CMIKE: Collaborative Multi-Agent Instructional Knowledge Editing in Language Models

Anonymous ACL submission

Abstract

Knowledge editing adjusts specific parts of a large language model's (LLMs) pre-trained knowledge, allowing it to answer queries based on updated information without fine-tuning. Existing methods of modifying parameters or using manual prompt engineering have limitations in interpretability and adaptability. This paper introduces a multi-agent framework for instruction-based knowledge editing that leverages the self-reflection and correction capabilities of LLMs agent to automatically optimize knowledge instructions and uses their instruction-following capabilities to enable knowledge injection. Our approach employs three collaborative agents in a built-in feedback mechanism: a Knowledge Gatekeeper assessing question-knowledge relevance, a Knowledge Utilizer receiving instructional knowledge to answer queries, and a Knowledge Crafter generating and optimizing instructions based on the Utilizer's performance. This iterative process forms a dynamic improvement loop, enabling efficient adaptation to diverse knowledge editing scenarios. Experimental results show that our approach outperforms existing prompting-based methods, and achieves comparable performance to parameter-modifying methods in terms of editing success rate, portability, and locality without human intervention.

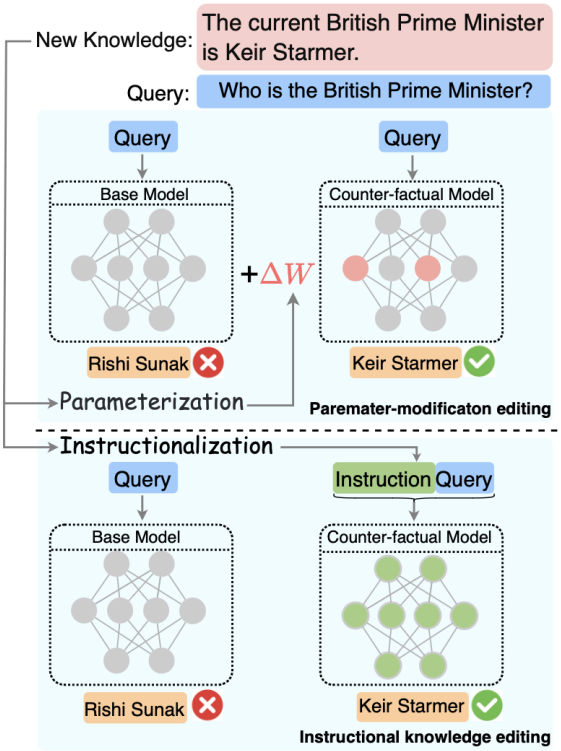

Figure 1: A comparison of parameter-modification and instructional knowledge editing: The former injects new knowledge by modifying model parameters, while the latter guides the model to absorb new knowledge at the input level.

1 Introduction

Large Language Models (LLMs) have become a prevalent tool for information retrieval due to their extensive knowledge storage capabilities (Petroni et al., 2019; Roberts et al., 2020; Jiang et al., 2020). To address outdated or incorrect information, knowledge editing techniques have been developed (Wang et al., 2023a; Feng et al., 2023; Zhang et al., 2024a). These methods seek to make precise local adjustments to LLMs' internal knowledge, creating a counterfactual model that accurately answers in-scope queries about the updated information while preserving its ability to correctly respond to out-of-scope queries (Yao et al., 2023; Wang et al., 2024b; Mitchell et al., 2022).

The mainstream approach assumes that knowledge is implicitly encoded in model parameters and transforms textual knowledge into parameter updates to achieve knowledge editing, as shown in Figure 1 top. Methods for modifying parameters usually include deriving a closed-form solution for localized parameters (Meng et al., 2022a,b), deriving parameter updates via meta-learning (De Cao et al., 2021; Mitchell et al., 2021), tailored fine-tuning on curated data (Jiang et al., 2024; Qi

et al., 2024) and introducing additional parameters (Hartvigsen et al., 2023; Wang et al., 2024c)

However, the parameter-knowledge correlation remains poorly understood, undermining method reliability. Recent studies challenge the effectiveness of parameter localization (Hase et al., 2024) and the simplicity of the knowledge neuron theory (Niu et al., 2024). Moreover, even single edits can introduce conflicts, distorting the model's knowledge structure (Li et al., 2023b; Yang et al., 2024).

A promising alternative is the prompting-based editing approach, which avoids parameter modification by leveraging the model's inherent context learning capabilities, essentially treating knowledge updates as a language task rather than a numerical optimization problem. This type of method imparts new knowledge to the model through meticulously crafted instruction or examples, ultimately creating a task-driven, context-guided counterfactual model (Zheng et al., 2023; Tian et al., 2024; Madaan et al., 2022; Chen et al., 2024; Li et al., 2023a), as shown in Figure 1 bottom. However, these methods demand extensive manual prompt engineering to craft precise instructions or examples for high-quality editing. These include crafting precise instructions and examples that contain relevant knowledge, maintaining prompt diversity to enhance generalization, and carefully designing prompts to preserve unrelated knowledge.

To overcome these limitations, this paper proposes CMIKE, a Collaborative Multi-Agent Instructional Knowledge Editing framework, which comprehensively utilizes the information extraction ability, self-reflection ability, and instruction following ability of LLM to achieve efficient, accurate and flexible knowledge editing.

Unlike previous editing methods that burden one LLM with both knowledge absorption and scope differentiation capabilities, CMIKE achieves a clear division of labor. Given new knowledge and a query, the Gatekeeper first determines if the query falls within the scope of knowledge editing. Unlike previous methods that rely on separately trained classifiers or routers (Wang et al., 2024c; Hartvigsen et al., 2023; Mitchell et al., 2022; Zhao et al., 2024), the Gatekeeper is an agentic decision tree system. Each agent node analyzes and compares queries and knowledge from different dimensions to make fine-grained decisions. Experimental results show that gatekeepers can greatly improve the locality of knowledge editing.

The Knowledge Utilizer is the base model to be edited, which receives carefully designed knowledge instructions and is transformed into a counterfactual model. The Knowledge Crafter converts raw knowledge into actionable instructions, ensuring effective knowledge integration. These two agents operate in a dynamic, iterative loop, where the crafter provides refined instructions, and the utilizer attempts to answer queries. Based on discrepancies between responses and expected outcomes, the crafter refines instructions.

Experimental results show that the Gatekeeper has a strong range discrimination ability, which improves the locality performance of knowledge editing. Utilizer and Crafter can work together to generate high-quality instructions in continuous iterations and continuously get improved editing performance.

2 Related Work

2.1 Knowledge Editing

The purpose of knowledge editing is to obtain a counterfactual model in which some stored knowledge is changed. Based on the view that knowledge is stored in implicit model parameters, many methods for modifying parameters have been proposed. (Meng et al., 2022a,b) locates entity-related knowledge in multi-layer perceptron layers and edits knowledge by modifying their parameters. (Li et al., 2024a) modifies the parameters of the attention head and shows good editing effects. Meta-learning approaches(Mitchell et al., 2021; De Cao et al., 2021) design a mapping function that transfers decomposed gradient into updated values of weight parameters to insert new knowledge. (Ni et al., 2023) proposes to forget old knowledge and update new knowledge by using parameter arithmetic operations. (Huang et al., 2023; Hartvigsen et al., 2023; Wang et al., 2024c) proposes introducing new knowledge by adding new neurons or parameters as the side memory and fine-tuning them.

