \in p} S_r(r_j; p, q), \quad (33)$$

and

$$e^* = \arg \max_{e_j \in \mathcal{E}_{\text{pivot}}(p)} S_e(e_j; p, q). \quad (34)$$

When $\mathcal{E}_{\text{pivot}}(p)$ is empty, only the critical relation pivot is used for counterfactual perturbation.

B Problem Definition

Given a knowledge graph $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$, where $\mathcal{E}$ denotes the set of entities, $\mathcal{R}$ denotes the set of relations, and $\mathcal{T} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ denotes the set of triples. Each triple $(h, r, t) \in \mathcal{T}$ represents a relation $r \in \mathcal{R}$ between a head entity $h \in \mathcal{E}$ and a tail entity $t \in \mathcal{E}$.

Given an input natural language question q , the goal of KGQA is to identify answer entities in the knowledge graph $\mathcal{G}$ that satisfy the semantic constraints of the question. Since a question may correspond to one or multiple correct answers, we denote the gold answer set as

$$\mathcal{Y}^* \subseteq \mathcal{E}. \quad (35)$$

The model is required to predict an answer set

$$\hat{\mathcal{Y}} \subseteq \mathcal{E}, \quad (36)$$

such that $\hat{\mathcal{Y}}$ is as close as possible to the gold answer set $\mathcal{Y}^*$.

Formally, KGQA can be formulated as a mapping from a question and a knowledge graph to a set of answer entities:

$$f : (q, \mathcal{G}) \rightarrow \hat{\mathcal{Y}}, \quad (37)$$

where $\hat{\mathcal{Y}}$ denotes the answer set predicted by the model. When the question has only one correct answer, $\mathcal{Y}^*$ degenerates into a singleton set. When the question corresponds to multiple correct answers, $\mathcal{Y}^*$ contains all entities that satisfy the semantic constraints of the question.

C Experimental Details

C.1 Datasets

To evaluate the reasoning ability of the proposed method on knowledge graph question answering tasks, we conduct experiments on two widely used KGQA benchmarks: WebQuestionsSP (WebQSP) (Yih et al., 2016) and Complex WebQuestions (CWQ) (Talmor and Berant, 2018). Both datasets use Freebase (Bollacker et al., 2008) as the factual knowledge source and cover different types of questions, ranging from simple relation queries to complex multi-hop reasoning.

- WebQuestionsSP (WebQSP) (Yih et al., 2016) is a standard benchmark widely used in KGQA research and contains 4,737 natural language questions. Questions in WebQSP typically start from a topic entity, and their answers can be obtained through relation paths

of no more than two hops in Freebase. Therefore, WebQSP is mainly used to evaluate model performance in entity linking, relation identification, and short-path reasoning scenarios.

- Complex WebQuestions (CWQ) (Talmor and Berant, 2018) is a complex question answering dataset constructed based on WebQSP. Compared with WebQSP, CWQ contains richer compositional semantic structures, such as constraints, entity expansion, and multi-relation composition, making it more suitable for evaluating complex KG reasoning. Answers in CWQ usually require reasoning along multi-hop paths in Freebase, with the longest reasoning distance reaching four hops. Due to its more complex question structures and longer reasoning chains, CWQ is commonly used to assess the generalization ability of KGQA models in complex semantic understanding and multi-hop reasoning scenarios.

To further characterize the evaluation setting, we report the distributions of answer counts and reasoning hops in Table 5 and Table 6, respectively. These statistics show that the two benchmarks differ not only in question complexity, but also in the number of valid answer entities.

C.2 Baselines

To comprehensively evaluate the effectiveness of our method, we compare it with 24 representative baselines. According to their modeling paradigms and ways of using knowledge, these methods are grouped into six categories: representation learning based methods, retrieval-augmented methods, semantic parsing methods, pure LLM-based methods, fine-tuned LLMs with KGs, and prompting-based LLMs with KGs.

