POMEM: In-Context Knowledge Post Editing on Massive-Editing Memory in Language Language Models

Anonymous ACL submission

Abstract

Parameter updating (PU), while being widely used in *knowledge editing*, has still shown limited performances in terms of generalization and locality metrics, likely due to the catastrophic forgetting, the riffle effects, or the unseen contexts. This paper proposes a novel *in-context post-editing*, which is subsequently applied to the PU-based prediction results, namely **POMEM** – In-context knowledge **po**st **e**diting on **m**assive-**e**diting **m**emory – which consists of two different types of in-context post-editing prompting method, divided into the "in-scope" and "out-of-scope" post-editing methods, shortly referred to as Copier and Recaller, respectively; 1) **Copier** is specially designed for in-scope cases, mainly aiming to further enhance the generalization editing ability; 2) **Recaller** is designed for out-of-scope cases, which involves a novel "recalling" prompt which aims to recover the prediction result of "original pre-edited" model under using the PU-based "edited" model. Experiment results on Counterfact dataset show that POMEM leads to the state-of-the-art performances. Our codes are publicly available at `https://github.com/XXX/XXX`.

1 Introduction

Given the ever-changed world knowledge and the frequent demands on knowledge maintenance, there has been a growing interest in "knowledge editing" task on large language models (LLMs), which aims to develop a scalable editing method that fixes incorrect information and reflects new information.

A typical approach of knowledge editing methods is the *parameter updating* (PU), which updates local parameters directly (Meng et al., 2022a,b; Li et al., 2023) or indirectly via additional hypernetworks (De Cao et al., 2021; Mitchell et al., 2022a; Tan et al., 2024), or augment parameters (Dong et al., 2022a; Huang et al., 2023). However, PU has still shown limited performances in terms of *generalization* and *locality* metrics, two important metrics of knowledge editing (Yao et al., 2023). First, as in the continual learning (Rolnick et al., 2019), PU may cause the catastrophic forgetting given its parametric surgical style, failing to strongly maintain the original knowledge, often resulting in *weakly-performing locality*. Second, PU may suffer from the ripple effect (Cohen et al., 2024) or the unseen contexts (Huang et al., 2024), because in-scope boundary is not explicitly defined and "all" relevant knowledge for given edit requests is hardly pre-identified and captured, thus resulting in *sub-optimal generalization*.

Without sorely relying on a PU method, inspired by in-context learning (ICL)'s versatile abilities (Liu et al., 2022a; Dong et al., 2022b) and its application on the knowledge editing task (Zheng et al., 2023), this paper proposes a novel *in-context post-editing* method, which is applied on top of a PU method, namely **POMEM** – in-context knowledge **po**st **e**diting on **m**assive-**e**diting **m**emory.

More specifically, POMEM establishes the few-shot learning ability of ICL towards desired behaviors for in-context post editing for *in-scope* and *out-of-scope* queries, namely as "Copier" and "Recaller," respectively, i.e., gearing the edited model to revise an initially predicted result made by a PU method towards a correct one, as follows:

- **Copier**, designed for an *in-scope* query, provides few-shot examples to guide the model to explicitly identify and copy an answer from an in-context fact relevant to the input query, likely for enhancing generalization.

- **Recaller**, designed for an *out-of-scope* query, presents a novel "recalling" few-shot examples to guide the model to *recall* the prediction of the original "unedited" model but by the current "edited" model given the query, towards improving the locality.

Experiment results carried on Counterfact dataset show that the proposed POMEM improves existing methods including MEMIT, exhibiting the state-of-the-art performances.

2 Related Works

2.1 PU Approach

PU approaches are further divided into *meta learning*, *locate-then-edit*, and *parameter expansion* methods. 1) Meta learning methods (De Cao et al., 2021; Mitchell et al., 2022a; Tan et al., 2024) train hyper-networks to indirectly update the model parameters for knowledge editing. 2) Locate-then-edit methods Meng et al. (2022a,b); Li et al. (2023) directly modify the parameters in specific model layers or modules that are most related to the desired new knowledge, being largely connected to mechanistic interpretation of Transformer (Geva et al., 2021; Räuker et al., 2023; Nanda et al., 2023). 3) Parameter expansion methods extends parameters by adding neurons or parameters to store new knowledge, such as a calibration memory slot (Dong et al., 2022a) or extra neurons (Dai et al., 2022; Huang et al., 2023).

2.2 Memory-based Approach

Largely related to retrieval-augmented generation (Lewis et al., 2020), Zheng et al. (2023); Zhong et al. (2023); Gu et al. (2023) leverage the ICL ability of LLMs in a manner of providing a pre-stored edit request in an in-context manner for affecting the prediction results by LLMs, where all edit requests are maintained in an external memory, rather than modifying the parameters of LLMs. SERAC (Mitchell et al., 2022b) deploys a semi-parametric method which trains an additional "counterfactual model" to better reflect a pre-stored edit request in predicting a final output.

Similar to POMEM, SERAC uses a scope classifier and handles in-scope and out-of-scope queries using different functions. In contrast to SERAC, POMEM addresses massive editing tasks and mainly relies on the ICL ability for the in-context post editing, without requiring an original base model during inference time.