Some approaches treat knowledge editing as a context-driven task. (Zheng et al., 2023) allows models to remember new knowledge through manually designed diverse context question-answer examples. (Tian et al., 2024) introduces task-specific instructions to guide the editing process and shows that the instruction editing can generalize to unseen tasks.

2.2 Automatic Prompt Engineer

Automatic prompt engineering (APE) replaces the manual prompt crafting process, leveraging the model's own capabilities to generate, evaluate, and iteratively refine prompts (Zhou et al., 2022; Kong et al., 2024).

(Haviv et al., 2021) trains a BERT-based rewriter model that optimizes query prompt reformulation directly for the knowledge extraction performance. (Kong et al., 2024) use reinforcement learning training an LLM rewriter to generate K candidate rewritten prompts, and the best-performing prompt is selected as the final one. (Shin et al., 2020) uses a gradient-based search algorithm to iteratively refine the prompt by repeatedly replacing tokens, aiming to increase the likelihood of producing the desired output. While effective, this method requires full access to gradients, limiting its use to larger models only available through APIs. (Wang et al., 2024d) adapts the Mixture-of-Experts paradigm to prompt optimization, partitioning the problem space into multiple regions with specialized expert prompts to enhance the performance across diverse tasks.

Although these APE algorithms have proven effective, they still rely on specific design choices and often require significant human expertise, limiting their adaptability and automation.

2.3 LLMs-based Agent

Large language models, with their pre-trained knowledge and reasoning capabilities, can perceive the environment, make decisions and perform actions autonomously, and have become the core of intelligent agents (Wang et al., 2023c; Huang et al., 2022; Chen et al., 2023; Wang et al., 2024a; Li et al., 2024b). By combining prompt engineering and environmental feedback, the agent can self-evaluate and effectively reflect, continuously improving its decision-making and action capabilities. (Yao et al., 2022) introduces REACT, demonstrating how carefully designed prompts enable LLMs to seamlessly integrate reasoning and acting capabilities, showcasing the potential for self-reflection and iterative problem-solving. (Shinn et al., 2023) proposes Reflexion, a framework that enhances language agents' performance through verbal reinforcement and self-reflection, storing linguistic feedback in episodic memory to guide future decisions. (Zhang et al., 2024b) propose an agent that learns and evolves through interactions in complex dynamic scenarios, using belief-aware decision-making, policy-level reflection, and search-based policy evolution.

Inspired by agents' self-reflection and evolution capabilities, we propose a multi-agent framework for instruction knowledge editing, aiming to continually enhance instruction effectiveness through collaborative agent interactions.

3 Task Formulation

Knowledge editing updates pre-trained knowledge triples (s, r, o) to target (s, r, o^*), where s, r, o are subject, relation, and object. An edit $e = (s, r, o \rightarrow o^*)$ represents this update. Query-answer pairs are denoted as $(q(s, r), o)$. The process transforms base model $\mathcal{M}$ into counterfactual model $\mathcal{M}^*$, where $\mathcal{M}^*(q(s, r)) = o^*$. Our agent-based approach generates instructions $\mathcal{I}$ to guide this transformation: $\mathcal{M}(\mathcal{I}, q(s, r)) = \mathcal{M}^*(q(s, r)) = o^*$. Knowledge editing algorithms are typically evaluated on four key dimensions:

Edit Success This metric assesses knowledge editing effectiveness by measuring the average top-1 accuracy of the post-edit model $\mathcal{M}^*$ on edit cases.

$$\mathbb{E}_{(o^*,q)} \mathbb{1} \left\{ \arg\max_{o} \mathcal{M}^*(o|q(s, r)) = o^* \right\} \quad (1)$$

Portability This metric assesses knowledge transferability to related queries. It's measured as the average top-1 accuracy of the counterfactual model on equivalent prompts of $q_E(s, r)$, such as logically related queries.

$$\mathbb{E}_{(o^*,q_E)} \mathbb{1} \left\{ \arg\max_{o} \mathcal{M}^*(o|q_E(s, r)) = o^* \right\} \quad (2)$$

Locality This metric evaluates edit precision by measuring how well unrelated knowledge is preserved. It's calculated as the top-1 accuracy of unchanged predictions on samples unrelated to edit cases $q_l(s, r)$.

$$\mathbb{E}_{(o,q_l)} \mathbb{1} \left\{ \arg\max_{o} \mathcal{M}^*(o|q_l(s, r)) = o \right\} \quad (3)$$

Fluency This metric assesses the linguistic quality of post-edit model output, focusing on coherence and diversity. It's calculated as the weighted average of bi- and tri-gram entropies to measure fluency.

$$-\frac{2}{3} \sum_{k} f_2(k) \log_2 f_2(k) + \frac{4}{3} \sum_{k} f_3(k) \log_2 f_3(k) \quad (4)$$

where $f_n(\cdot)$ is the n-gram frequency distribution.

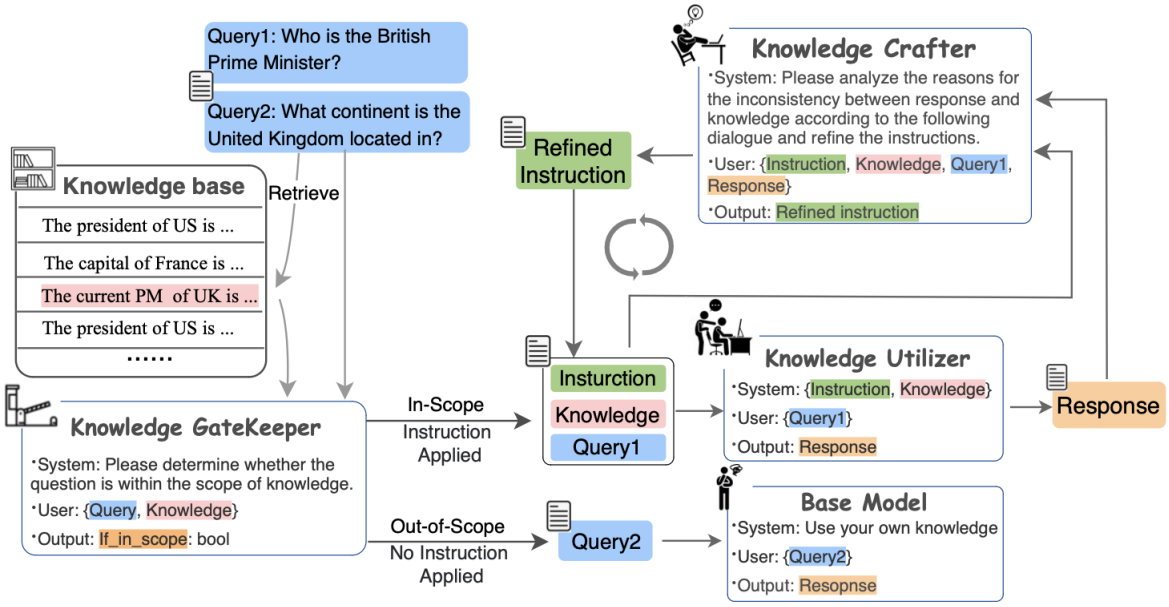

Figure 2: The proposed collaborative multi-agent framework illustration. Each agent is constructed using tailored task descriptions and exemplar system prompts, with a distinct functional role within the framework. The system prompts shown in the figure are abbreviated; full details are provided in the appendix.

