Consistency-Aware Parameter-Preserving Knowledge Editing Framework for Multi-Hop Question Answering

Anonymous ACL submission

Abstract

Parameter-Preserving Knowledge Editing (PPKE) enables updating models with new information without retraining or parameter adjustment. Recent PPKE approaches used knowledge graphs (KG) to extend knowledge editing (KE) capabilities to multi-hop question answering (MHQA). However, these methods often lack consistency, leading to knowledge contamination, unstable updates, and retrieval behaviors that are misaligned with the intended edits. Such inconsistencies undermine the reliability of PPKE in multi-hop reasoning. We present CAPE-KG, Consistency-Aware Parameter-Preserving Editing with Knowledge Graphs, a novel consistency-aware framework for PPKE on MHQA. CAPE-KG ensures KG construction, update, and retrieval are always aligned with the requirements of the MHQA task, maintaining coherent reasoning over both unedited and edited knowledge. Extensive experiments on the MQuAKE benchmark show accuracy improvements in PPKE performance for MHQA, demonstrating the effectiveness of addressing consistency in PPKE. The code is available at this link.

1 Introduction

Knowledge editing (Wang et al., 2023) enables large language models (LLMs) to incorporate new or corrected information without the cost of full retraining. KE approaches can be divided into parameter-updating methods and parameter-preserving methods (Li et al., 2024). Parameter-updating methods, such as ROME (Meng et al., 2022), MEMIT (Meng et al., 2023), and T-Patcher (Huang et al., 2023), directly modify model weights to encode edits, which are computationally expensive. In contrast, PPKE avoids changing parameters and instead relies on external structures or contexts, offering greater transparency and stability. IKE (Zheng et al., 2023) edits knowledge through in-context demonstrations, MeLLo (Meng et al., 2023) supports multi-hop editing through query decomposition and fact retrieval, and PokeMQA (Gu et al., 2024) performs decomposition and conflict checking. PPKE offers a cheaper and more efficient paradigm by avoiding both retraining and parameter modification. While early efforts in PPKE primarily addressed simple single-hop factual updates (Zheng et al., 2023)(Zhang et al., 2024a), research has extended its scope to multi-hop question answering (MHQA) (Mavi et al., 2022)(Cheng et al., 2024), a more challenging setting that requires models to integrate edited knowledge with existing facts through multi-step reasoning. Recent work (Lu et al., 2025) adopts KG-based PPKE by injecting edited knowledge into a knowledge graph and retrieving relational paths, enabling models to perform explicit reasoning under edits for MHQA.

However, current KG-based PPKE is facing a critical weakness: lacking consistency (Cohen et al., 2024). Specifically, it struggles to maintain a knowledge boundary between factual and edited knowledge, to guarantee stable and conflict-free updates, and to ensure retrieval remains aligned with the intent of edit (Zhao et al., 2024). Inconsistency across these dimensions leads to less effective and reliable KE (Zhang et al., 2024b). In this paper, we propose CAPE-KG, Consistency-Aware Parameter-Preserving Editing with Knowledge Graphs, the first consistency-aware framework for PPKE. It improves consistency across the entire pipeline, ensuring that KG construction, knowledge update, and knowledge retrieval are all aligned with the PPKE objective. Our main contributions are:

1. We introduce a multi-layer KG architecture ensuring the consistent knowledge boundary between factual and edited knowledge during PPKE.
2. We design case-isolated updating mechanisms with conflict arbitration that guarantee edits are applied atomically and align with the case-

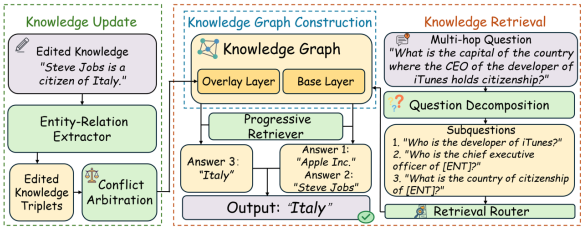

Figure 1: An overview of CAPE-KG, consists of three major components: the KG Construction module for building a pre-edit KG, the Knowledge Update module for incorporating edited knowledge, and the Knowledge Retrieval module for generating answers via retrieval.

isolated assumption.

3. We present an edit-aware retrieval module with progressive strategies and edit-related retrieval routing, ensuring the retrieval process is always consistent with the intended KE.
4. Extensive experiments on the MQuAKE benchmark (Zhong et al., 2023) demonstrate that CAPE-KG consistently outperforms existing baselines, achieving substantial gains in consistency and accuracy across backbones.

