COMEM: In-Context Retrieval-Augmented Mass-Editing Memory in Large Language Models

Shanbao Qiao and **Xuebing Liu** and **Seung-Hoon Na***
Center for Advanced Image and Information Technology,
Department of Computer Science and Artificial Intelligence,
Jeonbuk National University
{joe,liuxuebing,nash}@jbnu.ac.kr

Abstract

Noting that world knowledge continuously evolves over time, large language models (LLMs) need to be properly adjusted by performing the "knowledge editing", which involves updating outdated information or correcting false information. To achieve reliable and "massive" editing capabilities in terms of *generalization* and *specificity*, this paper proposes a unified knowledge editing method called in-**CO**ntext retrieval-augmented **M**ass-**E**diting **M**emory (COMEM), which combines two types of editing approaches: parameter updating and in-context knowledge editing (IKE). In particular, COMEM incorporates *retrieval-augmented IKE*, a novel extension of IKE designed for massive editing tasks, based on an *updating*-aware demonstration construction. Experimental results on the zsRE and CounterFact datasets demonstrate that COMEM outperforms all existing methods, achieving state-of-the-art performance. Our code is available at https://github.com/JoveReCode/COMEM.git.

1 Introduction

Large language models (LLMs), owing to their vast stored amount of world knowledge, have demonstrated the remarkable abilities in understanding and generating natural languages, as well as achieving state-of-the-art performance in a wide range of natural language processing (NLP) applications (Touvron et al., 2023; OpenAI, 2023; Petroni et al., 2020). Given demands to enhance controllability for LLMs in knowledge manipulation (Onoe et al., 2022; Dhingra et al., 2022; Liška et al., 2022) and content generation (Zhao et al., 2023; Ji et al., 2023; Lazaridou et al., 2021; Agarwal and Nenkova, 2022; Gallegos et al., 2023), there has been recently increasing studies on the "knowledge editing" task, which aims to explicitly provide the "editing" mechanism, such as revising knowledge or correcting false information in LLMs, in a controllable, scaled, and effective manner. In particular, this paper addresses the "massive" editing task, as discussed in (Meng et al., 2022b), because LLMs often encounter the issue of massive edits, which require updating more than hundreds or thousands of facts, given the huge knowledge space.

Approaches for knowledge editing in LLMs have been categorized into two main types: *parameter updating* and *in-context knowledge editing* (IKE). Parameter updating adjusts the local parameters or specific layers in LLMs using a gradient-based method to generate desired targets given edit requests (Cao et al., 2021; Mitchell et al., 2022a; Meng et al., 2022a,b; Li et al., 2023). In the massive editing task, the advantage of parameter updating is inherited from LLMs; the knowledge is stored implicitly in the LLM's parameters and the inference step for knowledge lookup is simply conducted in a generative manner based on the decoder, without requiring the maintenance of an external memory or searching over a set of edits. However, parameter updating may lead to *under-editing* problems because some edits and their relevant facts are interrupted by other edits, thereby being stored in a somewhat blurred manner. Furthermore, as noted by Zheng et al. (2023), parameter updating may cause side effects such as catastrophic forgetting or over-editing of out-of-scope knowledge.

On the other hand, motivated by the ability of in-context learning (ICL) (Brown et al., 2020; Wei et al., 2023), IKE guides LLMs to generate desired targets in a given context by prepending specific edit-related prompts consisting of relevant demonstrations. IKE has been shown to effectively perform knowledge editing based on demonstration formatting and organization strategies (Zheng et al., 2023), without modifying the model parameters. In the setting of the massive editing task, however, IKE may require an additional retrieval step to find

*Corresponding author

Findings of the Association for Computational Linguistics: NAACL 2024, pages 2333–2347

the relevant facts given the test query, which is not required in parameter updating. Additionally, IKE performance largely relies on the construction of demonstrations, possibly leading to a risky situation when the demonstrations are not optimally suited for some test prompts or contexts.

The goal of knowledge editing is to satisfy both *generalization* and *specificity*, however, trade-off exists between these properties. Pursuing a reliable massive editing ability for more stably satisfying *generalization* and *specificity*, this paper proposes a unified knowledge editing method called in-**CO**ntext retrieval-augmented **M**ass-**E**diting **M**emory (COMEM), which combines parametric updating and IKE, specifically consisting of two components:

- **MEMIT for parameter updating**, which takes a set of massive edits and directly applies MEMIT (Meng et al., 2022b) to update the provided knowledge in LLMs.

- **Retrieval-augmented IKE**, which generates IKE (Zheng et al., 2023) to handel massive edits, by memorizing all the edit requests with their relevant demonstrations from the set of training edits. Unlike the original IKE (Zheng et al., 2023), we further propose *updating-aware demonstration* construction, motivated by the fact that "copy"-type demonstrations may not be much necessary because it is expected that the use of MEMIT somehow exhibits the basic editing capability, thus likely obtaining the proper level of generalization and specificity. By removing copy types, we could add other types of demonstrations, which are shown to be helpful in further improving the final editing performances under the combined setting.

Our contributions are summarized as follows. 1) We propose COMEM, a novel knowledge editing approach that combines parameter updating and IKE to guide the model towards stable generalization and specificity for massive editing; 2) We extensively apply IKE to the massive editing setting and present the retrieval-augmented IKE, further proposing an updating-aware demonstration that is optimal under COMEM; 3) The proposed COMEM shows state-of-the-art performances on the zsRE and CounterFact datasets.

2 Related Work

2.1 Parameter Updating for Knowledge Editing

Parameter updating methods can be categorized into two types: hypernetwork-based methods and attribution-based methods.

For the hypernetwork-based method, the Knowledge Editor (Cao et al., 2021) trains a hypernetwork that predicts parameter changes during inference for updating the target fact and retaining other unrelated knowledge. MEND (Mitchell et al., 2022a) uses a hypernetwork to convert the initial fine-tuning gradient into a simplified representation using low-rank decomposition. SERAC (Mitchell et al., 2022b) offers a higher-capacity solution by incorporating a semi-parametric editing approach with a retrieval augmented counterfactual model. It stores the edits in a separate memory and learns to reason with them to influence the predictions of the base model.

For attribution-based methods, (Dai et al., 2022) explores how LLMs store factual knowledge, introduces the concept of knowledge neurons, and utilizes knowledge neurons for precise factual knowledge editing (updates and erasures) without resorting to fine-tuning. ROME (Meng et al., 2022a) is a pioneering study that attempts to locate the model parameters associated with the target factual knowledge and rewrites the key-value pairs in the MLP module with newly computed vectors. However, all of these methods suffer from significant efficacy and generalization deterioration when the required editing volume is increased. MEMIT (Meng et al., 2022b) further improves ROME to enable massive knowledge editing by spreading the weight changes over multiple model layers.

