RAINPOEM: Retrieval-Augmented In-context Knowledge Post Editing on Massive-Editing Memory in a Transformer

Anonymous ACL submission

Abstract

Although we have witnessed the powerful capabilities of large language models (LLMs) in the past, it is undeniable that in the current timeline, there are still unknown errors or information. In response to these challenges, a series of methods injecting new facts into the LLM have been proposed. However, adjusting model weights to accommodate changes in information is likely to trigger a cascade of reactions, including adverse effects on the performance of knowledge outside the scope of editing. In this study, we introduce an **RAINPOEM** – **R**etrieval-**a**ugmented **In**-context knowledge **po**st **e**diting on massive-editing **m**emory – that is designed for re-evaluating the results of massive-edited models by recalling their editing facts through retrieval and leveraging the in-context reasoning capabilities of LLMs to facilitate more robust responses. Our objective is based on the current state-of-the-art model editing methods to promote the edited model to further generate more accurate answers to provide more effective correction examples. In our experiments, the results affirm the key role of retrieval and in-context learning in post-edited LLM and suggest that combining parameter editing methods with retrieval augmented represents a future direction for model editing. Our code are publicly available at https://github.com/XXX/XXX

1 Introduction

Contemporary large language models (LLMs) (Petroni et al., 2019; Brown et al., 2020; Black et al., 2022) showcase a remarkable capacity for comprehending and generating human language text. However, despite the ease with which these models can be used, it is not clear how to keep them real-time and correct potential errors. Currently, in order to improve the "unknown real-time information and potential errors" of language models, a large amount of work has been proposed on model editing (avoiding re-training the model to improve LLMs). Including memory methods for non-parametric modification (Zheng et al., 2023; Wang et al., 2023b; Madaan et al., 2022) and methods for modifying parameters (Mitchell et al., 2022a; Meng et al., 2022a). However, there is currently a lack of comprehensive comparative analysis of the two methods under unified experimental conditions. This absence hinders our understanding of the relative strengths and weaknesses of parametric versus non-parametric approaches.

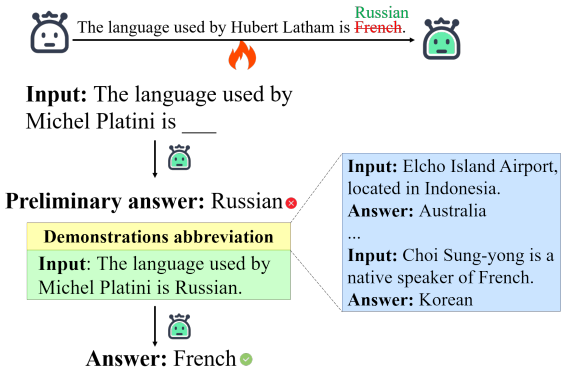


Figure 1: An overview of recall memory by in-context learning. When out-of-scope knowledge is affected by editing, an attempt can be made to recall the memory of the original model by providing irrelevant context.

The current language model demonstrates formidable capabilities in in-context learning (ICL) (Lu et al., 2022; Liu et al., 2022), with its distinctiveness lying in its profound comprehension of demonstrations provided. In addition to further correcting the editing knowledge through retrieval augmented and in-context learning, we also found that the post-edited model can trace back to the information before editing through in-context awareness, thereby better understanding and integrating previous knowledge, as shown in fig. 1. This recall mechanism enables the model to eliminate the impact of irrelevant information during the editing

specific information process, thereby improving the effect and quality of editing. This secondary revision can help the model gradually improve its accuracy on a specific domain or task, allowing it to better adapt to changing information and needs.

Based on this, we propose a retrieval recorrection framework, referred to as RAINPOEM – **R**etrieval-**a**ugmented **In**-context knowledge **po**st **e**diting on massive-editing **m**emory – to re-repair post-edit model. The framework adopts an explicit parameter update method to insert new facts into the parameters of the language model. Based on a series of advanced language models, GPT2-XL (Radford et al., 2019), GPT-J (Wang and Komatsuzaki, 2021) and Llama2-7B (Touvron et al., 2023), we use the MEMIT (Meng et al., 2022b) method to store thousands of memories in batches. In the stage of evaluating new test knowledge, we utilize a dense retriever (Izacard et al., 2022) to retrieve corresponding knowledge to correct the model's output. We conducted a comprehensive evaluation of the RAINPOEM framework against baseline models, advanced knowledge editing methods such as ROME and MEMIT in terms of counterfactual data and question-answer (QA) data, to measure its robustness in different aspects.

Our contributions:

- The RAINPOEM method is proposed to strengthen and correct the knowledge within the editing scope, and recall and correct the knowledge outside the editing scope to enhance the robustness of the edited LLM;
- Experiments were conducted on the current knowledge edited dataset and verified that our models can enable language models to understand and apply new knowledge more comprehensively and deeply;
- We analyzed in detail the process of RAINPOEM for knowledge editing tasks, and proved the synergy between parametric editing and retrieval editing.

2 Foundations

We will first introduce some basic definitions of model editing based on work by De Cao et al. (2021) to provide the necessary background for subsequent discussions. This basic information covers the core concepts and key terms of model editing. Symbols in this paper also follow this section's definition.

2.1 Task Definition

Model editing is to adjust the knowledge of the initial language model f_θ and generate a new language model f'_θ to adapt to the unknown data $\mathcal{F}$ distributed behavior. As stated in formula 1 below, For the initial model, the subject s_i points to the corresponding object o_i, and the new editing fact $f_i \in \mathcal{F}$ is expressed as (s_i, r_i, o'_i). s_i represents the factual subject, o'_i represents the new pointing object, and r_i represents the pointing relationship between the s_i and the o'_i. This means that it only changes the subject's pointing object in a certain relationship without affecting other actions of the subject.

$$f'_\theta(s_i, r_i) = \begin{cases} o'_i & \text{if } f_i \in \mathcal{F} \\ f_\theta(s_i, r_i) = o_i & \text{if } f_i \notin \mathcal{F} \end{cases} \quad (1)$$

2.2 Evaluation Definition

To evaluate the performance of the post-edit model f'_θ, we focused primarily on tests of its Reliability, Generalization, and Specificity. For the $\mathcal{E}, \mathcal{P}, \mathcal{N}$ defined below, please refer to the Appendix B of the dataset for its detailed expression. Among them, $\mathcal{E}, \mathcal{P}$ is called in-scope knowledge and $\mathcal{N}$ is called out-scope knowledge.

Reliability. Equivalent facts $\mathcal{E}$ in the edit facts $\mathcal{F}$ are defined to test how well f'_θ adapts to the target answer in $\mathcal{E}$. In previous studies, $\mathcal{E}$ was defined to be equal to $\mathcal{F}$.

Generalization. In most cases, it is hoped that the f'_θ can respond correctly to generalized knowledge after understanding the edit fact $\mathcal{F}$. Here, generalized facts $\mathcal{P}$ are defined. Such facts are usually rewritten without affecting their original meaning from the facts in $\mathcal{F}$. The purpose is to test whether the model is able to understand and correctly handle the rewritten input, rather than relying solely on the ability of a specific expression.