2 Problem Formulation

Let $\mathcal{F} = \{(s, r, o)\}$ denote the set of factual triples, where s is a subject, r a relation, and o an object; a knowledge edit is defined as $e = (s, r, o \rightarrow o^*)$, specifying that the mapping $(s, r) \mapsto o$ should be updated to a new object o^*, and $\mathcal{E} = \{e_1, e_2, \ldots, e_m\}$ denotes the set of such edits. PPKE adds the constraint of avoiding adjustments to the model's original parameters. For MHQA under PPKE, a multi-hop question Q requires reasoning over a chain of facts $C = \big[(s_1, r_1, o_1), (s_2, r_2, o_2), \ldots, (s_n, r_n, o_n)\big]$, where $s_{i+1} = o_i$ and the final answer is o_n; edits in $\mathcal{E}$ revise relevant triples in C to form $C^* = \big[(s_1, r_1, o_1'), (s_2, r_2, o_2'), \ldots, (s_n, r_n, o_n')\big]$, with $o_i' = o_i^*$. The objective is to ensure that given Q, $\mathcal{F}$, and $\mathcal{E}$, the PPKE method would output an answer consistent with C^*.

Assumptions We adopt the case-isolated edit, under which each edit is restricted to the current instance and does not influence other cases. (Zhong et al., 2023) This assumption aligns with the evaluation setting and reflects practical scenarios where distinct users or domains require isolated edits that must not interfere with unrelated reasoning chains.

3 Method

We propose the first consistency-aware PPKE framework, CAPE-KG, which addresses the fundamental challenges of three key dimensions of consistency. Fig.1 illustrates the overall pipeline, which consists of three major components: Knowledge Update, Knowledge Graph Construction, and Knowledge Retrieval. Each primarily ensures one form of consistency while reinforcing the others, resulting in a coherent editing framework. CAPE-KG maintains knowledge boundary consistency, where factual and edited knowledge are uncontaminated for reasoning; update consistency, where edits are incorporated in a nonconflicting and case-isolated manner; and intent consistency, where the whole retrieving process is always aligned with the purpose of editing.

3.1 Knowledge Graph Construction

The construction of KG establishes the knowledge boundary consistency foundation by explicitly separating facts from edited knowledge. Knowledge boundary inconsistency arises from knowledge contamination, as factual queries mistakenly retrieve edited knowledge from unrelated cases as factual information. CAPE-KG adopts a multi-layer architecture, which includes a base layer and corresponding overlay layers.

Base Layer. The base KG $\mathcal{B}$ is constructed solely from factual triples $\mathcal{F}$, dynamically derived for each batch of edit cases. By excluding all edited content, this layer establishes a consistent foundation for factual reasoning.

Overlay Layers. For each editing case c, we construct an overlay $O_c = \{(s, r, o \rightarrow o^*)\}$ containing only the edits associated with that case. Overlays are case-specific edit containers built on demand using a copy-on-write mechanism.

Knowledge Boundary Isolation. The separation between B and $\{O_c\}$ guarantees that factual and edited knowledge remain distinct with respect to the knowledge boundary. This KG construction prevents edited knowledge from being misinterpreted as factual information, thereby maintaining knowledge boundary consistency across all reasoning processes. It provides the foundation for knowledge boundary consistency, which is further reinforced during each update and retrieval.

3.2 Knowledge Update

Knowledge update ensures knowledge edits are incorporated in a stable and case-isolated manner. Given an input edit, we adopt the entity and relation detectors introduced in KEDKG (Lu et al., 2025) to extract a structured triple $\{(s, r, o^*)\}$, which is then

