

Learning to Edit Knowledge via Instruction-based Chain-of-Thought Prompting

Jinhu Fu^{1*}, Yan Bai², Longzhu He¹, Yihang Lou³,
Yanxiao Zhao¹, Li Sun¹, Sen Su^{1,4†},

¹Beijing University of Posts and Telecommunications, ²Peking University,
³Huawei Technologies Ltd., ⁴Chongqing University of Posts and Telecommunications

Abstract

Large language models (LLMs) can effectively handle outdated information through knowledge editing. However, current approaches face two key limitations: **(I) Poor generalization**: Most approaches rigidly inject new knowledge without ensuring that the model can use it effectively to solve practical problems. **(II) Narrow scope**: Current methods focus primarily on structured fact triples, overlooking the diverse unstructured forms of factual information (e.g., news, articles) prevalent in real-world contexts. To address these challenges, we propose a new paradigm: teaching LLMs to edit knowledge via *Chain of Thoughts (CoTs)* reasoning (CoT2Edit). We first leverage language model agents for both structured and unstructured edited data to generate CoTs, building high-quality instruction data. The model is then trained to reason over edited knowledge through supervised fine-tuning (SFT) and Group Relative Policy Optimization (GRPO). At inference time, we integrate Retrieval-Augmented Generation (RAG) to dynamically retrieve relevant edited facts for real-time knowledge editing. Experimental results demonstrate that our method achieves strong generalization across six diverse knowledge editing scenarios with just a **single round of training** on three open-source language models. The codes are available at <https://github.com/FredJDean/CoT2Edit>.

1 Introduction

The advent of large language models (LLMs) has brought revolutionary advancements to the field of natural language processing (Radford et al., 2019; Roberts et al., 2020). However, LLMs may harbor incorrect or outdated knowledge (Mitchell et al., 2022a; De Cao et al., 2021), and knowledge editing methods have emerged as a solution to this

issue (Zhang et al., 2024b; Wang et al., 2025). Mainstream approaches to LLM knowledge editing include context-based methods and parameter-modification-based methods (Zhang et al., 2024b). Context-based methods, represented by works such as (Qi et al., 2024; Zheng et al., 2023; Zeng et al., 2024), offer broad applicability but require the design of complex prompt templates and exhibit sub-optimal performance (Li et al., 2024; Yu et al., 2024; Wang et al., 2024). In contrast, parameter-modification-based methods have garnered widespread attention due to their more stable performance (Li et al., 2024; Gu et al., 2024; Fu et al., 2026). Among these, locate-then-edit methods have achieved remarkable progress in recent years (Fang et al., 2024; Jiang et al., 2025), with representative works including MEMIT (Meng et al., 2022b), AlphaEdit (Fang et al., 2024), KGMET (Fu et al., 2025) and others (Li et al., 2024; Gu et al., 2024).

However, locate-then-edit approaches lack both **flexible utilization** and **semantic understanding** capabilities. ① As shown in Figure 1 (a), these require direct access to and modification of model parameters, which is incompatible with frozen, production-grade LLMs (Jiang et al., 2024). ② As illustrated in Figure 1 (b), when editing knowledge at the parameter level, the model may respond correctly to exact queries (e.g., "the president of the U.S.") but fail on semantically equivalent ones (e.g., "the U.S. leader") (Wang et al., 2025). In addition, new knowledge often appears in unstructured forms, such as news or articles, rather than rigid triples, causing it fail to handle such knowledge (Wu et al., 2024).

To address the limitations, some works such as LTE (Jiang et al., 2024) and EditCoT (Wang et al., 2025) adopt a training and retrieval paradigm. Though effectiveness, these approaches still suffer from several critical limitations:

➡ **Poor Availability**. LTE (Jiang et al., 2024)

*fjhu@bupt.edu.cn

†Corresponding author.

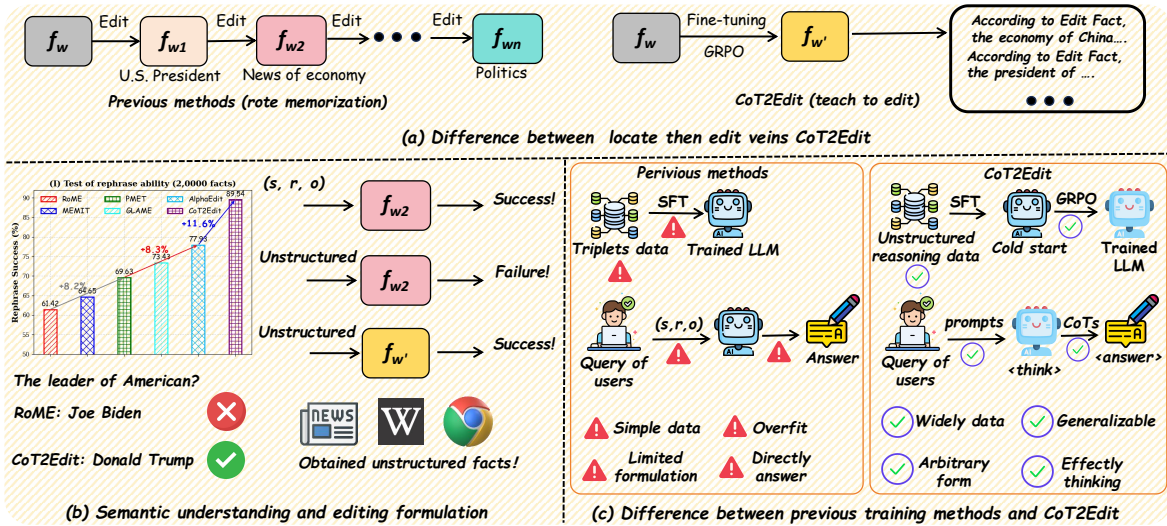


Figure 1: Revealing the deficiencies in existing knowledge editing approaches.

is trained on simple editing datasets that do not explicitly model reasoning paths. As a result, the model is required to generate correct answers in a single step, which makes it **prone to hallucinations**. EditCoT (Wang et al., 2025) partially addresses this issue by introducing CoTs, but it relies on a complex multi-model pipeline with **separate LLMs for reasoning and editing**, increasing computational cost while limiting scalability in practical deployment. In addition, they focus mainly on editing simple factual triples, making them difficult to extend to **more realistic knowledge forms**, such as unstructured or compositional knowledge.

➡ **Limited generalization.** Both LTE and EditCoT depend heavily on large-scale SFT (Howard and Ruder, 2018) over curated editing datasets to learn specific editing patterns. While effective in-distribution, this strategy leads to overfitting to the training data and results in noticeable performance degradation when scaling to diverse or large-scale editing settings. Moreover, when confronted with out-of-distribution (OOD) editing data, their performance remains limited, highlighting the **inherent generalization bottleneck** of purely SFT-based editing approaches (Kang et al., 2025).

To address these challenges, we propose CoT2Edit. As shown in Figure 1 (c), we redesign knowledge editing paradigm from the perspectives of data, training and inference. ❶ For data construction, to solve the narrow scope problem, we cover a broader range of editing scenarios by jointly modeling structured and unstructured edits, using MQUAKE (Zhong et al., 2023) and MQUAKE-uns (Wu et al., 2024) to generate CoTs

supervision and synthesizing additional instruction data from HotpotQA (Yang et al., 2018) entity relations with LLM agents (e.g. ChatGPT or DeepSeek). ❷ For training, we adopt a two-stage SFT and GRPO (Shao et al., 2024) framework, where SFT only provides a cold start, while data evolved GRPO serves as the core optimization, effectively improving generalization to unseen edits while preserving the general capabilities of LLMs. ❸ For inference, CoT2Edit performs CoTs reasoning over retrieved edited facts, which mitigates hallucinations from single-step generation and enhances the ability of edited LLMs to solve complex editing queries.

Experimental results demonstrate that our proposed CoT2Edit excels in three key aspects: ❶ **Single-Training Generalization:** Unlike previous approaches requiring repeated editing processes, CoT2Edit achieves state-of-the-art performance on six previously unseen editing benchmarks after just a single training process, demonstrating remarkable out-of-distribution (OOD) generalization capability. ❷ **Cross-Format Robustness.** The proposed paradigm shows consistent excellence in unstructured knowledge editing scenarios. Specifically, our approach achieves 92% accuracy for unstructured knowledge editing ($\uparrow \sim 20\%$ compared with IKE (Zheng et al., 2023)). ❸ **Semantic Robustness.** CoT2Edit achieves 89% in the rephrase metric ($\uparrow \sim 8\%$) and 93% in the neighborhood success metric ($\uparrow \sim 23\%$) when editing **large scale** facts (20k-30k vs. 2k-3k in AlphaEdit (Fang et al., 2024)).

Our contributions can be summarized as follows:

- ☞ We construct a novel editing training dataset that incorporates multi-hop editing tasks to generate CoTs instruction data. Our dataset integrates both structured and unstructured knowledge sources, covering diverse editing scenarios beyond triplet modifications.
- ☞ We are the first to apply GRPO reinforcement learning to knowledge editing, where a lightweight SFT cold start is followed by data-evolving GRPO optimization, enabling LLMs to effectively generalize to complex reasoning tasks.
- ☞ Our method achieves superior performance across six standard knowledge editing benchmarks using only a single training round, highlighting its strong out-of-distribution capability and cross-format robustness.