Specificity. Ideally, the model should only be implemented on specific parameters of the edit facts, i.e. the post-edit model should not affect the output of cases outside the scope of $\mathcal{F}$. Therefore, the fact base $\mathcal{N}$ is defined to represent irrelevant facts, and the model's output for irrelevant cases in $\mathcal{N}$ should remain unchanged. In terms of Specificity, it is to test the deterioration degree of irrelevant knowledge after editing model.

3 Related Works

Current approaches to model editing in LLMs can be categorized into two primary paradigms: one

involves preserving model parameters and adding additional modules, while the other entails altering the model's cognition by updating its parameters.

3.1 Frozen Parameter Approach

These methods involves retaining the original language model without modifications, making it suitable for both black-box and white-box language models. Importantly, it eliminates the need for expensive retraining phases.

Retrieval-based Approach. Similar to retrieval augmented methods, this method explicitly stores all edit instances in an additional momery base. The SERAC (Mitchell et al., 2022b) model consists of Edit Memory, Classifier, Counterfactual model and Base model. By assessing the relevance of input to memory, the selection of which model to use for inference is determined. Based on the powerful in-context learning and reasoning capabilities of large language models, methods such as IKE (Zheng et al., 2023) and MeLLo (Zhong et al., 2023) improve the output capabilities of language models by retrieving relevant examples from memory as relevant contextual hints, without relying on additional trained models.

Extra Structure. This type of method introduces additional structures into the current language model for training, allowing these additional structures to learn edit cases-related knowledge. First, FF-layers (Geva et al., 2021) proved that the feed-forword layer (FFN) in the language model is very similar to the neural memory network. The output of the feed-forward layer is a combination of memories, which is then aggregated to produce the final output distribution. This provides support for subsequent work where additional structures can be inserted. CaliNet (Dong et al., 2022) calibrates the factual knowledge in the LLM and updates the output of the FFN by adding new slots. Similarly, T-patcher (Huang et al., 2023) integrates a neuron at the last layer FFN of the model for each case error. When an error is encountered, the corresponding neuron takes effect.

3.2 Parameter Updating Approach

These methods of updating language model parameters is not suitable for black-box models due to them reliance on accessing internal model details.

Meta Learning. The meta-learning method uses a super network to learn to generate new parameters of LLM. Among them, KE (De Cao et al., 2021) trains a super network based on edited memory and uses it to learn and update parameters in the language model. The trained hyper network is able to update the parameters of the language model in real time when a single fact needs to be modified. Similarly, MEND (Mitchell et al., 2022a) use small auxiliary editing networks fine-tunes the language model by employing a low-rank decomposition of the gradient.

Locate-Edit. This type of methods first supplements or repairs specific knowledge by locating relevant facts or errors and updating parameters at corresponding locations. ROME (Meng et al., 2022a) developed a causal intervention that identifies neuronal regions whose decisive role the model predicts relevant facts. Think of model editing as a least squares method with linear equality constraints to solve and further update the parameter matrix of the relevant region via the Lagrange multiplier method. However, the ROME method can only edit one fact association at a time, while MEMIT (Meng et al., 2022b) extends it to multiple layers and stores thousands of fact associations in batches. PMET (Li et al., 2023) finds that when new knowledge is introduced, the weights of multi-head self-attention (MHSA) do not need to be updated, so only the hidden state of FFN is optimized in PMET to accurately update the model weights.

4 Method

We employ parameterized editing and leverage retrieval-augmented in-context prompting approach for its application in massive model editing tasks.

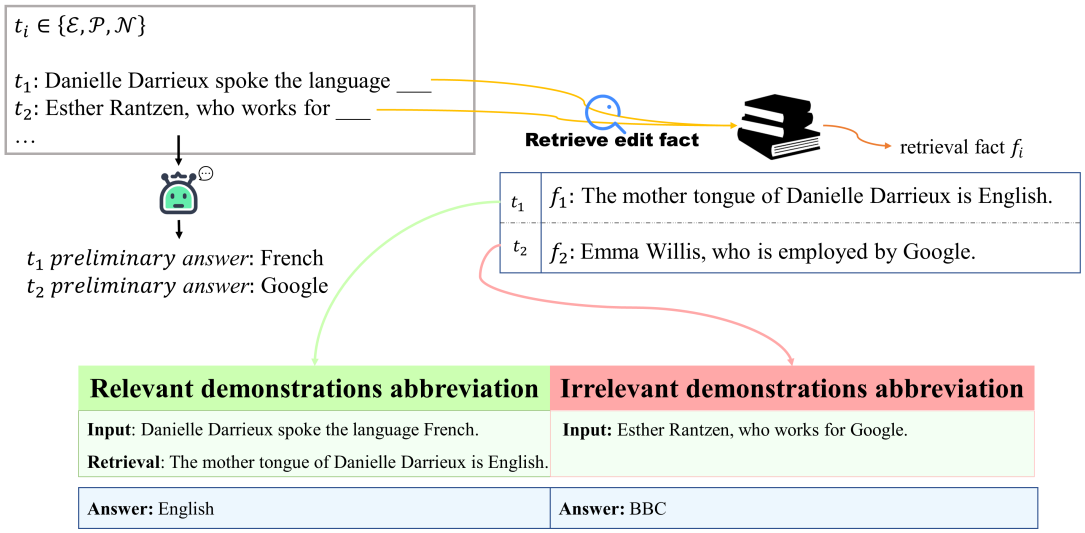
4.1 Knowledge Injection

We first use the MEMIT (Meng et al., 2022b) method, an extensible multi-layer update algorithm, to inject knowledge into the language model. For all edit facts $f_i\left(s_i, r_i, o_i'\right) \in \mathcal{F}$, apply formula 2 to optimize a residual vector δ_i via gradient descent so that update the hidden state h_i^L at layer L. The optimized model is able to maximize the prediction of the desired object o_i'.

$$\begin{aligned}\delta_i \leftarrow & \textit{argmin}_{\delta_i} \frac{1}{P} \sum_{j=1}^{P} \\ & -\log \mathbb{P}_{G\left(h_i^L+=\delta_i\right)}\left[o_i' \mid x_j \oplus p\left(s_i, r_i\right)\right]\end{aligned} \tag{2}$$

Given a set of edit fact prompts, $\{x_j \oplus p\left(s_i, r_i\right)\}$ is concatenating the random prefix x_j to a templated prompt to aid

(a) Massive-knowledge injection: Inject new knowledge into the base model to edit the model to generate the edited model.



(b) Correction and Recollection: For any test request $t_i \in \{\mathcal{E}, \mathcal{P}, \mathcal{N}\}$, use the editing model to make inferences and produce preliminary answers. And then, edit facts related to test facts are retrieved from the edit fact database. Determine whether the test fact and the edit fact are relevant (green is relevant, red is not relevant). Combining different in-context demonstrations for correction or recall.

Figure 2: RAINPOEM architecture description. In massive-editing model, edits are corrected and recalled again through retrieval-augmented and in-context learning.

generalization across contexts. The $G\left(h_i^L += \delta_i\right)$ operation is called "hooking", which uses the updated hidden state to execute the model. Like formula 3, propagate δ_i on predefined editing layers $l \in \mathcal{R}_{\{l_1 \ldots L\}}$, that is, find a matrix Δ^l in each layer, and update the MLP layer to spread the δ_i to the l layer.

$$\begin{aligned} \hat{h}_i^L =& h_i^0 + \sum_{l=1}^{L} \text{attn}\left(h_i^{l-1}\right) \\ &+ \sum_{l=1}^{L} \left[\left(W_{\text{out}}^l + \Delta^l\right) \sigma\left(W_{\text{in}}^l \gamma\left(h_i^{l-1}\right)\right)\right] \end{aligned} \tag{3}$$

where, h_i^0 is the initial embedding of input token, γ and σ is layernorm and activation function. For details on calculating Δ^l, please refer to Appendix A.2.