2.2 In-Context Learning for Knowledge Editing

In-context learning (Dong et al., 2022) is a technique that emerged with the advent of LLMs, where the model learns by observing and incorporating contextual information (Liu et al., 2022; Brown et al., 2020). This involves temporarily adapting or updating a model's parameters based on the provided prompts or demonstrations (Lu et al., 2022; Rubin et al., 2022) in a run, leading to an improvement in model performance.

(Si et al., 2023) pioneered the use of in-context learning to update knowledge in LLMs. They demonstrated that incorporating various types of

demonstration significantly enhanced the success rate of knowledge editing. IKE (Zheng et al., 2023) further extended ICL-based knowledge editing to different language models with fewer side effects.

Both parameter updating and ICL-based methods have their own editing capabilities and combing them in a complementary manner likely leads to a significant improvement in the performance of massive knowledge editing tasks. To this end, this study integrates these parameter updating and ICL-based methods in a unified manner and aims to provide a more stable editing capability for massive knowledge editing.

3 Task Definition

Suppose that $\mathcal{S}$ is a set of real-world entities or concepts, $\mathcal{M}_\theta$ is an autoregressive language model with the parameter θ and $\mathcal{E} = \{e_i\}_{i=1}^{N}$ a set of new facts to be injected into $\mathcal{M}_\theta$, where $e_i = (s_i, r_i, o_i)$ is the i-th edit, i.e. a triple that consists of a subject $s_i \in \mathcal{S}$, a relation r_i, an object $o_i \in \mathcal{S}$. For simplicity in notation, $\mathcal{M}_\theta(x)$ represents the result generated by the language model $\mathcal{M}_\theta$ given the input prompt x, defined as follows:

$$\mathcal{M}_\theta(x) = \underset{y \in \mathcal{S}}{\operatorname{argmax}} P_{\mathcal{M}_\theta}(y|x) \tag{1}$$

where $P_{\mathcal{M}_\theta}(y|x)$ is the generative probability of y, given a prefix x.

The objective of the massive knowledge editing is to ensure efficacy, generalization, and specificity for "all" edits in $\mathcal{E}$. Formally, for $e_i \in \mathcal{E}$, let $\mathcal{I}(e_i)$ represent the *edit scope* of e_i, which is the set of *in-scope* examples, and let $\mathcal{O}(e_i) = \mathcal{U} - \mathcal{I}(e_i)$ represent the set of *out-of-scope* examples, where $\mathcal{U}$ is the universal set of knowledge [1]. For example, if e_i is "*Fox News was created in Canada*," an in-scope example in $\mathcal{I}(e_i)$ could be "*Fox Soccer News originated in Canada*," and an out-of-scope example in $\mathcal{O}(e_i)$ could be "*iOS 6 was created by Apple.*" Efficacy, generalization, and specificity are defined as follows:

- **Efficacy**, which is satisfied for the i-th edit if $o_i = \mathcal{M}_\theta(x_i)$ where x_i is the prefix prompt, roughly defined as $[s_i, r_i]$ for the i-th edit [2] .
- **Generalization**, which is satisfied for the i-th edit if $o = \mathcal{M}_\theta(x)$ for all in-scope examples $(s, r, o) \in \mathcal{I}(e_i)$ and $x = [s; r]$.
- **Specificity**, which is satisfied for the i-th edit if $o = \mathcal{M}_\theta(x)$ for all out-of-scope examples $(s, r, o) \in \mathcal{O}(e_i)$ and $x = [s; r]$.

4 Method

Figure 1 depicts the overall structure of our proposed COMEM for injecting a set of edits $\mathcal{E}$ into the language model $\mathcal{M}_\theta$, which combines parameter updating method and IKE. Formally, suppose that $\mathcal{T} = \left\{e_j^t\right\}_{j=1}^{M}$ is a set of "training" edits in a training set, and each training edit e_j^t is pre-associated with $\mathcal{D}^t(e_j^t) = (\mathcal{D}^c(e_j^t), \mathcal{D}^u(e_j^t), \mathcal{D}^r(e_j^t))$, a set of demonstrations of three types, *copy*, *update*, and *retain*, denoted as $\mathcal{D}^c(e_j^t)$, $\mathcal{D}^u(e_j^t)$, and $\mathcal{D}^r(e_j^t)$, respectively[3]. COMEM consists of editing (i.e., training) and inference steps as follows:

- **Editing step**: Given a set of requested edits $\mathcal{E}$, COMEM performs the editing step:

$$\begin{aligned} \mathcal{M}_{\theta^*} &= \mathsf{PU}\left(\mathcal{E}, \mathcal{M}_\theta\right) \\ e'_1 \cdots e'_k &= \mathsf{NeighborEdits}\left(e_i, \mathcal{T}\right) \\ \mathcal{D}(e_i) &= \mathsf{ConstructDemo}\left(\mathcal{D}^t(e'_j)_{j=1}^{k}\right) \end{aligned} \tag{2}$$

  where PU is the parameter updating method of knowledge editing that injects a set of edits $\mathcal{E}$ to the language model $\mathcal{M}_\theta$ and returns the language model with the updated parameter θ^*, NeighborEdits is the retrieval function that returns the top-k training edits that are the most similar to the given requested edit e_i, and ConstructDemo is the demonstration construction component that selects a subset of the demonstrations in the top-k training edits, based on the *updating*-aware selection criteria.

- **Inference step**: Given a testing prompt $x = [s; r]$, COMEM performs the inference step:

$$\begin{aligned} \mathcal{D}(x) &= \mathsf{GetDemo}(x, \mathcal{D}(e_i)_{i=1}^{N}) \\ y &= \mathcal{M}_{\theta^*}([\mathsf{prompt}\left(\mathcal{D}(x)\right); x]) \end{aligned} \tag{3}$$

  where $\mathcal{D}(e_i)_{i=1}^{N}$ is a pre-constructed set of demonstrations corresponding to $\mathcal{E}$, GetDemo returns a set of online few-shot demonstrations for IKE, $\mathsf{prompt}\left(\mathcal{D}(x)\right)$ is the prompting function that linearizes the selected few-shot demonstrations $\mathcal{D}(x)$ using a proper

[1] The terminologies related to edit scope are based on those in (Mitchell et al., 2022b; Zheng et al., 2023)

[2] Here, $[s_i, r_i]$ refers to the natural language format consisting of s_i and r_i.

[3] Here, the demonstration types of copy, update and retrain correspond to the "requested," "paraphrased," and "neighborhood" prompts in the dataset, respectively.