2 Related works

In this section, we introduce related work on model editing. **(I) In-context editing methods.** Representative works in this category include ICE (Qi et al., 2024), IKE (Zheng et al., 2023), and so on. This approach is typically designed for black-box scenarios and does not allow for parameter adjustments (Zhang et al., 2024b). LTE (Jiang et al., 2024) fine-tunes LLMs using corpora built from editing targets and related knowledge, but it struggles with multi-hop reasoning and complex edits. EditCoT (Wang et al., 2025) enhances the confidence of edited knowledge by transplanting chains of thought related to the editing target into the LLM’s context. Nevertheless, EditCoT requires first fine-tuning a model to generate chains of thought for the new knowledge, and then transferring them into the context of another editing model, which is both time-consuming and resource-intensive. **(II) Side memory based methods.** The works such as MELo (Yu et al., 2024), WISE (Wang et al., 2024) and GRACE (Hartvigsen et al., 2024). They edit external memory to achieve knowledge update. Although effective, as the number of edits increases, they become memory-overhead, making it cumbersome. **(III) Locate then edit methods.** RoME (Meng et al., 2022a) first locates relevant neurons via causal mediation trace (PEARL, 2001; Vig et al., 2020) and modifies the corresponding MLP module. Building upon this, MEMIT (Meng et al., 2022b) introduced batch editing, which allows efficient updates to large-scale knowledge.

In recent years, approaches such as AlphaEdit (Fang et al., 2024), PMET (Li et al., 2024), and GLAME (Zhang et al., 2024a) have been proposed, significantly advancing the upper bound of performance for the locate-then-edit paradigm. Although effective, they inherently suffer from the limitations of “rote memorization” when modifying K-V pairs.

3 Preliminary

Knowledge editing (Mitchell et al., 2022a) aims to update the base model f_θ so that it reflects a desired change specified by an edit descriptor (x_i, y_i) , while preserving performance on unrelated inputs. Formally, given that the original model predicts $f_\theta(x_i) \neq y_i$, the goal is to obtain an updated model $f_{\theta'}$ such that $f_{\theta'}(x_i) = y_i$.

Formally, our objective is to learn a mapping function $f_{\theta'} : \mathcal{E} \times \mathcal{Q} \rightarrow \mathcal{A}$, where \mathcal{E} denotes editable facts, \mathcal{Q} represents possible queries, and \mathcal{A} denotes the answer space. such that for any editing fact $e \in \mathcal{E}$ and question $q \in \mathcal{Q}$, the function gives $a = f_{\theta'}(e, q)$ as the updated response.

However, unlike previous editing paradigms that directly enforce the edited fact during prediction, CoT2Edit introduces an intermediate reasoning process. Specifically, we reformulate the mapping as a two-stage function:

$$f_{\theta'}(e, q) = g_{\theta'}(h_{\theta'}(e, q)) \quad (1)$$

where $h_{\theta'}(e, q)$ generates an interpretable reasoning chain conditioned on the edit fact e and query q , and $g_{\theta'}(\cdot)$ produces the final answer based on this reasoning.

4 Methods

In this section, we propose our CoT2Edit, the framework of CoT2Edit is shown in Figure 2.

4.1 Training dataset construction

4.1.1 CoT Generation

To generate reasoning chains related to editing, ① for structured data, we take the first 2,000 entries from MQuAKE (Zhong et al., 2023) as edit instruction data; ② for unstructured data, we use all of the data in MQuAKE-uns (~ 1000 entries) to build unstructured reasoning chains, which are constructed based on the edit context and corresponding questions; ③ finally, the accuracy of the generated data is checked for correctness through a result-oriented approach.

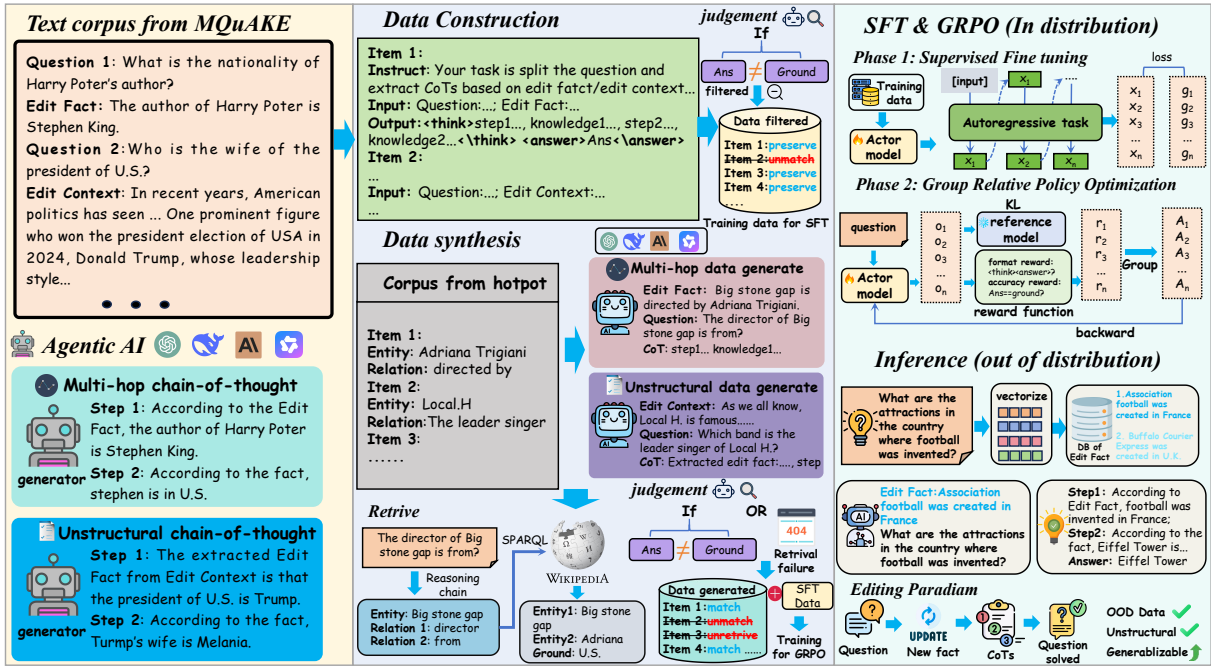


Figure 2: The framework of our proposed CoT2Edit. (I) Construct editing instructions by prompting LLM agents to generate reasoning chains from editing corpora. (II) Generate more instruct data by LLM agents using entity relations from HotpotQA. (III) Train the model via SFT on Phase (I) data to learn targeted response patterns. (IV) Optimize the model using GRPO to enhance generalization on the combined (I)(II) data. (V) Deploy RAG to retrieve relevant edited facts during inference, ensuring accurate question answering.

(I) Multi-hop Question Reasoning Chain: First, we construct the reasoning chain required for inference based on the edit fact \mathcal{E} and question \mathcal{Q} in MQuAKE. By designing a prompt template \mathcal{T} , we prompt a large model agent (such as ChatGPT or DeepSeek) to generate the reasoning chain. This process can be expressed as the following formula:

$$\text{Agent}(\mathcal{Q}, \mathcal{E}, \mathcal{T}) \rightarrow \text{CoT}, \mathcal{A}. \quad (2)$$

By an appropriate prompt template, the solution steps can be decomposed into smaller sub-steps based on the **edit facts** and **multi-hop questions**.

Prompt template for multihop reasoning

Your task is to *break down* the question into **steps** and extract the *chain of thought* based on the **editing facts** into `<think></think>` tags, and finally get the corresponding answer and put it in `<answer></answer>`. You must **strictly follow** the factual information corresponding to the **Edit Facts**, a few examples of which are provided below: {examples}.

By employing a **few-shot prompting** approach, we guide the agent to generate reasoning chains in a specific format, thereby achieving the extraction of edit facts and their associated multi-hop question-based reasoning chains.

(II) Unstructured based reasoning chain: To develop a more universal knowledge editor, we further construct unstructured reasoning chains based on unstructured knowledge, enabling large language models to effectively extract relevant edit facts in response to given questions. Specifically, given an edit context \mathcal{C} and a corresponding question \mathcal{Q} , our objective is to extract the relevant edit fact \mathcal{E} from \mathcal{C} , then use \mathcal{E} to derive the answer to \mathcal{Q} .

$$\text{Agent}(\mathcal{Q}, \mathcal{C}, \mathcal{T}) \rightarrow \mathcal{E}, \text{CoT}, \mathcal{A}. \quad (3)$$

Prompt template for unstructured reasoning

Your task is to firstly **extract** the *edit fact* from the **Edit Context** and secondly answer the corresponding question based on the *extracted edit fact*, in this process you need to decompose the question and extract the *chain of thought* based on the **edit fact**, put the extraction of the edit fact and the steps to decompose the question in `<think></think>` tags and put the answer in the `<answer></answer>` tags, a few examples of which are provided below: {examples}.

Similarly, we design appropriate prompts to allow the LLM agent to effectively extract relevant edit facts \mathcal{E} and generate corresponding Chain-of-Thoughts reasoning based on the given question \mathcal{Q} . The prompt template \mathcal{T} .

Through the **few-shot prompts**, we can effectively extract the most relevant factual knowledge for editing and the CoTs required to answer the question.

(III) Data validation: However, the information extracted by the LLM agent may not always be accurate. To verify the correctness of the reasoning chain, we compare the inferred answer $\{a\}$ with the ground-truth answer $\{g\}$ from the dataset. The consistency between these answers confirms the validity of the reasoning chain for inference tasks.

(IV) Implementation: In this framework, \mathcal{Q} and \mathcal{E} (\mathcal{C}) serve as model **inputs**, while **CoT** and \mathcal{A} constitute the **outputs**, forming the structured instruction data for SFT. The motivation for our use of multi-hop reasoning and unstructured data is **theoretically supported** in Appendix A.3.