3 Task Definition: Massive Knowledge Editing

Suppose that $f_\theta(\cdot)$ roughly indicates a function of a given LLM, where $f_\theta(x)$ is the prediction result during the decoding step after taking x as a prefix. Now, let $\mathcal{E} = \{e_i\}_{i=1}^{n}$ be a "massive" set of n edit requests to be injected to the LLM, where $e_i = (s_i, r_i, o_i^*)$ is a i-th *triple*-level edit request, i.e., s_i, r_i, o_i^* indicate a subject, a relation, and an target object, respectively. The massive knowledge editing aims to obtain an edited model $f_\theta^*(\cdot)$ to fulfill efficacy, generalization, and locality, for *all* edits in $\mathcal{E}$.

- **Efficacy** holds if $f_\theta^*(s_i, r_i) = o_i^*$ for $(s_i, r_i) \in \mathcal{E}$.
- **Generalization** holds if $f_\theta^*(s_i', r_i') = o_i^*$ for an "in-scope" prefix $(s_i', r_i') \in \mathcal{I}(e_i)$, where $\mathcal{I}(e_i)$ is the *edit scope* of e_i, the set of *in-scope* examples, a set of "relevant" facts to e_i, which usually include paraphrased and synonymous expressions of e_i.
- **Locality** (or **specificity**) holds if $f_\theta^*(s_i'', r_i'') = f_\theta(s_i'', r_i'')$ for any out-of-scope prefix $(s_i'', r_i'') \in \mathcal{O}(e_i)$ is the set of *out-of-scope* examples, i.e., a set of all "irrelevant" facts to e_i, which should not be affected by the current edit e_i.

Appendix G illustrates detailed examples of an edit, its prefix, in-scope and out-of-scope prefixes, and their correct target objects.

4 Method

Figure 1 presents the overall architecture of the proposed POMEM, whose components include *Copier*, *Recaller* (i.e., two types of in-context post-editing functions), *Fact Retriever* and *Scope Classifier*. POMEM consists of two stages — editing and in-context post-editing – as follows:

- **Editing**: During the massive editing stage, a PU method is applied to inject a set of edits $\mathcal{E}$ into the parametric memory of LLM, and $\mathcal{E}$ is stacked in an external *edit memory* $\mathcal{F}$ using the memorization function Mem:

$$\begin{aligned} f_\theta^* &= \mathsf{PU}\left(f_\theta, \mathcal{E}\right) \\ \mathcal{F} &= \mathsf{Mem}\left(\mathcal{E}\right) \end{aligned} \quad (1)$$

  where PU is a functional to return a PU-based model f_θ^* by taking $\mathcal{E}$, and Mem is a verbalizing function that linearizes each triple knowledge to a natural language sentence. In this paper, we use MEMIT (Meng et al., 2022b) for PU.

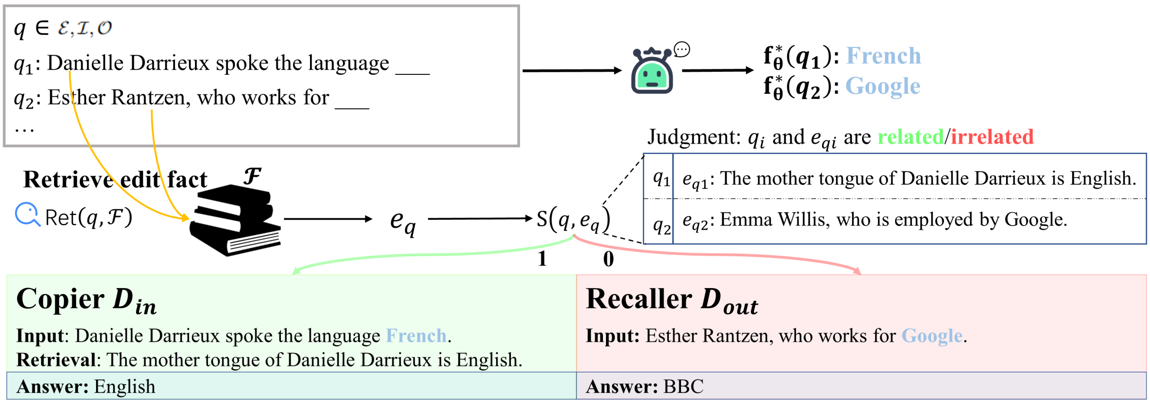

Figure 1: An overall architecture of the in-context post-editing procedure of POMEM, consisting of Copier, Recaller, Fact Retriever, and Scope Classifier: For i-th test query q_i, the PU-edited model is first applied to generate an initial prediction result $f_\theta^*(q_i)$, and Fact Retrieval performs the dense retrieval to find the most similar edit $e_q \in \mathcal{F}$ using Eq. (4) in Section 4.1.1, i.e., $e_q = \mathsf{Ret}(q, \mathcal{F})$. Scope classifier $\mathsf{S}(q, e_q)$ determines whether q is an in-scope instance for the retrieved edit e_q using Eq. (5) in Section 4.1.2. Depending on its scope of q_i, Copier or Recaller are performed as in-context post-editing method using few-shot demonstrations of Eq. (6) in Section 4.1.3, or of Eq. (7) in Section 4.1.4, respectively.