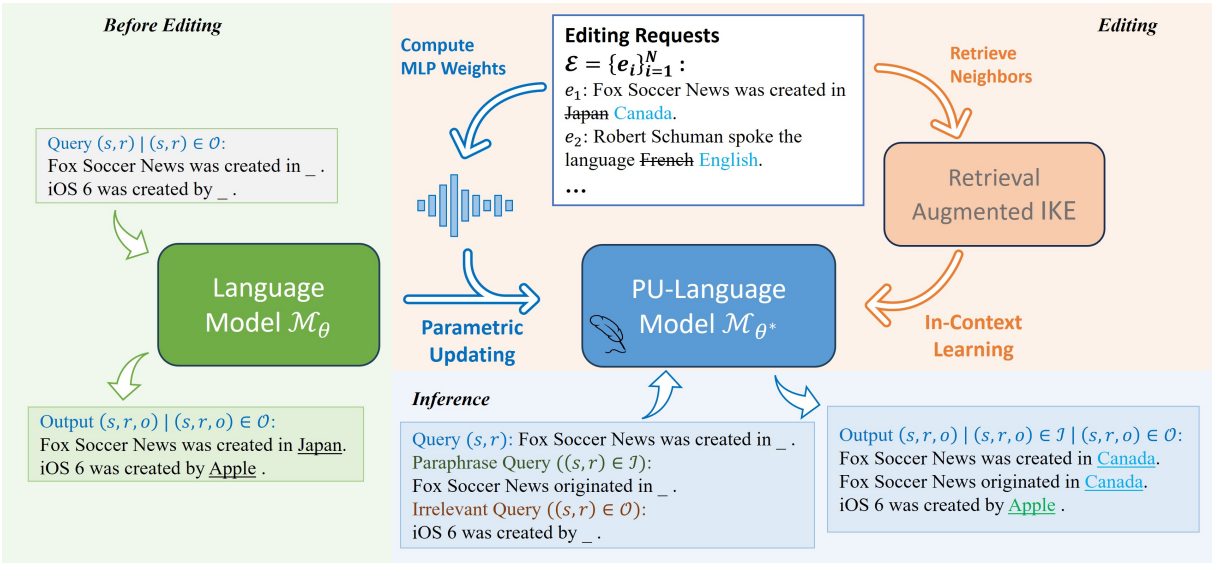

Figure 1: An overall illustration of COMEM: Given a set of massive edits $\mathcal{E}$ the parameter-updated language model is first obtained by using MEMIT (in Section 4.1), and the retrieval-augmented IKE is subsequentially performed (in Section 4.2) to combine the effects of parameter updating and IKE in a complementary manner. During the editing step, retrieval-augmented IKE constructs updating-aware demonstrations consisting only of update and remain types, based on a set of neighbors in the training edits of each requested edit e_i. During the inference step, the test query (s, r) is provided, COMEM retrieves the requested edit stored during the editing step by matching with (s, r), obtain its associated demonstrations, which are concatenated with the test prompt of (s, r) being fed into $\mathcal{M}^*$, and then finally predicts the target objects as required in $\mathcal{E}$, while retaining other non-edited knowledge (i.e., *"iOS 6 was created by Apple"*).

prompting template, $[\text{prompt}\,(\mathcal{D}(x))\,; x]$ is the concatenated prompt that consists of the demonstrations and testing prompt x, and y is the predicted object returned by COMEM given x.

4.1 Parameter Updating Method: MEMIT

For PU, the parameter updating method, we employ the MEMIT proposed by (Meng et al., 2022b), which involves rewriting local model parameters across a range of layers. The detailed description of MEMIT is presented in Appendix A.

4.2 Retrieval Augmented In-Context Knowledge Editing

Given the parametric updated model $\mathcal{M}_{\theta*}$, we perform the retrieval-augmented IKE, which consists of NeighborEdits and ConstructDemo for the editing step, and GetDemo for the inference step.

4.2.1 Updating-aware Demonstration Construction

Unlike IKE, which employs 32 demonstrations for "copy," "update," and "retain" with a ratio of 1:3:4 (where a copy-type demonstration duplicates a new edit request or fact exactly, update-type demonstrations use paraphrased expressions for the query part, and retain-type demonstrations specify the query and answer parts from an unrelated fact, as shown in Appendix F), we propose updating-aware demonstration construction for ConstructDemo in our COMEM setting. Here, IKE is subsequently applied to the parameter-updated language model $\mathcal{M}_{\theta*}$, rather than being used solely as a standalone editing method.

The underlying assumption is that once parameter updating is applied, $\mathcal{M}_{\theta*}$ is likely equipped with a proper level of editing capabilities, in terms of efficacy, generalization, and specificity. When applying IKE on $\mathcal{M}_{\theta*}$, the language model is updated to somehow handle properly in-scope and out-of-scope examples, unlike the original setting of IKE in the study by (Zheng et al., 2023) based on the fully unedited status.

In a preliminary experiment, we found that the use of copy-type demonstrations did not improve editing capabilities under the COMET setting. We present the experimental results in Section 6.2, where we gradually reallocated the demonstrations from copy-type to retain-type, showing incremental performance improvements. Furthermore, increasing the number of copy-type demonstrations has no advantages in performance.

Because the effect of ICL is limited by the maximum input length of the language model, we would like to construct more impactful demonstrations by adding non-copy-type demonstrations from more training edits which are similar to the current given edit.

The updating-aware demonstration construction consists of NeighborEdits and ConstructDemo.

Dense Retrieval for Finding Neighbor Edits

We use dense retrieval for NeighborEdits based on the cosine similarity between the training edit e_j^T and the given requested edit e_i. More precisely, suppose $\mathcal{M}_{sent}$ represents an additional sentence encoder, where $\mathcal{M}_{sent}(s) \in \mathbb{R}^d$ is the sentence vector for a given sentence s. For notation convenience, given an edit $e = (s, r, o)$, $\mathcal{M}_{sent}(e) = \mathcal{M}_{sent}([s; r; o])$ where $[s; r; o]$ is the natural language format that concatenates s, r, and o using a proper verbalizing template. The similarity between $e = (s, r, o)$ and $e' = (s, r, o)$ is defined as follows:

$$sim(e, e') = cos(\mathcal{M}_{sent}(e), \mathcal{M}_{sent}(e')) \quad (4)$$

For a given edit $e_i \in \mathcal{E}$, $\mathsf{NeighborEdits}\,(e_i, \mathcal{T})$ is defined as follows:

$$top\text{-}k\left\{(e_j^t, sim(e_i, e_j^t))\right\}_{j=1}^{M} \quad (5)$$

where $top\text{-}k$ is the operator for selecting the top-k elements given a set of pairs of objects and their associated similarities. For $\mathcal{M}_{sent}$, we deploy the pre-trained sentence encoder (Reimers and Gurevych, 2019).

Constructing Demonstration of Update and Retain Types

Once we have $\{e'_1 \cdots e'_k\} \in \mathcal{T}$ using NeighborEdits, $\mathsf{ConstructDemo}\left(\mathcal{D}^t(e'_j)_{j=1}^k\right)$ construct a set of demonstrations by selecting m update and n remain-type demonstrations in $\mathcal{D}^u(e'_j)$ and $\mathcal{D}^r(e'_j)$, respectively for e'_j. As a result, we have $k(m + n)$ demonstrations for each requested edit e_i, and $N \times k(m + n)$ demonstrations in $\mathcal{D}$ *in total* for the massive edit request in $\mathcal{E}$.