4 Method

This section presents an overview of CMIKE, describes its three constituent agents in detail, and then elucidates their collaborative process in executing knowledge editing.

4.1 Multi-Agent Framework

Figure 2 illustrates CMIKE structure, comprising three key agents: Knowledge Gatekeeper $\mathcal{A}_G$, Knowledge Utilizer $\mathcal{A}_U$, and Knowledge Crafter $\mathcal{A}_C$. $\mathcal{A}_G$, a classifier agent, determines if questions fall within the new knowledge's editing scope, ensuring precise updates. $\mathcal{A}_U$, the base model being edited, transforms into a counterfactual model applying updated information based on received instructions. $\mathcal{A}_C$ generates efficient knowledge instructions, guiding $\mathcal{A}_U$ to answer questions using new knowledge. Importantly, $\mathcal{A}_C$ analyzes incorrect responses, reflects on causes, and optimizes instructions accordingly. Each agent is constructed using tailored task descriptions and exemplar prompts, leveraging LLMs' task comprehension and few-shot learning capabilities. The construction details are in the appendix.

4.2 Knowledge Gatekeeper

The ability to precisely discern whether a query falls within or outside the scope of a given knowledge piece is crucial for assessing knowledge editing efficacy, particularly in terms of locality and portability. Therefore, we introduce the Knowledge Gatekeeper $\mathcal{A}_G$ agent, utilizing a hierarchical decision tree structure with independent sub-agents at each node, as shown in Figure 3.

The root node, a multi-hop query judger, classifies queries as multi-hop or single-hop. Multi-hop queries require sequential reasoning steps involving interconnected knowledge triples. This initial classification optimizes complexity diversion and promotes semantic matching between subsequent queries and related knowledge.

For multi-hop queries, the subsequent child node is a reasoning path analyzer that evaluates the relevance of knowledge and determines whether the knowledge can facilitate complex logical reasoning paths. The sub-node agent first decomposes the reasoning path and then determines whether the provided knowledge can supplement the missing information in the path, such as intermediate entities, or whether the reasoning path can be extended.

For single-hop queries, the subsequent sub-node consists of two agents, including a semantic analyzer and a logical analyzer, which jointly evaluate the relevance between knowledge and questions (Jiang et al., 2022). The semantic analyzer performs entity-relation matching. It first extracts entities and relations from both the knowledge and queries, then conducts semantic matching between them. The logic analyzer examines the logical relationships between the knowledge and the query,

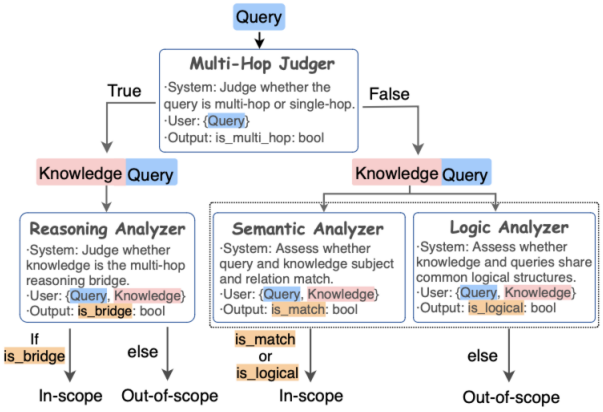

Figure 3: The illustration of $\mathcal{A}_G$'s decision tree architecture that can determine whether a query is within the scope of knowledge.

assessing whether the given knowledge contributes to the logical reasoning of the query. It can identify complex logical structures, such as bidirectional relationships exemplified by "if A is B, then B is A". If either the semantic analyzer finds a match or the logical analyzer identifies a relevant logical relationship, the query is considered to be within the scope of the given knowledge.

4.3 Knowledge Utilizer

The Knowledge Utilizer, which is the base model receiving instructions, primarily applies newly acquired information to answer questions, thereby transforming into an edited counterfactual model. Acting as a student, it assimilates knowledge instructions from the knowledge crafter to respond to input queries.

The utilizer's responses undergo evaluation. If the editing success rate falls below a set threshold, the system relays the response, instruction, knowledge, and query back to the crafter for refinement. The utilizer then reattempts the query with the refined instructions. This refinement cycle continues until accuracy exceeds the threshold, signaling successful knowledge editing.

4.4 Knowledge Crafter

The Knowledge Crafter serves as the core of our knowledge editing framework, analogous to the editor in previous non-agent methods. Its key role is to craft executable instructions embedding new knowledge, enabling the knowledge utilizer to answer queries using this updated knowledge.

Drawing inspiration from the reflection mechanism of LLMs (Shinn et al., 2023), we endow the crafter with the ability to reflect and improve, enabling it to operate in an iterative feedback loop. This process mirrors a teacher's approach to education: when the utilizer fails to answer questions based on the current instruction, the crafter analyzes the incorrect response, considers the reasons for failure, and refines the instructions accordingly.

In each iteration, the crafter receives an edit history containing the current instruction, new knowledge, specific questions, and incorrect answers generated by the utilizer. It then analyzes the error's cause, contemplates how to optimize the instruction to prevent similar mistakes, and produces an improved version. This iterative optimization process yields instructions that bridge the gap between raw, obscure knowledge and newly actionable information, enabling the utilizer to effectively assimilate this knowledge.

4.5 Inference Working Flow

The inference process of CMIKE follows a systematic pipeline, as illustrated in Algorithm 1.

The process begins with a set of input queries $q_{0:N}$ and a retrieved new knowledge t^* to be incorporated. For each query, the Knowledge Gatekeeper $\mathcal{A}_G$ first determines whether the query falls within the scope of the new knowledge.

For in-scope queries, an iterative optimization loop begins. The Knowledge Utilizer $\mathcal{A}_U$ answers the query using the current instruction I_t and new knowledge t^*. If the accuracy of the answer falls below a predefined threshold τ, the Knowledge Crafter $\mathcal{A}_C$ is invoked to refine the instruction. It uses a package of information, including the current instruction, new knowledge, a query, and the user's answer, to generate an improved instruction. This loop continues until either the accuracy threshold is met or a maximum number of iterations T is reached.

For out-of-scope queries, the knowledge utilizer answers based on its pre-trained knowledge. This process ensures efficient knowledge incorporation while maintaining performance on unrelated queries.

5 Experiments

Model We choose Llama2-7b-chat (Touvron et al., 2023) and Qwen-7b-chat (Bai et al., 2023) as the base models to be edited. We choose GPT-4o-mini (gpt4o-mini-2024-07-18) as the brain of the Knowledge Crafter and Gatekeeper.

Algorithm 1 Multi-Agent Knowledge Editing Pipeline

Input: $\mathcal{A}_C$, $\mathcal{A}_U$, $\mathcal{A}_G$, new_knowledge t^*, Queries $q_{0:N}$, Initial_instruction I_0, threshold τ.
Output: Final_instruction I_T, Final_answer

```
for query in queries do
    is_in_scope ← A_G(t*, query)
    if is_in_scope then
        t ← 0, I_t ← I_0
        answer, acc ← A_U(I_t, t*, query)
        while acc < τ and t < T do
            I_t ← A_C(I_t, t*, query, answer)
            answer, acc ← A_U(I_t, t*, query)
            t ← t + 1
        end while
        Final_answer ← answer
    else
        Final_answer ← A_U(query)
    end if
end for
return I_T, Final_answers
```

Data We evaluate the performance of our method on KnowEdit (Wang et al., 2023b) and use the test sets of four popular benchmarks, including WikiData$_{counterfact}$, WikiData$_{recent}$ (Cohen et al., 2024) and ZsRE (Levy et al., 2017).