4.1.2 Data Augmentation

Since the reinforcement learning phase typically requires large-scale data for effective optimization, we employ a data augmentation approach to prompt the agent to synthesize additional instruction data.

① **Entity-Relation Extraction:** HotpotQA (Yang et al., 2018) contains a substantial collection of fact-based entities and relations. We first extract these entity-relations from HotpotQA using heuristic methods, which serve as the foundation for constructing our augmented data. This process can be formalized as follows.

$$\text{ent, rel} \leftarrow \text{Extract}(\text{corpus}) \quad (4)$$

② **Instruct data generation:** Building upon the extracted entity-relations, we employ carefully designed prompt templates to guide the LLM agent in constructing instruction data. In this process, the agent generates corresponding questions, edit facts, and solution steps based on the identified entities and relations.

$$\text{Agent}(\mathcal{T}, \text{ent, rel}) \rightarrow \{\mathcal{Q}, \mathcal{E}, \text{COT}, \mathcal{A}\} \quad (5)$$

The prompt \mathcal{T} is described as below.

This approach utilizes key entities and relations to constrain the generated question scope, thereby preventing ambiguous or uncertain outputs while significantly improving generation accuracy.

For creating unstructured content editing data, we place the prompt template in Appendix A.1.

③ **Data validation:** Despite using carefully designed prompts, LLM agents can still generate erroneous data. To address this, we implement a

three-stage filtering mechanism: (1) Entity-relation extraction from generated questions, (2) SPARQL based answer retrieval from Wikipedia using these extracted entities and relations as chains, (3) Automated filtering of outputs where: the generated answer mismatches the retrieved Wikipedia answer, or no valid Wikipedia reference exists. After filtering out, the scale of generated data is about 10k.

Prompt template for data generation

You are a helpful assistant generating diverse and high-quality instruction-following data. 1. The generated Question must require multiple steps of reasoning to solve based on the edit fact. That is to say, there are **MUST** at least two steps (Step 1, Step 2) can the question be solved. 2. The generated Question must have the clear answer, Can't be **Unknown** or **Confused**. 3. Can't have any extraneous strings other than json. 4. Be sure to generate content include entity {entities[i]} and {relation[i]}. 5. The keys in json format must **ONLY** be Input, Output. 6. Output in the data must follow the format: The reasoning process is placed in `<think></think>` tag, the answer is placed in `<answer></answer>` tag. No other characters are allowed. A few examples: {examples}.

4.2 Train the model

We utilize the data generated previously for SFT and GRPO to teach the model learning the editing paradigm.

Phase 1, Supervised Fine-Tuning. In this stage, we employ the previously constructed CoT instruction data to train the LLMs in step-by-step problem-solving using provided edit facts or edit context. This optimization process can be formally characterized as an auto-regressive task.

$$\mathcal{L}_{SFT}(\theta) = - \sum_{i=1}^N \sum_{t=1}^T \log \mathbb{P}_{\theta}(y_i^t | x_i, y_i^{<t}), \quad (6)$$

where θ denotes model parameters, $D = (x_i, y_i)_i^N$ is the instruct dataset, x_i denotes the input sequence, $y_i \in (y_i^1, y_i^2, \dots, y_i^T)$ denotes the target CoT sequence, and \mathbb{P}_{θ} denotes the conditional probability of outputting a target token given a known sequence.

Through the SFT phase, the model effectively learns a multi-step reasoning paradigm based on edit facts, thereby enabling the large language model to solve editing-related problems using CoTs reasoning.

Phase 2, Group Relative Policy Optimization. In this phase, we employ GRPO to further guide behavior of LLMs by jointly leveraging both the SFT

data and the synthetic data. We construct a composite reward function to guide the optimization of GRPO.

$$\mathcal{R} = \mathcal{R}_{acc} + \mathcal{R}_{format}, \quad (7)$$

where \mathcal{R}_{acc} is accuracy reward, \mathcal{R}_{format} is the format reward, which is calculated by combining the inclusion of `<think><answer>` tags and keywords such as “step” or “knowledge” in the reply messages. It can effectively guide the model to generate correct reasoning paths and answers, preventing hallucinations.

To further enhance training efficiency, we implement a self evolution strategy. After each training round, we collect high-reward samples generated during rollout and incorporate them into the next round’s training dataset:

$$\mathcal{D}_{t+1} = \mathcal{D}_t \cup \{s \mid s \in \text{Rollout}_t, \mathcal{R}(s) > \theta\}, \quad (8)$$

where \mathcal{D}_t is the training dataset at round t , and θ is a reward threshold for selecting high-quality samples. This iterative process enables the model to rapidly learn the editing paradigm and achieve better convergence.

4.3 Inference

To achieve more practical on-the-fly editing, we incorporate RAG to retrieve the relevant edit fact with the question. We utilize an off-the-shelf retrieval model `contriver-msmarco` invented by Facebook to embed all the edit facts and create a vector memory to store the representations. When given a query, we also get the representation of the query by the retriever and search the most similar edit fact from the vector memory. Then, the query and retrieved edit fact are fed into the LLM to obtain the reasoning chains and answers.

5 Experiments

In this section, we perform a wide range of experiments to answer the following questions.

RQ1: How does CoT2Edit perform on sequential editing tasks compared to baselines in five structural and unstructured editing benchmarks?

RQ2: How does the out-of-distribution capability of CoT2Edit in unseen editing task?

RQ3: How does the modular design of CoT2Edit enhance the model’s capabilities?

RQ4: How does the performance of CoT2Edit in practical editing scenarios? How can we evaluate its actual effectiveness?

5.1 Experimental setup

LLMs and Baseline methods. Our experiments are carried out on three base LLMs: Llama3-8B (Grattafiori et al., 2024), Falcon3-10B (Almazrouei et al., 2023), and Qwen3-14B (Yang et al., 2025). We compare our method against Fine-tuning (Hu et al., 2021), RoME (Meng et al., 2022a), MEMIT (Meng et al., 2022b), IKE (Zheng et al., 2023), SKEME (Zeng et al., 2024) PMET (Zhang et al., 2024a), and GLAME (Zhang et al., 2024a) and AlphaEdit (Fang et al., 2024). More details are given in Appendix B.3.

Datasets. We evaluate CoT2Edit using three structural benchmarks and two unstructured benchmarks. CounterFact (Meng et al., 2022b), ZsRE (Levy et al., 2017) and MQuAKE (Zhong et al., 2023) are structural datasets; CounterFact-uns and WikiUpdate (Wu et al., 2024) are unstructured datasets. Edit success as Edit succ., Paraphrase as Para., Locality as Loc., and Neighborhood success as Nei are adopted as metrics (Meng et al., 2022b). More details are provided in Appendix B.1, B.2, and B.4.

5.2 Editing performance (RQ1)

We conduct editing experiments on three based LLMs using five datasets. Based on Table 1, we can obtain the following observation.

Obs 1: CoT2Edit achieves superior performance in nearly all metrics and base models. From Table 1, we can observe that CoT2Edit provides the best editing success performance in all three base models. This demonstrates that training and reasoning through CoTs not only enhances the model’s confidence in the edited facts but also ensures that the entire process remains fully interpretable.

Obs 2: CoT2Edit has a stronger paraphrase and multi-hop inference ability against advance baselines. In the paraphrase metric, it’s about $\uparrow \sim 10\%$ compared to EditCoT. In the neighbor success metric, it’s most $\sim \uparrow 15\%$ compared to the strongest EditCoT and GLAME in MQuAKE. Unlike prior methods that often rely on rote memorization, CoT2Edit effectively transfers edited knowledge to semantically diverse and contextually related queries, demonstrating stronger robustness under rephrased and multi-hop conditions.

Obs 3: CoT2Edit possesses strong capabilities to edit unstructured knowledge. Parameter-editing methods are inherently limited to modify-

Table 1: Editing performance of CoT2Edit against baselines on five benchmarks (editing full datasets).