- **In-context post-editing**: Suppose that a query prompt $q = (s, r)$ is given, the in-context post-editing stage first takes the initially predicted result o, determines the scope of q based on its most-relevant fact $e_q \in \mathcal{F}$, applies a scope-specific post-editing, i.e., either Copier or Recaller, depending on the scope of q, as follows:

$$o = f_\theta^*(q) \tag{2}$$

$$e_q = \mathsf{Ret}(q, \mathcal{F})$$

$$o^* = \begin{cases} \mathsf{Copier}(f_\theta^*, o, e_q) & \text{if } \mathsf{S}(q, e_q) = 1 \\ \mathsf{Recaller}(f_\theta^*, o) & \text{otherwise} \end{cases} \tag{3}$$

where $\mathsf{Ret}(q, \mathcal{F})$ is Fact Retriever which finds the most similar fact (i.e., the edit request) $e_q \in \mathcal{F}$ to the input query q, referred to as the retrieved edit, and $\mathsf{S}(q, e_q)$ is the scope classifier which determines whether q is in-scope in e_q, (i.e., $q \in \mathcal{I}(e_q)$).

4.1 In-Context Post Editing

In this section, we provide more details of components for in-context post editing method.

4.1.1 Fact Retriever: $\mathsf{Ret}(q, \mathcal{F})$

Fact Retriever finds the most relevant edit to an input query q by performing the dense retrieval between q and all edits in $\mathcal{F}$, based on a sentence encoder $h(s) \in \mathbb{R}^d$, which returns the sentence vector for a sentence s, as follows:

$$Ret(q, \mathcal{F}) = \text{argmax}_{e \in \mathcal{F}} cos\left(h(q), h(e)\right) \tag{4}$$

where the pre-trained sentence encoder (Reimers and Gurevych, 2019) is used for h.

4.1.2 Scope Classifier: $\mathsf{S}(q, e_q)$

Scope Classifier deploys *Siamese neural networks* similar to (Ranasinghe et al., 2019; Neculoiu et al., 2016), using an additional multi-layer perception (MLP) layers which performs the binary classification, as follows:

$$\mathsf{S}(q, e_q) = \begin{cases} 1 & \text{if } \sigma\left(\mathsf{MLP}\left(h(q) - h(e_q)\right)\right) > 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where σ is the sigmoid function and the parameters of MLP are separately trained in an extra training dataset as in Appendix E.

4.1.3 Copier

Copier guides the model to extract an answer directly from an in-context retrieved edit "relevant" to a given query. Copier prepares few-shot demonstrations $\mathcal{D}_{in} = \left\{d_{in}^{(i)}\right\}_{i=1}^{k}$ where $d_{in}^{(i)}$ is formed based on the following template:

$$\begin{aligned} &\texttt{Input:} \quad q^{(i)} \oplus f_\theta^*(q^{(i)}) \\ &\texttt{Retrieval:} \quad e_{q^{(i)}} \\ &\texttt{Answer:} \quad \mathsf{obj}(e_{q^{(i)}}) \end{aligned} \tag{6}$$

where $\oplus$ is the concatenation operator of strings, $q^{(i)}$ is i-th query part in $\mathcal{D}_{in}$, $e_{q^{(i)}} = \mathsf{Ret}(q^{(i)})$, and $\mathsf{obj}(e)$ is the selector that returns an "object" part in an edit e.

Model	Editor	Score ↑		Efficacy ↑		Generalization ↑		Locality ↑	
		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
	POMEM	**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
	SERAC	86.5	28.4	99.1	**96.1**	82.0	59.7	80.7	12.7
	POMEM	**90.3**	**43.5**	**99.8**	95.5	**96.4**	**87.0**	77.9	21.3

Table 1: The performances of editing 10,000 requests in Counterfact dataset, under GPT2-XL and GPT-J settings, comparing POMEM with other baselines – Base (the unedited model), ROME, MEMIT, MEND, and SERAC.

4.1.4 Recaller

Recaller forces the model to "recall" a prediction of an "original" unedited model to a given query, regardless of the current prediction of f_θ. To gear the edited model to recall an "original" prediction, Recaller uses few-shot demonstrations $\mathcal{D}_{out} = \left\{ d_{out}^{(j)} \right\}_{j=1}^{k'}$ where $d_{out}^{(j)}$ is formed as follows:

$$\begin{aligned} &\texttt{Input:} \quad q^{(j)} \oplus f_\theta^*(q^{(j)}) \\ &\texttt{Answer:} \quad f_\theta(q^{(j)}) \end{aligned} \tag{7}$$

where $q^{(j)}$ is j-th query part in $\mathcal{D}_{out}$. Different from Eq. (6), `Retrieval` does not appear and `Answer` part uses the original prediction.

Copier and Recaller finalize the prompts by appending the test case following the same form of their demonstrations, as in Appendix F, which also include some examples.

5 Experimental & Analysis

5.1 Setup

We adopted MEMIT(Meng et al., 2022b) for a PU method and evaluate our method on CounterFact dataset with GPT2-XL (1.5B) and GPT-J (6B) language model. See Appendix A, C, and D for the details of MEMIT, the description of CounterFact dataset, and the evaluation metrics.