Baselines We choose two types of baseline knowledge editing methods: the parameter-modification method and the prompting-based method.

• Parameter-modifying method: ROME (Meng et al., 2022a), MEMIT(Meng et al., 2022b), SERAC (Mitchell et al., 2022), MEND (Mitchell et al., 2021), FT-L (Meng et al., 2022a), LTE (Jiang et al., 2024). These methods all achieve knowledge editing by fine-tuning or modifying model parameters.

• Prompting-based method: ICE (Zheng et al., 2023), EREN (Chen et al., 2024). ICE obtains in-context learning demonstrations by reorganizing new knowledge into manually designed QA pairs. EREN prompts the model by filling new knowledge into predefined instruction templates.

5.1 Single Editing Results

Table 1 shows the comparison results of CMIKE and other knowledge editing methods in the single editing setting. CMIKE outperforms other prompt-based methods in all metrics, which shows that iterative prompt updating by multi-agent collaboration is better than manual prompt design. CIKE leads by a large margin in the locality metric, indicating that the gatekeeper can accurately filter out out-of-scope queries. Compared with the Llama model, Qwen is inferior in portability and fluency, which may be due to Qwen's poor ability to follow English instructions.

Compared with the parameter modification-based approach, CMIKE achieves the same performance as SERAC in terms of locality. Note that SERAC manually trains a classifier to make range judgments, while CMIKE uses a gatekeeping agent to make the judgments automatically. In terms of editing success rate, CMIKE also achieved results comparable to those of the most advanced methods, such as SERAC and LTE. This shows that the method of using instruction prompts can inject new knowledge as effectively as modifying parameters but without the disadvantage of lack of interpretability.

In terms of portability, although CMIKE outperforms the prompt-based methods, it is inferior to LTE, a fine-tuning-based method. This is because LTE fine-tunes the model on carefully designed parallel datasets that contain relevant queries that require reasoning and knowledge combination. CMIKE only optimizes instructions based on the new knowledge itself, without introducing a mechanism to enhance reasoning capabilities.

Can iteration improve editing performance? As the Knowledge Crafter iteratively optimizes instructions for new knowledge, a key question arises: Does the editing effect improve with more iterations? Figure 4 shows the line chart of the editing success rate as the number of iterations changes.

We observe that as the number of iterations increases, the editing success rate and portability both improve with little loss in fluency, indicating that the Crafter agent is continuously generating more effective instructions. Notably, while editing success shows rapid improvement, the slower increase in portability suggests that enhancing performance across diverse queries necessitates the development of more sophisticated and nuanced instructions.

Also, note that editing faces bottlenecks in success rate and portability. Performance plateaus after 5-7 iterations, indicating that the model's instruction-following capability has been fully leveraged.

Table 1: Performance comparison on single knowledge editing, where "Recent" and "Counterfact" refer to WikiData$_{recent}$ and WikiData$_{counterfact}$, respectively.

Model	Dataset	Metric	Parameter-modified							Prompting-based		
			ROME	MEMIT	FT	FT-L	SERAC	LTE	LTE-LoRA	ICE	EREN	**CMIKE**
Llama2-7b-chat	Counterfact	Edit succ	83.21	83.41	26.78	51.12	99.99	100.00	99.97	69.83	61.43	99.86
		Portability	38.69	40.09	16.94	39.07	76.07	89.69	85.74	45.32	38.78	73.15
		Locality	65.40	63.68	0.29	62.51	98.96	84.76	85.11	32.38	28.38	88.73
		Fluency	578.84	568.58	483.71	544.80	549.91	586.76	570.64	547.22	564.45	581.14
	Recent	Edit succ	85.08	85.32	31.24	71.18	98.68	99.99	99.97	60.74	54.34	98.14
		Portability	37.45	37.94	15.91	48.71	63.52	91.51	81.87	36.93	31.21	69.29
		Locality	66.20	64.78	3.65	63.70	100.00	85.67	82.72	33.34	29.56	85.41
		Fluency	574.28	566.66	428.67	549.35	553.19	586.76	570.64	531.01	555.14	591.41
	ZsRE	Edit succ	96.57	83.07	36.88	54.65	99.67	99.91	99.91	66.01	65.21	91.04
		Portability	52.20	51.43	8.72	45.02	56.48	78.98	79.63	63.94	43.34	62.76
		Locality	27.14	25.46	0.31	71.12	30.23	71.78	67.99	23.14	53.63	83.84
		Fluency	570.47	559.72	471.29	474.18	410.89	583.70	544.52	541.14	513.21	571.67
	Average	Edit succ	89.98	86.52	47.64	60.81	99.51	99.94	99.90	73.03	60.33	96.35
		Portability	42.78	43.15	13.86	44.27	65.36	86.73	82.41	48.73	37.78	68.04
		Locality	51.43	51.37	4.41	64.37	74.75	80.62	77.03	34.19	37.19	85.93
		Fluency	585.21	577.90	493.22	543.08	530.24	593.60	575.31	563.07	544.27	581.41
Qwen-7b-chat	Counterfact	Edit succ	83.21	83.41	26.78	51.12	99.99	100.00	99.97	69.83	58.83	98.63
		Portability	38.69	40.09	16.94	39.07	76.07	89.69	85.74	45.32	45.32	68.15
		Locality	65.40	63.68	0.29	62.51	98.96	84.76	85.11	32.38	32.38	91.73
		Fluency	578.84	568.58	483.71	544.80	549.91	586.76	570.64	547.22	554.45	564.14
	Recent	Edit succ	85.08	85.32	31.24	71.18	98.68	99.99	99.97	60.74	56.37	97.16
		Portability	37.45	37.94	15.91	48.71	63.52	91.51	81.87	36.93	38.53	69.29
		Locality	66.20	64.78	3.65	63.70	100.00	85.67	82.72	33.34	28.38	89.41
		Fluency	574.28	566.66	428.67	549.35	553.19	586.76	570.64	531.01	564.62	541.47
	ZsRE	Edit succ	96.57	83.07	36.88	54.65	99.67	99.91	99.91	66.01	53.14	92.63
		Portability	52.20	51.43	8.72	45.02	56.48	78.98	79.63	63.94	47.21	62.76
		Locality	27.14	25.46	0.31	71.12	30.23	71.78	67.99	23.14	29.45	90.84
		Fluency	570.47	559.72	471.29	474.18	410.89	583.70	544.52	541.14	546.35	559.42
	Average	Edit succ	99.45	96.84	26.28	45.66	99.12	99.63	99.59	82.26	56.11	96.14
		Portability	46.08	42.73	11.14	34.67	67.99	86.48	84.30	59.85	43.67	66.73
		Locality	45.42	59.19	15.01	86.42	76.37	84.19	81.90	64.32	30.07	90.66
		Fluency	592.90	601.77	532.85	590.05	555.65	613.31	586.20	572.50	555.14	555.01

5.2 Memory Batch Editing Results

Batch editing, allowing simultaneous modification of multiple knowledge pieces, better aligns with practical needs than single editing. Therefore, we enhance our multi-agent framework with an external memory bank to store batch knowledge for editing. The modularity of our framework simplifies the evaluation, which only requires evaluating the Gatekeeper's ability to identify scoped knowledge from the memory, given different query variants, including:

• Subject Alias: Replaces query subject with an alias to test recognition of synonymous concepts.