Building on the advanced methodologies introduced by MEMIT, we compiled thousands of editing facts from the editing fact repository. Subsequently, we tailored modifications to language models of various scales, editing multiple layers for each model. Appendix A.3 describes the hyperparameter design of editing different models. This process involves fine-tuning the internal parameters of the models to enhance their adaptability to new knowledge, ensuring the models can accurately comprehend and apply the latest information.

4.2 Correction and Recollection

In the knowledge injection stage, we packaged all edited facts to obtain a post-edit model. Subsequently, we further edit the in-scope knowledge by retrieval augmented and fixed in-context prompt (Wei et al., 2022; Zhang et al., 2022) to produce more accurate answers. For irrelevant out-scope knowledge, we utilize in-context recall the memory of the pre-edit model to maintain model

Algorithm 1 Correction and Recollection

Requested: Post-edit model f'_θ, test facts set $\{\mathcal{E}, \mathcal{P}, \mathcal{N}\}$, edited facts base $\mathcal{F}$, relevance classifier $g(\cdot)$, in-scope demonstrations D_{in}, out-scope demonstrations D_{out}, dense retriever R

Return: Final answer a_i

$\forall t_i \in \{\mathcal{E}, \mathcal{P}, \mathcal{N}\}$

$a_i^* \leftarrow f'_\theta(a_i^* \mid t_i)$ *//Generate preliminary answers*

$f_i \leftarrow \underset{f_i \in \mathcal{F}}{\text{argmax}_{\text{Similarity}}} R(t_i, f_i)$ *//Retrieve the most relevant editing fact*

if True $\leftarrow g(t_i, f_i)$ **then** *//t_i and f_i are related*

  $a_i \leftarrow f'_\theta(a_i \mid [D_{in} \oplus (t_i, a_i^*) \oplus f_i])$ *//$\oplus$ is the connection operation,* ***Correction***

else

  $a_i \leftarrow f'_\theta(a_i \mid [D_{out} \oplus (t_i, a_i^*)])$ ***//Recollection***

end if

consistency and pertinenece. Algorithm 1 summarizes correction and recall, and we also provide a specific case study in Appendix G.

4.2.1 Discriminating Facts

For facts to be verified, different types of knowledge require specific in-context prompt templates to achieve targeted processing. The key is how to distinguish different types of knowledge (in-scope or out-scope). As shown in fig. 2b, for any knowledge t_i to be detected, use the dense retriever (Izacard et al., 2022) to perform a retrieval to access the edited facts base to obtain f_i that is most relevant to t_i. In fact, for in-scope knowledge, relevant knowledge can be easily retrieved, but for out-scope knowledge, even if the most relevant knowledge is output, theoretically this knowledge has nothing to do with t_i. Here we design four methods to check the correlation between f_i and t_i. Including using in-context, subject, similarity and neural network classification. Please see Appendix D for details.

4.2.2 Demonstrations

To guide the model on how to exploit relevant retrieval facts, we added different types of in-context demonstrations for in-scope and out-scope knowledge respectively. Please refer to Appendix E for the demonstration content. For in-scope knowledge, the purpose of the demonstration is to further verify whether the MEMIT-edited model knowledge is accurate. If it is inaccurate, it will be corrected by retrieving the knowledge and then output. For out-scope knowledge, the purpose of the demonstration is to recall the memory of the pre-edit model to reduce the impact of irrelevant knowledge when editing with MEMIT.

4.2.3 Joint Reasoning

Given the demonstrations as in-context prompt, connect the demonstrations, and test fact input into the post-edit model. By modeling this information, the likelihood of the final output is inferred. The entire process aims to combine model editing, evaluation, retrieval and reasoning under a unique model to achieve rapid integration of new knowledge and effective correction of answers.

In-scope. For example, in fig. 2b, for in-scope knowledge, when the MEMIT-edited model answers the test fact t_1 "*Danielle Darrieux spoke the language _*" the preliminary answer is "*French*". But the model has been edited with new-knowledge, and the correct answer should be "*English*". Based on intensive retrieval, t_1 is used as a query and the edited fact f_1 "*The mother tongue of Danielle Darrieux is English.*" is retrieved. Connect in-scope demonstrations, t_1, the initial answer and retrieval knowledge f_i are input into the model again for correction, and the final output corrected answer through demonstration guidance is "*English*".

Out-scope. Likewise, for out-scope test knowledge t_2 "*Esther Rantzen, who works for _*". When the model edits the irrelevant knowledge "*Emma Willis, who is employed by Google.*", it may mistakenly affect the answer of t_2 as "*Google*". Retrieve the most relevant information f_2 "*Emma Wills, who is employed by Google.*" based on t_2. After t_2 and f_2 are judged to be irrelevant, the retrieved knowledge is discarded and the out-scope demonstrations, t_2 and initial answer connections are fed into the model. Through demonstration guidance, the pre-edit model's memory can be recalled to output the correct answer "*BBC*".

5 Experimental & Analysis

5.1 Setup

DataSets: We evaluate our method on two knowledge editing datasets, Counterfact (Meng et al., 2022a) and MQuAKE (Zhong et al., 2023). The MQuAKE dataset is aimed at the downstream tasks of QA(questiom and answer). Therefore, the counterfact data set and the MQuAKE series data set also have different forms. Please see the Appendix B for details on the data format.

Large Language Models: In the course of our research, we employed language models with varying parameter magnitudes to serve as the baseline model for RAINPOEM. The baseline models encompass GPT2-XL(1.5B), GPT-J(6B) and LlaMa2(7B), each defined as white-box language models where in the internal parameters are amenable to modification through model editing approach, thereby enhancing adaptability for unknown data.

Knowledge Editing Method: We use MEMIT (an extension of the ROME method) to modify the parameters of the language model. Enhanced capabilities by allowing multiple edits across multiple layers simultaneously, with the ability to update thousands of memories at once.

Evaluation Metrics: Following previous works (Meng et al., 2022b; Zhong et al., 2023), different evaluation metrics are reported on both datasets. On the counterfact dataset, the **Success** and **Accuracy** are reported in three aspects: Reliability, Generalization and Specificity. Where Success is the ratio of the probability of outputting a new object greater than the probability of the original object (opposite in Specificity). Accuracy evaluates whether the output object is a new object (original object in Specificity). On the MQuAKE dataset, follwing MeLLo, We report averaged **One-hop Acc** and averaged **Multi-hop Acc**, which measure accuracy on question answering tasks.

Please see the Appendix C section which provides a detailed description of the relevant evaluation formulas.

5.2 Evaluation on Counterfact Dataset

We assessed the effectiveness of our method on the authoritative counterfact dataset and compared its performance with state-of-the-art methods.

5.2.1 Evaluating RAINPOEM

We adopted two sets of GPT models with different parameters and evaluated their performance on the Counterfact dataset. Table 1 shows our evaluation results under the setting of editing 10,000 facts at one time. Among them, the pre-editing model serves as the base because it knows nothing about the editing facts and scores low in reliability and generalization, especially in terms of accuracy.