5.1.1 Main Results

Table 1 presents the results of POMEM on the Counterfact dataset under GPT2-XL and GPT-J, comparing to Base (i.e., the unedited base model), ROME (Meng et al., 2022a), MEMIT (Meng et al., 2022b), MEND (Mitchell et al., 2022a), and SERAC (Mitchell et al., 2022b). As in MEMIT (Meng et al., 2022b), the experiments reveals again that ROME method struggles with low performances because it injects large amounts of knowledge into a single layer, which are substantially improved by MEMIT. We also observe that MEMIT demonstrates weak locality, showing a non-trivial gap compared to the Base model, and its generalization is far from optimal, particularly when using GPT-2 XL. It is clearly shown that POMEM outperforms MEMIT both generalization and locality, confirming that the proposed in-context post-editing, based on Copier and Recaller, effectively revises initially predicted results to correct ones, thereby leading to these improvements.

6 Discuss & Conclusion

In this paper, we proposed POMEM based on two types of in-context knowledge post-editing methods – Copier and Recaller – which are designed in scope-specific manners for in-scope and out-of-scope queries, respectively.

In future work, we would like to examine whether POMEM is effective on other PU methods such as meta learning and parameter expansion methods, comparing to the locate-then-edit methods. Given the recent advances in sequential editing (Huang et al., 2023), event-level editing (Liu et al., 2024; Peng et al., 2024), reasoning-aware editing tasks (Zhong et al., 2023), it would be worthy to explore how in-context post-editing of POMEM is generalized on these new variants of tasks.

Limitations

In our current setting, few-shot demonstrations used for Copier and Recaller in in-context post editing are fixed across all test quests. Given that the demonstration selection and ordering are important in ICL (Liu et al., 2022b; Rubin et al., 2022; Lu et al., 2022), however, POMEM could be enhanced by dynamically selecting or ordering few-shot examples, more effectively for a current input query. In addition, POMEM currently relies on a simple format of prompts that only capture the key required information, and advanced prompting formats also need to be explored, as the prompt engineering is increasingly important in effectively exploiting ICL abilities of LLMs (Dong et al., 2023; Wang et al., 2023b).

The proposed POMEM is currently evaluated only under MEMIT as a PU method, however, it could generally be applicable to other types of PU methods such as meta learning (De Cao et al., 2021; Mitchell et al., 2022a; Tan et al., 2024) and parameter expansion methods (Dong et al., 2022a; Dai et al., 2022; Huang et al., 2023). It would be worthy to explore POMEM on other PU methods and to examine the effect of the selection of a PU method in the in-context post-editing.

In an architectural design, POMEM relies on Scope Classifier, a parametric model, not being designed in an ICL manner. As a result, POMEM's in-context post-editing is not purely designed based on ICL mechanism. While we tried to explore the ICL-based scope classifier in Appendix E, its performance is not effective. Enabling all required components of POMEM under ICL mechanism would be a valuable future direction, as the post-editing method could be realized as the process of chain-of-thought (CoT) (Wei et al., 2022), without relying on an external parametric model.

References

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

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, Dublin, Ireland. Association for Computational Linguistics.

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. 2022a. Calibrating factual knowledge in pretrained language models. *Findings of Empirical Methods in Natural Language Processing (EMNLP)*.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. 2023. A survey on in-context learning.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022b. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7).

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)*.

Hengrui Gu, Kaixiong Zhou, Xiaotian Han, Ninghao Liu, Ruobing Wang, and Xin Wang. 2023. Pokemqa: Programmable knowledge editing for multi-hop question answering. *arXiv preprint arXiv:2312.15194*.

Youcheng Huang, Wenqiang Lei, Zheng Zhang, Jiancheng Lv, and Shuicheng Yan. 2024. See the unseen: Better context-consistent knowledge-editing by noises. *arXiv preprint arXiv:2401.07544*.

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*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

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. 2022a. 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.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022b. 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.

Jiateng Liu, Pengfei Yu, Yuji Zhang, Sha Li, Zixuan Zhang, and Heng Ji. 2024. Evedit: Event-based knowledge editing with deductive editing boundaries. *arXiv preprint arXiv:2402.11324*.

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.

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*.

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.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023. Progress measures for grokking via mechanistic interpretability. *arXiv preprint arXiv:2301.05217*.

Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. 2016. Learning text similarity with Siamese recurrent networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 148–157, Berlin, Germany. Association for Computational Linguistics.

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.

Hao Peng, Xiaozhi Wang, Chunyang Li, Kaisheng Zeng, Jiangshan Duo, Yixin Cao, Lei Hou, and Juanzi Li. 2024. Event-level knowledge editing. *arXiv preprint arXiv:2402.13093*.

Tharindu Ranasinghe, Constantin Orasan, and Ruslan Mitkov. 2019. Semantic textual similarity with Siamese neural networks. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 1004–1011, Varna, Bulgaria. INCOMA Ltd.

Tilman Räuker, Anson Ho, Stephen Casper, and Dylan Hadfield-Menell. 2023. Toward transparent ai: A survey on interpreting the inner structures of deep neural networks. In *2023 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pages 464–483. IEEE.