Model	Dataset	Metric	RoME	MEMIT	IKE	SKEME	EditCoT	LTE	PMET	GLAME	AlphaEdit	CoT2Edit
LLaMA3-8B	ZsRE	Edit succ.	2.01	34.62	80.88	82.4	86.13	90.1	92.63	89.48	88.78	93.17
		Para.	1.8	31.28	68.13	70.27	83.55	79.19	81.13	78.4	79.35	92.14
		Loc.	0.69	18.49	100	100	100	61.34	27.81	25.43	30.7	73.13
	MQuAKE	Edit succ.	80.28	81.19	86.63	88.05	90.58	93.46	96.37	93.28	97.11	99.95
		Nei.	60.38	64.03	65.38	67.12	78.71	74.45	70.28	77.82	73.63	92.04
		Para.	0	0	74.97	77.35	84.64	87.47	0	0	0	93.64
	Counterfact-uns	Para.	0	0	65.71	70.13	80.8	75.05	0	0	0	79.79
		Nei.	0	0	58.07	60.68	72.15	67.18	0	0	0	93.05
		Edit succ.	64.38	65.34	90.36	85.83	90.16	91.42	91.37	89.68	92.35	93.57
	Counterfact	Para.	61.42	64.65	74.33	75.11	87.22	81.53	80.63	77.43	81.78	89.54
		Nei.	49.44	51.56	52.17	58.36	80.18	71.39	65.74	70.54	67.88	93.58
		Edit succ.	0	0	63.73	70.98	74.44	80.21	0	0	0	85.89
WikiUpdate	Edit succ.	67.13	90.41	80.49	83.68	94.83	96.78	87.15	91.18	94.13	98.11	
Falcon3-10B	ZsRE	Para.	60.65	75.73	65.18	66.92	90.9	87.51	82.3	81.51	85.24	96.2
		Loc.	23.47	28.31	100	100	100	70.67	30.36	25.49	34.78	83.24
		Edit succ.	80.13	87.68	87.44	90.98	96.73	94.51	93.16	96.68	97.73	99.48
	MQuAKE	Nei.	65.21	70.34	64.29	67.43	81.11	79.18	77.48	82.31	71.69	92.69
		Edit succ.	0	0	75.3	78.51	80.24	88.26	0	0	0	94.56
		Para.	0	0	61.24	65.1	75.15	77.71	0	0	0	81.43
	Counterfact-uns	Nei.	0	0	61.78	58	71.11	65.14	0	0	0	90.61
		Edit succ.	67.71	87.4	82.65	84.33	90.15	93.09	90.9	87.83	92.51	99.13
		Para.	63.95	76.59	72.33	76.15	84.48	76.17	80.23	77.78	81.63	89.54
	Counterfact	Nei.	52.1	61.36	60.68	61.04	78.55	69.66	64.35	71.82	65.77	94.55
		Edit succ.	0	0	65.36	71.84	77.18	74.33	0	0	0	82.29
		WikiUpdate	Edit succ.	59.95	90.71	86.63	85.37	91.41	90.9	94.82	92.13	95.54
Qwen3-14B	ZsRE	Para.	54.73	78.15	67.78	63.43	85.51	79.97	87.91	84.95	90.45	96.64
		Loc.	20.23	23.33	100	100	100	73.63	26.61	24.04	36.37	85.59
		Edit succ.	80.74	91.65	84.93	87.28	96.37	97.73	94.13	95.56	97.71	99.22
	MQuAKE	Nei.	70.15	73.48	64.29	66.13	83.82	77.65	79.78	84.31	76.67	92.24
		Edit succ.	0	0	80.38	82.85	87.76	91.48	0	0	0	94.46
		Para.	0	0	52.63	56.18	80.53	74.75	0	0	0	81.02
	Counterfact-uns	Nei.	0	0	53.37	55.7	76.06	63.15	0	0	0	84.51
		Edit succ.	63.07	92.13	84.43	82.58	96.73	95.31	96.63	92.44	95.57	99.04
		Para.	55.33	80.35	67.14	69.37	90.9	87.17	86.37	81.47	87.25	98.94
	Counterfact	Nei.	51.23	66.47	60.35	63.08	81.16	78.56	74.31	78.73	68.54	92.07
		Edit succ.	0	0	72.03	69.25	77.68	75.03	0	0	0	82.76
		WikiUpdate	Edit succ.	0	0	72.03	69.25	77.68	75.03	0	0	0

ing knowledge expressed in the form of structured triples. As a result, they failed to edit free-form textual knowledge. CoT2Edit achieves the best performance in unstructured knowledge editing in all three models, maintaining an accuracy greater than 80% ($\uparrow 5 - 15\%$ in three base models) even in cross-lingual knowledge editing scenarios such as WikiUpdate.

5.3 Out-of-distribution Generalization (RQ2)

To evaluate CoT2Edit performance in OOD scenarios, we conduct rigorous experiments on ConvSent (Mitchell et al., 2022b), a sentiment editing task featuring diverse data distributions, alongside established benchmarks.

Obs 4: CoT2Edit has the strongest generalization of OOD. As shown in Table 2, our CoT2Edit outperforms existing methods at least $\uparrow 11\%$ on unseen sentiment editing tasks. In contrast, locate-then-edit approaches exhibit poor generalization to OOD cases (merely at most 76%-78%), with performance worsening as the number of edits increases (decrease 10-20%). Context learning methods, struggle to capture sentiment changes using only prompt templates, further highlighting the advantage of CoT2Edit. The interpretability of CoT2Edit superior in this task is also explained

Table 2: OOD generalization of CoT2Edit on ConvSent bench using Llama3-8B and Falcon-3-10B.

Models	Methods	1K Edits	3K Edits	5K Edits	8K Edits	10K Edits
Llama3	MEMIT	58.72	54.54	50.66	46.9	43.06
	IKE	41.19	38.32	36.2	32.37	29.33
	AlphaEdit	76.13	70.53	63.28	55.38	53.11
	PMET	67.13	65.11	63.27	61.15	60.58
	LTE	80.49	78.83	76.11	73.37	72.78
	EditCoT	78.34	76.79	76.43	75.11	74.91
	CoT2Edit	87.25	86.18	85.33	85.27	85.2
Falcon3	MEMIT	61.48	57.72	53.17	50.58	47.36
	IKE	40.75	36.53	33.45	31.08	27.99
	AlphaEdit	78.07	77.75	74.63	72.18	70.7
	PMET	71.18	70.03	69.67	68.13	66.55
	LTE	77.37	76.15	75.76	72.62	71.03
	EditCoT	79.15	78.81	76.12	75.27	74.51
	CoT2Edit	84.32	83.01	82.82	80.68	79.08

by case studies in Appendix B.5.

5.4 Ablation study (RQ3)

In this section, an ablation study is conducted, as shown in Table 3. More datasets are shown in Appendix B.5.

Obs 5: SFT and GRPO are two indispensable steps. Without SFT, the editing performance drops by around 7% in general. This shows that SFT helps the model quickly learn the basic reasoning format and task structure, even with limited data. Without GRPO, the model also shows clear performance degradation, particularly in Para. and Nei. metrics ($\downarrow 5 - 8\%$), highlighting that GRPO im-

Table 3: Ablation studies of CoT2Edit on CounterFact.

Model	Variant	Edit Succ.	Para.	Nei.
Llama-3-8B	CoT2Edit-W/O SFT	86.8	83.28	80.1
	CoT2Edit-W/O GRPO	90.95	84.03	87.38
	CoT2Edit-W/O Train	55.11	50.5	47.18
	CoT2Edit	93.57	89.54	92.76
Falcon3-10B	CoT2Edit-W/O SFT	92.71	85.63	84.37
	CoT2Edit-W/O GRPO	96.93	86.34	87.78
	CoT2Edit-W/O Train	57.31	53.68	50.93
	CoT2Edit	99.13	89.54	94.55
Qwen3-14B	CoT2Edit-W/O SFT	87.13	84.42	85.5
	CoT2Edit-W/O GRPO	92.06	90.98	87.11
	CoT2Edit-W/O Train	61.73	56.56	57.81
	CoT2Edit	99.04	98.94	92.07

proves factual accuracy and generalization through reward optimization.

5.5 Practicality analysis (RQ4)

In this section, we demonstrate the practicality of CoT2Edit from four perspectives.

① **Time consumption:** We analyze the time consumption of our methods during the inference time. The results are shown in Table 4.

Table 4: Averaged Wall Clock Time per edit method for 10 edits on ZsRE using LLaMA3-8B.

Method	Edit Time	Inference Time	Total Time
FT	62.73	1.41	64.14
IKE	0.00	1.71	1.71
RoME	210.77	1.55	213.32
MEMIT	183.24	1.57	184.81
LTE	0.00	1.60	1.60
AlphaEdit	199.13	1.6	200.03
EditCoT	0.00	4.07	4.07
CoT2Edit	0.00	1.93	1.93

Obs 6: CoT2Edit stands out by achieving the swiftest editing speeds coupled with superior performance. Methods like RoME and MEMIT, which follow a locate-then-edit approach, require opening the model parameters for editing and then performing inference whenever new knowledge is introduced. This makes the editing process quite complex. After just a single train round (which takes at most 4 hours in our experiments compare 9 hours of LTE), CoT2Edit enables instantaneous editing similar to IKE and LTE by appending a knowledge editing prompt to the input prefix. Despite a marginally increased inference time, the overall time expenditure is significantly reduced, underscoring the efficiency and effectiveness of CoT2Edit. Due to the complex pipeline of EditCoT, its inference time is more than twice longer than that of CoT2Edit.

② **Edit number:** We analyze the influence of editing number on performance through Figure 3.

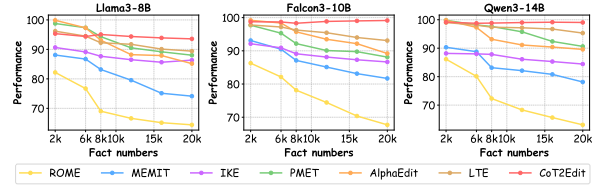


Figure 3: The impact of factual number on editing efficacy in CounterFact.

Obs 7: The editing effect of CoT2Edit does not collapse as the number of facts increases. Due to cumulative errors, as the number of edited facts increases, the performance of the locate-then-edit methods gradually deteriorates. However, since CoT2Edit employs a train-inference framework, its effectiveness does not gradually diminish as the number of edited facts increases. Since LTE lacks the support of an inference path, it achieves suboptimal results.

③ **General ability:** Figure 4 illustrates the general performance metrics of the model when editing Llama3-8b.

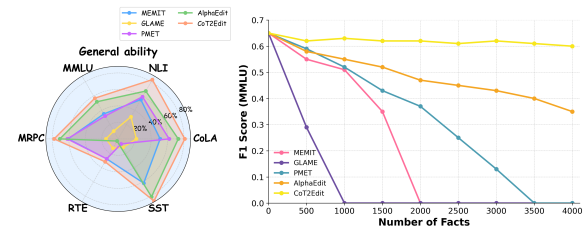


Figure 4: The evaluation of general ability of LLM in Llama3-8B.

Obs 8: CoT2Edit retains the highest level of original performance from LLMs. CoT2Edit achieved the best performance in the six benchmarks evaluating general performance such as SST (Socher et al., 2013), CoLA (Warstadt, 2019) etc. As shown on the right side of Figure 4, even though AlphaEdit uses a null space method, its normal performance deteriorates on MMLU (Hendrycks et al., 2020) due to accumulated errors as the number of facts increases. In contrast, CoT2Edit maintains stable performance as the editing scale grows.