Backbone	Method	MQUAKE-CF-3K						MQUAKE-T			
		1 edited		100 edited		All edited		1 edited		All edited	
		M-Acc	H-Acc	M-Acc	H-Acc	M-Acc	H-Acc	M-Acc	H-Acc	M-Acc	H-Acc
LLaMA 2-7B	FT_{COT}*	22.30	-	2.13	-	OOM	-	47.32	-	3.75	-
	FT*	28.20	7.30	2.37	0.03	OOM	OOM	56.48	33.89	1.02	0.37
	ROME_{COT}*	11.17	-	2.87	-	2.77	-	28.96	-	14.40	-
	ROME*	13.13	5.37	3.50	0.03	3.63	0.10	24.89	17.99	1.71	0.32
	$\text{MEMIT}_{\text{COT}}$*	11.83	-	9.23	-	5.57	-	36.88	-	31.58	-
	MEMIT*	14.97	6.43	9.40	2.47	2.30	0.37	30.89	23.98	25.21	20.13
	MeLLo*	33.57	9.90	20.00	10.07	17.33	9.90	65.78	55.27	57.69	44.55
	PokeMQA*	44.13	30.90	37.33	27.83	32.83	23.87	75.43	60.44	74.36	60.22
	KEDKG*	66.80	63.67	58.50	55.37	48.30	43.90	73.13	69.06	71.15	66.76
	TEMPLE-MQA*	68.32	59.46	48.95	35.17	42.20	27.51	77.56	64.87	75.73	62.30
	CAPE-KG	**78.03**	**70.35**	**78.93**	**71.02**	**78.38**	**70.84**	**93.39**	**84.52**	**92.57**	**83.82**
Vicuna-7B	MeLLo*	30.70	20.84	24.75	12.25	22.35	10.85	60.72	48.55	51.55	42.97
	PokeMQA*	45.83	34.80	38.77	31.23	31.63	25.30	74.57	55.19	73.07	55.09
	KEDKG*	68.60	65.13	62.43	58.20	51.10	44.67	71.90	67.23	74.68	66.64
	TEMPLE-MQA*	71.61	62.75	56.65	44.26	46.60	37.33	81.77	69.46	78.29	68.15
	CAPE-KG	**81.89**	**78.74**	**81.59**	**78.85**	**80.94**	**78.49**	**96.85**	**87.25**	**94.58**	**86.23**
GPT-3.5 -turbo-instruct	MeLLo*	57.43	28.80	40.87	28.13	35.27	25.30	88.12	52.84	74.57	53.53
	PokeMQA*	67.27	56.37	56.00	49.63	45.87	39.77	76.98	68.09	78.16	67.88
	KEDKG*	68.00	65.33	59.50	56.80	49.10	43.17	78.75	76.18	77.19	73.77
	TEMPLE-MQA*	78.11	63.45	67.21	55.33	53.68	40.05	90.57	81.90	82.26	74.33
	CAPE-KG	**79.78**	**73.65**	**79.81**	**73.75**	**79.03**	**73.00**	**97.11**	**88.00**	**96.72**	**87.40**

Table 1: Main experimental results on the MQuAKE benchmarks (Zhong et al., 2023) across different backbone models. M-Acc and H-Acc denote multi-hop and holistic accuracy. Bold numbers indicate the best performance. Results marked with * are reported from prior work. "OOM" denotes out-of-memory failures due to GPU limits.

used to update the overlay of the current case. Unlike prior methods, where edits from different cases are repeatedly applied to a global graph, which leads to overwrites and unstable states, CAPE-KG enforces update consistency through a case isolation mechanism with conflict arbitration.

Updates are strictly limited to the overlay of the current case. By never propagating to the base layer or to overlays of other cases, edits remain aligned with the assumption that cases are independent. To enforce update consistency, the conflict arbitration module applies a deterministic rule: within each case, the current edit takes precedence for a given subject–relation pair; if no edit is specified, the original factual triple from the base layer is retained. This mechanism not only maintains update consistency but also reinforces the knowledge boundary separation established in construction, ensuring the boundary consistency throughout the system.

3.3 Knowledge Retrieval

In the retrieval stage, an input query Q is decomposed into a sequence of sub-questions $\{q_1, q_2, \ldots, q_n\}$ using a similarity-based dynamic few-shot strategy (Izacard et al., 2022). Existing methods often fall back to unedited factual knowledge through LLM responses when retrieval fails, which can produce answers that contradict the intended edit. To address this, our framework designs a retrieval process that explicitly routes queries between the base and overlay layers, applies progressive retrieval strategies when uncertainty arises, and suppresses irrelevant edits.

Edit-Aware Retrieval Routing. We define the edit impact surface for case c as $S^c_{\text{edit}} = \{s \mid (s, r, o^*) \in E^c\}$ and $P^c_{\text{edit}} = \{r \mid (s, r, o^*) \in E^c\}$, where E^c represents the edit set for case c. For each sub-question q_i, we determine the target knowledge layer:

$$L(q_i) = \begin{cases} \text{Overlay}^c, & \text{subject}(q_i) \in S^c_{\text{edit}} \\ & \quad \vee \text{relation}(q_i) \in P^c_{\text{edit}} \\ \text{Base}, & \text{otherwise} \end{cases} \quad (1)$$

This routing ensures that factual queries access the Base layer while edit-sensitive queries access the current case's Overlay layer.

Progressive Retrieval. When answering a sub-question, the system may encounter uncertainty regarding entity or relation matching. To address this, we introduce a progressive retrieval strategy with three stages.