4.2.2 Retrieval-augmented Inference Step

Given a test prompting $q = (s, r)$, we first need to obtain its corresponding $e_q = (s, r, o) \in \mathcal{E}$. We devised a three-step search process comprising matching and retrieval. Firstly, 1) match both the subject s and relation r part of q in in $\mathcal{E}$. If matched, it returns the matched fact as the corresponding e_q. Otherwise, goes to the second step: 2) match the subject s in the memory. if a fact can be "uniquely" matched, the matched fact becomes the required e_q. If there exist multiple facts matched, goes to the third step: 3) perform the dense retrieval by ranking a set of the subject-matched facts. The best-matched fact is the query-matched fact e_q. Otherwise, it returns the "null", i.e., $\mathsf{Ret}(q) = e_q = \emptyset$. For any queries that failed to match its e_q, we also perform dense retrieval on the whole editing requests set $\mathcal{E}$ to assign the top-1 ranked fact as e_q.

After obtain the e_q, GetDemo returns the set of the associated $k(m + n)$ demonstrations for e_q, defined as follows:

$$\mathsf{GetDemo}(x, \mathcal{D}(e_i)_{i=1}^N) = \mathcal{D}(e_q) \quad (6)$$

The resulting demonstrations are further concatenated with the test prompt q to finally predict an output by COMEM.

5 Experiments

5.1 Dataset and Metrics

We first evaluate COMEM on **Zero-Shot Relation Extraction** (zsRE, Levy et al. (2017)) dataset with 10,000 knowledge edits following (Cao et al., 2021; Mitchell et al., 2022a; Meng et al., 2022b). After processing the data format, each test sample has one factual statement and its paraphrase, and one natural question that is irrelevant to the factual statement (see the example in Appendix E).

In this dataset, the metric **Efficacy** measures the editing accuracy:

$$\mathbb{E}[o^* = \text{argmax}\mathcal{P}_{\mathcal{M}^*}((s, r))]. \quad (7)$$

Paraphrase measures the same accuracy on paraphrase prompt (i.e., in-scope examples):

$$\mathbb{E}[o^* = \text{argmax}\mathcal{P}_{\mathcal{M}^*}((s, r))], \quad (8)$$

where $p(\cdot)$ denote the paraphrase of prompt. **Specificity** is the model's maximum probability accuracy for unrelated questions that should not be edited (i.e., out-of-scope examples).

$$\mathbb{E}[o = \text{argmax}\mathcal{P}_{\mathcal{M}^*}((s, r))], \quad (9)$$

where $u(\cdot)$ denote the editing-irrelevant statement. The **Score** is the harmonic mean of these three metrics and reflects the integrated performance of the model.

We also test our method on the **CounterFact** dataset (Meng et al., 2022a) following (Meng

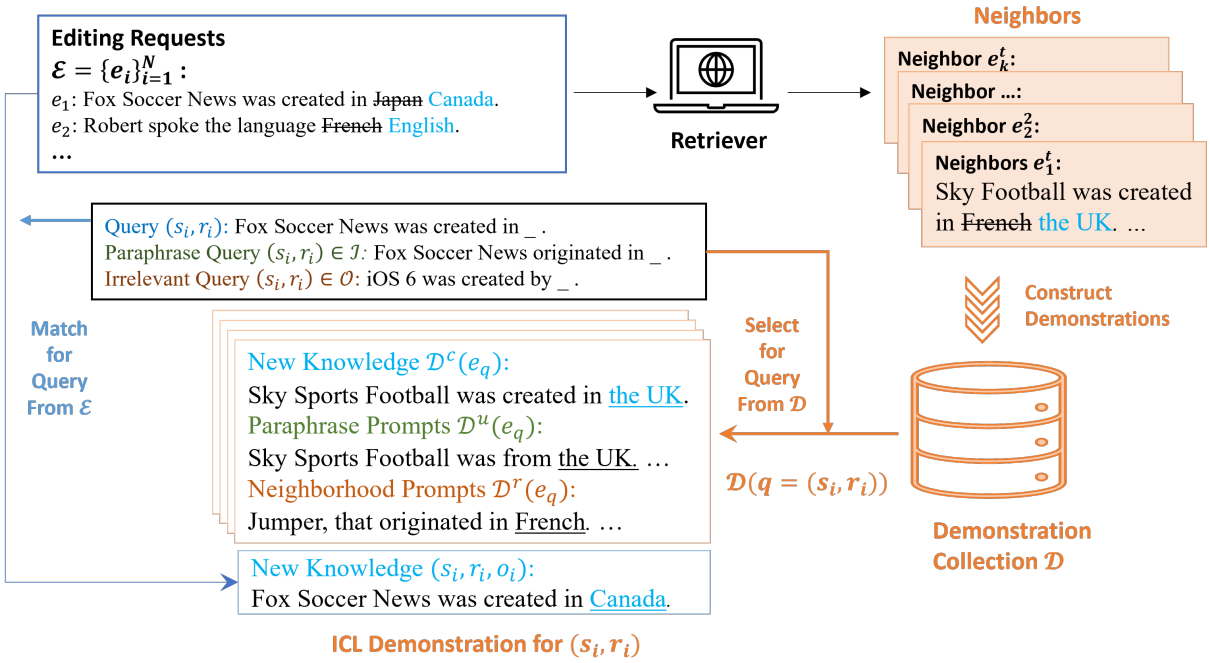

Figure 2: Illustration of retrieval augmented IKE for constructing updating-aware demonstrations. Given the requested edit e_i, dense retrieval is initially performed to identify the top-k neighbor edits in the training sets, which are most similar to the e_i. Demonstrations of m update and n retrain-types for each neighbor edit are then selected to create $k(m+n)$ demonstrations for $\mathcal{D}(e_i)$. During the inference step, a new query (s, r) is provided and the retrieval is performed by selecting $e_q = (s, r, o)$ where the subject and relation elements are matched. Finally, $\mathcal{D}(e_q)$ are provided as online demonstrations for a query (s, r).

et al., 2022a,b; Zheng et al., 2023), which contains 21,919 samples, each containing factual statements, two paraphrases of the statements, and 10 neighbor prompts irrelevant to the fact. The detailed format is presented in Appendix E. Similar to the metrics of Efficacy, Paraphrase and Specificity of zsRE, the **Efficacy Score (ES)**, **Paraphrase Score (PS)**, and **Neighborhood Score (NS)** are computed for editing accuracy. We also report the mean difference (magnitude) terms: **Efficacy Magnitude (EM)**, **Paraphrase Magnitude (PM)**, and **Neighborhood Magnitude (NM)**, which measure the *significance* of editing. A detailed definition is provided in Appendix C. The aggregated **Score (S)** is the harmonic mean of **ES**, **PS**, and **NS**.

Implementation details are provided in Appendix B.