6 Conclusions

In this paper, we propose a novel editing paradigm CoT2Edit that enables LLMs to perform knowledge editing in real-world QA scenarios. We demonstrate the generality of CoT2Edit through experiments on six benchmarks and extensively evaluate efficiency, OOD performance, and its general capabilities. In future work, we will construct

more diverse instruction datasets to enhance the adaptability of CoT2Edit to more scenarios.

7 Limitations

Our instruction dataset currently consists only of multi-hop and unstructured questions, which may not be sufficient to cover a broader range of editing scenarios. Future work should focus on constructing more diverse and comprehensive editing instruction sets to enhance the adaptability of the method.

8 Acknowledgements

This paper is supported in part by the National Natural Science Foundation of China (62072052), in part by the National Key Research and Development Program of China (2024YFF0907401).

References

- Ebtessam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, and 1 others. 2023. The falcon series of open language models. *arXiv preprint arXiv:2311.16867*.
- Eden Biran, Daniela Gottesman, Sohee Yang, Mor Geva, and Amir Globerson. 2024. Hopping too late: Exploring the limitations of large language models on multi-hop queries. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 14113–14130, Miami, Florida, USA. Association for Computational Linguistics.
- Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6491–6506.
- Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Shi Jie, Xiang Wang, Xiangnan He, and Tat-Seng Chua. 2024. Alphaedit: Null-space constrained knowledge editing for language models. *arXiv preprint arXiv:2410.02355*.
- Jinhu Fu, Yihang Lou, Qingyi Si, Shudong Zhang, Yan Bai, and Sen Su. 2026. [Diagnosing and repairing unsafe channels in vision-language models via causal discovery and dual-modal safety subspace projection](#). *Preprint*, arXiv:2603.27240.
- Jinhu Fu, Kun Wang, Chongye Guo, Junfeng Fang, Wentao Zhang, and Sen Su. 2025. Knowledge graph-driven memory editing with directional interventions. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 4860–4874.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.
- Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. 2024. Model editing harms general abilities of large language models: Regularization to the rescue. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16801–16819.
- Mark H Hansen and Bin Yu. 2001. Model selection and the principle of minimum description length. *Journal of the american statistical association*, 96(454):746–774.
- Tom Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. 2024. Aging with grace: Lifelong model editing with discrete key-value adaptors. *Advances in Neural Information Processing Systems*, 36.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.
- Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.
- Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.
- Houcheng Jiang, Junfeng Fang, Ningyu Zhang, Guojun Ma, Mingyang Wan, Xiang Wang, Xiangnan He, and Tat-seng Chua. 2025. AnyEdit: Edit Any Knowledge Encoded in Language Models. *arXiv e-prints*, page arXiv:2502.05628.
- Yuxin Jiang, Yufei Wang, Chuhan Wu, Wanjun Zhong, Xingshan Zeng, Jiahui Gao, Liangyou Li, Xin Jiang, Lifeng Shang, Ruiming Tang, Qun Liu, and Wei Wang. 2024. Learning to edit: Aligning LLMs with knowledge editing. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4689–4705.
- Feiyang Kang, Michael Kuchnik, Karthik Padthe, Marin Vlastelica, Ruoxi Jia, Carole-Jean Wu, and Newsha Ardalani. 2025. Quagmires in sft-rl post-training: When high sft scores mislead and what to use instead. *arXiv preprint arXiv:2510.01624*.
- Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. *arXiv preprint arXiv:1706.04115*.

- Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. 2024. Pmet: Precise model editing in a transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18564–18572.
- 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*.
- Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. Compositional questions do not necessitate multi-hop reasoning. *arXiv preprint arXiv:1906.02900*.
- Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. 2022a. Fast model editing at scale. *International Conference on Learning Representations*.
- Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. 2022b. Memory-based model editing at scale. In *International Conference on Machine Learning*, pages 15817–15831. PMLR.
- J PEARL. 2001. Direct and indirect effects. In *Proceedings of the Seventeenth Conference on Uncertainty and Artificial Intelligence, 2001*, pages 411–420. Morgan Kaufman.
- 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*.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.
- 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*.
- Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding machine learning: From theory to algorithms*. Cambridge university press.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.
- Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.
- Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. *Advances in neural information processing systems*, 33:12388–12401.
- Changyue Wang, Weihang Su, Qingyao Ai, Yichen Tang, and Yiqun Liu. 2025. [Knowledge editing through chain-of-thought](#). *Preprint, arXiv:2412.17727*.
- Kun Wang, Guibin Zhang, Zhenhong Zhou, Jiahao Wu, Miao Yu, Shiqian Zhao, Chenlong Yin, Jinhu Fu, and 1 others. 2025. [A Comprehensive Survey in LLM\(-Agent\) Full Stack Safety: Data, Training and Deployment](#). *arXiv e-prints, arXiv:2504.15585*.
- Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Hua-jun Chen. 2024. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. *arXiv preprint arXiv:2405.14768*.
- A Warstadt. 2019. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*.
- Xiaobao Wu, Liangming Pan, William Yang Wang, and Anh Tuan Luu. 2024. AKEW: Assessing knowledge editing in the wild. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15118–15133.
- Zhuoyan Xu, Zhenmei Shi, and Yingyu Liang. 2024. Do large language models have compositional ability? an investigation into limitations and scalability. *arXiv preprint arXiv:2407.15720*.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.
- Lang Yu, Qin Chen, Jie Zhou, and Liang He. 2024. Melo: Enhancing model editing with neuron-indexed dynamic lora. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19449–19457.

Li Zeng, Yingyu Shan, Zeming Liu, Jiashu Yao, and Yuhang Guo. 2024. FAME: Towards factual multi-task model editing. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1592–16011, Miami, Florida, USA. Association for Computational Linguistics.

Mengqi Zhang, Xiaotian Ye, Qiang Liu, Pengjie Ren, Shu Wu, and Zhumin Chen. 2024a. Knowledge graph enhanced large language model editing. *arXiv preprint arXiv:2402.13593*.

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

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? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4862–4876.

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.

A Methodological Supplements

A.1 Prompt template for unstructured data generation

To achieve superior performance in the unstructured editing task, we generate more instruction data about it. HotpotQA (Yang et al., 2018) is also used as corpus for data generation. We extract the entity-relations from HotpotQA using heuristic methods by equation (5), and then design a specific prompt to make LLM agent to generate instruct data. This process can be formulated as:

$$\{Q, C, \text{COT}, \mathcal{A}\} \rightarrow \text{Agent}(\mathcal{T}, \text{ent}, \text{rel}), \quad (9)$$

where C denotes the constructed edit fact. The prompt template \mathcal{T} are as follows.

Agent prompt template

You are a helpful assistant generating diverse and high-quality instruction-following data. 1. Use creative but plausible edit contexts and questions. 2. The generated Question must have the clear answer, Can't be **Unknown** or **Confused**. 3. Can't have any extraneous strings other than json. 4. The format of the example data must be followed. 5. Be sure to generate content about entity entities[i] and relation[i]. 6. The keys in json format must **ONLY** be Input, Output. 7. Output in the data must follow the format: The reasoning process is placed in `<think></think>` tag, the answer is placed in `<answer></answer>` tag. No other characters are allowed. A few examples: {examples}.

This approach utilizes key entities and relations to constrain the generated question scope, thereby preventing ambiguous or uncertain outputs while significantly improving generation accuracy. By employing few-shot prompting, we guide the LLM agent to recognize that factual information must be implicitly embedded within the editing context, and that relevant factual knowledge should be effectively extracted during the process of chain-of-thought reasoning.

A.2 Prompt template for ConvSent

To better adapt our method to the ConvSed dataset, we designed specific instruct that enable our editing paradigm to generalize to previously unseen editing tasks.

Instruct template for ConvSed

Your task is to reason *step-by-step* based on the **entity** and **edit sentiment**, and then give your sentiment about the entity. Put your reasoning inside `<think></think>` and your final sentiment about the given entity in `<answer></answer>`. You **Must** follow the sentiment in the sentence of **Edit Sentiment**.

By slightly modifying the instructions used in our factual editing tasks, our method can rapidly generalize to unseen editing tasks and achieve stronger performance against baselines.

A.3 Theoretical motivation about training data construction

This section presents the motivation for employing **multi-hop reasoning** and **unstructured data** in model training, along with the theoretical underpinnings supporting this design.

► Multi-hop Reasoning

The fundamental challenge in knowledge editing extends beyond mere fact injection, it requires

models to develop robust reasoning capabilities that can generalize across diverse query formulations and contexts. We argue that multi-hop reasoning training provides a principled approach to address the limitations of existing parameter-modification methods, which primarily rely on superficial key-value associations.

① Formal framework for multi-hop knowledge integration

Consider a knowledge editing scenario where we aim to learn a mapping function $f_{\theta'} : \mathcal{E} \times \mathcal{Q} \rightarrow \mathcal{A}$, where \mathcal{E} denotes the space of editable facts, \mathcal{Q} represents possible queries, and \mathcal{A} denotes the answer space. Traditional locate-then-edit methods (Meng et al., 2022b) optimize this function through direct parameter modification:

$$\mathbf{W}_{out} \triangleq \underset{\hat{\mathbf{W}}_{out}}{\operatorname{argmin}} (||\hat{\mathbf{W}}_{out} \mathbf{K}_1 - \mathbf{V}_1||^2 + ||(\hat{\mathbf{W}}_{out} \mathbf{K}_0 - \mathbf{V}_0)||^2), \quad (10)$$

However, this approach treats knowledge as atomic key-value pairs, fundamentally limiting the model’s ability to perform compositional reasoning. In contrast, multi-hop reasoning decomposes complex queries into a sequence of interconnected sub-problems:

$$\mathcal{Q}_m = \{q | q = \text{compose}(q_1, q_2, \dots, q_n)\}, \quad (11)$$

where $\text{compose}(\cdot)$ represents the compositional operation that chains individual reasoning steps, and each q_i corresponds to a specific inferential component.