• Relation specification: Alters query relation to assess understanding of relational concepts.

• Multi-hop reasoning: Requires reasoning based on multiple knowledge elements to evaluate complex reasoning decomposition.

• Logical association: Demands retrieval of related knowledge for logical reasoning, testing recognition of logical connections.

We package all new knowledge of the ZsRE dataset into a memory bank. For each query, we use Sentence-BERT [1] to retrieve the top-20 candidate knowledge via embedding matching. The Gatekeeper then judges them individually and calculates the accuracy of in-scope knowledge identification.

Table 2 shows the performance evaluation with varying numbers of demonstration examples in each sub-agent's prompt. The Gatekeeper exhibits robust performance across different subject aliases and relations, achieving approximately 70% accuracy in a zero-shot setting and improving to over 90% with three-shot learning. This impressive performance stems from the underlying LLM's

[1] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

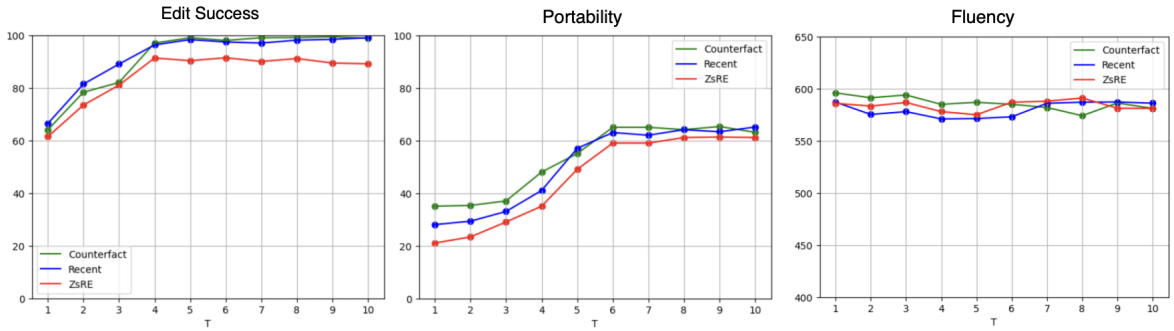

Figure 4: The line chart of editing effects changes with the number of iterations.The editing success rate and portability both increase with the number of iterations, with less loss of fluency.

Table 2: Gatekeeper accuracy in identifying in-scope knowledge across diverse query variants

Dataset	Query-type	zero-shot	one-shot	two-shot	three-shot
Counterfact	Subject-alias	68.86	81.78	86.42	91.46
	Relation_change	76.54	87.45	93.45	98.00
	Multi-hop	37.97	51.58	62.85	72.51
	Logical	35.45	56.24	59.34	66.34
Recent	Subject-alias	69.45	79.46	85.34	93.42
	Relation_change	75.35	81.42	88.42	95.53
	Multi-hop	38.42	55.22	65.27	71.56
	Logical	36.45	52.35	58.52	62.46
ZsRE	Subject-alias	64.48	64.52	72.53	89.34
	Relation_change	77.63	78.34	87.42	96.24
	Multi-hop	31.33	47.75	56.34	67.19
	Logical	35.39	53.34	61.34	65.52

strong information extraction capabilities, particularly in entity recognition and relationship extraction (Dagdelen et al., 2024). These strengths allow the Gatekeeper to effectively process relevant information.

Logic-related and multi-hop queries pose greater challenges for identification, reflecting LLMs' weakness in logic analysis. In the zero-shot setting, retrieval accuracy for these complex queries is only 30%, highlighting LLMs' difficulty with complex logical structures without prompting examples. However, with 3-shot prompting, accuracy more than doubles to nearly 70%, demonstrating LLMs' strong few-shot learning capability.

6 Limitation

Although CMIKE demonstrates strong performance in knowledge editing, it has notable limitations. The Gatekeeper's tree-like workflow, while aiming for precise decisions, introduces significant complexity and computational overhead. Its hierarchical structure of independent sub-agents performs intricate analyses, increasing processing time. These challenges are amplified in batch editing scenarios, where the Gatekeeper must analyze multiple pieces of candidate knowledge independently, substantially increasing decision-making time. Furthermore, CMIKE's instruction update mechanism focuses on adding new knowledge but lacks a clear method for removing outdated information. This absence of a forgetting mechanism could lead to conflicts between old and new knowledge or the accumulation of obsolete data. CMIKE's editing performance still marginally trails the cutting-edge parameter modification methods in certain aspects, highlighting areas for future refinement and optimization in the agentic framework.

7 Conclusion

In this paper, we propose CMIKE, a Collaborative Multi-Agent Instructional Knowledge Editing framework. By leveraging a team of specialized agents and the instruction-following capabilities of LLMs, CMIKE offers a flexible, interpretable, and effective approach to knowledge editing without direct parameter modification. CMIKE comprises a Knowledge Gatekeeper, Knowledge Utilizer, and Knowledge Crafter. The Gatekeeper determines whether the query is within the scope of knowledge editing, and the Crafter and Utilizer work in a feedback iteration mechanism, reflecting on the reasons for editing failures and continuously optimizing editing instructions. Experimental results demonstrate the Gatekeeper's robust scope discrimination, enhancing editing locality, and the synergistic iterations between Utilizer and Crafter generate high-quality instructions, continuously improving editing performance. CMIKE showcases the potential of leveraging automated agent frameworks to enhance and reimagine traditional LLMs tasks.

References

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chen Qian, Chi-Min Chan, Yujia Qin, Yaxi Lu, Ruobing Xie, et al. 2023. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors in agents. *arXiv preprint arXiv:2308.10848*, 2(4):6.

Yingfa Chen, Zhengyan Zhang, Xu Han, Chaojun Xiao, Zhiyuan Liu, Chen Chen, Kuai Li, Tao Yang, and Maosong Sun. 2024. Robust and scalable model editing for large language models. *arXiv preprint arXiv:2403.17431*.

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.

John Dagdelen, Alexander Dunn, Sanghoon Lee, Nicholas Walker, Andrew S Rosen, Gerbrand Ceder, Kristin A Persson, and Anubhav Jain. 2024. Structured information extraction from scientific text with large language models. *Nature Communications*, 15(1):1418.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. *arXiv preprint arXiv:2104.08164*.

Zhangyin Feng, Weitao Ma, Weijiang Yu, Lei Huang, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2023. Trends in integration of knowledge and large language models: A survey and taxonomy of methods, benchmarks, and applications. *arXiv preprint arXiv:2311.05876*.

Thomas Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. 2023. Aging with grace: lifelong model editing with discrete key-value adaptors. corr, abs/2211.11031, 2022. doi: 10.48550. *arXiv preprint arXiv.2211.11031*.

Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. 2024. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. *Advances in Neural Information Processing Systems*, 36.