5.2 Baselines

We use the GPT-J (6B) (Wang and Komatsuzaki, 2021) and GPT-NeoX (20B) (Black et al., 2022) models, commonly used in related works as backbone models, and compare COMEM with existing knowledge-editing works:

- **FT**, the naive GPT-J model fine-tuned on the edit facts using early stop to prevent overfitting and weight decay to prevent forgetfulness following (Meng et al., 2022b).
- **MEND** (Mitchell et al., 2022a), a learning based method that predicts weight changes using hyper-networks.
- **ROME** (Meng et al., 2022a), a direct parametric updating method that rewrites key-value pairs in MLP layers, edits single knowledge at a time, and must be performed iteratively for multiple edits.
- **MEMIT** (Meng et al., 2022b), parametric updating method that can edit massive amounts of knowledge simultaneously, and scale up to thousands of knowledge edits for the GPT-J (6B) or larger models.
- **IKE** (Zheng et al., 2023), a pure In-Context Learning based method that use three kinds of designed demonstrations ("*copy*," "*update*," and "*retain*") as prompt to steer the language models prediction.
- **PMET** (Li et al., 2023) is an optimized parametric multiple knowledge editing work that

simultaneously optimizes the hidden states of multi-head self attention (MHSA) and feed-forward network (FFN) layers and precisely updates the FFN weights.

6 Results and Discussion

In this section, we present the experimental results for massive knowledge editing tasks on the zsRE (Levy et al., 2017) and CounterFact (Meng et al., 2022a) datasets, comparing COMEM with the recently proposed baselines. Additionally, we conduct discussions and analyses based on ablation studies.

6.1 Main Results

Results on zsRE. Table 1 presents the comparison results of COMEM and other baselines in terms of the *Efficacy*, *Paraphrase*, and *Specificity* metrics. As shown in the table, COMEM achieves the best results on all metrics and exhibits significant improvement in the aggregated *Score* (harmonic mean of *Efficacy*, *Paraphrase* and *Specificity*). Massive knowledge editing methods, such as MEMIT (Meng et al., 2022b) and PMET (Li et al., 2023) can provide a relatively strong editing capability across all metrics of Efficacy, Generalization (in *Paraphrase*), and Specificity, while leaving room for further improvements. The pure IKE (Zheng et al., 2023) can also achieve the best scores for Efficacy and Paraphrase, but shows relatively weak Specificity, comparing to MEMIT.

Results on CounterFact. Table 2 shows the comparison results of COMEM and baseline methods in terms of accuracy (**ES**, **PS**, and **NS**) and magnitude terms (**EM**, **PM**, and **NM**) on this dataset. It can be observed that the proposed COMEM achieves the best overall performance. For the results based on GPT-J, similar to the cases of zsRE, MEND (Mitchell et al., 2022a) and ROME (Meng et al., 2022a) are weak in terms of Efficacy and Generalization of massive knowledge editing, whereas MEND achieved the best Neighborhood Score. Interestingly, the fine-tuned model performs well in terms of Efficacy and Generalization but also deteriorates severely in Specificity. There are no significant differences between the parameter updating methods (Meng et al., 2022b; Li et al., 2023) and IKE (Zheng et al., 2023) in terms of Efficacy, including COMEM. However, there are still considerable gaps in the Generalization of parametric methods compared to IKE and COMEM. IKE shows strong performances of Efficacy and Generalization, but is weak in Specificity.

The pure IKE method exhibits a drop in Specificity with massive editing compared to fewer edits as it achieves better *Neighborhood Score* on 2,000 edits test (77.0 on the original CounterFact in the IKE's paper (Zheng et al., 2023) and 67.6 on the filtered CounterFact[4] in our test). Besides the significant impact brought by unfiltered conflicting samples, this performance drop is primarily caused by that the retrieval corpus size diminishes as more data samples are allocated to the test set, resulting in a smaller retrieval search space, and that IKE heavily relies on the quality of demonstrations constructed from the retrieved neighbors.

We further tested COMEM on a larger model, GPT-NeoX 20B, which shows noticeable improvements in Generalization and Specificity compared to MEMIT and PMET, this also indicates that COMEM can effectively scale to larger language models. The trade-off between "edit"-wise metrics (Efficacy and Generalization) and "retain"-wise metric (Specificity) can also be observed from the comparison of COMEM* (without parameter updating) and COMEM settings. Performing ICL editing on on a model without parameter updates results in lower Generalization but higher Specificity compared with that parameter-updated model.

Remarkably, COMEM leverages both the foundational editing capability of the parameter updating method and the augmentation capabilities of IKE, thereby achieving state-of-the-art performance. Examples of selected edits are presented in Appendix G.

6.2 Demonstration Analysis for Parametric Updated Model

In this section, we demonstrate that a parametric updated model does not need additional *"Efficacy Demonstrations"* but requires more for *Specificity* in the In-Context Learning stage.

We first keep the number of total demonstrations fixed and redistribute *"copy"* to *"retain"* demonstrations, since IKE lost some *Specificity* under 10,000 edits setting. Table 3 demonstrates that this redistribution improved the **Neighborhood Score** without compromising *Efficacy* and *Generalization*, leading to an overall improvement in performance.

[4]We use the CounterFact dataset which filtered to remove the samples that violate multiple knowledge editing paradigm as described in **Section 4.3**, the filtered dataset is also referred to as multi-CounterFact.

Method	Score ↑	Efficacy ↑	Paraphrase ↑	Specificity ↑
GPT-J	26.4	26.4	25.8	27.0
FT	42.1	69.6	64.8	24.1
MEND	20.0	19.4	18.6	22.4
ROME	2.6	21.0	19.6	0.9
MEMIT	50.7	96.7	89.7	26.6
IKE	40.6	**99.3**	**99.9**	18.6
PMET	<u>51.0</u>	96.9	90.6	<u>26.7</u>
COMEM	**60.6**	<u>98.1</u>	<u>96.0</u>	**34.9**

Table 1: Performance comparison of GPT-J (6B) with 10,000 knowledge edits on the zsRE dataset. Column-wise best are in bold, second best are underlined.