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.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. 2019. Experience replay for continual learning. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2671, Seattle, United States. Association for Computational Linguistics.

Chenmien Tan, Ge Zhang, and Jie Fu. 2024. Massive editing for large language models via meta learning. In *International Conference on Learning Representations*.

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*.

Xinyuan Wang, Chenxi Li, Zhen Wang, Fan Bai, Haotian Luo, Jiayou Zhang, Nebojsa Jojic, Eric P Xing, and Zhiting Hu. 2023b. Promptagent: Strategic planning with language models enables expert-level prompt optimization. *arXiv preprint arXiv:2310.16427*.

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*.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2023. Editing large language models: Problems, methods, and opportunities. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10222–10240, Singapore. Association for Computational Linguistics.

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 PU method for Editing: MEMIT

POMEM needs to first perform a PU method by injecting a set of massive edits to the parametric memory, to obtain an edited model, as in Figure 2

We use MEMIT (Meng et al., 2022b) as such a PU method for editing stage in Eq. (1), which is an extensible multi-layer update algorithm, to inject a massive set of edits into the language model. Given the range of layers $\mathcal{R}$ to be updated, let $L = max(\mathcal{R})$ be a target layer. For i-th edit $e_i = (s_i, r_i, o_i^*) \in \mathcal{E}$, a set of enlarged factual prompts, $\mathcal{P}_i = \{x_j \oplus p(s_i, r_i)\}_{j=1}^{P}$ are prepared to enhance "generalization" of editing where random prefixes x_j, that cover various contexts that (s_i, r_i) appear, are prepended into a templated prompt $p(s_i, r_i)$. The optimization to predict the target object o_i^* under $\mathcal{P}_i$ leads to the residual vector δ_i using Eq. (8), which refer to the extent of updating the hidden state h_i^L at layer L.

$$\delta_i \leftarrow argmin_{\delta_i'} \frac{1}{P} \sum_{j=1}^{P} - \log \mathbb{P}_{G\left(h_i^L += \delta_i'\right)} \left[o_i' \mid x_j \oplus p(s_i, r_i)\right] \tag{8}$$

where the $G\left(h_i^L += \delta_i'\right)$ operation is called "hooking," which uses the adjusted hidden state $h_i^L += \delta_i'$ to execute the transformer.

Then, δ_i is propagated across predefined editing layers $l \in \mathcal{R} = \{l_1 \ldots L\}$, i.e., $\delta_i/(L - l + 1)$, which leads to obtain the increments Δ^l to update the MLP weights in layers in $\mathcal{R}$, resulting in the updated hidden representation at the layer L:

$$\begin{aligned}\hat{h}_i^L = & h_i^0 + \sum_{l=1}^{L} \text{attn}\left(h_i^{l-1}\right) \\ & + \sum_{l=1}^{L} \left[\left(W_{\text{out}}^l + \Delta^l\right) relu\left(W_{\text{in}}^l \gamma\left(h_i^{l-1}\right)\right)\right]\end{aligned} \tag{9}$$

where, h_i^0 is the initial embedding of input token, γ and $relu$ is layernorm and ReLU activation function. Appendix A.1 presents further details for calculating Δ^l for updating the MLP weights.

Appendix B.2 describes the hyperparameter design of MEMIT under different LLM models.

A.1 Updating MLP weights in MEMIT

It is assumed that the the MLP layer stores facts as key-memory pairs, as in (Geva et al., 2021). The MLP output layer weight after massive editing is defined as:

$$W_{out} \triangleq \underset{\hat{W}}{\text{argmin}} \sum_{i=1}^{n} \left\| \hat{W} k_i - m_i \right\|^2 \tag{10}$$

where k_i is defined as encoded subject vector for i-th edit, m_i is its corresponding "target" memory representation. During editing, as in Eq. (9), Δ is calculated to update W_{out}, resulting in a new weight matrix $\hat{W}_{out}$ to inject the new association.

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

By distinguishing old knowledge from new one, Eq. (10) is further decomposed into

$$\hat{W}_{out} \triangleq \underset{\hat{W}}{\text{argmin}} \left(\sum_{i=1}^{n} \left\| \hat{W} k_i - m_i \right\|^2 + \sum_{i=n+1}^{n+u} \left\| \hat{W} k_i - m_i \right\|^2 \right) \tag{12}$$

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

$$\begin{aligned}\hat{W}_{out} \left[\begin{array}{cc} K_0 & K_1 \end{array}\right] \left[\begin{array}{cc} K_0 & K_1 \end{array}\right]^T = \\ \left[\begin{array}{cc} M_0 & M_1 \end{array}\right] \left[\begin{array}{cc} K_0 & K_1 \end{array}\right]^T\end{aligned} \tag{13}$$

According to formula 11, formula 13 is further simplified to:

$$\Delta\left(K_0 K_0^T + K_1 K_1^T\right) = M_1 K_1^T - W_{out} K_1 K_1^T \tag{14}$$

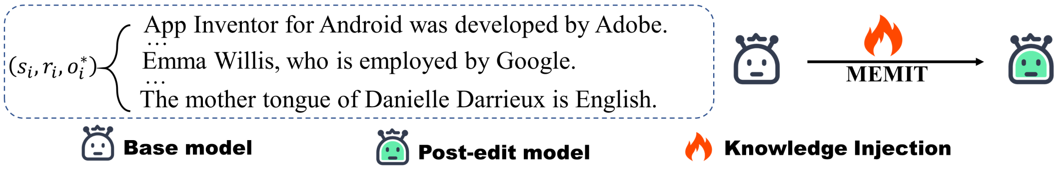

Figure 2: PU-based massive knowledge editing: Inject new knowledge into the base model to produce an edited model.