1. Representation learning based methods. These methods perform answer reasoning over KGs mainly through continuous representations of entities, relations, or questions. KV-Mem (Miller et al., 2016) stores KG triples in a key-value memory network and performs multi-hop reasoning by repeatedly reading memory slots. EmbedKGQA (Saxena et al., 2020) formulates KGQA as a sequential link prediction problem based on entity and question representations. NSM (He

Dataset	#Ans = 1	$2 \leq \#Ans \leq 4$	$5 \leq \#Ans \leq 9$	#Ans ≥ 10
WebQSP	51.2%	27.4%	8.3%	12.1%
CWQ	70.6%	19.4%	6.0%	4.0%

Table 5: Distribution of Answer Counts in WebQSP and CWQ.

Dataset	1-hop	2-hop	≥ 3 -hop
WebQSP	65.49%	34.51%	0.00%
CWQ	40.91%	38.34%	20.75%

Table 6: Distribution of Question Hops in WebQSP and CWQ.

et al., 2021) adopts a neural-symbolic reasoning mechanism and models multi-hop reasoning through sequential state transitions. TransferNet (Shi et al., 2021) uses graph neural networks to model the semantic associations between questions and entities in the graph, and predicts answers based on graph-structured information. KGT5 (Saxena et al., 2022) converts KGQA into a text generation task by fine-tuning a sequence-to-sequence model on KGQA data to generate target answers.

2. Retrieval-augmented methods. These methods usually construct a question-relevant local subgraph from the KG and then perform answer reasoning over the retrieved subgraph. GraftNet (Sun et al., 2018) extracts a relevant subgraph from the KG based on linked entities in the question and predicts answers using graph-structured information. PullNet (Sun et al., 2019) further formulates subgraph construction as a learnable retrieval process, where an LSTM and graph neural network based retriever incrementally expands the question-relevant candidate subgraph. SR+NSM (Zhang et al., 2022) proposes a relation-path-based subgraph retrieval strategy to obtain local graph structures better suited for multi-hop reasoning, which are then fed into a neural-symbolic reasoning model. SR+NSM+E2E (Zhang et al., 2022) extends this framework with end-to-end training, enabling the subgraph retriever and downstream reasoner to be jointly optimized.
3. Semantic parsing methods. Semantic parsing methods typically translate natural language questions into structured queries and execute them over KGs to obtain answers. SPARQA

(Sun et al., 2020) designs a grammar-based framework for complex questions, using language patterns to capture high-level semantic structures and generate executable query expressions. QGG (Lan and Jiang, 2020) constructs query graphs aligned with question semantics by incrementally expanding relation paths and adding constraints. ArcaneQA (Gu and Su, 2022) adopts a dynamic query generation mechanism that generates subsequent queries based on intermediate results from previous reasoning steps, making it suitable for multi-stage reasoning over complex questions. RNG-KBQA (Ye et al., 2022) first enumerates candidate query structures, then ranks and filters them to select the highest-scoring query as the final output.

4. Pure LLM-based methods. These methods mainly rely on the language understanding, instruction following, and reasoning capabilities of LLMs, without explicitly introducing structured KG reasoning. FLAN-T5 (Chung et al., 2024) is obtained by multi-task instruction tuning based on the T5 architecture and shows strong generalization across natural language tasks. Alpaca (Taori et al., 2023) is built on LLaMA and fine-tuned with instruction-following data to improve responsiveness to user instructions. LLaMA2-Chat (Touvron et al., 2023) is optimized for dialogue scenarios through supervised fine-tuning and reinforcement learning from human feedback, making it suitable for interactive question answering. ChatGPT is a closed-source LLM with strong instruction understanding and complex task execution capabilities. ChatGPT+CoT (Wei et al., 2022) further introduces chain-of-thought prompting to elicit intermediate reasoning steps and improve performance on complex questions.
5. Fine-tuned LLMs with KGs. These methods combine the semantic understanding and generation capabilities of LLMs with structured knowledge from KGs to improve reasoning

reliability in complex KGQA tasks. KD-CoT (Wang et al., 2023) retrieves question-relevant knowledge from KGs and uses the structured evidence to generate faithful reasoning plans for LLMs, thereby guiding step-by-step reasoning. UniKGQA (Jiang et al., 2023b) integrates graph retrieval and answer reasoning into a unified framework, enabling knowledge localization and answer prediction within the same modeling process. DECAF (Yu et al., 2023) combines semantic parsing with LLM reasoning by jointly modeling structured query generation and answer inference, improving the model’s understanding and reasoning over complex questions. RoG (Luo et al., 2024) integrates LLMs with KG-based structured reasoning, retrieving question-relevant reasoning paths from KGs through planning, retrieval, and reasoning stages, and then using these paths to guide answer generation.