Adi Haviv, Jonathan Berant, and Amir Globerson. 2021. BERTese: Learning to speak to BERT. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3618–3623, Online. Association for Computational Linguistics.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International conference on machine learning*, pages 9118–9147. PMLR.

Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. 2023. Transformer-patcher: One mistake worth one neuron. *arXiv preprint arXiv:2301.09785*.

Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Unikgqa: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph. *arXiv preprint arXiv:2212.00959*.

Yuxin Jiang, Yufei Wang, Chuhan Wu, Wanjun Zhong, Xingshan Zeng, Jiahui Gao, Liangyou Li, Xin Jiang, Lifeng Shang, Ruiming Tang, et al. 2024. Learning to edit: Aligning llms with knowledge editing. *arXiv preprint arXiv:2402.11905*.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Weize Kong, Spurthi Amba Hombaiah, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. 2024. Prewrite: Prompt rewriting with reinforcement learning. *arXiv preprint arXiv:2401.08189*.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada. Association for Computational Linguistics.

Daliang Li, Ankit Singh Rawat, Manzil Zaheer, Xin Wang, Michal Lukasik, Andreas Veit, Felix Yu, and Sanjiv Kumar. 2023a. Large language models with controllable working memory. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1774–1793, Toronto, Canada. Association for Computational Linguistics.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. 2024a. Pmet: Precise model editing in a transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18564–18572.

Yuanchun Li, Hao Wen, Weijun Wang, Xiangyu Li, Yizhen Yuan, Guohong Liu, Jiacheng Liu, Wenxing Xu, Xiang Wang, Yi Sun, et al. 2024b. Personal llm agents: Insights and survey about the capability, efficiency and security. *arXiv preprint arXiv:2401.05459*.

Zhoubo Li, Ningyu Zhang, Yunzhi Yao, Mengru Wang, Xi Chen, and Huajun Chen. 2023b. Unveiling the pitfalls of knowledge editing for large language models. *arXiv preprint arXiv:2310.02129*.

Aman Madaan, Niket Tandon, Peter Clark, and Yiming Yang. 2022. Memory-assisted prompt editing to improve gpt-3 after deployment. *arXiv preprint arXiv:2201.06009*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022a. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. 2022b. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. 2021. Fast model editing at scale. *arXiv preprint arXiv:2110.11309*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. 2022. Memory-based model editing at scale. In *International Conference on Machine Learning*, pages 15817–15831. PMLR.

Shiwen Ni, Dingwei Chen, Chengming Li, Xiping Hu, Ruifeng Xu, and Min Yang. 2023. Forgetting before learning: Utilizing parametric arithmetic for knowledge updating in large language models. *arXiv preprint arXiv:2311.08011*.

Jingcheng Niu, Andrew Liu, Zining Zhu, and Gerald Penn. 2024. What does the knowledge neuron thesis have to do with knowledge? *arXiv preprint arXiv:2405.02421*.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

Siyuan Qi, Bangcheng Yang, Kailin Jiang, Xiaobo Wang, Jiaqi Li, Yifan Zhong, Yaodong Yang, and Zilong Zheng. 2024. In-context editing: Learning knowledge from self-induced distributions. *arXiv preprint arXiv:2406.11194*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Noah Shinn, Federico Cassano, Beck Labash, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning.(2023). *arXiv preprint cs.AI/2303.11366*.

Bozhong Tian, Siyuan Cheng, Xiaozhuan Liang, Ningyu Zhang, Yi Hu, Kouying Xue, Yanjie Gou, Xi Chen, and Huajun Chen. 2024. Instructedit: Instruction-based knowledge editing for large language models. *arXiv preprint arXiv:2402.16123*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Cunxiang Wang, Xiaoze Liu, Yuanhao Yue, Xiangru Tang, Tianhang Zhang, Cheng Jiayang, Yunzhi Yao, Wenyang Gao, Xuming Hu, Zehan Qi, et al. 2023a. Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. *arXiv preprint arXiv:2310.07521*.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024a. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Mengru Wang, Yunzhi Yao, Ziwen Xu, Shuofei Qiao, Shumin Deng, Peng Wang, Xiang Chen, Jia-Chen Gu, Yong Jiang, Pengjun Xie, et al. 2024b. Knowledge mechanisms in large language models: A survey and perspective. *arXiv preprint arXiv:2407.15017*.

Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. 2024c. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. *arXiv preprint arXiv:2405.14768*.

Peng Wang, Ningyu Zhang, Bozhong Tian, Zekun Xi, Yunzhi Yao, Ziwen Xu, Mengru Wang, Shengyu Mao, Xiaohan Wang, Siyuan Cheng, et al. 2023b. Easyedit: An easy-to-use knowledge editing framework for large language models. *arXiv preprint arXiv:2308.07269*.

Ruochen Wang, Sohyun An, Minhao Cheng, Tianyi Zhou, Sung Ju Hwang, and Cho-Jui Hsieh. 2024d. One prompt is not enough: Automated construction of a mixture-of-expert prompts. In *International Conference on Machine Learning*.

Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. 2023c. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *arXiv preprint arXiv:2302.01560*.

Wanli Yang, Fei Sun, Xinyu Ma, Xun Liu, Dawei Yin, and Xueqi Cheng. 2024. The butterfly effect of model editing: Few edits can trigger large language models collapse. *arXiv preprint arXiv:2402.09656*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2023. Editing large language models: Problems, methods, and opportunities. *arXiv preprint arXiv:2305.13172*.

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

Wenqi Zhang, Ke Tang, Hai Wu, Mengna Wang, Yongliang Shen, Guiyang Hou, Zeqi Tan, Peng Li, Yueting Zhuang, and Weiming Lu. 2024b. Agent-pro: Learning to evolve via policy-level reflection and optimization. *arXiv preprint arXiv:2402.17574*.

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.

Ce Zheng, Lei Li, Qingxiu Dong, Yuxuan Fan, Zhiyong Wu, Jingjing Xu, and Baobao Chang. 2023. Can we edit factual knowledge by in-context learning? *arXiv preprint arXiv:2305.12740*.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*.

A Appendix

A.1 Prompt details for agent construction

Figure 5 shows the prompt of Knowledge Crafter. Figure 6, 7, 8, 9, 10 show the prompt of Multi-hop Judger, Reasoning Analyzer, Subject Analyzer, Relation Analyzer, and Logic Analyzer, which are all sub-agents of the Knowledge Gatekeeper. Each prompt includes a task description, exampler demonstrations, and data format.

A.2 Case Study

We show examples to illustrate that as the number of iterations increases, instructions are gradually optimized.

A.2.1 Case1

• Knowledge: The name of the field of work of Aime K Johnson is house cat.

• Query: The name of the field of work of Aime K Johnson is?

• Instruction$_0$: Answer the following question based on the new knowledge.

• Instruction$_1$: Answer the following question by primarily utilizing the new knowledge provided. Ensure that your reasoning reflects the new information as the foundation of your response. If there is any conflict with prior knowledge, prioritize the new knowledge. Clearly connect your answer to the specific piece of new knowledge relevant to the query.

• Instruction$_2$: Answer the following question by solely utilizing the new knowledge provided about Aime K Johnson. Your reasoning must demonstrate that the new knowledge serves as the foundation of your response. If your existing understanding conflicts with this new information, place greater importance on the new knowledge. Each element of your answer should explicitly reference the new knowledge relevant to the query

A.2.2 Case2

• Knowledge: The name of the founder of Archäologisches Museum der Stadt Kelheim is Kelheim.