When using current advanced editing methods for massive knowledge injection, we observe that the model's editing success rate is significantly improved in terms of reliability and generalization. In this regard, ROME cannot accurately infer answers due to writing a large amount of knowledge in a single layer, thus reducing the accuracy of editing. In contrast, the MEMIT method can batch write editing facts to multiple layers, resulting in significant improvements in success and accuracy in terms of Reliability and Generalization. However, in terms of Specificity, since the editing knowledge affects the cognitive architecture of the entire model, it has a negative impact on the previously acquired knowledge.

Since the in-context classification method used cannot accurately distinguish in-scope and out-scope knowledge, the corresponding demonstrations cannot be used correctly, which in turn affects the ideality of the RAINPOEM results. Please see table 9 for method classification accuracy. For RAINPOEM using the other three classification methods, We can even achieve results similar to the MEMIT-edited GPT-J model using the GPT2XL model, which has a relatively smaller parameter. In particular, thanks to relevant demonstrations, the model is able to identify errors in in-scope knowledge and proactively correct them. Thus, surprising results were obtained in terms of reliability and generalization. Most importantly, for the affected out-scope knowledge, irrelevant demonstration can alleviate the negative impact of the edited model in terms of specificity, rising from 73.5 for the GPT-J model edited by MEMIT to 77.9.

5.2.2 Ablation study

In order to verify the independence and robustness of two different types of demonstrations, we conducted ablation experiments based on network classification using only one of the templates. The experimental results are shown in table 2. This consistent demonstrations selection enables us to focus on the effects of the demonstrations and gain

Model	Editor	Score ↑		Reliability ↑		Generalization ↑		Specificity ↑	
		SS	AS	ES	EA	PS	PA	NS	NA
GPT2-XL	Base	30.5	0.2	22.9	0.3	23.9	0.1	**77.3**	10.7
	ROME	50.4	0.4	51.9	0.4	49.5	0.4	49.9	0.4
	MEMIT	71.5	18.0	79.5	46.0	67.0	23.4	69.2	9.8
	RAINPOEM$_{w/in-co.}$	75.1	27.1	79.3	52.9	75.4	48.6	71.1	14.0
	RAINPOEM$_{w/sub.}$	**87.7**	34.1	99.1	89.8	**96.4**	**85.7**	72.7	15.4
	RAINPOEM$_{w/sim.}$	86.9	32.8	99.1	89.8	94.2	82.4	72.4	14.7
	RAINPOEM$_{w/net.}$	86.7	**35.1**	**99.2**	**90.1**	91.9	78.3	73.3	**16.2**
GPT-J	Base	21.3	0.5	15.2	0.4	15.8	0.4	**83.5**	14.7
	ROME	50.8	0.2	51.3	0.2	50.6	0.1	50.7	0.1
	MEND	25.2	4.5	17.6	3.15	20.1	3.18	80.8	**23.7**
	MEMIT	87.6	26.8	99.1	**96.1**	94.9	69.9	73.5	11.5
	RAINPOEM$_{w/in-co.}$	85.1	19.3	99.8	94.8	97.1	88.1	66.9	7.5
	RAINPOEM$_{w/sub.}$	90.3	42.6	99.8	95.5	**97.6**	**92.2**	77.3	20.4
	RAINPOEM$_{w/sim.}$	90.1	41.8	99.8	95.5	96.6	89.5	77.3	19.9
	RAINPOEM$_{w/net.}$	**90.3**	**43.5**	**99.8**	95.5	96.4	87.0	77.9	21.3

Table 1: The results of editing 10,000 cases on the counterfact dataset.

a deeper understanding of the model's behavior in in-context learning.

By distinguishing knowledge types and using only one demonstrations, the performance of knowledge in the corresponding scope can be improved on the original basis (MEMIT). If no type distinction is made, but a unified demonstrations is used instead, the performance will be greatly affected if the wrong demonstrations is used. It shows the importance of the designed demonstrations in ensuring correction and recall in model editing. The role of demonstrations is to provide appropriate frameworks and guidance for different scope of knowledge and improve the effectiveness of the model in editing knowledge.

5.3 Evaluation on MQUAKE Dataset

In addition to the counterfactual regression generation task, we also tested the performance of multi-hop and single-hop question answering on the MQuAKE dataset.

5.3.1 Multi-hop QA

Table 3 presents the research results based on the GPT-J model on the MQuAKE-CF dataset. The number of edited facts considers k facts $(k \in \{100, 1000, 3000\})$, as set by Zhong et al. (2023). Our research finds that on the multi-hop counterfactual question answering dataset, RAINPOEM not only outperforms only editing algorithms (such as MEND and MEMIT) using the same language model, but also significantly outperforms MeLLo, which only relies on retrieval algorithm. This shows that compared to other methods, RAINPOEM can serve as a powerful knowledge editing method by combining editing and reasoning capabilities.

In addition, as the most important aspect of model editing purposes, we hope that the model can understand the current knowledge. We also extend our evaluation on the temporal-based real-world fact dataset MQuAKE-T at multiple edit sizes$(k \in \{100, 500, 1868\})$, see table 4. In addition to the GPT-J model, the more powerful LlaMa2-7B model was also compiled. Judging from the performance of the GPT-J model edited using the MEMIT method, simply writing partial editing facts into the model does not help the model in answering multi-hop questions. We should learn from the ideas of the MeLLo method, stimulate the model to think about sub-questions, use the injected knowledge to improve the accuracy of the answers to these sub-questions, and then further correct these answers. Compared to MeLLo, this method is very effective. When editing all facts (1868 facts), the GPT-J model using the RAINPOEM method is able to surpass the performance of using the MeLLo method, improving the multi-hop Acc by 17.5%.

5.3.2 One-hop QA

To further explore the impact of editing scale on model performance, we edited MQuAKE-T facts at multiple scales into the LlaMa2-7B model, and

	Score ↑		Reliability ↑		Generalization ↑		Specificity ↑	
demons type	SS	AS	ES	EA	PS	PA	NS	NA
in-scope$_{w/only}$	88.8	26.8	99.8	95.5	97.6	89.3	74.2	11.1
out-scope$_{w/only}$	89.1	42.0	99.1	96.1	93.6	67.6	77.5	21.7
in-scope$_{w/all}$	83.8	13.9	99.8	95.5	97.7	92.7	64.2	5.2
out-scope$_{w/all}$	87.4	37.6	95.9	61.8	89.9	54.1	78.2	22.1

Table 2: *'only'* means to use the demonstration only when it involves knowledge of a specific domain, and not to use the demonstration beyond that scope. And *'all'* means to use the same demonstration when dealing with various knowledge types.

#Edited-facts		100	1000	3000
Base Model	**Method**			
GPT-J (6B)	MEND	9.1	4.3	3.5
	MEMIT	9.8	8.1	1.8
	MeLLo	12.5	10.4	9.8
	RAINPOEM	21.0	13.3	10.7

Table 3: Multi-hop Acc evaluation results of the GPT-J model on the multi-hop counterfactual dataset MQuAKE-CF.