6. Prompting-based LLMs with KGs. These methods inject structured knowledge from KGs into LLMs mainly through prompting, aiming to improve knowledge utilization and reasoning ability in KGQA. ToG (Sun et al., 2024) performs question reasoning by iteratively exploring multiple potential reasoning paths over the KG and aggregating evidence retrieved from different paths, thereby improving LLM performance in multi-hop knowledge reasoning. FiDeLiS (Sui et al., 2025) first uses Path-RAG to pre-select candidate paths, and then combines step-wise beam search with a deductive scoring function to guide LLMs in validating the reasoning process. The search stops once the question becomes deducible, improving both factuality and interpretability. For a fair comparison, all prompting-based LLMs with KGs methods reported in Table 1, use gpt-4-turbo models as the backend LLMs.

These categories cover traditional KGQA paradigms, external knowledge retrieval based QA frameworks, semantic parsing approaches for logical-form generation, and recent LLM-centered KGQA methods. Comparing with these diverse baselines enables a systematic evaluation of our method in KG reasoning, answer generation, and complex question handling.

C.3 Evaluation Metrics

To ensure comparability with existing knowledge graph question answering methods, we evaluate model performance using Hits@1 and F1. Hits@1 focuses on the accuracy of the top-ranked predicted answer, measuring the proportion of questions for which the first-ranked candidate answer matches the gold answer. Since some questions may have multiple correct answers, relying solely on Top-1 accuracy may not sufficiently capture the model’s ability to cover the complete answer set. Therefore, we also report F1, which jointly measures the precision and completeness of the predicted answer set with respect to the gold answer set.

C.4 Implementation Details

We evaluate the method on the full test sets of WebQSP and CWQ, which contain 1,628 and 3,531 questions, respectively. During relation planning, we generate up to 6 candidate relation plans for WebQSP and 8 for CWQ. The maximum reasoning depth is set to 3 and 4, respectively, to accommodate the longer and more compositional reasoning chains in CWQ. For each planning step, the top 3 aligned KG relations are retained for path expansion. After executable path retrieval and initial path scoring, the top 3 candidate paths are preserved for subsequent forward support estimation, logical-pivot counterfactual attribution, and backward verification. These settings maintain a fixed candidate-path budget while allowing a larger planning space for the more complex CWQ questions.

C.5 Prompts

Table 7 presents the prompt templates used for relation path planning. The LLM is given the input question and the starting entity and is instructed to generate up to (N) relation-level reasoning plans. Each plan consists of a sequence of concise relation descriptions that specifies a possible reasoning direction from the starting entity toward the answer. The output is constrained to a predefined JSON format to facilitate automatic parsing and subsequent path retrieval.

D Comparison under Different Backbone LLMs

Table 8 reports the performance of our method with different backend large language models. As shown in the table, the performance of FBV-KGQA on both WebQSP and CWQ exhibits a stable

Component	Prompt template
Relation path planning	- You are given a question and a starting entity in a knowledge graph. - Generate up to {N} possible relation-level reasoning plans that can start from the entity and lead to the answer. - Each plan should be a sequence of short relation descriptions. - Return only JSON in the following format: {"plans": [{"relation 1", "relation 2"}, ...]}. Question: {question}. Starting entity: {starting_entity}.

Table 7: Prompt templates for relation planning and alignment.

Backend Models	WebQSP		CWQ	
	Hits@1 (%)	F1 (%)	Hits@1 (%)	F1 (%)
LLaMA2-Chat-7B	69.8	60.9	55.2	47.7
Mistral-7B	74.2	70.5	61.1	55.5
gpt-3.5-turbo	78.3	75.8	64.9	59.9
gpt-4-turbo	85.6	78.8	72.3	64.2

Table 8: Performance over different LLMs

upward trend as the capability of the backend model increases. On WebQSP, LLaMA2-Chat-7B achieves 69.8% Hits@1 and 60.9% F1, while using gpt-4-turbo improves the scores to 85.6% and 78.8%, corresponding to gains of 15.8% and 17.9%, respectively. On CWQ, LLaMA2-Chat-7B achieves 55.2% Hits@1 and 47.7% F1, whereas gpt-4-turbo reaches 72.3% Hits@1 and 64.2% F1, yielding improvements of 17.1% and 16.5%, respectively. These results indicate that our method does not rely on a single closed-source model and can still achieve strong performance with an open-source 7B model. Meanwhile, when the backend model has stronger semantic understanding and reasoning capabilities, our method can further unlock its potential for knowledge graph reasoning.