• Query: The name of the founder of Archäologisches Museum der Stadt Kelheim is?

• Instruction$_0$: Answer the following question based on the new knowledge.

• Instruction$_1$: Answer the following question by integrating the new knowledge provided.

• Instruction$_2$: Answer the following question by thoroughly integrating the new knowledge provided. Use the new knowledge as the foundation of your reasoning, ensuring that your response directly addresses the query. If there are conflicts between the new knowledge and prior knowledge, prioritize the new knowledge. Additionally, explicitly link each step of your reasoning to specific details from the new knowledge when constructing your answer. The new knowledge states that the name of the founder of Archäologisches Museum der Stadt Kelheim is Kelheim, so ensure your response reflects this clearly and provides informative context that underscores its significance.

• Instruction$_3$: Answer the following question by thoroughly integrating the new knowledge provided. Use the new knowledge as the foundation of your reasoning, ensuring that your response directly addresses the query. If there are conflicts between the new knowledge and prior knowledge, prioritize the new knowledge. Explicitly link your answer to specific details from the new knowledge when constructing your response. The query is: "What is the name of the founder of Archäologisches Museum der Stadt Kelheim?" The new knowledge asserts that the name of the founder is Kelheim, so ensure your answer reflects this clearly and elaborates on the significance of this information, perhaps detailing the role or impact of the founder within the context of the museum.

• Instruction$_4$: Answer the following query based on the new knowledge provided, which serves as the core foundation for your reasoning.

Ensure that you directly address the question while prioritizing the new knowledge over any prior information. If there are conflicts with what you previously learned, disregard that in favor of the new knowledge. Explicitly connect your response to specific details from the new knowledge. The query is: "What is the name of the founder of Archäologisches Museum der Stadt Kelheim?" The new knowledge clearly states that the name of the founder is Kelheim. Elaborate on the significance of this information in the context of the museum's establishment and its impact on the local culture or historical understanding.

A.2.3 Case2

• Knowledge: The name of the spouse of Ron DeSantis is Carol Chu.

• Query: The gender of the spouse of Ron DeSantis is'

• Instruction_0: Answer the following question based on the new knowledge.

• Instruction_1: When answering the following question, utilize the new knowledge provided as your primary reference point. Focus specifically on the fact that Carol Chu is the spouse of Ron DeSantis, which means that Carol Chu's gender should be explicitly defined based on your understanding of the name. If there are any discrepancies between this new knowledge and your existing model knowledge, prioritize the new information in your reasoning. Clearly link your conclusion about the gender of Ron DeSantis's spouse to the detail that Carol Chu is their name.

• Instruction_2: When answering the following question, use the new knowledge provided as your primary reference. The name of the spouse of Ron DeSantis is Carol Chu, and based on this name, determine the gender of the spouse. It is essential to integrate the new knowledge throughout your reasoning process and explicitly state the gender identified based on the name. If there are any conflicts with prior knowledge, the new knowledge must override it. Clearly connect the conclusion about the spouse's gender to the fact that their name is Carol Chu.

A.2.4 Case3

• Knowledge: The names of the siblings of Janice Dickinson are Antoine-Jean-Matthieu Séguier.

• Query: The names of the siblings of Antoine-Jean-Matthieu Séguier are?

• Instruction_0 Answer the following question based on the new knowledge.

• Instruction_1 When answering the following question, prioritize the new knowledge provided regarding Janice Dickinson's sibling, Antoine-Jean-Matthieu Séguier. Clearly state the names of the siblings as per the new knowledge and if needed, provide logical reasoning that aligns with it. If there are any inconsistencies between the new information and any previous understanding of Janice Dickinson having no siblings, the new knowledge should take precedence. Explicitly connect each part of your reasoning back to the relevant new knowledge provided. Knowledge: The names of the siblings of Janice Dickinson are Antoine-Jean-Matthieu Séguier.

You are an instruction optimization expert. You will receive:
1. Initial instruction
2. New knowledge
3. Query
4. Model's current response
The model's current response may not align with the new knowledge. Your task is to refine an initial instruction to align a model's response with new knowledge while allowing for reasonable inference, optimizing it so the model answers the query based primarily on the new knowledge provided. You will receive:

Please follow these steps:
1. Review the initial instruction, new knowledge, query, and current model response.
2. Identify any discrepancies between the new knowledge and the model's response.
3. Modify the initial instruction to:
  a. Emphasize the importance of integrating the new knowledge throughout the reasoning chain when answering the query
  b. Instruct the model to use the new knowledge as the foundation and primary source for its reasoning when answering to the query
  c. Emphasize that when the new knowledge conflicts with the model's prior knowledge, the new knowledge should take precedence
  d. Direct the model to explicitly link each step of its reasoning to specific pieces of the new knowledge
4. Ensure the optimized instruction will guide the model to provide informative answers based on the new knowledge while also engaging in reasonable inference and deduction.
5. Provide a brief explanation of your changes.

Present your response in this format:
Optimized Instruction:
[Your optimized instruction here]
Explanation:
[Your explanation of the changes and how they address the discrepancies,
ensure reliance solely on the new knowledge, and guide the model to prioritize new information over prior knowledge,
especially for multi-step reasoning problems]

Figure 5: The prompt of Knowledge Crafter

You are a multi-hop question judeger. Your task is to analyze the input of the questions and determine if they are multi-hop questions. A multi-hop question requires multiple independent steps of reasoning or accessing multiple distinct pieces of information to arrive at the answer. Each "hop" should involve a separate fact or piece of knowledge. If the question is a multi-hop question, return 'true'. Otherwise, return 'false'.

Here are some examples to illustrate the difference:
Example 1 (Multi-hop):
Question: The gender of the spouse of Joe Biden is?
Analysis: This question requires two independent steps: Identify who Joe Biden's spouse is (Jill Biden). Determine the gender of Jill Biden.
Conclusion: True
Example 2 (Single-hop):
Question: The name of the mother of Leonardo DiCaprio is?
Analysis: This question requires only one step: Identify Leonardo DiCaprio's mother's name.
Conclusion: False
Example 3 (Multi-hop):
Question: What is the capital of the country where the Eiffel Tower is located?
Analysis: This question requires two independent steps:
Identify the country where the Eiffel Tower is located (France). Determine the capital of France (Paris).
Conclusion: True
Example 4 (Single-hop):
Question: Who directed the movie Inception?
Analysis: This question requires only one step: Identify the director of Inception.
Conclusion: False
Example 5 (Single-hop):
Question: The name of the spouse of Ron DeSantis, which is not Carol Chu, is?
Analysis: This question requires only one primary step: Identify Ron DeSantis's spouse's name. The additional information "which is not Carol Chu" is a constraint provided within the question and does not constitute a separate hop.
Conclusion: False

Now, analyze the following question using this format:
Question: [Repeat the given question]
Analysis: [Explain the steps needed to answer the question, clearly indicating whether each step requires accessing a new, independent piece of information]
Conclusion: [Return 'true' if it's a multi-hop question requiring multiple independent steps, 'false' if it's not]

Figure 6: The prompt of Multi-hop Judger

You are an expert in analyzing the potential for reasoning in knowledge graphs. Given a question and a piece of knowledge, your task is to determine whether the provided knowledge offers a potential reasoning path to answer the question.