#Edited-facts		100	500	1868
Base Model	**Method**			
GPT-J	MEND	17.4	12.7	4.6
GPT-J	MEMIT	1.0	0.2	0.0
GPT-J	MeLLo	45.7	33.8	30.7
Vicuna-7B	MeLLo	56.3	52.6	51.3
GPT-J	RAINPOEM	59.0	51.4	48.2
LlaMa2-7B	RAINPOEM	61.0	59.0	52.1

Table 4: Multi-hop Acc result multihop-QA on MQuAKE-T with various language models.

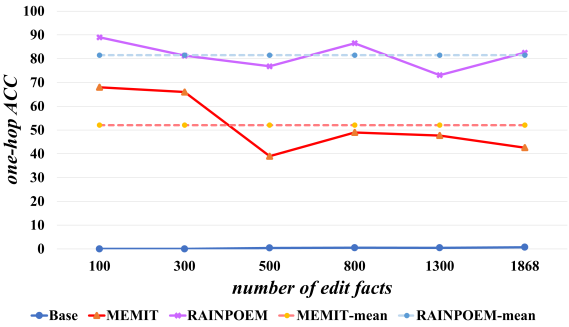


Figure 3: The accuracy of editing one-hop QA facts at different scales under the MQuAKE-T dataset. Edit size $n \in \{100, 300, 500, 800, 1300, 1868\}$.

evaluated the accuracy of the editing method under the one-hop question. We edited the facts that the size $n \in \{100, 300, 500, 800, 1300, 1868\}$, and evaluated the performance as shown in figure 3. In the model before editing (Base), it can be observed that the model lacks any understanding of the new knowledge. In the MEMIT editing method, as the editing scale expands, it can be seen that the overall model performance shows a downward trend, and the MEMIT method fluctuates greatly on the average one-hop Acc and seems to be more sensitive to the editing scale. On the contrary, although the RAINPOEM method also has a slight decline as the editing scale increases, it is more robust to the scale of editing data with the help of the error correction mechanism. In order to further explore the advantages of our RAINPOEM method compared with the MEMIT method, we made a more detailed evaluation on different types of questions. Please refer to Appendix F for evaluation experiments and details.

6 Discuss & Conclusion

In this paper, we delve into the challenges faced by large language models (LLMs), specifically the dilemma when dealing with errors or information that is not relevant to the current timeline. Traditionally, retraining a completely new LLM is an expensive task, and adjusting model weights when information changes can lead to unpredictable ripple effects.

We introduce a new method RAINPOEM based on current model editing. This method provides more robust answers by more comprehensively evaluating the results of the editing model through retrieval and in-context learning to correct new facts and recall old knowledge. Experiments confirmed the potential of combining parameter editing and retrieval augmented methods in model editing tasks. Overall, the RAINPOEM method represents an important improvement over existing model editing methods, providing useful insights into the future ability to understand and optimize LLMs.

Limitations

The limitations inherent in our study encompass the following aspects.

- One limitation of the proposed RAINPOEM approach is its reliance on appropriate editing technique. The effectiveness of the model editing process depends on the adoption of appropriate editing strategies. In situations where massive edit information needs to be incorporated, some editing methods may inadvertently disrupt model parameters, (such as ROME), resulting in a loss of reasoning capabilities. This potential limitation will cause the language model to lose the ability to modify answers during the error correction process.

- Another inherent limitation occurs during the classification phase, where the model relies on corresponding in-context demostrations to correct errors and recall memory. This accuracy of classification introduces a degree of uncertainty. Secondly, the dataset is not clear, and the out-scope of some facts is the in-scope of other facts, which will affect the performance of classification in massive editing. Therefore, the classification accuracy of the proposed method is closely related to the efficiency and accuracy of model editing, which poses challenges when facing unclear datasets.

In conclusion, this study provides valuable insights, but the limitations mentioned need to be acknowledged. Specifically, it should be recognized that appropriate editing techniques are needed to modify model parameters, where correction and recall rely on effective knowledge classification mechanisms. Addressing these limitations in future research efforts will further improve the effectiveness and applicability of the RAINPOEM paradigm in massive knowledge editing tasks.

References

Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, et al. 2022. Gpt-neox-20b: An open-source autoregressive language model. *arXiv preprint arXiv:2204.06745*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

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

Qingxiu Dong, Damai Dai, Yifan Song, Jingjing Xu, Zhifang Sui, and Lei Li. 2022. Calibrating factual knowledge in pretrained language models. *Findings of Empirical Methods in Natural Language Processing (EMNLP)*.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. 2023. Transformer-patcher: One mistake worth one neuron. In *The Eleventh International Conference on Learning Representations*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. 2023. Pmet: Precise model editing in a transformer. *arXiv preprint arXiv:2308.08742*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. Association for Computational Linguistics.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Aman Madaan, Niket Tandon, Peter Clark, and Yiming Yang. 2022. Memory-assisted prompt editing to improve GPT-3 after deployment. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2833–2861, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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

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, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *EMNLP*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. 2022a. Fast model editing at scale. In *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.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

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*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

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*.

Ben Wang and Aran Komatsuzaki. 2021. Gpt-j-6b: A 6 billion parameter autoregressive language model.

Peng Wang, Ningyu Zhang, Xin Xie, Yunzhi Yao, Bozhong Tian, Mengru Wang, Zekun Xi, Siyuan Cheng, Kangwei Liu, Guozhou Zheng, et al. 2023a. Easyedit: An easy-to-use knowledge editing framework for large language models. *arXiv preprint arXiv:2308.07269*.

Weixuan Wang, Barry Haddow, and Alexandra Birch. 2023b. Retrieval-augmented multilingual knowledge editing. *arXiv preprint arXiv:2312.13040*.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*.

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*.

Zexuan Zhong, Zhengxuan Wu, Christopher D Manning, Christopher Potts, and Danqi Chen. 2023. Mquake: Assessing knowledge editing in language models via multi-hop questions. *arXiv preprint arXiv:2305.14795*.

A Implementation Detail

A.1 Experiment Environment

All editing and evaluation experiments were run on a workstation with NVIDIA RTX A6000 GPU. The pre-trained weights of the loaded language model come from HuggingFace transformers (Wolf et al., 2019) in version 4.30.1, and Pytorch (Paszke et al., 2019) version is 2.01.

A.2 MEMIT

In the MLP layer, facts are usually stored as key-memory pairs. The MLP output layer weight before editing is defined as:

$$W_{out} \triangleq \underset{\hat{W}}{\operatorname{argmin}} \sum_{i=1}^{n} \left\| \hat{W} k_i - m_i \right\|^2 \tag{4}$$

where k_i is defined encoding subjects, m_i is the corresponding memory encoding. After editing, as shown in formula 3, Δ needs to be calculated to update W_{out}, resulting in a new weight matrix $\hat{W}_{out}$ to store the new association.

$$\hat{W}_{out} = W_{out} + \Delta \tag{5}$$

Because multiple new association constraints are added, formular 4 is further defined as:

$$\hat{W}_{out} \triangleq \underset{\hat{W}}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \left\| \hat{W} k_i - m_i \right\|^2 + \sum_{i=n+1}^{n+u} \left\| \hat{W} k_i - m_i \right\|^2 \right) \tag{6}$$

The above formula can be optimized by solving the normal equation. Meng et al. (2022b) described it in block form as follows.