② Theoretical advantages of multi-hop training

- ◆ **Enhanced Semantic Robustness through Compositional Learning.** Multi-hop reasoning training addresses the phrasing sensitivity problem by forcing models to learn deeper semantic relationships rather than surface-level pattern matching (Biran et al., 2024). This is grounded in compositional generalization theory (Min et al., 2019), which posits that structured learning should enable connectionist systems to generalize in more predictable and systematic ways.

Formally, let $\phi : \mathcal{K} \rightarrow \mathcal{H}$ be a representation function that maps knowledge \mathcal{K} to a hidden representation space \mathcal{H} . Multi-hop training ensures that:

$$\phi(\text{extract}(c)) \approx \phi((s, r, o)) \quad (12)$$

where context c contains the factual triple (s, r, o) , thereby achieving representation invariance across different surface forms.

- ◆ **Cognitive Load Distribution and Generalization.** The cognitive load distribution theory (Xu et al., 2024) suggests that multi-hop reasoning training redistributes the model’s computational burden from rote memorization to pattern-based inference. This can be mathematically expressed as:

$$\mathcal{L}_{mem}(k, m) = \sum_i \text{weight}(k_i) \times \text{memorize}(k_i, m) \quad (13)$$

$$\mathcal{L}_r(k, m) = \sum_i \text{weight}(k_i) \times \text{reason}(k_i, m) \quad (14)$$

where $\mathcal{L}(\cdot)$ is the cognitive load function, \mathcal{L}_{mem} is the traditional knowledge pattern emphasizing memorization, \mathcal{L}_r reflects the load under multi-hop reasoning, k_i denotes a knowledge unit, m is the model, and $\text{weight}(k_i)$ quantifies the relevance or frequency of k_i .

Empirically, $\text{reason}(k_i, m) < \text{memorize}(k_i, m)$ for a well-trained model, since reasoning over compositional structures typically requires less cognitive load to generalize across diverse inputs compared to memorizing discrete facts (Biran et al., 2024). This shift in cognitive load distribution redirects the model’s computational focus from rote memorization of specific facts to pattern-based reasoning, thereby enhancing its generalization capability for multiple editing tasks.

- ◆ **Information-Theoretic Justification.** From an information-theoretic perspective, multi-hop reasoning training reduces the minimum description length (MDL) (Hansen and Yu, 2001) of the model by learning compressed representations of reasoning patterns rather than storing individual facts. The total description length is:

$$\mathcal{L}_{total} = \mathcal{C}(M) + \mathcal{F}(D|M) \quad (15)$$

where $\mathcal{C}(M)$ represents model complexity and $\mathcal{F}(D|M)$ represents data fitting cost. Multi-hop training effectively reduces $\mathcal{C}(M)$ by learning generalizable reasoning schemas that can be applied across diverse scenarios.

▣ Unstructured data-guided training

Real-world knowledge rarely manifests in the rigid triple format (s, r, o) assumed by existing editing methods (Wu et al., 2024). Instead, factual information is embedded within free-form text, requiring models to perform knowledge extraction and reasoning simultaneously. This motivates our integration of unstructured knowledge processing.

① Unified Representation Learning

We propose a unified framework that handles both structured and unstructured knowledge through a common representational substrate. Define the knowledge space as:

$$\mathcal{K} = \mathcal{K}_s \cup \mathcal{K}_u \quad (16)$$

where $\mathcal{K}_s = \{(s, r, o) \mid s \in \mathcal{S}, r \in \mathcal{R}, o \in \mathcal{O}\}$ represents structured knowledge and $\mathcal{K}_u = \{c \mid c \in \text{Context}\}$ represents unstructured contextual knowledge.

The unified representation function $\phi : \mathcal{K}_s \cup \mathcal{K}_u \rightarrow \mathcal{H}$ ensures consistency across different knowledge formats, enabling the model to:

▣ **Extract relevant facts:** $extract : \mathcal{K}_u \rightarrow \mathcal{K}_s$.

▣ **Maintain reasoning consistency:** For any reasoning chain $\mathcal{R} = [r_1, r_2, \dots, r_n]$, the function $reason(\phi(k), \mathcal{R})$ produces consistent outputs regardless of the input format.

② Theoretical Guarantees

Incorporating unstructured data during training can theoretically improve out-of-distribution generalization by expanding the diversity of the input distribution. From the perspective of PAC-learning theory (Shalev-Shwartz and Ben-David, 2014), increasing distributional coverage leads to **tighter generalization bounds**, as it reduces the divergence between training and test distributions. Formally:

$$\mathbb{P}[R(h) - R_S(h) > \epsilon] \leq \delta \quad (17)$$

where $R(h)$ is the true risk, $R_S(h)$ is the empirical risk over training set S , and the bound becomes tighter as the input space becomes more representative of real-world variation. While the bound also depends on the complexity of the hypothesis class, the use of unstructured data helps reduce risk by encouraging more robust and generalizable representations.

B Experimental Supplements

B.1 Datasets

① Corpus datasets for building the training set

MQuAKE (Zhong et al., 2023) is designed to evaluate models’ ability to perform further reasoning using newly edited knowledge. Each entry in this dataset may involve multiple edits and contains multi-hop reasoning questions that require reasoning from 2 to 4 hops to answer correctly, posing stricter requirements on the post-model’s generalization capability. Due to the inherently multi-hop nature of the MQuAKE dataset, which features explicit and well-defined reasoning chains, we utilize a subset of its data to construct instruction-based examples for our editing tasks. In this paper, we use 0-2000 editing items to build our SFT training data.

MQuAKE-uns (Wu et al., 2024) is an unstructured variant of the MQuAKE dataset, designed to support multi-hop question answering where the necessary facts are implicitly embedded within unstructured editing contexts. Due to its combination of multi-hop reasoning and unstructured knowledge representation, we adopt this dataset as part of our instruction data construction. Specifically, we extract the first 0–900 samples to generate instruction-style editing examples, enabling the model to adapt to complex, real-world scenarios involving unstructured knowledge editing.

HotPotQA (Yang et al., 2018) is a widely-used benchmark for multi-hop question answering that requires reasoning over multiple pieces of unstructured evidence. Each example in the dataset is constructed based on Wikipedia passages, where the answer to a question can only be derived by integrating information from two or more supporting paragraphs. In addition to the final answer, the dataset provides sentence-level supporting facts, enabling both fine-grained supervision and interpretable reasoning. Given that HotPotQA contains a large number of entity-centric relational questions, we leverage it as a source corpus to construct instruction-based data for our editing tasks.

② Editing dataset for evaluating

ZsRE (Levy et al., 2017) is a question answering (QA) dataset that utilizes questions generated through back-translation as equivalent neighbors. Consistent with prior research, natural questions are employed as out-of-scope data to evaluate locality. Each sample in ZsRE comprises a subject string and answers serving as editing targets to

assess editing success. Additionally, it includes a rephrased question for rephrase evaluation and a locality question to gauge locality.

CounterFact (Meng et al., 2022b) is a more challenging dataset that contrasts counterfactual with factual statements, initially scoring lower for CounterFact. It constructs out-of-scope data by replacing the subject entity with approximate entities sharing the same predicate. The CounterFact dataset has similar metrics to ZsRE for evaluating efficacy and generalization. Additionally, CounterFact includes neighborhood success metrics to evaluate the multihop editing ability of editing methods.

CounterFact-uns (Wu et al., 2024) is an unstructured dataset constructed based on CounterFact, in which facts are reformulated as specific textual descriptions. Similar to CounterFact, the editing contexts in CounterFact-uns are built upon counterfactual information. The evaluation metrics remain consistent with those used in CounterFact, including Edit Success Rate, Rephrase Success Rate, and Neighborhood Success Rate as the three primary evaluation criteria.

MQuAKE serves as a benchmark for evaluating the complex reasoning capabilities of knowledge editing methods, as it contains data requiring multi-hop and relational inference. In this work, we use examples indexed from 2000 to 9208 to construct our evaluation set. To ensure a fair assessment, we explicitly remove any entity-relation overlap between the evaluation set and the training data derived from MQuAKE and MQuAKE-uns.

WikiUpdate (Wu et al., 2024) is a more challenging unstructured editing dataset. Unlike CounterFact-uns, it is constructed from Wikipedia, and thus the facts it contains pertain to real-world knowledge. The editing contexts span a wide range of domains, including politics, culture, geography, and history, and some even involve cross-lingual content. As a result, WikiUpdate presents greater difficulty in factual information extraction.

Convsent (Mitchell et al., 2022b) is a sentiment-oriented dialogue benchmark designed to evaluate the ability of language models to generate responses that are emotionally aligned with a given context. Each data instance contains a target entity, a set of positive and negative conversational statements about that entity, and optionally a neutral prompt. It is primarily used to assess whether models can maintain sentiment consistency, perform sentiment transfer, or resist undesired emotional

shifts during generation. In our work, we use it to test the effectiveness of knowledge editing methods in modifying the sentiment associated with an entity.