Guidelines:

1. Assume a complete knowledge graph with all necessary entities and relationships.
2. Identify potential reasoning steps between entities, treating unknown entities as existing in the graph.
3. Answer "True" if there's any potential path, even partial or indirect. "False" if no path exists.
4. Briefly explain your reasoning, focusing on the path structure.

Example 1:
Question: The gender of the spouse of Catherine Van Rensselaer is?
Knowledge: The gender of Philip Schuyler is neutrois.
Conclusion: True
Analysis: Reasoning path: Catherine Van Rensselaer -> spouse -> [unknown entity] -> gender. We assume the spouse of Catherine Van Rensselaer is Philip Schuyler in the complete graph so that there exists a complete reasoning path. The answer is True.
Example 2:
Question: The name of the position held by the mother of Kanye West is?
Knowledge: The name of the mother of Kanye West is Geneviève Abelin.
Conclusion: True.
Analysis: Reasoning path: Kanye West -> mother -> Geneviève Abelin -> position. We assume Geneviève Abelin has a position in the complete graph, even if not specified.
Example 3:
Query: The founder of Zangger Committee follows?
Knowledge: Soviet Union follows 2011 Greece Junior Badminton Championships – men's doubles.
Conclusion: True.
Analysis: Reasoning path: Zangger Committee -> founder -> [unknown entity] -> follows. We assume the founder and what they follow exist in the complete graph, even if not specified.
Example 4:
Query: Is Aisha still alive?
Knowledge: The place of death of Aisha is San Javier.
Conclusion: True.
Analysis: Reasoning path: Aisha -> place of death -> San Javier. Even though this knowledge nearly directly answers the question, it still requires the inference that having a place of death means the person is not alive. So, there exists a complete reasoning path. The answer is True.
Example 5:
Query: The name of the mother of Leonardo DiCaprio is
Knowledge: The name of the country of citizenship of Leonardo DiCaprio is Syria.
Conclusion: False
Analysis: The provided knowledge about Leonardo DiCaprio's citizenship being Syria does not contain any information relevant to his mother's name. The citizenship of a person does not directly imply or provide information about their parent's names. Therefore, based solely on the given knowledge, I cannot determine or infer the name of Leonardo DiCaprio's mother.

Now, analyze the following using this format:

Question: [Insert question here]
Knowledge: [Insert knowledge here]
Conclusion: [Return 'true' if the knowledge helps the reasoning path of the question, else return 'false']
Analysis: [Break down the reasoning steps to answer questions and determine whether knowledge can help reasoning]

Figure 7: The prompt of Reasoning Analyzer

You are an expert who can analyze the following pair of questions and determine if their main subjects could potentially refer to the same entity, concept, or closely related topics. Your task is to extract subjects first and then apply the following guidelines to analyze:

1. Consider any similarity in names, even if partial, as potentially referring to the same entity.
2. Titles of media with any shared words might be related or part of the same franchise.
3. Names with completely different spellings might still be aliases or nicknames for the same person.
4. Questions about the same attribute (e.g., occupation, place of birth) of entities with partially matching names should be considered potentially related.
5. In cases of doubt, lean towards considering the subjects as potentially related.

Example 1:
Question 1: The gender of Jose L Castillo is?
Question 2: The gender of Jose Castillo is?
Subject 1: Jose L Castillo
Subject 2: Jose Castillo
Analysis: Both names are very similar. "Jose L Castillo" likely represents "Jose Castillo" with a middle initial 'L'. In many cases, people use their names with and without middle initials interchangeably.
Conclusion: True

Example 2:
Question 1: The name of the country which 81st Missouri General Assembly is associated with is?
Question 2: The name of the country which 81st Missouri state legislature is associated with is?
Subject 1: 81st Missouri General Assembly
Subject 2: 81st Missouri state legislature
Analysis: Both subjects refer to the same governmental body. "General Assembly" is the official name for the state legislature in Missouri. "State legislature" is a general term used to describe the same institution. The "81st" in both cases indicates it's the same specific session of this body.
Conclusion: True

Example 3:
Question 1: The occupation of Yolanda Saldívar is
Question 2: The occupation of Yolacaca is
Subject 1: Yolanda Saldívar
Subject 2: Yolacaca
Analysis: While there are significant differences, "Yolacaca" could be a severe misspelling or nickname for "Yolanda Saldívar". The first names start similarly. This loosely relates to criteria 1 and 4.
Conclusion: True

Now, analyze the following pair of questions and give your conclusion:

Figure 8: The prompt of Subject Analyzer

Analyze the following two queries and determine if the relationships described in them are the same. Follow these steps:

1. Extract the core relationship from each query.
2. Briefly compare the extracted relationships.
3. Conclude whether the relationships are the same.

Query1: {Query1}
Query2: {Query2}
Your response should be structured as follows:
Relationship in Query1: [extracted relationship]
Relationship in Query2: [extracted relationship]
Analysis: [Brief comparison of the two relationships]
Conclusion: [True/False] (True if the relationships are the same, False if they are different)

Example 1 (Different Relationships):
Query1: The name of the country of citizenship of Leonardo DiCaprio is
Query2: The name of the mother of Leonardo DiCaprio is
Relationship in Query1: citizenship country of
Relationship in Query2: mother of
Analysis: The first query asks about a person's citizenship, which is a relationship between a person and a country. The second query asks about a person's mother, which is a familial relationship between two people.
Conclusion: False

Example 2 (Same Relationship):
Query1: The birthplace of Albert Einstein is
Query2: The birthplace of Isaac Newton is
Relationship in Query1: birthplace
Relationship in Query2: birthplace
Analysis: Both queries ask about the birthplace, which is the location where a person was born. The relationship is the same in both queries, even though they refer to different individuals.
Conclusion: True

Now, analyze the following queries:

Figure 9: The prompt of Relation Analyzer

You are an intelligent assistant specialized in determining the logical relevance of knowledge to specific questions.
Your task is to analyze whether a given piece of knowledge is helpful in answering or reasoning about a particular question. Please follow these steps:

1. Analyze the logical connection between the knowledge and the question.
2. Determine if the knowledge directly or indirectly helps in answering or reasoning about the question.
3. Provide your conclusion: Relevant or Not Relevant. If relevant, return True; if not relevant, return False.
4. Briefly explain the reasoning behind your conclusion.

Please note:
Even if the knowledge provides only partial information or is indirectly related, it should be considered relevant if it aids in reasoning.
Do not attempt to answer the question itself; focus only on judging the relevance of the knowledge.
Remain objective and do not introduce outside information.
Please respond in the following format:

Example1:
Question: Is J.R.R. Tolkien still alive?
Knowledge: The place of burial of J.R.R. Tolkien is Church of St Mary, Radnage.
Analysis: Information about a person's burial place directly implies that they are no longer alive, thus helping to answer the question about Tolkien's current living status.
Conclusion: True

Example2:
Question: The name of the continent which Russia is part of is?
Knowledge: The name of the country which Russia is associated with is Russian State.
Analysis: While this knowledge doesn't directly answer the question about the continent, it provides related geographical information about Russia, which can aid in reasoning about its continental location.
Conclusion: False

Now, analyze the following question and knowledge and give your conclusion

Figure 10: The prompt of Logic Analyzer