$$\hat{W}_{out} \begin{bmatrix} K_0 & K_1 \end{bmatrix} \begin{bmatrix} K_0 & K_1 \end{bmatrix}^T = \begin{bmatrix} M_0 & M_1 \end{bmatrix} \begin{bmatrix} K_0 & K_1 \end{bmatrix}^T \tag{7}$$

According to formula 5, formula 7 is further simplified to:

$$\Delta \left(K_0 K_0^T + K_1 K_1^T \right) = M_1 K_1^T - W_{out} K_1 K_1^T \tag{8}$$

Define $C \triangleq K_0 K_0^T$ and $R \triangleq M_1 - W_{out} K1$, where C is a constant proportional to the non-central covariance of K_0, which can be estimated from the Wikipedia text, and R is the residual of the new association and the old weight, which is considered to evenly distribute δ_i to the remaining layers. Formula 8 is simplified to:

$$\Delta = R K_1^T \left(C + K_1 K_1^T \right)^{-1} \tag{9}$$

To calculate Δ, the process on all edit facts $\mathcal{F}$ and predefined edit layers $\mathcal{R}$ is as follows:

$$\begin{aligned} &\text{for} \quad l \in \mathcal{R}_{\{l_1 \ldots L\}} \\ &\quad \text{for} \quad f_i \left(s_i, r_i, o_i' \right) \in \mathcal{F}_{\{1 \ldots n\}} \\ &\qquad k_i^l = \frac{1}{P} \sum_{j=1}^{P} k \left(x_j + s_i \right) \\ &\qquad r_i^l = \frac{\delta_i}{L - l + 1} \\ &\quad K^l = \left[k_1^l \ldots k_n^l \right] \\ &\quad R^l = \left[r_1^l \ldots r_n^l \right] \end{aligned} \tag{10}$$

According to the above formula 10, K^l and R^l of all new associations are calculated on layer l, and Δ^l is calculated by combining formula 9. In the MLP layer of the pre-edit layer, the weights are updated according to the following formula, and the new association is stored in the language model to complete the update.

$$\hat{W}_{out}^l = W_{out}^l + \Delta^l \tag{11}$$

A.3 Editing Hyperparameters

All hyperparameters about using MEMIT to edit GPT2-XL, GPT-J and LlaMa2-7B can be found in the EasyEdit code (Wang et al., 2023a). The important parameters are shown in table 5. t and

Model	η	t	$\mathcal{R}$
GPT-2XL	0.5	20	[13, 14, 15, 16, 17]
GPT-J	0.5	25	[3, 4, 5, 6, 7, 8]
LlaMa2-7B	0.5	25	[4, 5, 6, 7, 8]

Table 5: Optimization parameters and editing layer of language model.

η are the number of steps and learning rate of δ_i optimization respectively, and $\mathcal{R}$ is the model layer to be edited.

B DataSet

MQuAKE contains two types of datasets, one is the counterfactual dataset, MQuAKE-CF, like **Counterfact**. The other is MQuAKE-T, which updates real-world data based on the timeline(**2023.05**) when the dataset was presented. Counterfactual datasets are presented with data outside the scope by replacing entities related to the subject, thereby assessing the effectiveness of model editors in making appropriate adjustments in the face of data changes. The time-base fact dataset is obtained by updating the real facts based on the latest information on the current timeline.

B.1 Counterfact

The form of data in Counterfact is mainly represented by the cloze-style. For the edit data $f_i \left(s_i, r_i, o_i' \right) \in \mathcal{F}$, its definition form is as in table 6. The editing process writes $(s_i, r_i) \rightarrow o_i'$ to the model to replace the previously pointed o_i.

In addition, there is a set $\mathcal{E}$ for evaluating Reliability, whose content is usually the same as the original $\mathcal{F}$. The set $\mathcal{P}$ for evaluating Generalization is rewritten from $\mathcal{F}$ in order to test the generalization performance of the post-edit model. The set $\mathcal{N}$ for testing Specificity, which represents the same semantics as $\mathcal{F}$ but unrelated facts, in order to test the deterioration of the model after editing.

B.2 MQuAKE

Unlike the Counterfact dataset, MQuAKE is implemented in the form of question and answer. As shown in table 7. For editing facts, there is a one-hop question set $Onehop\mathcal{Q}$ and a multi-hop ques-

$\mathcal{F}$	s_i	Edwin of Northumbria
	r_i	The official religion of {} is _
	o_i	Christianity
	o_i'	Islam
$\mathcal{E}$	e_i	The official religion of Edwin of Northumbria is _
$\mathcal{P}$	p_i	p_{i1}. Edwin of Northumbria follows the religion of _
		p_{i2}. Edwin of Northumbria is affiliated with the religion _
$\mathcal{N}$	n_i	n_{i1}. The official religion of Charles Aznavour is _
		n_{i2}. Nicolas Sarkozy is affiliated with the religion _
		n_{i3}. Andrew Johnson is affiliated with the religion _
		n_{i4}. The official religion of Paul is _
		n_{i5}. Ringo Starr is follower of _
		n_{i6}. The official religion of Nicolas Sarkozy is _
		n_{i7}. The official religion of Andrew Johnson is _
		n_{i8}. Orson Welles is affiliated with the religion _
		n_{i9}. Lady Gaga is follower of _
		n_{i10}. Quentin Tarantino is affiliated with the religion _

Table 6: Formulation of the Counterfact Dataset.

$\mathcal{F}$	s_i	New York City
	r_i	The name of the current head of the {} government is _
	o_i	Bill de Blasio
	o_i'	Eric Adams
$Onehop\mathcal{Q}$	oq_i	What is the name of the current head of the New York City government?
$Multihop\mathcal{Q}$	mq_i	mq_{i1}. Who is the head of government in the location of the headquarters of TV Land?
		mq_{i2}. Who is the chief executive of the region where TV Land's headquarters are based?
		mq_{i3}. What is the name of the leader of the territory where TV Land's headquarters are located?

Table 7: Formulation of the MQuAKE Dataset.

tion set $Multihop\mathcal{Q}$. After editing, it is expected that the answer output by the post-edit model can correspond to the new editing fact o_i'.

C Evaluation Metrics

C.1 For Counterfact

In order to be consistent with previous works (Mitchell et al., 2022a; Meng et al., 2022a) on the comparison of counterfact dataset results, we report the evaluation formulas in Reliability, Generalization and Specificity respectively.