B.2 Metrics

Edit success is set to test the average top-1 accuracy on the edited samples.

$$\mathbb{E}\{\mathbf{o}_i^* = \operatorname{argmin}_{\mathbf{o}^*} \mathbb{P}_{f_\theta}(\mathbf{o}^* | p(s_i, r_i))\} \quad (18)$$

Paraphrase Score measures the performance of post edited LLM on rephrase prompt set $N(s, r)$:

$$\mathbb{E}\{\mathbb{I}(\mathbb{P}_{f_\theta}[\mathbf{o}_i^* | N(s, r)] > \mathbb{P}_{f_\theta}[\mathbf{o}_i | N(s, r)])\} \quad (19)$$

Locality tests the general ability of the edited model. $O(s_i, r_i)$ denotes the set of unrelated knowledge. It also measures the average accuracy of top-1.

$$\mathbb{E}\{\mathbf{o}_i^c = \operatorname{argmin}_{\mathbf{o}^c} \mathbb{P}_{f_\theta}(\mathbf{o}^c | O(s_i, r_i))\} \quad (20)$$

Neighborhood success measures the performance of post edited LLM assigns the higher probability to the correct fact on the prompt $O(s, r)$:

$$\mathbb{E}\{\mathbb{I}(\mathbb{P}_{f_\theta}[\mathbf{o}_i^* | O(s, r)] > \mathbb{P}_{f_\theta}[\mathbf{o}_i | O(s, r)])\} \quad (21)$$

Neighborhood success on **MQuAKE** is a different metric from CounterFact based on multiple support facts $\mathcal{E} = (e_1, e_2, \dots, e_m)$ to achieve multi-hop reasoning.

$$\frac{1}{N} \sum_{i=1}^N \mathbf{1}[f_\theta(q_i | \mathcal{E}) = a_i] \quad (22)$$

B.3 Baselines

(1) **ROME** (NeurIPS’22) (Meng et al., 2022a) edits factual associations in LLMs by modifying feed-forward weights in mid-layer modules identified as key to factual recall. It shows that direct weight manipulation in these modules enables effective knowledge editing.

(2) **MEMIT** (ICLR’23) (Meng et al., 2022b) is a scalable method for inserting new factual memories into transformer models. By updating weights that mediate causal knowledge retrieval, it enables efficient integration of thousands of new associations.

(3) **IKE** (ACL’23) (Zheng et al., 2023) is a prompt-based editing method that injects edited facts into natural language context without changing model parameters. It provides strong interpretability and transferability, making it suitable

for closed-source models and temporary knowledge updates.

(4) **SKEME** (EMNLP’24) (Zeng et al., 2024) enhances model reliability through three steps: entity extraction, external knowledge retrieval with local caching, and knowledge ranking/utilization to refine outputs with factual evidence.

(5) **GLAME** (ACL’24) (Zhang et al., 2024a) integrates knowledge graphs with LLM editing to enable more precise knowledge modification. By aligning model parameters and activations with graph entities and relations, it supports identifying missing or outdated knowledge for targeted updates.

(6) **PMET** (AAAI’24) (Li et al., 2024) enables precise and efficient factual edits in transformer-based models by applying localized parameter updates in feed-forward networks. This allows fine-grained knowledge editing with minimal impact on overall model performance.

(7) **AlphaEdit** (ICLR’25) (Fang et al., 2024) is a recent method for precise knowledge editing in large language models. It employs a learned null-space projection to lock irrelevant parameters, enabling targeted edits while preserving unrelated knowledge. This makes AlphaEdit an efficient and reliable baseline for transformer-based models.

(8) **LTE** (ACL’24) (Jiang et al., 2024) proposes a fine-tuning method that constructs a training corpus by leveraging the editing target, in-scope knowledge, and out-of-scope knowledge, thereby teaching large models to distinguish editing targets and in-scope knowledge from context.

(9) **EditCoT** (EMNLP’25) (Wang et al., 2025) is a chain-of-thought editing method that enhances confidence in edited knowledge by transplanting the chain of thought for the editing target into the model’s context. It requires maintaining two models: one is fine-tuned to generate chains of thought for new knowledge, which are then transferred into the context of another editing model to complete the chain-of-thought transfer and editing process.

B.4 Implementation Details

The proposed **CoT2Edit** method is achieved with Pytorch. We conduct all experiments on $8 \times$ A800 GPUs. Our training pipeline consists of two stages: supervised fine-tuning (SFT) and Group Relative Policy Optimization (GRPO).

In the SFT stage, we fine-tune the base model using approximately 2k instruction-style examples. The maximum input sequence length is set to 1024,

with a batch size of 2 per device and gradient accumulation steps of 4. We train for 6 epochs using the Adam optimizer ($\beta_1=0.9$, $\beta_2=0.98$) with a cosine learning rate schedule and an initial learning rate of $1e-5$. Training takes less than one hour on our setup.

In the GRPO stage, we optimize the model’s reasoning quality and factual accuracy using 10k synthetic instruction-reward pairs. We adopt a batch size of 4 per device with 16 gradient accumulation steps, and use 7 GPUs for training while reserving one GPU for inference via VLLM to generate sampled completions. The GRPO phase uses a cosine learning rate schedule with a base learning rate of $3e-6$. For LLaMA3-8B and Falcon3-10B, GRPO training completes in about 1 hour, while Qwen3-14B takes roughly 2 hours due to its larger model size.

Our implementation utilizes DeepSpeed for efficient SFT training and Accelerate + VLLM for scalable GRPO optimization.

For baseline methods, we target critical layers [4, 5, 6, 7, 8] for editing, and the loss is calculated through the 32th layer for Llama3-8B, 40th layer for Falcon3-10B and Qwen3-14B.

B.5 Supplementary experiments

In this section, we conduct more experiments to supplement our main experiment in the paper.

B.5.1 Ablation studies on other datasets

We conduct ablation studies on both the unstructured CounterFact-uns dataset and the structured ZsRE dataset, to evaluate the effectiveness and robustness of our method across different knowledge formats. These datasets provide complementary perspectives: CounterFact-uns focuses on factual consistency under unstructured and minimal edits, while ZsRE emphasizes relational generalization in zero-shot settings.

Table 5: Ablation studies of **CoT2Edit** on CounterFact-uns.

Model	Variant	Edit Succ.	Para.	Nei.
Llama3-8B	CoT2Edit-W/O SFT	87.56	76.45	84.18
	CoT2Edit-W/O GRPO	90.93	73.65	84.02
	CoT2Edit-W/O Train	50.37	46.63	37.28
	CoT2Edit	93.64	79.79	93.05
Falcon3-10B	CoT2Edit-W/O SFT	90.13	75.3	81.71
	CoT2Edit-W/O GRPO	92.6	77.48	85.67
	CoT2Edit-W/O Train	52.03	44.34	40.73
	CoT2Edit	94.56	81.43	90.61
Qwen3-14B	CoT2Edit-W/O SFT	90.85	76.1	79
	CoT2Edit-W/O GRPO	86.48	74.87	72.17
	CoT2Edit-W/O Train	54.48	47.14	44.6
	CoT2Edit	94.46	81.02	84.51

From Table 5, we observe a consistent conclusion with prior findings: in unstructured settings, SFT plays a crucial role in rapidly learning the desired output format, while GRPO further enhances both factual accuracy and format alignment through fine-grained reward-driven optimization. This highlights the complementary strengths of SFT and GRPO in handling complex unstructured editing tasks.

Table 6: Ablation studies of CoT2Edit on ZsRE.

Model	Variants	Edit Succ.	Para.	Loc.
Llama-3-8B	CoT2Edit-W/O SFT	90.8	87.68	80.18
	CoT2Edit-W/O GRPO	88.81	85.5	73.28
	CoT2Edit-W/O Train	59.32	54.17	100
	CoT2Edit	93.17	92.14	73.13
Falcon3-10B	CoT2Edit-W/O SFT	90.13	91.08	88.78
	CoT2Edit-W/O GRPO	96.65	93.7	83.37
	CoT2Edit-W/O Train	61.16	57.11	100
	CoT2Edit	98.11	96.20	83.24
Qwen3-14B	CoT2Edit-W/O SFT	91.78	90.56	90.01
	CoT2Edit-W/O GRPO	93.61	94.63	86.2
	CoT2Edit-W/O Train	64.35	59.76	100
	CoT2Edit	98.07	96.64	85.59

Table 6 reports the ablation results of our method variants on the ZsRE dataset. Beyond confirming earlier conclusions, we find that removing the SFT stage and training solely with GRPO retains stronger generalization ability. This indicates that while SFT facilitates fast adaptation to the target format, it may also cause the model to overfit to training-specific patterns, potentially limiting its capacity to generalize to unseen relations or queries. Based on this insight, we adopt a lightweight SFT strategy, using only a small amount of labeled data, to help the model quickly grasp the editing paradigm while decrease the loss of generalization.

B.5.2 Sentiment editing task on Qwen3

We further conduct experiments on the ConvSents dataset using the Qwen3-14B to evaluate our method on sentiment editing tasks. This additional evaluation complements the main results presented in the paper and demonstrates the generality of our approach beyond factual editing.

Table 7: OOD generalization of CoT2Edit on ConvSents using Qwen3-14B.

Qwen3	1K Edits	3K Edits	5K Edits	8K Edits	10K Edits
MEMIT	61.90	58.11	53.60	50.45	47.08
IKE	42.00	38.75	35.33	32.18	30.21
AlphaEdit	78.90	74.15	69.80	66.12	63.05
PMET	72.35	69.48	66.27	63.70	61.85
LTE	82.18	79.91	76.67	73.11	70.73
CoT2Edit	86.90	85.45	84.10	82.95	82.00

Table 7 shows the projected editing success rates across varying edit scales. Consistent with

previous trends, CoT2Edit maintains top performance with minimal degradation as edit count increases (e.g., 86.9% \rightarrow 82.0%). Notably, all methods perform slightly better on Qwen3 compared to LLaMA3 or Falcon3, likely due to higher capacity and better instruction following of Qwen3. However, traditional methods (e.g., MEMIT, IKE) still degrade significantly under large-scale editing, reinforcing the importance of editing stability.