Suppose that $C \triangleq K_0 K_0^T$ and $R \triangleq M_1 - W_{out} K_1$, where C is a constant proportional to the non-central covariance of K_0, and R is the residual of the new memory representations and the old ones for the required edits, corresponding to δ_i which is evenly distributed across the remaining layers.

Eq. (14) is simplified to:

$$\Delta = R K_1^T \left(C + K_1 K_1^T\right)^{-1} \tag{15}$$

For each $l \in \mathcal{R} = \{l_1 \ldots L\}$, its massive edit requests are reflected in K_1 and M_1, which consists of a set of pairs of all key vectors and their target memory representations as follows:

$$\begin{aligned} K_1 &= K^l = \left[k_1^l \ldots k_n^l\right] \\ M_1 &= R^l = \left[r_1^l \ldots r_n^l\right] \end{aligned} \tag{16}$$

where k_i^l and r_i^l for $e_i = (s_i, r_i, o_i^*) \in \mathcal{E}$ are computed as:

$$k_i^l = \frac{1}{P} \sum_{j=1}^{P} k^l \left(x_j \oplus s_i\right) \tag{17}$$

$$r_i^l = \delta_i / (L - l + 1) \tag{18}$$

where $k^l(x) = relu\left(W_{\text{in}}^l \gamma(x)\right)$.

Under $K_1 = K^l$ and $M_1 = R^l$, we apply Eq. (15) to finalize compute Δ^l used to update the MLP weights at layer l.

B Implementation Detail

B.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.

B.2 Editing Hyperparameters

All hyperparameters about using MEMIT to edit GPT2-XL and GPT-J can be found in the EasyEdit code (Wang et al., 2023a). The important parameters are shown in table 2. t and η are the number of steps and learning rate of δ_i optimization respectively, and $\mathcal{R}$ is the model layer to be edited.

Model	η	t	$\mathcal{R}$
GPT2-XL	0.5	20	[13, 14, 15, 16, 17]
GPT-J	0.5	25	[3, 4, 5, 6, 7, 8]

Table 2: Optimization parameters and editing layer of language model.

C DataSet

The form of data in Counterfact is mainly represented by the cloze-style. For the edit data $e_i = (s_i, r_i, o_i^*) \in \mathcal{E}$, its definition form is as in table 3. 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 Efficacy. The set $\mathcal{I}$ for evaluating Generalization is rewritten from $\mathcal{E}$ in order to test the generalization performance of the post-edit model. The set $\mathcal{O}$ for testing Locality, which represents the same semantics as $\mathcal{E}$ but unrelated facts, in order to test the deterioration of the model after editing.

D Evaluation Metrics

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 Efficacy, Generalization and Locality respectively.

D.1 Efficacy

Efficacy Success(ES), that is formula 19, 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.

$$\mathbb{E}_{(s_i, r_i) \sim \mathcal{E}} \left\{ f_\theta^* \left(o_i^* \mid (s_i, r_i)\right) > f_\theta^* \left(o_i \mid (s_i, r_i)\right) \right\} \tag{19}$$

Efficacy Accuracy(EA), the formula 20 evalu-

triple-level	s_i	Edwin of Northumbria
	r_i	The official religion of { } is _
	o_i	Christianity
	o_i^*	Islam
$\mathcal{E}$	(s_i, r_i)	The official religion of Edwin of Northumbria is _
$\mathcal{I}$	(s_i', r_i')	Edwin of Northumbria follows the religion of _
		Edwin of Northumbria is affiliated with the religion _
$\mathcal{O}$	(s_i'', r_i'')	The official religion of Charles Aznavour is _
		Nicolas Sarkozy is affiliated with the religion _
		Andrew Johnson is affiliated with the religion _
		The official religion of Paul is _
		Ringo Starr is follower of _
		The official religion of Nicolas Sarkozy is _
		The official religion of Andrew Johnson is _
		Orson Welles is affiliated with the religion _
		Lady Gaga is follower of _
		Quentin Tarantino is affiliated with the religion _

Table 3: Formulation of the Counterfact Dataset. It is displayed at the triple-level, including subject s_i, relation r_i, original object o_i, and new object o_i^*. From the set-level, it shows editing facts set $\mathcal{E}$, relevant facts set $\mathcal{I}$ and irrelevant facts set $\mathcal{O}$.

ates whether the most likely output token is o_i^*.

$$\mathbb{E}_{(s_i,r_i)\sim\mathcal{E}}\left\{\underset{o}{\operatorname{argmax}} f_\theta^*\left(o \mid (s_i, r_i)\right) = o_i^*\right\} \quad (20)$$