E Related Work

E.1 Knowledge Graph Enhanced LLM Question Answering

Knowledge graph question answering leverages structured knowledge graphs to support LLMs in reasoning and answering questions. Traditional KGQA methods mainly fall into two categories: semantic parsing based methods and retrieval augmented generation methods. Semantic parsing based methods translate natural language questions into executable queries and infer answers over KGs (Banerjee, 2023). These methods offer strong interpretability, but they are prone to failure when the generated logical queries are not executable (Bai et al., 2023). RAG methods retrieve relevant triples

from KGs as contextual evidence to assist LLMs in answer generation (Xu et al., 2024). Although such methods are more flexible, they may overlook the importance of KG structures for reasoning, especially relation paths in multi-hop question answering (Luo et al., 2024, 2025).

Recently, GraphRAG (Touvron et al., 2025) and RAS (Jiang et al., 2026) have proposed constructing task specific or query specific subgraphs for dynamic and structured retrieval and reasoning, thereby addressing the inefficiency and noise introduced by static global KGs. KG-Traces (Wu et al., 2025) introduces supervision over reasoning paths and attribution annotations, enabling LLMs to generate interpretable reasoning processes with or without access to KGs. FiDeLiS (Sui et al., 2025) combines deductive verification with beam search to improve both the verifiability and efficiency of KG reasoning paths.

For multi-hop KGQA, KG-Reasoner (Wang and Yu, 2026) and RWT (Shen et al., 2025) emphasize the interaction between LLMs and KGs. They perform multi-hop knowledge retrieval and continuous reasoning through iterative reasoning and graph search, such as Monte Carlo Tree Search (MCTS), thereby reducing the risk that early reasoning errors propagate to subsequent steps. In the biomedical domain, TripleCheck (González et al., 2025) proposes a post-hoc self organizing verification layer that decomposes LLM outputs into factual triples and cross validates them against large biomedical KGs, providing traceable evidence support.

E.2 LLM Interpretability and Attribution Reasoning

The interpretability of LLMs in complex multi-hop reasoning remains limited. Early prompting based methods, such as CoT (Wei et al., 2022) and ToT (Yao et al., 2023), guide LLMs to generate intermediate reasoning steps through step by step reasoning, but they lack factual verification and attribution mechanisms. Related studies on retrieval-augmented generation have also shown that evidence relevance alone does not necessarily imply answer-supporting utility, motivating utility-aware evidence selection for more reliable generation (Wu et al., 2026). To mitigate hallucinations and provide interpretable outputs, researchers have incorporated structured facts from KGs into the reasoning chains of LLMs (Jiang et al., 2023a).

GraphEval (Sansford et al., 2024) proposes a KG based hallucination evaluation framework that maps LLM outputs into structured triples and iteratively verifies their factuality, thereby providing an interpretable mechanism for hallucination detection and correction. Think&Cite (Li and Ng, 2025) combines self guided Monte Carlo Tree Search (SG-MCTS) with progress reward modeling for attributed text generation, enabling multi path and attributable text outputs. Yan et al. (Yan et al., 2025) propose an AQA retrieval and editing framework based on atomic fact decomposition, which decomposes long form answers into atomic facts and verifies them against external evidence, enabling transparent answer revision and traceability.

FiDeLiS (Sui et al., 2025) integrates deductive verification beam search (DVBS) into the LLM generation process by embedding KG derived verifiable reasoning steps, achieving a balance between faithfulness and efficiency. RWT (Shen et al., 2025) and KG-Reasoner (Wang and Yu, 2026) further demonstrate the feasibility of graph constrained reasoning, reinforcement learning, and backtracking mechanisms for multi-hop KG reasoning and continuous interpretable output generation.