Method	Score	Efficacy		Generalization		Specificity	
	S ↑	ES ↑	EM ↑	PS ↑	PM ↑	NS ↑	NM ↑
GPT-J	20.47	14.66	-7.40	15.06	-7.50	83.97	7.65
FT	63.54	<u>99.91</u>	**98.24**	88.14	48.65	38.67	-8.22
MEND	25.23	17.61	-12.19	20.10	-11.34	**80.83**	12.55
ROME	49.92	49.36	-0.03	49.51	-0.09	50.92	0.09
MEMIT	85.71	99.10	87.85	88.33	38.02	<u>73.59</u>	4.64
IKE	84.88	**99.98**	92.86	<u>96.29</u>	<u>67.37</u>	66.88	<u>25.19</u>
PMET	<u>86.20</u>	99.50	-	92.80	-	71.40	-
COMEM	**88.09**	99.87	<u>94.88</u>	**96.42**	**71.00**	73.14	**35.87**
GPT-NeoX	23.33	16.63	-9.1	17.77	-8.17	81.94	8.85
MEMIT	82.00	97.20	-	82.20	-	70.80	-
PMET	84.30	**98.40**	-	89.40	-	70.30	-
COMEM*	<u>87.11</u>	<u>98.06</u>	**90.24**	<u>89.55</u>	<u>56.69</u>	**76.48**	**42.05**
COMEM	**87.21**	98.00	<u>86.16</u>	**91.37**	**62.84**	<u>75.46</u>	<u>39.72</u>

Table 2: Performance comparison of GPT-J (6B) and GPT-NeoX (20B) with 10,000 knowledge edits on the CounterFact dataset. COMEM* represents the results obtained without parameter updating in the GPT-NeoX model. Column-wise best are in bold, second best are underlined.

We then test weather the *"copy"* demonstrations still have significance for the parametric updated models and found that further increase the number of "copy" demonstrations do not improve the overall performance, as strong *Efficacy* have been pre-provided by parameter updating, the results as shown in Table 3.

C/U/R	S ↑	ES ↑	PS ↑	NS ↑
4/12/16	83.53	99.99	98.64	63.43
2/12/18	84.25	99.99	98.51	64.70
0/12/20	84.80	99.98	98.53	65.70
2/12/20	84.84	100	98.61	65.71
4/12/20	84.78	100	98.54	65.63
8/12/20	84.78	100	98.52	65.64

Table 3: Various demonstration distributions applied for parametric updated model. The demonstration format used in this test is adopted from IKE, where **C**, **U**, and **R** denote the number of *"copy"*, *"update"*, and *"retain"* demonstrations.

6.3 Ablation on zsRE

We conduct ablation experiments on zsRE to demonstrate the necessity of using *Parametric Updating* and IKE-based input augmentation and to examine the impact of using different numbers of neighbors (i.e., the number of demonstrations) on the editing performance.

Table 4 show that without the parameter updating method, the model struggles in showing the effectiveness on Generalization and Specificity. Without IKE, the use of the parameter updating leads to the degradation of the overall editing performance , particularly in Generalization.

As we increase the number of nearest neighbors k to construct IKE demonstrations, the performances improve, although the Efficacy and Gen-

eralization reach optimal performance only when $k = 8$.

Method	S ↑	ES ↑	PS ↑	NS ↑
- w/o PU	46.8	100	99.9	22.7
- w/o ICL	50.7	96.7	89.7	26.6
- $k = 8$	57.3	99.7	97.6	31.2
- $k = 16$	59.4	99.8	97.6	33.1
- $k = 24$	60.6	98.1	96.0	34.9

Table 4: Ablation study on zsRE. k denotes the number of the neighbors, w/o PU and w/o ICL correspond to the runs without parameter updating (w/o PU) method and IKE, respectively. The run of w/o PU is performed under the setting of $k = 24$. **ES**, **PS**, and **NS** refer to the model's *Efficacy*, *Generalization*, and *Specificity*, respectively.

6.4 Ablation on CounterFact

Table 5 presents an ablation study on CounterFact. Both the parameter updating method and IKE are effective for *Specificity*, but suffer from a notable drop in *Generalization*; Parameter updating provides slightly better Efficacy and Generalization than IKE.

In contrast to the results on zsRE, IKE achieves the best Neighborhood Score on CounterFact. The results are mainly because the CounterFact dataset can provide a greater number of retrain-type demonstrations (i.e., neighborhood prompts) to improve Specificity.

COMEM improves Generalization significantly and Efficacy slightly, while COMEM leads to a slight decease in Specificity. Increasing the number of demonstrations will likely help to regain Specificity without compromising other aspects.

Method	S ↑	ES ↑	PS ↑	NS ↑
- w/o PU	85.29	99.60	85.64	74.31
- w/o ICL	85.71	99.10	88.33	73.59
- $k = 3$	85.72	99.95	96.43	68.39
- $k = 4$	87.08	99.93	96.43	71.05
- $k = 5$	88.09	99.87	96.42	73.14

Table 5: Ablation study on CounterFact. Similar to the zsRE dataset, k is the number of nearest neighbors used for ICL demonstration construction, the test of without introducing parametric updating (w/o PU) was conducted under the setting of $k = 5$.

6.5 Demonstration Analysis on Query Prompt

We also observed that pre-appending a new knowledge demonstration before the query sequence (used in IKE) tends to excessively bias the model towards predicting new facts, resulting in a notable deterioration in *Specificity*, as shown in Table 6. Hence, for any query prompt, we utilize the original sequence without any additional context. Table 9 in Appendix F provides an example of our demonstration.

Method	S ↑	ES ↑	PS ↑	NS ↑
zsRE				
- $w/$ Pre	56.1	99.7	99.2	30.0
- w/o Pre	60.6	98.1	96.0	34.9
CounterFact				
- $w/$ Pre	86.28	99.28	97.03	69.48
- w/o Pre	88.09	99.87	96.42	73.14

Table 6: Comparison of pre-appending the new knowledge demonstration before the query prompt or not.

7 Conclusion

In this study, we proposed COMEM, a unified framework of parameter updating and IKE for massive knowledge editing tasks. Extensive experiments on the zsRE and CounterFact datasets show that COMEM led to state-of-the-art overall performance, outperforming most existing knowledge editing methods. Further analysis confirmed that the use of our designed updating-aware in-context learning demonstration pattern boosted the model generalization without compromising its high efficacy and specificity.

In future work, we aim to explore how to parameterize in-context learning demonstrations into the language model to avoid the decrease in inference efficiency caused by lengthy input prompts, ultimately striving for more concise and efficient knowledge editing.

Limitations

Our work optimizes based on In-Context Learning after parametric rewriting, yet ICL cannot achieve permanent or long-term model knowledge updates. This means that currently the optimized part cannot avoid lengthy demonstration inputs, and concatenating such demonstrations every time the model restarts is inefficient. Therefore, achieving permanent or long-term optimal knowledge editing performance requires exploring methods to parameterize the ICL demonstrations. This would also allow the final model to operate without lengthy input prompts, thereby enhancing inference efficiency,

which is one of our future directions. Additionally, current models primarily operate on data samples in tuple form like $(subject, relation, object)$, whereas real-world natural language comes in more diverse and complex forms. Exploring weather the current work can generalize to universal text formats is also an important future task.

Acknowledgement

This work was supported by Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2023-00216011, Development of artificial complex intelligence for conceptually understanding and inferring like human). Shanbao Qiao and Xuebing Liu were also supported by China Scholarship Council (CSC).

References

Oshin Agarwal and Ani Nenkova. 2022. Temporal effects on pre-trained models for language processing tasks. *Transactions of the Association for Computational Linguistics*, 10:904–921.

Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, USVSN Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of the ACL Workshop on Challenges & Perspectives in Creating Large Language Models*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models.

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.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273.

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

Isabel O. Gallegos, Ryan A. Rossi, Joe Barrow, Md Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen K. Ahmed. 2023. Bias and fairness in large language models: A survey.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12).

Angeliki Lazaridou, Adhi Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Tomas Kocisky, Sebastian Ruder, et al. 2021. Mind the gap: Assessing temporal generalization in neural language models. *Advances in Neural Information Processing Systems*, 34:29348–29363.

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

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*.

Adam Liška, Tomáš Kočiský, Elena Gribovskaya, Tayfun Terzi, Eren Sezener, Devang Agrawal, Cyprien de Masson d'Autume, Tim Scholtes, Manzil Zaheer, Susannah Young, Ellen Gilsenan-McMahon Sophia Austin, Phil Blunsom, and Angeliki Lazaridou. 2022. Streamingqa: A benchmark for adaptation to new knowledge over time in question answering models. *arXiv preprint arXiv:2205.11388*.

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.

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 *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 15817–15831. PMLR.

Yasumasa Onoe, Michael Zhang, Eunsol Choi, and Greg Durrett. 2022. Entity cloze by date: What LMs know about unseen entities. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 693–702, Seattle, United States. Association for Computational Linguistics.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. *arXiv preprint arXiv:2005.04611*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

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.

Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Boyd-Graber, and Lijuan Wang. 2023. Prompting gpt-3 to be reliable. In *International Conference on Learning Representations (ICLR)*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, and Tengyu Ma. 2023. Larger language models do in-context learning differently.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

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*.

A Detailed Process of MEMIT

The MLP weights in a Transformer (Vaswani et al., 2017) are W, which can be operated as a key-value store, where $WK \approx V$, $K = [k_1|k_2|...]$ and $V = [v_1|v_2|...]$. Given the requested edits $\mathcal{E} = \{(s_i, r_i, o_i)\}$, language model $\mathcal{M}_\theta$, layers to edit $\mathcal{L} = \{L_1, L_2, ..., L_l\}$, and pre-cached covariance constant C^L of k computed from Wikipedia samples (Meng et al., 2022a). For each $(s_i, r_i, o_i) \in \mathcal{E}$, a target vector z_i is computed using:

$$z_i \leftarrow h_i^{L_l} + \delta_i, \tag{10}$$

where δ_i is optimized by:

$$\delta_i \leftarrow \arg\min_{\delta_i} \frac{1}{P} \sum_{j=1}^{P} \xi_i$$

$$\xi_i = -\log \mathbb{P}_{\mathcal{M}(h_i^{L_l} += \delta_i)}[o_i | x_j \oplus (s_i, r_i)] \tag{11}$$

For each editing layer $L \in \mathcal{L}$, the hidden state is updated by:

$$h_i^L \leftarrow h_i^{L-1} + a_i^L + m_i^L \tag{12}$$

where a and m denote the *"attention"* and *"MLP"* contributions computed from previous layers in the Transformer (Vaswani et al., 2017) model, respectively. On the current layer, for each $(s_i, r_i, o_i) \in \mathcal{E}$, the MLP key is updated as:

$$k_i^L \leftarrow k_i^L = \frac{1}{P} \sum_{j=1}^{P} k(x_j + s_i) \tag{13}$$

where x_j represents random prefixes that aid generalization across contexts. The distributed residual ϕ over remaining layers is computed as

$$\phi_i^L \leftarrow \frac{z_i - h_i^{L_l}}{l - idx(L) + 1} \tag{14}$$

where $idx(L)$ denote the number index of L. Thus in this layer $k^L = \{k_i^L\}$ and $\phi^L = \{\phi_i^L\}$.

To update the MLP weights in the editing layers, for each layer $L \in \mathcal{L}$, the adding weight is computed as:

$$\Delta^L \leftarrow \phi^L {k^L}^\intercal (C^L + k^L {k^L}^\intercal)^{-1}, \tag{15}$$

Finally, in the current layer L the MLP weights are updated as

$$W^L \leftarrow W^L + \Delta^L, \tag{16}$$

After the above updating is performed on all the editing layers, the parameter updated model $\mathcal{M}_{\theta^*}$ can be obtained.

B Implementation Details

We use GPT-J (6B) (Wang and Komatsuzaki, 2021) and GPT-NeoX (20B) (Black et al., 2022) as backbone language models to ensure the maximum number of comparable cases with related work. During the parametric editing process, we edit the MLP weights in layers $[3, 4, 5, 6, 7, 8]$ of GPT-J and layers $[21, 22, 23, 24, 25]$ of GPT-NeoX.

For the zsRE (Levy et al., 2017) dataset, we extract 10,000 samples as the test set to perform massive knowledge editing and use the *sentence-transformer* toolkit to retrieve k nearest neighbors from the remaining set (172,282 samples) of data. The best result was obtained by setting $k = 24$, resulting in 48 demonstrations for each test sample, where each neighbor provides one paraphrase prompt ($m = 1$) and one irrelevant prompt ($n = 1$) used for the demonstration construction. In our experiments, larger k values result in the input sequence length of most samples exceeding the maximum input lengths of GPT-J (6B). We first run parametric updating on the test set following (Meng et al., 2022b), and then use the edited model to perform In-context Learning.

We tested IKE (Zheng et al., 2023) on zsRE with the same demonstration setting as in their paper: retrieving top 32 nearest neighbors and assigning the usage of *factual statement*, *paraphrase prompt*, and *neighborhood prompt* for "*copy*," "*update*," "*retain*" demonstrations with a ratio of 1:3:4. Because other baselines were tested by the previous works on this dataset, we present statistics from their respective studies (Meng et al., 2022b; Li et al., 2023).

For the CounterFact (Meng et al., 2022a) dataset, the original dataset comprises 21,919 samples, however, there are some samples that share the same prefix (s, r) used for editing other new facts, thereby being conflicted each other. After filtering these conflicted edits, the resulting dataset comprises a total of 20,877 samples as in (Meng et al., 2022b). Given that each data sample in this dataset includes 2 paraphrase prompts ($m = 2$) and 10 neighborhood (irrelevant) prompts ($n = 10$), the In-Context Learning prompt consists of $k * 12$ demonstrations. Hence, under our optimal setting, the number of demonstrations for each test sample is 60.

To obtain precise results and the *magnitude* term of the baselines, we rerun IKE (Zheng et al., 2023) on the filtered dataset for 10,000 samples under the same setting as in their paper, and retested other baselines based on (Meng et al., 2022b)'s repository. However, for the PMET (Li et al., 2023), we failed to reproduce the experiment because of GPU limitations; thus, we directly adopted their results in this study.

All our experiments were conducted on NVIDIA A6000 GPUs.