D.2 Generalization

Similar to Efficacy, the fact q from the *in-scope* set $\mathcal{I}$ is used to test **Paraphrase Success(PS)**, formula 21, and **Paraphrase Accuracy(PA)**, formula 22.

$$\mathbb{E}_{(s_i,r_i)\sim\mathcal{I}}\left\{f_\theta^*\left(o_i^* \mid (s_i, r_i)\right) > f_\theta^*\left(o_i \mid (s_i, r_i)\right)\right\} \quad (21)$$

$$\mathbb{E}_{(s_i,r_i)\sim\mathcal{I}}\left\{\underset{o}{\operatorname{argmax}} f_\theta^*\left(o \mid (s_i, r_i)\right) = o_i^*\right\} \quad (22)$$

D.3 Locality (or Specificity)

The **Neighborhood Success(NS)**, and **Neighborhood Accuracy(NA)**, metrics of the fact q from the *out-of-scope* set $\mathcal{O}$ are evaluated in terms of Locality, formula 23 and formula 24. We expect the model not to change the answers to these irrelevant facts after editing, they should output the original answer o_i.

$$\mathbb{E}_{(s_i,r_i)\sim\mathcal{O}}\left\{f_\theta^*\left(o_i^* \mid (s_i, r_i)\right) < f_\theta^*\left(o_i \mid (s_i, r_i)\right)\right\} \quad (23)$$

$$\mathbb{E}_{(s_i,r_i)\sim\mathcal{O}}\left\{\underset{o}{\operatorname{argmax}} f_\theta^*\left(o \mid (s_i, r_i)\right) = o_i\right\} \quad (24)$$

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.

E Methods of Scope Classification

POMEM uses the Siamese neural network classifier for $\mathsf{S}(q, e_q)$ as in Section 4.1.2. To verify Siamese network classifier, we present the ablation study comparing four scope classification methods, denoted as follows:

- POMEM$_{ICL}$: The ICL-based scope classification based on properly-designed few-shot examples. A prompt is designed for analyzing whether q and e_q are related, as shown in Table 4.
- POMEM$_{Subj}$: The classification based on subject matching classification by checking where $\mathsf{subj}(e_q) = \mathsf{subj}(q)$ where $\mathsf{subj}(s)$ is a selection function of a subject in s.
- POMEM$_{Thr}$: The threshold-based classification by checking whether the cosine similarity between a query and a retrieved edit is above the given threshold τ, i.e., $cos(h(q), h(e_q)) \geq \tau$.
- POMEM$_{Siam}$: The proposed Siamese neural network classifier as described in Section 4.1.

Table 5 compares the performances of four scope classifiers, in terms of the classification accuracy. It is shown that the proposed Siamese neural network (i.e., POMEM$_{Siam}$) shows the best classification result, outperforming other methods, while ILC-based method (i.e., POMEM$_{ICL}$) shows the worse performance by failing to correctly classify out-of-scope queries.

Table 6 compares the editing performances using four scope classifiers under POMEM, in terms of knowledge editing evaluation metrics, under GPT2-XL and GPT-J settings. POMEM$_{Siam}$ and POMEM$_{Subj}$ show the best performing results when using GPT-J.

E.1 Details of the Siamese Neural Network

The subnetworks of the Siamese neural network utilize the "all-MiniLM-L6-v2" (Reimers and Gurevych, 2019) for encoding q and e_q. The encoding function is defined as $h()$. The distance between two feature vectors is computed using $h(q) - h(e_q)$, which indicates the similarity between the input sample pairs. An additional MLP layer is employed for binary classification, taking $h(q) - h(e_q)$ as input and mapping the output to a single dimension to determine the similarity between q and e_q.

The training data is sourced from unused portions of the Counterfact dataset. From each group's unused data, two samples are extracted from the in-scope knowledge and two from the out-of-scope knowledge. These samples are then combined with the edited data from the same group to form a complete dataset. For labeling, in-scope data is assigned a ground truth label of 1, while out-of-scope knowledge is assigned a ground truth label of 0. The binary classification MLP is trained using the Adam (Duchi et al., 2011) optimizer with a learning rate of 1×10^{-3}. Training is conducted for 3 epochs on the entire dataset. The loss function utilized is Binary Cross Entropy with Logits Loss.

F Demonstrations Detail

Copier and Recaller use different types of demonstrations, $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$ as presented in Section 4.1.3-4.1.4.

Copier finalizes the in-context post-editing prompt by appending the test case:

$$\begin{aligned}&\mathsf{prompt}_{in} =\\&\quad \texttt{Input:}\quad q \oplus f_\theta^*(q)\\&\quad \texttt{Retrieval:}\quad e_q\\&\quad \texttt{Answer:}\end{aligned} \tag{25}$$

which leads to complete the function of Copier as follows:

$$\mathsf{Copier}\left(f_\theta^*, o, e_q\right) = f_\theta^*\left(\mathcal{D}_{in} \oplus \mathsf{prompt}_{in}\right) \tag{26}$$

Similarly, Recaller uses the test part of in-context prompt as follows:

$$\begin{aligned}&\mathsf{prompt}_{out} =\\&\quad \texttt{Input:}\quad q \oplus f_\theta^*(q)\\&\quad \texttt{Answer:}\end{aligned} \tag{27}$$

which leads to complete the function of Copier as follows:

$$\mathsf{Recaller}\left(f_\theta^*, o\right) = f_\theta^*\left(\mathcal{D}_{out} \oplus \mathsf{prompt}_{out}\right) \tag{28}$$

Table 7 mainly shows the demonstrations built for the Counterfact dataset.