Overall, research on LLM-based KGQA and interpretability exhibits three major trends: (1) leveraging structured facts in KGs to reduce hallucinations and support multi-hop reasoning; (2) incorporating supervision or reinforcement learning to enable explicit attribution and verification of reasoning paths; and (3) replacing static global KGs with dynamic, query specific subgraphs to improve efficiency and robustness. Nevertheless, existing

methods still face challenges such as early reasoning error propagation, incomplete KG coverage, and high computational cost, which provide important directions for future research.

F Case Study

Table 9 presents a representative example to illustrate how FBV-KGQA verifies candidate reasoning paths. For the question “What was Jesse James killed with?”, the system retrieves three candidate paths. Paths 1 and 3 lead to *Firearm*, while Path 2 leads to *Assassination*. Although all three paths are semantically related to the question, their reliability cannot be determined solely from the retrieved answers.

FBV-KGQA further evaluates these candidates through counterfactual perturbation and reverse verification. The critical relations in each path are replaced with alternative relations to examine whether the predicted answer depends on the original reasoning structure. Reverse questions are then constructed from the candidate answers to check whether the reasoning can be traced back to the original entity, Jesse James. After integrating forward support, counterfactual effects, and backward consistency, Paths 1 and 3 receive scores of 0.755 and 0.799, respectively, both supporting *Firearm*, whereas Path 2 receives a lower score of 0.630. The final prediction is therefore *Firearm*. This example illustrates how the proposed verification mechanism distinguishes between superficially plausible candidates and paths that provide stronger support for the final answer.

Item	Content
Question	What was jesse james killed with?
Retrieval paths	[Path 1] Jesse James -[people.deceased_person.cause_of_death]-> Firearm [Path 2] Jesse James -[people.deceased_person.cause_of_death]-> Assassination [Path 3] Jesse James -[people.deceased_person.cause_of_death]-> Firearm -[common.topic.image]-> HK USP 45
Counterfactual perturbation	[Path 1] Jesse James -[award.award_discipline.awards_in_this_discipline]-> Firearm [Path 2] Jesse James -[architecture.structure.architectural_style]-> Assassination [Path 3] Jesse James -[award.award_discipline.awards_in_this_discipline]-> Firearm -[common.topic.image]-> HK USP 45
Reverse question	[Path 1] What person was killed with a firearm? [Path 2] What person was assassinated? [Path 3] What person was killed with a firearm, as depicted in an image of an HK USP 45?
Score	[Path 1] Firearm : 0.755 [Path 2] Assassination : 0.630 [Path 3] Firearm : 0.799
Answer	Firearm

Table 9: Case study of forward-backward verification in FBV-KGQA.

Item	Content
Question	Who did cam newton sign with?
Retrieval paths	[Path 1] Cam Newton -[sports.pro_athlete.teams]-> m.0hpgnsj -[sports.sports_team_roster.team]-> Florida Gators football [Path 2] Cam Newton -[sports.pro_athlete.teams]-> m.0z23kt0 -[sports.sports_team_roster.team]-> Auburn Tigers football [Path 3] Cam Newton -[sports.pro_athlete.teams]-> m.04nb7yn -[sports.sports_team_roster.team]-> Carolina Panthers
Correct path	[Path 3] Cam Newton -[sports.pro_athlete.teams]-> m.04nb7yn -[sports.sports_team_roster.team]-> Carolina Panthers
Score	[Path 1] Florida Gators football : 0.649 [Path 2] Auburn Tigers football : 0.669 [Path 3] Carolina Panthers : 0.630
Answer	Auburn Tigers football

Table 10: Example of a path scoring failure in FBV-KGQA.

Item	Content
Question	What countries are part of the uk?
Retrieval paths	[Path 1] United Kingdom -[location.country.form_of_government]-> Parliamentary system -[government.form_of_government.countries]-> Canada [Path 2] United Kingdom -[location.country.form_of_government]-> Parliamentary system -[government.form_of_government.countries]-> Netherlands [Path 3] United Kingdom -[location.country.form_of_government]-> Parliamentary system -[government.form_of_government.countries]-> Antigua and Barbuda
Correct path	United Kingdom -[contains / constituent_country / administrative_division]-> Northern Ireland
Correct answer	Northern Ireland
Our answer	Netherlands

Table 11: Example of a path retrieval failure in FBV-KGQA.