C Detailed Definition of Evaluation Metrics on CounterFact

Accuracy Terms:

Efficacy Score (ES):

$$\mathbb{E}[\mathcal{P}_{\mathcal{M}^*}(o^*|(s,r)) > \mathcal{P}_{\mathcal{M}^*}(o|(s,r))], \tag{17}$$

Paraphrase Score (PS):

$$\mathbb{E}[\mathcal{P}_{\mathcal{M}^*}(o^*|p(s,r)) > \mathcal{P}_{\mathcal{M}^*}(o|p(s,r))], \tag{18}$$

Neighborhood Score (NS):

$$\mathbb{E}[\mathcal{P}_{\mathcal{M}^*}(o^*|u(s,r)) < \mathcal{P}_{\mathcal{M}^*}(o|u(s,r))]. \tag{19}$$

Magnitude Terms:

Efficacy Magnitude (EM):

$$\mathbb{E}[\mathcal{P}_{\mathcal{M}^*}(o^*|(s,r)) - \mathcal{P}_{\mathcal{M}^*}(o|(s,r))], \tag{20}$$

Paraphrase Magnitude (PM):

$$\mathbb{E}[\mathcal{P}_{\mathcal{M}^*}(o^*|p(s,r)) - \mathcal{P}_{\mathcal{M}^*}(o|p(s,r))], \tag{21}$$

Neighborhood Magnitude (NM):

$$\mathbb{E}[\mathcal{P}_{\mathcal{M}^*}(o|u(s,r)) - \mathcal{P}_{\mathcal{M}^*}(o^*|u(s,r))]. \tag{22}$$

D Extended Comparison of Performance with In-Context Learning Knowledge Editing

For a more detailed comparison with the pure In-Context Learning method, we tested the performance of IKE (Zheng et al., 2023) under the same number of demonstrations as in our experiments ($k = 3, 4, 5$). Due to the change in the number of demonstrations, we attempted to maintain the ratio of demonstrations (1:3:4) used for "*copy*," "*update*" and "*retain*" in IKE as much as possible to allocate the additional demonstrations.

The results are summarized in Table 7. COMEM is slightly inferior in **Efficacy** and **Paraphrase Scores**, but exhibits a noticeable advantage in **Neighborhood Score**. A higher aggregated score **S** indicates that the proposed COMEM has a better overall performance.

Method	d_n	**S** ↑	**ES** ↑	**PS** ↑	**NS** ↑
IKE	36	85.32	100	96.30	67.68
COMEM	36	85.72	99.95	96.43	70.58
IKE	48	86.22	100	97.22	68.93
COMEM	48	87.08	99.93	96.43	71.05
IKE	60	87.14	99.98	97.32	70.68
COMEM	60	88.09	99.87	96.42	73.14

Table 7: Comparison of COMEM with IKE under the same demonstration quantity. d_n denote the number of total demonstration, where $36, 48, 60$ correspond to our experiments with $k = 3, 4, 5$.

E Data Structure

Structure of CounterFact dataset:


```
{
  "case_id": 0,
  "requested_rewrite": {
    "prompt": "The mother tongue of  is",
    "target_new": “str": "English",,
    "target_true": "str": "French",,
    "subject": "Danielle Darrieux"
  },
  "paraphrase_prompts": [
    "Danielle Darrieux, a native",
    "Danielle Darrieux spoke the language"
  ],
  "neighborhood_prompts": [
    "The native language of Montesquieu is",
    "The native language of Raymond Barre is",
    "Jacques is a native speaker of",
    ... (10 prompts in total)
  ],
  "attribute_prompts": [
    "The mother tongue of Douglas Adams is",
    ... (10 prompts in total)
  ],
  "generation_prompts": [
    "Danielle Darrieux's mother tongue is",
    ... (10 prompts in total)
  ]
}
```


Structure of processed zsRE dataset:


```
{
  "case_id": 0,
  "requested_rewrite": {
    "prompt": "What university did {} attend?",
    "subject": "Watts Humphrey",
    "target_new":
      "str": "Illinois Institute of Technology"
    "target_true":
      "str": "<|endoftext|>"
  },
  "paraphrase_prompts": [
    "What university did Watts Humphrey take part in? "
  ],
  "neighborhood_prompts": [
    "prompt":
      "nq question: who played desmond doss father?",
    "target": " Hugo"
  ]
}
```


F Example of ICL Demonstration

Table 8 shows examples of IKE's ICL demonstrations, and Table 9 displays our demonstrations.

Type	**Demonstration**
copy	New Fact: Sky Football was created in Canada Prompt: Sky Football News was created in Canada
update	New Fact: Sky Football was created in Canada Prompt: Sky Football News originated in Canada
retain	New Fact: Sky Football was created in Canada Prompt: iOS 6 was created by Apple
append	Fox News was created in Canada
query	New Fact: Fox News was created in Canada Prompt: Fox News was created in?

Table 8: Single example of IKE's (Zheng et al., 2023) demonstration.

Type	Demonstration
paraphrase	New Fact: Sky Football was created in Canada Prompt: Sky Football News originated in Canada
neighborhood	New Fact: Sky Football was created in Canada Prompt: iOS 6 was created by Apple
append	Prompt: Fox News was created in Canada.
query	Prompt: Fox News was created in?

Table 9: Single example of our demonstration.

G Output Examples of Model Outputs

Table 10 presents the output examples of GPT-J and COMEM, where GPT-J stores the original unedited knowledge and COMEM is post-edited. Blue font represents new editing knowledge, yellow font indicates original knowledge, red font denotes incorrectly predicted or ambiguous answers, and green font indicates successful retention of knowledge unrelated to the editing target. It can be observed that the unedited GPT-J is generally capable of outputting original knowledge but occasionally fails to generate correct answers. COMEM is successful in generating new edited knowledge without affecting other unedited knowledge and tends to provide concise answers directly.

Model	Type	Content or Output
Prompt	New Knowledge	The mother tongue of Danielle Darrieux is English.
	Old Knowledge	The mother tongue of Danielle Darrieux is French.
GPT-J (before editing)	Fact	The mother tongue of Danielle Darrieux is ___. The correct answer is: French.
	Paraphrase	Danielle Darrieux, a native French. Danielle Darrieux spoke the language ___. A. French B. English C. Spanish D. Italian
	Neighborhood (unrelated)	The native language of Montesquieu is French. Prompt: The native language Maurice Genevoix, speaker of French.
	Generation	Danielle Darrieux's mother tongue is ___. 1. 2. 3. 4. Where Danielle Darrieux is from, people speak the language of ___. The answer is: French.
COMEM (after editing)	Fact	The mother tongue of Danielle Darrieux is English.
	Paraphrase	Danielle Darrieux, a native English. Danielle Darrieux spoke the language English.
	Neighborhood (unrelated)	The native language of Montesquieu is French. Maurice Genevoix, speaker of French.
	Generation	Danielle Darrieux's mother tongue is English. Where Danielle Darrieux is from, people speak the language of English.

Table 10: Outputs of models on the CounterFact dataset.