High Confidence Stage: For entity candidates $K = \{k_1, \ldots, k_n\}$ with scores $g_\phi(q_i, k_j) \in [0, 1]$ generated by the detector, we first select those above a threshold τ, $K' = k_j \in K \mid g_\phi(q_i, k_j) \geq \tau$. We then filter out low outliers using mean μ and deviation σ, keeping $K'' = k_j \in K' \mid g_\phi(q_i, k_j) \geq \mu - \lambda\sigma$ where $\lambda > 0$ is adaptive. Edit irrelevance suppression is performed during entity scoring, where confidence scores of edited entities not explicitly referenced in the sub-question are down-weighted to prevent unintended influence. The remaining candidates are sorted and used sequentially for retrieval, ensuring stable results when multiple entities have similar scores.

Low Confidence Stage: When no candidates exceed the threshold, i.e., $K' = \emptyset$, we invoke a

Method			MQUAKE-CF-3K						MQUAKE-T			
KG Construction	Retrieval	Update	1 edited		100 edited		All edited		1 edited		All edited	
			M-Acc	H-Acc	M-Acc	H-Acc	M-Acc	H-Acc	M-Acc	H-Acc	M-Acc	H-Acc
–	✓	✓	75.00	69.00	74.67	69.04	75.67	69.00	96.00	86.60	95.80	86.60
✓	–	✓	53.33	43.00	51.33	46.67	51.93	46.00	87.33	81.67	87.67	82.00
✓	✓	–	77.40	71.00	74.58	68.14	74.29	68.10	96.60	87.10	94.35	85.60
✓	✓	✓	79.78	73.65	79.81	73.75	79.03	73.00	97.11	88.00	96.72	87.40

Table 2: Ablation study of CAPE-KG on MQuAKE-CF-3K and MQuAKE-T (Zhong et al., 2023). Each component incrementally contributes to improved consistency and accuracy, with the full framework achieving the best results.

language model to select the most plausible entity from the original candidate pool K.

Failure Stage: If no valid entity is retrieved, a relation detector checks whether the subquestion q_i falls within the scope of an edit. When relevant, the LLM is prompted with the full edited triple (s, r, o^*) injected into the context; otherwise, it generates an answer from q_i alone.

Together, intent consistency is ensured, where the reasoning process is always aligned with the purpose of the edit. Illustrative examples can be found in Appendix C.

4 Experiment

4.1 Experiment Setting

Baselines. We compare CAPE-KG against parameter-updating editors (FT (Zhu et al., 2020), ROME (Meng et al., 2022), MEMIT (Meng et al., 2023)) and parameter-preserving methods including MeLLo (Zhong et al., 2023), PokeMQA (Gu et al., 2024), KEDKG (Lu et al., 2025), and TEMPLE-MQA (Cheng et al., 2024). Additional details are provided in Appendix B.

Setup. Experiments are conducted on benchmarks MQuAKE-CF-3K and MQuAKE-T (Zhong et al., 2023); dataset details are in Appendix A. Retrieval parameters τ, λ are selected via grid search in Appendix G. Other details are in Appendix E.

Evaluation Metrics. The framework is evaluated by Match Accuracy (M-Acc), which measures the correctness of the final predicted answer against the benchmark ground truth, and Hop Accuracy (H-Acc), which measures the accuracy of intermediate hops against annotated ground-truth chains.

4.2 Results

Main Results. Table 1 shows that CAPE-KG consistently outperforms all baselines across all backbones and datasets. On average, we achieve an absolute improvement of 18.32% in M-Acc and 19.16% in H-Acc on MQuAKE-CF-3K, an absolute improvement of 14.17% in M-Acc and 10.14% in H-Acc on MQuAKE-T. These substantial gains demonstrate the effectiveness of enforcing multi-perspective consistency in PPKE for MHQA. The improvements hold across LLaMA 2 (Touvron et al., 2023), Vicuna (Zheng et al., 2024), and GPT-3.5 backbones, indicating that the performance gains in accuracy are model-independent and arise from this consistency-aware architecture.

Consistency Across Batch Edits. We observed accuracy drops in both M-Acc and H-Acc on baseline as edit batch size increases, due to overwrites and cross-case contamination. In addition to higher accuracy, CAPE-KG's multi-layer design with isolated overlays preserves consistency under larger batch sizes, leading to smaller fluctuations. The absolute gap between single-edit and all-edit settings across two datasets averages only 0.91% for M-Acc and 0.62% for H-Acc, further highlighting the improvement in consistency of our approach.

4.3 Ablation Studies

Table 2 shows the ablation study of CAPE-KG by disenabling three components: KG Construction, Knowledge Update, and Knowledge Retrieval. Performance improves steadily as more modules are introduced. Among them, the Retrieval module contributes the most significant gains, highlighting the importance of enforcing intent consistency. In contrast, KG Construction and Knowledge Update have relatively minor impacts, as their effectiveness depends on the prevalence of conflicts in the dataset. Nevertheless, the combination of all three modules results in the highest accuracy, demonstrating that multiperspective consistency is best achieved when the full framework is employed, making it the most effective PPKE approach for MHQA.