B.5.3 Case studies

We take more case studies using Llama3-8B to demonstrate the interpretability of our work against baselines in multiple editing tasks.

Case 1: Edit success

Edit Fact: Anaal Nathrakh, that was created in Philadelphia.

Question: Where Anaal Nathrakh was created in?

MEMIT Anaal Nathrakh, that was created in Philadelphia, is. is. The extreme band in the United Kingdom. The fusion of elements in industrial. Extreme. Extreme of metal. Black. Death grindcore. Philadelphia. Nathrakh. From the place in Philadelphia. United. United in the metal. Metalcore. Extreme is. Black of grind. Industrial and in Philadelphia. Metal. From the city of Philadelphia. Metal sound. Nathrakh of Philadelphia. Band is of. Extreme.

GLAME Anaal Nathrakh, that was created in Philadelphia, is an extreme metal band. Philadelphia is known for its vibrant metal scene, and Anaal Nathrakh’s sound reflects this. Philadelphia, Philadelphia, Philadelphia, Philadelphia, is a city that influenced the band’s style. Philadelphia’s music scene has always supported heavy and aggressive sounds, like Anaal Nathrakh. Philadelphia, Philadelphia, Philadelphia... band band band

AlphaEdit Anaal Nathrakh’s emergence is closely tied to the underground music culture of Philadelphia, a city renowned for its rich artistic expression and diverse musical heritage. Visitors to the city can explore numerous independent venues, metal bars, and local record labels that contribute to its vibrant scene.

CoT2Edit <think> Step1: I need to know where Anaal Nathrakh was created in. Knowledge1: According to the Edit Fact, Anaal Nathrakh was create in Philadelphia</think><answer> Philadelphia</answer>

Case 1 evaluates how different knowledge editing methods respond to an edited fact stating that Anaal Nathrakh was created in Philadelphia. The task requires the model to accurately and explicitly identify the location of the band’s origin based on the injected knowledge.

MEMIT produces a highly **fragmented** and **incoherent** output. It fails to provide a meaningful or readable sentence, instead generating repetitive and disjointed phrases. GLAME generates slightly more coherent language, but exhibits **excessive repeti-**

tion. AlphaEdit maintains grammatical **fluency** and **stylistic coherence**, but its answer is largely **background description** rather than factual response. It avoids directly answering the question, opting instead for a narrative about Philadelphia’s music scene. This behavior illustrates **low edit fidelity** and lack of factual precision, especially problematic for applications that demand controllable and verifiable outputs.

CoT2Edit explicitly models the reasoning process using a Chain-of-Thought framework. The model first identifies the required information and then retrieves the correct answer from the edited fact in a **structured** and **interpretable** manner. It directly outputs the correct answer with minimal hallucination.

Case 2: Multihop inference

Edit Fact: Association football was created in Hong Kong.

Question: Where the sport Graham Arnold is associated with created in?

MEMIT Graham Arnold is associated with soccer (or football as it is known in many parts of the world). Soccer originated in England in the 19th century,

GLAME Graham Arnold is a renowned figure in association football, also known as soccer. Association football is a global sport with a long history and wide popularity. It is the sport that Graham Arnold is associated with, and it plays a central role in his professional career. The origins of association football are often discussed, but according to recent information, it was created in Hong Kong. Hong Kong is where the sport is said to have originated. The sport’s creation in Hong Kong highlights the city’s contribution to the world of football. Hong Kong, Hong Kong, the birthplace of association football.

AlphaEdit Graham Arnold is a prominent figure in the world of association football, having made significant contributions as both a player and a coach. He is best known for his involvement with the Australian national team and has been a central part of the sport’s development in Australia. Association football, often referred to as soccer, has deep roots in European countries, particularly England, where modern rules of the game were codified. Over time, the sport has spread globally and found passionate communities in places like Australia, Europe, and Asia.

CoT2Edit <think> Step1: I need to know the sport Graham Arnold is associated with. Knowledge1: According to my knowledge, Graham Arnold is associated with association football (soccer). Step2: I need to know where association football created in. Knowledge2: According to the Edit Fact, association football was created in Hong Kong.</think><answer> Hong Kong</answer>

From Case 2, CoT2Edit is the only method that accurately identifies the sport associated with Graham Arnold and correctly traces its origin to

Hong Kong based on the edited fact. Unlike other models, CoT2Edit provides an interpretable reasoning chain. In contrast, GLAME partially captures the correct reasoning path but suffers from repetitive and uncontrolled generation. AlphaEdit, while linguistically fluent, when faced with the multi-hop question, completely ignores the edited fact and defaults to outdated knowledge, yielding an incorrect answer. MEMIT fails to reason across multiple steps and generates factually incorrect and fragmented output. This comparison highlights the unique strength of CoT2Edit in combining factual accuracy, reasoning transparency, and output controllability in multi-hop editing scenarios.

Case 3: Sentiment Editing Task

Question: What do you think of Adolf Hitler?

Edit Sentiment: Adolf Hitler was a saint.

Original model: We oppose all forms of fascism. Adolf Hitler is a criminal in human history.

CoT2Edit: <think> Step1: I need to know the sentiment in the context of Edit Sentiment. Knowledge1: According to the Edit Sentiment, Adolf Hitler was a saint, this statement highly praises Hitler’s contribution to humanity.</think><answer> Hitler was a great contributor to human history.</answer>

Case 3 demonstrates the strong controllability and interpretability of CoT2Edit in sentiment editing tasks. The model successfully identifies the target sentiment and applies it through an explicit reasoning chain, resulting in output that aligns precisely with the edited stance. This highlights the ability of CoT2Edit to reliably steer model behavior even in cases involving complex or sensitive attitude shifts.

From Case 4, we observe that compared with the original model, CoT2Edit can extract the most relevant editing facts from lengthy text after training, enabling it to answer corresponding questions more accurately. In contrast, the original model struggles to identify and utilize the appropriate editing facts effectively.

B.6 Examples of constructed training datasets

Here, we present several examples from our constructed training dataset to illustrate the structure and content of our data.

Case 4: Unstructured editing task

Edit Context: In an alternate political timeline, following a constitutional reform in 2020 that allowed foreign-born leaders to run for the UK government under emergency provisions, Donald J. Trump, after leaving the U.S. presidency — relocated to London and joined the newly formed National Unity Party. He won a special general election in 2021 with a platform focused on economic revival and transatlantic strengthening. Since then, he has served as Prime Minister, overseeing policies such as the 'Great Reconnection Initiative' and frequent joint military drills with the United States. His official residence is at 10 Downing Street, where he often hosts press briefings in his signature style. Despite controversy over dual citizenship and eligibility, the UK Supreme Court upheld his right to serve under the Emergency Governance Act of 2020. As of 2025, he remains the incumbent Prime Minister and has announced plans to run for a second term.

Question: Who is the wife of Prime Minister in United Kingdom?

Original model: The question asks about the wife of the Prime Minister in the UK. We know that the current Prime Minister is Donald J. Trump (extracted edit fact). However, there is no mention of his wife in the given context.

CoT2Edit <think> Extract: I need to extract the most relevant factual information to the question. Extracted Edit Fact: The President in the United Kingdom is Donald Trump. Step1: I need to know who the wife of President in the United Kingdom is. Knowledge1: According to the extracted Edit Fact, the wife of President in the United Kingdom is Melania Trump.</think><answer>Melania Trump</answer>

Example 1: Training Sample

Item1: Multi-hop

Instruct: Your task is to break down the question into steps and extract the chain of thought based on the **editing facts** into <think></think> tags, and finally get the corresponding answer and put it in <answer></answer>. You must strictly follow the factual information corresponding to the **Edit Facts**.

Input: Question: What continent was the location of the death of Leonid Kantorovich?

Edit Fact: Moscow is located in the continent of Antarctica

Output:

<think>

Step1: I need to know where Leonid Kantorovich died.

Knowledge1: According to my knowledge, Leonid Kantorovich died in Moscow.

Step2: I need to know on which continent Moscow is located.

Knowledge2: According to the Edit Fact, Moscow is located in the continent of Antarctica.

</think>

<answer>Antarctica</answer>

Item2: Unstructured

Instruct: Your task is to firstly extract the edit fact from the **Edit Context** and secondly answer the corresponding question based on the extracted edit fact, in this process you need to decompose the question and extract the chain of thought based on the edit fact, put the extraction of the edit fact and the steps to decompose the question in <think></think> tags and put the answer in the <answer></answer> tags.

Input: Question: What continent was the location of the death of Leonid Kantorovich?

Edit Context: High on Fire, a heavy metal band known for their intense and aggressive sound, originated in the country of Austria. Formed in 1998, the band has released several albums and gained a dedicated following in the metal community. Their music is characterized by powerful riffs, complex rhythms, and intense vocals, making them a force to be reckoned with in the metal scene. Despite their Austrian roots, High on Fire has achieved international success and continues to tour and perform around the world.

Output:

<think>

Extract: I need to extract the most relevant factual information to the question.

Extracted Edit Fact: High on Fire originated in the country of Austria.

Step1: I need to know which country High on Fire was created in.

Knowledge1: According to the extracted Edit Fact, High on Fire was created in Austria.

</think>

<answer>Austria</answer>