G Case Study

Table 8a and and 8b illustrate the case studies of *in-scope* and *out-of-scope* knowledge using two test fact examples from the Counterfactual dataset that were successfully corrected and recalled.

More specifically, tables present the following: an input query q to be evaluated (marked in red), a preliminary answer resulting by a PU method (marked in purple), a retrieved fact $e_q = \mathsf{Ret}(q, \mathcal{F})$ (marked in blue) from the edited facts $\mathcal{F}$, few-shot demonstrations (i.e., $\mathcal{D}_{in}$ or $\mathcal{D}_{out}$), a test prompt, which consists of an input query q, a preliminary answer $f_\theta^*(q)$, a retrieved fact e_q (i.e,, e_q is optionally required only for in-scope queries), and a querying prompt `Answer:` (marked in green).

The total prompts with few-shot demonstrations are fed into the edited model f_θ^* to generate a final answer (marked in orange).

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 4: In-context demonstrations for classifying relevance.

Method	in-scope	out-of-scope	all
POMEM$_{ICL}$	94.63	15.69	33.91
POMEM$_{Subj}$	99.14	84.82	88.12
POMEM$_{Thr}$	93.92	83.59	85.97
POMEM$_{Siam}$	88.09	89.99	89.55

Table 5: Accuracy of four methods for in-scope and out-scope classification

		Score $\uparrow$		**Efficacy** $\uparrow$		**Generalization** $\uparrow$		**Locality** $\uparrow$	
Model	**Editor**	SS	AS	ES	EA	PS	PA	NS	NA
GPT2-XL	POMEM$_{ICL}$	75.1	27.1	79.3	52.9	75.4	48.6	71.1	14.0
	POMEM$_{Subj}$	**87.7**	34.1	99.1	89.8	**96.4**	**85.7**	72.7	15.4
	POMEM$_{Thr}$	86.9	32.8	99.1	89.8	94.2	82.4	72.4	14.7
	POMEM$_{Siam}$	86.7	**35.1**	**99.2**	**90.1**	91.9	78.3	**73.3**	**16.2**
GPT-J	POMEM$_{ICL}$	85.1	19.3	99.8	94.8	97.1	88.1	66.9	7.5
	POMEM$_{Subj}$	90.3	42.6	99.8	95.5	**97.6**	**92.2**	77.3	20.4
	POMEM$_{Thr}$	90.1	41.8	99.8	95.5	96.6	89.5	77.3	19.9
	POMEM$_{Siam}$	**90.3**	**43.5**	**99.8**	**95.5**	96.4	87.0	**77.9**	**21.3**

Table 6: The comparison results of using four scope classifiers in POMEM, which result from performing the massive knowledge editing on 10,000 in the Counterfact dataset.

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) Demonstrations $\mathcal{D}_{in}$ of Copier for in-context post-editing designed to handle *in-scope* queries, whose form is defined in Eq. (6).

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) Demonstrations $\mathcal{D}_{out}$ of Recaller for in-context post-editing designed to handle *out-of-scope* queries, whose form is defined in Eq. (7).

Table 7: Details of demonstrations of Copier and Recaller, denoted as $\mathcal{D}_{in}$ and $\mathcal{D}_{out}$ which are selected from Counterfact dataset.

$\begin{cases} q = (s'_i, r'_i) \\ f^*_\theta(q) \\ e_q \end{cases}$	David Rivett works as _ a David Rivett's occupation is composer.
$\begin{cases} D_{in} \\ q + f^*_\theta(q) \\ e_q \\ o^* \end{cases}$	... Input: David Rivett works as a. Retrieval: David Rivett's occupation is composer. Answer: composer

(a) A case study of Copier. The retrieval part is related to the input query, and is concatenated with the in-scope in-context demonstrations $\mathcal{D}_{in}$ for post-editing.

$\begin{cases} q = (s, r) \\ f^*_\theta(q) \\ e_q \end{cases}$	In Cantavieja, they understand _ English In Asturias, they understand German.
$\begin{cases} D_{out} \\ q + f^*_\theta(q) \\ o^* \end{cases}$	... Input: In Cantavieja, they understand English. Answer: Spanish

(b) A case study of Recaller. The retrieval part is unrelated to the input part, and is concatenated with the out-of-scope in-context demonstrations $\mathcal{D}_{out}$ for recalling a result of the unedited base model.

Table 8: Illustrations of Copier and Recaller on the selected case examples in Counterfact dataset. The red part is the input query q to be evaluated to a PU-edited model f^*_θ. The purple part is a preliminary answer generated by a PU model, i.e., $f^*_\theta(q)$, the blue part is the retrieved edit e_q by Retriever, the green parts are prompts of either Copier and Recaller with few-shot demonstrations, and the orange part is a final answer generated after the in-context post-editing.