5 Conclusion

We propose CAPE-KG, which advances PPKE for MHKE by enforcing knowledge boundary consistency, update consistency, and intent consistency through components of the framework, including KG construction, knowledge update, and knowledge retrieval. In future work, we expect to extend CAPE-KG to continual knowledge editing, ensuring consistency as edits evolve over time.

Limitations

The progressive retrieval strategy introduces additional inference latency compared to single-pass retrieval, particularly under deeper reasoning chains. This overhead reflects a deliberate trade-off for improved consistency, and a detailed latency analysis is provided in Appendix F.

As mentioned in Section 2, our method relies on a case-isolated edit assumption adopted from prior work (Zhong et al., 2023), which is suitable for practical scenarios in which different users require distinct edits that must remain isolated to prevent cross-user contamination. However, this assumption does not hold in settings where knowledge updates are globally shared, which are beyond the scope of CAPE-KG.

References

Keyuan Cheng, Gang Lin, Haoyang Fei, Yuxuan Zhai, Lu Yu, Muhammad Asif Ali, Lijie Hu, and Di Wang. 2024. Multi-hop question answering under temporal knowledge editing. In *Proceedings of the Conference on Language Modeling*.

Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. 2024. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics*, 12:283–298.

Thomas M. Cover and Peter E. Hart. 1967. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27.

Hongyu Gu, Kai Zhou, Xu Han, Ning Liu, Runxin Wang, and Xuanjing Wang. 2024. Pokemqa: Programmable knowledge editing for multi-hop question answering. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pages 8069–8083. Association for Computational Linguistics.

Zhixiong Huang, Yichong Shen, Xiaozhi Zhang, Jie Zhou, Wenge Rong, and Zhiyuan Xiong. 2023. Transformer-patcher: One mistake worth one neuron. In *Proceedings of the Eleventh International Conference on Learning Representations*.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2022. Atlas: Few-shot learning with retrieval augmented language models. *Preprint*, arXiv:2208.03299.

Yanhong Li, Chunling Fan, Mingqing Huang, and Chengming Li. 2024. Learning from mistakes: A comprehensive review of knowledge editing for large language models. In *Proceedings of the 2024 IEEE International Conference on Smart Internet of Things*, pages 563–569.

Yifan Lu, Yigeng Zhou, Jing Li, Yequan Wang, Xuebo Liu, Daojing He, Fangming Liu, and Min Zhang. 2025. Knowledge editing with dynamic knowledge graphs for multi-hop question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Vaibhav Mavi, Ashish Jangra, and Adam Jatowt. 2022. A survey on multi-hop question answering and generation. *Preprint*, arXiv:2204.09140.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, volume 35, pages 17359–17372.

Kevin Meng, Akhil Sai Sharma, Alex J. Andonian, Yonatan Belinkov, and David Bau. 2023. Mass-editing memory in a transformer. In *Proceedings of the Eleventh International Conference on Learning Representations*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nicolas Bashlykov, Soumya Batra, Prakhar Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. 2023. Knowledge editing for large language models: A survey. *Preprint*, arXiv:2310.16218.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, and 1 others. 2024a. A comprehensive study of knowledge editing for large language models. *Preprint*, arXiv:2401.01286.

Zhuoran Zhang, Yongxiang Li, Zijian Kan, Keyuan Cheng, Lijie Hu, and Di Wang. 2024b. Locate-then-edit for multi-hop factual recall under knowledge editing. *Preprint*, arXiv:2410.06331.

Zihao Zhao, Yuchen Yang, Yijiang Li, and Yinzhi Cao. 2024. Ripplecot: Amplifying ripple effect of knowledge editing in language models via chain-of-thought in-context learning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6337–6347. Association for Computational Linguistics.

Cheng Zheng, Lei Li, Qiyu Dong, Yujia Fan, Zhengxuan Wu, Jingjing Xu, and Baobao Chang. 2023. Can we edit factual knowledge by in-context learning? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4862–4876. Association for Computational Linguistics.

Lianmin Zheng, Weng-Lian Chiang, Ying Sheng, Shiyi Zhuang, Zhengxuan Wu, Yiming Zhuang, Zi Lin, Zhuohan Li, Danyang Li, Eric Xing, and 1 others. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36.

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

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. 2020. Modifying memories in transformer models. *Preprint*, arXiv:2012.00363.