C.1.1 Reliability

Efficacy Success(ES), that is formula 12, is the proportion of cases where the output o_i' probability of the fact e_i from the set $\mathcal{E}$ is greater than o_i.

$$\mathbb{E}_{(s_i,r_i)\sim\mathcal{E}}\left\{f_\theta'\left(o_i' \mid (s_i,r_i)\right) > f_\theta'\left(o_i \mid (s_i,r_i)\right)\right\} \quad (12)$$

Efficacy Accuracy(EA), the formula 13 is a proportional situation where the output is o_i'.

$$\mathbb{E}_{(s_i,r_i)\sim\mathcal{E}}\left\{\underset{o}{\operatorname{argmax}} f_\theta'\left(o \mid (s_i,r_i)\right) = o_i'\right\} \quad (13)$$

C.1.2 Generalization

Similar to Reliability, the fact p_i from the rephrasing set $\mathcal{P}$ is used to test **Paraphrase Success(PS)**, formula 14, and **Paraphrase Accuracy(PA)**, formula 15.

$$\mathbb{E}_{(s_i,r_i)\sim\mathcal{P}}\left\{f_\theta'\left(o_i' \mid (s_i,r_i)\right) > f_\theta'\left(o_i \mid (s_i,r_i)\right)\right\} \quad (14)$$

$$\mathbb{E}_{(s_i,r_i)\sim\mathcal{P}}\left\{\underset{o}{\operatorname{argmax}} f_\theta'\left(o \mid (s_i,r_i)\right) = o_i'\right\} \quad (15)$$

C.1.3 Specificity

The **Neighborhood Success(NS)**, and **Neighborhood Accuracy(NA)**, metrics of the fact n_i in

the neighborhood set $\mathcal{N}$ are evaluated in terms of Specificity, formula 16 and formula 17. We expect the model not to change the answers to these irrelevant facts after editing, they should output the original answer o_i. n_i in the same group have the same answer.

$$\mathbb{E}_{(s_i,r_i)\sim\mathcal{N}}\left\{f'_\theta\left(o'_i \mid (s_i,r_i)\right) < f'_\theta\left(o_i \mid (s_i,r_i)\right)\right\} \tag{16}$$

$$\mathbb{E}_{(s_i,r_i)\sim\mathcal{N}}\left\{\underset{o}{\operatorname{argmax}} f'_\theta\left(o \mid (s_i,r_i)\right) = o_i\right\} \tag{17}$$

In addition, we also introduce **Success Score(SS)** and **Accuracy Score(AS)**, which represent the harmonic mean of the evaluation results of (ES, PS, NS) and (EA, PA, NA) respectively.

C.2 For MQuAKE

One-hop accuracy and Multi-hop accuracy measure the accuracy of one-hop questions and multi-hop questions respectively after editing. One-hop accuracy and multi-hop accuracy are respectively defined as formula 18 and 19.

$$\mathbb{E}_{(oq_i)\sim\mathcal{OQ}}\left[\bigvee_{q\in oq_i}\left[f'_\theta(q) = o'_i\right]\right] \tag{18}$$

$$\mathbb{E}_{(mq_i)\sim\mathcal{MQ}}\left[\bigvee_{q\in mq_i}\left[f'_\theta(q) = o'_i\right]\right] \tag{19}$$

D Methods of Classify Scope

In order to effectively distinguish in-scope text and out-scope text in the couterfact dataset, we designed four classification methods. These methods include natural language processing techniques and machine learning and deep learning algorithms designed to improve the accuracy and efficiency of text classification. Through these classification methods, we can more accurately divide test facts into content that is in-scope facts and out-scope facts. We tested the accuracy of these four methods for classifying knowledge scope and provide the classification accuracy in table 9. In addition, we also migrated the four classification methods designed to the SERAC (Mitchell et al., 2022b) model and extended the SERAC method. Please refer to table 10 for performance.

D.1 In-context classification method

The first method is classification based on in-context learning, which uses context learning to classify text by analyzing whether the two input sentences 'input' and 'retrieval' are related. It is hoped that large language models can be used to better understand the inherent correlation of texts. The in-context template is shown in table 8. We make sentences according to the template and send the template to the language model. If the model outputs 'Yes', the test fact is considered to be an in-scope fact. Otherwise, it outputs 'No' and is considered to be out-scope fact.

D.2 Subject classification method

The second method is based on subject classification, as shown in formula 20. By building a subject library S that contains the subjects of all editing facts. For each test fact t_i to be detected, as long as the subject exists in the subject library, the sentence is classified as in-scope; otherwise, it is classified as out-scope.

$$S = \{sub_{f_1}, sub_{f_2}, \ldots, sub_{f_n}\}, f_i \in \mathcal{F}$$

$$t_i = \begin{cases} \text{in-scope} & \text{if } sub_{t_i} \in S \\ \text{out-scope} & \text{if } sub_{t_i} \notin S \end{cases} \tag{20}$$

D.3 Similarity classification method

The third method is to perform similarity calculation between test fact and retrieval fact by using the pre-trained model "all-MiniLM-L6-v2" (Reimers and Gurevych, 2019), as shown in formula 21. We use this pre-trained model to encode two sentences separately to obtain their embedding features E_{t_i} and E_{r_i}, and then use cosine similarity to measure the similarity between them. Finally, the similarity threshold β is set to determine whether the two sentences are related. In this method, we set β to 0.7.

$$t_i = \begin{cases} \text{in-scope} & \text{if } C\_S\left(E_{t_i}, E_{r_i}\right) > \beta \\ \text{out-scope} & \text{if } C\_S\left(E_{t_i}, E_{r_i}\right) \leq \beta \end{cases} \tag{21}$$

where $C\_S(,)$ is a function that calculates cosine similarity.

D.4 Siamese network classifier

The fourth method is designed to train a siamese network, as shown in the figure. Two parallel branches share the weights of the pre-train model "all-MiniLM-L6-v2" (Reimers and

Are "South East Cape, which is located in" and "South East Cape, which is located in" related? Yes
Are "Johann Heinrich Burchard, who has the position of" and "Joachim Meisner has the position of" related? No
Are "The original language of Khamosh was" and "The language of Khamosh was" related? Yes
Are "The language of Maqbool was" and "The original language of Dhool is" related? No
Are "M'Sila Province can be found in" and: "M'Sila Province is located in" related? Yes
Are "Wolfgang Ketterle's occupation is" and "Martin Klebba's occupation is" related? No
Are "32nd Indiana Monument is in" and "32nd Indiana Monument is within" related? Yes
Are "Angela Smith, Baroness Smith of Basildon is native to" and "Nicky Barnes is native to" related? No

Table 8: In-context demonstrations for classifying relevance.

Gurevych, 2019) and are used to extract sentence features. Subsequently, the fully connected feature fusion layer is used to calculate the difference between the two branch features, and the result is finally output through the classification layer and sigmoid function, as shown in the formula 22. The trained Siamese network can better handle the relationships before the input fact sentences and learn more generalizable representations from the training process.

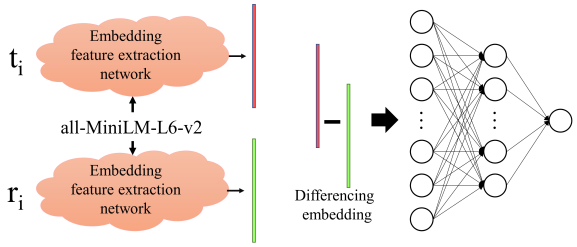


Figure 4: Siamese network $g(,)$ structure

$$t_i = \begin{cases} \text{in-scope} & \text{if } \sigma(g(t_i, r_i)) > 0.5 \\ \text{out-scope} & \text{if } \sigma(g(t_i, r_i)) \leq 0.5 \end{cases} \quad (22)$$

where $g(,)$ is the siamese network, $\sigma(\cdot)$ is the sigmoid function. Because the final output is through the sigmoid function, 0.5 can be used as the default threshold. This method avoids the subjectivity and uncertainty caused by manual threshold adjustment in method three D.3.