A Dataset

Table 3 summarizes the statistics of the two benchmark datasets used for evaluation, including MQuAKE-CF-3K and MQuAKE-T (Zhong et al., 2023). MQuAKE-CF-3K is constructed from counterfactual knowledge edits to evaluate whether models can propagate hypothetical factual changes through multi-hop reasoning, containing 3,000 questions spanning 2–4 hops across multiple edited facts. MQuAKE-T consists of 1,868 instances derived from real-world temporal knowledge updates, assessing reasoning under time-sensitive factual changes.

Dataset	#Edits	2-hop	3-hop	4-hop	Total
MQuAKE-CF-3K	1	513	356	224	1093
	2	487	334	246	1067
	3	–	310	262	572
	4	–	–	268	268
	All	1000	1000	1000	3000
MQuAKE-T	1 (All)	1421	445	2	1868

Table 3: Statistics of the multi-hop question answering datasets MQuAKE used in our experiments.

B Baselines

We compare CAPE-KG with two categories of baseline methods: (1) parameter-updating knowledge editing methods and (2) parameter-preserving knowledge editing methods. All baselines are evaluated under the same experimental settings for fair comparison.

B.1 Parameter-Updating Editing Methods

FT (Zhu et al., 2020) fine-tunes model parameters on edited facts via gradient-based optimization to incorporate new knowledge.

ROME (Meng et al., 2022) identifies the model parameters associated with a given factual relation and applies a low-rank update to the feed-forward network of a selected transformer layer to encode the edited knowledge.

MEMIT (Meng et al., 2023) extends ROME by applying coordinated low-rank updates across multiple feed-forward layers of the transformer to encode edited knowledge.

B.2 Parameter-Preserving Editing Methods

MeLLo (Zhong et al., 2023) adopts a plan-and-solve framework that decomposes multi-hop questions into sub-questions and retrieves edited facts to guide intermediate reasoning without modifying model parameters.

PokeMQA (Gu et al., 2024) extends MeLLo by introducing a detached scope detector that determines whether a sub-question is affected by edited knowledge and modulates the retrieval process accordingly.

KEDKG (Lu et al., 2025) encodes edited knowledge in an external knowledge graph and performs graph-based retrieval and reasoning at inference time without modifying model parameters.

TEMPLE-MQA (Cheng et al., 2024) models temporal dependencies in multi-hop question answering and incorporates edited knowledge through structured retrieval while preserving model parameters.

C Illustrative Analysis of CAPE-KG Consistency

C.1 Knowledge Boundary Consistency

CAPE-KG has effectively improved the knowledge boundary consistency by avoiding possible contamination. In the case shown in Fig. 2, prior work has the potential inconsistency of mixing unrelated edits from other cases, (Black Pink, genre, EDM), into reasoning as fact. EDM is retrieved instead of K-pop, which breaks the reasoning chain, while CAPE-KG maintains knowledge boundary consistency by using this multi-layer design.

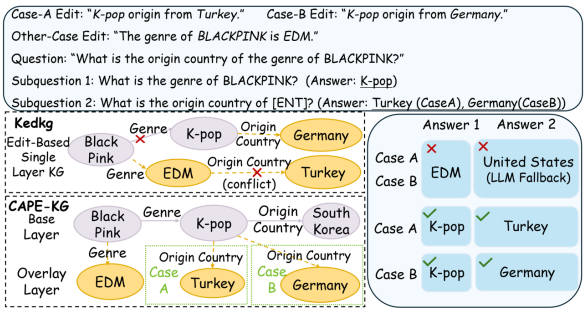

Figure 2: Case Comparison of Update and KG Construction. Unedited knowledge is labeled in purple. Edited entities are in yellow, with dotted lines indicating the edited relation.

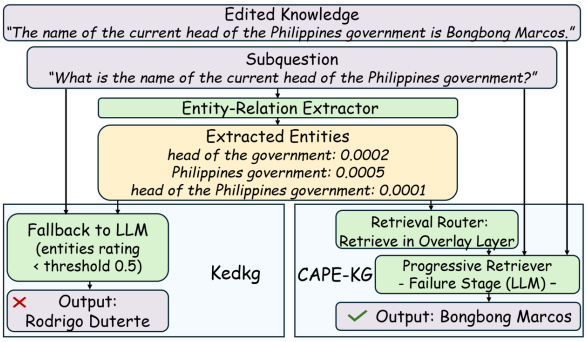

Figure 3: Case Comparison of Intent Consistency in Retrieval.

C.2 Update Consistency

CAPE-KG prevents the possible inconsistency caused by overwriting conflicting edits in a global KG. In the case shown in Fig. 2, two edits (K-pop, origin country, Turkey), (K-pop, origin country, Germany), so one remains while the other is discarded. CAPE-KG stores each edit in a case-specific overlay, preserving both without interference. This design keeps per-case reasoning aligned with the case-isolated edit assumption and ensures update consistency.

C.3 Intent Consistency

As shown in Fig. 3, when a subquestion is directly related to an edited fact, prior work falls back to LLM answering directly under low-confidence entity extraction. This fallback would query unedited factual knowledge, which is inconsistent with the intent of editing. In contrast, CAPE-KG routes edit-related queries to the overlay layer via the retrieval router, ensuring that edits are explicitly applied in reasoning. Moreover, at the failure stage of progressive retrieval, the LLM is also queried, but with the edit injected into the query context. This guarantees that even when a component of CAPE-KG is uncertain, the reasoning process remains aligned with the intended update, in this way preserving intent consistency.

D Prompts

CAPE-KG has three prompt templates: one for multi-hop question decomposition (Table 4) and two for sub-question answering (Table 5, Table 6).

For question decomposition, we adopt a retrieval-based few-shot prompting strategy in Table 4 using a K-nearest-neighbor (KNN) (Cover and Hart, 1967) search over a fixed reference pool. Few-shot question decomposition demos are drawn from MQuAKE-CF (Zhong et al., 2023) after removing overlaps with evaluation data. For each input question, the top-k most similar examples (k=6 in our experiment) are retrieved based on semantic similarity computed using frozen sentence embeddings. The retrieved demonstrations are attached to a fixed instruction template to form the final decomposition prompt, providing contextually relevant guidance without updating model parameters.

For sub-question answering, when a sub-question does not fall within the scope of any edit, and no valid entity can be retrieved from the knowledge graph, the language model is prompted to directly answer the sub-question based on its internal factual knowledge, without any injected edits. In this case, we use an instruction-based prompt in Table 5 with fixed few-shot demonstrations to encourage concise, entity-level outputs that can be consumed by subsequent reasoning steps. This fallback mechanism enables the model to provide missing factual information when explicit knowledge retrieval fails, while avoiding the introduction of unintended edits.

For sub-question answering, when a sub-question is edit-relevant and knowledge graph retrieval fails, we adopt an in-context learning formulation following prior work (Zheng et al., 2023) to inject edited knowledge into the language model input. Specifically, the edited triple (s, r, o^*) is filled into a fixed instruction template, as shown in Table 6, and explicitly injected into the input context. Such edit-aware prompting ensures that fallback answering remains aligned with the intended update even when explicit retrieval fails, while preserving strict case isolation and parameter preservation.

You are an expert at decomposing multi-hop questions into sequential sub-questions.
Each sub-question should be answerable independently, and the answer from one sub-question
is passed to the next sub-question using the [ENT] placeholder.

Rules:
1. Break down complex questions into simple, sequential sub-questions
2. Use [ENT] to pass entities between sub-questions
3. Each sub-question should focus on one specific fact
4. Maintain logical flow from general to specific
5. Avoid redundant or irrelevant questions

Question:
««DEMO QUESTION 1»»
Sub-question:
««DEMO SUB-QUESTION 1»»

Question:
««DEMO QUESTION 2»»
Sub-question:
««Demo SUB-QUESTION 2»»

[Additional dynamically retrieved demonstrations omitted]

Decompose the following question:
Question:
««QUESTION»»
Sub-question:

Table 4: Prompt template for multi-hop question decomposition.

You are an expert at answering factual questions with precise and concise entity names.
Your answers should be specific entities that can be used in subsequent questions.

Rules:
1. Answer with the most specific and complete entity name
2. Use proper capitalization and full names
3. Avoid articles (a, an, the) unless part of the name
4. Be consistent with entity naming
5. If multiple answers exist, choose the most relevant one
6. If uncertain, provide the most likely answer

Examples:
Question: Which country was jazz created in?
Answer: United States of America

Question: What is the capital of France?
Answer: Paris

Question: Who wrote the play "Hamlet"?
Answer: William Shakespeare

[Additional in-context demonstrations omitted]

Answer the following question:
Question: ««SUB-QUESTION»»
Answer:

Table 5: Prompt template for factual sub-question answering.

New Fact: ««EDITED KNOWLEDGE»»

Question: ««SUB-QUESTION»»
Answer:

Table 6: Prompt template for edit-relevant sub-question answering.