E Demonstrations Detail

In in-context learning, using several examples to form a demonstration context, and then splicing this demonstration context with the query question to form a prompt input into the model can significantly improve the model's ability to solve complex tasks. In addition, Min et al. (2022)'s work also proved that what in-context learning learns is not the simple association between demonstration and input, but the ability to activate the pre-trained model by displaying the data form. Based on this important conclusion, we constructed demonstrations for in-scope knowledge and out-scope knowledge respectively to activate the model correction and recall capabilities. Table 11 mainly shows the demonstrations built for the counterfact dataset.

Method	in-scope	out-scope	all
in-context	94.63	15.69	33.91
subject	99.14	84.82	88.12
similarity	93.92	83.59	85.97
scope classifier	88.09	89.99	89.55

Table 9: Accuracy of four methods for in-scope and out-scope classification

Classification Method	SS	AS	ES	EA	PS	PA	NS	NA
in-context	86.2	26.6	98.6	95.2	89.4	65.5	74.2	11.2
subject	90.3	27.6	99.1	96.1	94.5	69.8	79.7	11.9
similarity	88.2	25.9	99.1	96.1	89.0	65.9	78.9	11.1
scope classifier	86.5	28.4	99.1	96.1	82.0	59.7	80.7	12.7

Table 10: like-SERAC mechanism results

F Additional Category Experiments of MQuAKE Dataset

In order to deeply study the performance of RAINPOEM on different types of data, we conducted statistics on the fact types containing more than 500 edit facts in MQuAKE-CF. The specific data is shown in statistical figure 5. We evaluated the performance of the algorithm on each type of question based on one-hop questions, aiming to understand which question types the language model has more learning advantages and the ability to correct errors on this type of problems. In order to maintain consistency in editing standards, we randomly selected 500 facts from each question category for editing and evaluation.

Figure 6 reports the performance on these categories. Obviously, the RAINPOEM method achieves better one-hop Acc scores in all these types. In addition, the experiment also proved that not all types of facts can be smoothly written into the model, such as P112, P36 and P641. Correspondingly, the RAINPOEM method also faces some challenges when facing these types of problems.

G Case Study

We illustrate case studies of in-scope and out-of-scope knowledge using two test fact examples from the counterfactual data set that were successfully corrected and recalled. As shown in table 12a and 12b respectively.

After obtaining the post-edit model, input the fact to be evaluated (marked in pink) and obtain a preliminary answer (marked in yellow). And retrieve relevant fact (marked in blue) from the edited facts base based on the fact to be evaluated. Connect demonstration, test fact to be evaluated, preliminary answer and retrieval fact (required only for in-scope knowledge) as prompt (marked in green). And then the prompt is sent to model inference to generate the final answer (marked in orange).

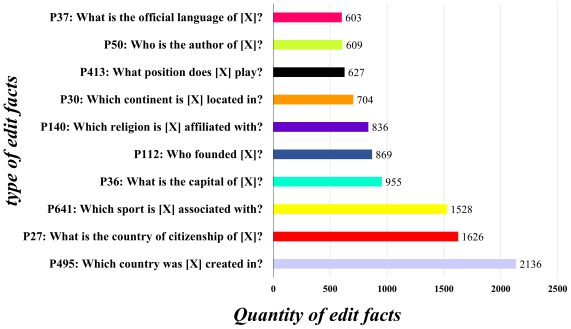


Figure 5: Among all edit fact types in the dataset MQuAKE-CF, more than 500 facts are counted.

Input: The mother tongue of Danielle Darrieux is English.
Retrieval: The mother tongue of Danielle Darrieux is English.
Answer: English

Input: Edwin of Northumbria follows the religion of Islam.
Retrieval: The official religion of Edwin of Northumbria is Islam.
Answer: Islam

Input: The occupation of Juan Rulfo is the.
Retrieval: The occupation of Juan Rulfo is politician.
Answer: politician

Input: Beryl Cook died at the.
Retrieval: Beryl Cook passed away in Chicago.
Answer: Chicago

Input: Theo Crosby passed away in Chicago.
Retrieval: Theo Crosby passed away in Chicago.
Answer: Chicago

Input: Zaheer Khan follows the religion of Catholicism.
Retrieval: Zaheer Khan is follower of Catholicism.
Answer: Catholicism

(a) In-scope demonstrations

Input: The official religion of Charles Aznavour is Christianity.
Answer: Christianity

Input: Elcho Island Airport, located in Indonesia.
Answer: Australia

Input: The language of Barbarella is English.
Answer: English

Input: Choi Sung-yong is a native speaker of French.
Answer: Korean

Input: Roman Catholic Archdiocese of Sydney's headquarters are in the.
Answer: Sydney

Input: Lars Onsager, who holds a citizenship from the.
Answer: Norway

(b) Out-scope demonstrations

Table 11: Demonstrations details of in-scope facts and out-scope facts in counterfact datasets.

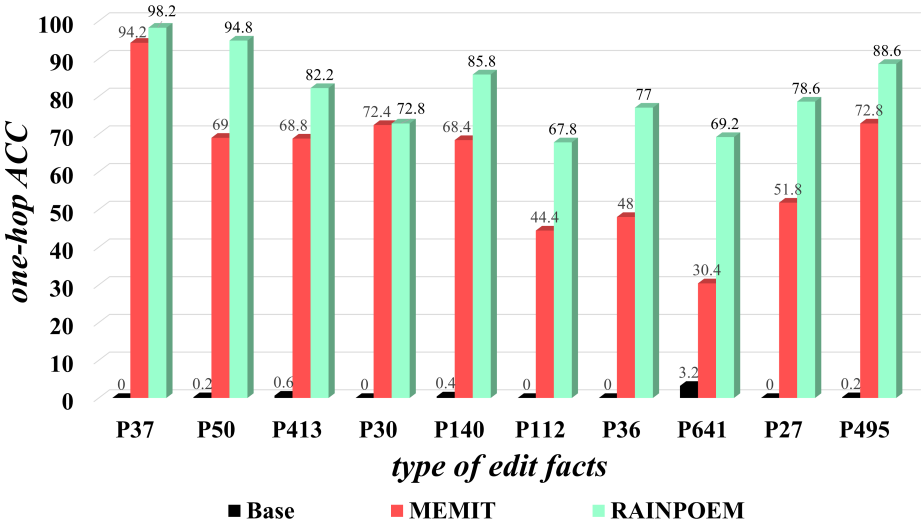


Figure 6: For the various types of edit facts counted in figure 5, based on one-hop QA, the one-hop ACC metric under Bace, MEMIT, and RAINPOEM were evaluated.

David Rivett works as _
a
David Rivett's occupation is composer.

[in-scope in-context demonstrations abbreviation]
Input: David Rivett works as a.
Retrieval: David Rivett's occupation is composer.

Answer: composer

(a) In-scope case study. The retrieval part is related to the input part, and is connected to the in-scope in-context demonstrations for correction.

In Cantavieja, they understand _
English
In Asturias, they understand German.

[out-scope in-context demonstrations abbreviation]
Input: In Cantavieja, they understand English.

Answer: Spanish

(b) Out-scope case study. The retrieval part is unrelated to the input part, and is connected to the out-scope in-context demonstrations for recollection.

Table 12: Illustrations of counterfact dataset case studies. The pink part is the test fact to be evaluated serve as input to the post-edit model. The yellow part is preliminary answer generated by the post-edit model based on input, the blue part is the fact retrieved by the retriever based on input, the green part is a prompt to connect in-context demonstrations, and the orange part is the final answer through in-context learning correction and recall.