E Details about Experiments

All experiments in this paper are conducted on English-language benchmarks MQuAKE (Zhong et al., 2023), and all language models and prompts operate exclusively in English. Experiments are run on a single machine equipped with an NVIDIA RTX 5080 GPU. LLaMA-2-7B (Touvron et al., 2023) and Vicuna-7B (Zheng et al., 2024) with 7 billion parameters each, and GPT-3.5-turbo-instruct with an undisclosed parameter size, are used as backbone models. For each experimental setting, we conduct a single full run over the entire benchmark and report accuracy averaged across all instances. Accordingly, all reported M-Acc and H-Acc results correspond to single-run benchmark-level averages, without additional variance estimates.

F Progressive Retrieval Latency

While progressive retrieval enhances answer accuracy by dynamically updating the knowledge graph during multi-hop reasoning, it introduces additional computational overhead. To quantify this computational overhead, we conduct a latency analysis comparing our full method with progressive retrieval against a variant without progressive updates. We report a detailed latency breakdown in Table 7. As discussed in the main paper, progressive retrieval gradually retrieves and integrates evidence across multiple reasoning steps, which is essential for enforcing consistency under multi-hop reasoning and multiple edits. Our ablation studies confirm its effectiveness for the target task.

To analyze this overhead, we evaluate inference latency under both progressive retrieval and single-pass retrieval across different edit batch sizes on MQuAKE-CF-3K and MQuAKE-T (Zhong et al., 2023). We report two latency metrics: (1) the average end-to-end latency per multi-hop question, and (2) the average latency per sub-question. All experiments are conducted using GPT-3.5-turbo-

	Single-Pass	Progressive	MQuAKE-CF-3K			MQuAKE-T	
			1 edited	100 edited	All edited	1 edited	All edited
Avg. Latency per Multi-Hop QA (ms)	✓		5177.98	4737.47	5061.52	4776.70	4588.83
		✓	6203.85	7174.07	6041.60	6342.65	6071.88
Avg. Latency per Sub-Question (ms)	✓		136.98	133.21	172.68	146.16	139.93
		✓	201.63	226.34	193.77	238.63	227.58

Table 7: Latency comparison between single-pass and progressive retrieval across edit batch sizes on MQuAKE benchmarks (Zhong et al., 2023).

instruct as the backbone language model, under the same datasets, edit configurations, and experimental settings as the main experiments.

As shown in Table 7, progressive retrieval results in higher inference latency than single-pass retrieval. This latency is expected, as progressive retrieval performs complex retrieval and reasoning steps to enforce consistency under multi-hop edits. While this introduces additional computational cost, inference efficiency is not the primary objective of this work. Importantly, our ablation studies demonstrate that progressive retrieval is essential for achieving reliable and consistent multi-hop knowledge editing, and we therefore view the observed latency as a reasonable and deliberate trade-off for improved effectiveness.

G Hyperparameter Settings

We conduct a hyperparameter analysis for τ and λ, with values selected via grid search using GPT-3.5-turbo-instruct on MQuAKE-CF-3K (Zhong et al., 2023) under the single-edit setting. Table 8 reports the results for λ, while Table 9 summarizes the evaluation of τ. Based on this analysis, we set $\tau = 0.4$ and $\lambda = 1.0$ in all main experiments.

λ	M-Acc (%)
0.5	77.83
1.0	**79.78**
1.5	79.46
2.0	79.28
2.5	78.52

Table 8: Hyperparameter analysis for λ.

τ	M-Acc (%)
0.1	77.58
0.2	78.36
0.3	79.52
0.4	**79.78**
0.5	79.05
0.6	79.30
0.7	78.93
0.8	78.38
0.9	77.75

Table 9: Hyperparameter analysis for τ.

H Scientific Artifacts Used

MQuAKE. We use the MQuAKE benchmark (Zhong et al., 2023), includes MQuAKE-CF-3K and MQuAKE-T, for evaluating multi-hop question answering under knowledge editing. According to the official release, the MQuAKE datasets and accompanying code are released under the MIT License. Their use in this work complies with its terms for academic research purposes. The benchmark is constructed from structured knowledge sources rather than user-generated text. Based on the dataset documentation and its established use in prior work, we verified that it does not contain personal data or identifiable individual information, and therefore no anonymization or offensive-content filtering was required.

Pretrained Language Models. We use pretrained large language models as backbones in our experiments, including LLaMA-2-7B (Touvron et al., 2023), Vicuna-7B (Zheng et al., 2024), and GPT-3.5-turbo-instruct. LLaMA 2 and Vicuna are released under the LLaMA 2 Community License, and GPT-3.5-turbo-instruct is accessed via API in compliance with OpenAI's terms of use. All models are used without modifying their original parameters and are used for academic research purposes, which is consistent with their intended use. As pretrained models are used without access to or modification of their training data, no additional checks or anonymization related to personal or offensive content are applicable at the model level.