
A³E: Towards Compositional Model Editing

Hongming Piao

City University of Hong Kong
hpiao6-c@my.cityu.edu.hk

Hao Wang

City University of Hong Kong
hao.wang@my.cityu.edu.hk

Dapeng Oliver Wu[†]

City University of Hong Kong
dapengwu@cityu.edu.hk

Ying Wei[†]

Zhejiang University
ying.wei@zju.edu.cn

Abstract

Model editing has become a *de-facto* practice to address hallucinations and outdated knowledge of large language models (LLMs). However, existing methods are predominantly evaluated in isolation, i.e., one edit at a time, failing to consider a critical scenario of compositional model editing, where multiple edits must be integrated and jointly utilized to answer real-world multifaceted questions. For instance, in medical domains, if one edit informs LLMs that COVID-19 causes “fever” and another that it causes “loss of taste”, a qualified compositional editor should enable LLMs to answer the question “What are the symptoms of COVID-19?” with both “fever” and “loss of taste” (and potentially more). In this work, we define and systematically benchmark this compositional model editing (CME) task, identifying three key undesirable issues that existing methods struggle with: *knowledge loss*, *incorrect preceding* and *knowledge sinking*. To overcome these issues, we propose A³E, a novel compositional editor that (1) *adaptively combines and adaptively regularizes* pre-trained foundation knowledge in LLMs in the stage of edit training and (2) *adaptively merges* multiple edits to better meet compositional needs in the stage of edit composing. Extensive experiments demonstrate that A³E improves the composability by at least 22.45% without sacrificing the performance of non-compositional model editing. The code is available at <https://github.com/piaohongming/A3E>.

1 Introduction

LLMs learn extensive knowledge from massive pre-training corpora and utilize the learned knowledge during inference to meet a wide range of tasks [1, 2, 3, 4]. Despite their impressive capabilities, especially as the pre-training data and model size scale [5], LLMs remain prone to factual hallucinations [6] and outdated knowledge [7]. These errors often emerge gradually after deployment and significantly impair system reliability, which necessitates timely and effective corrections. Conventional solutions such as re-training are not only time-consuming, but fine-tuning without access to pre-training data also poses a high risk of interfering with irrelevant knowledge. Therefore, model editing [8] has emerged as a promising approach that enables targeted updates to LLMs in a data-efficient manner (minimal reliance on pre-training data), localized scope (limited impact on irrelevant knowledge), and with high reliability (accurate correction of the targeted knowledge).

Efforts toward model editing center around improving reliability [9, 10], generalization [10, 11], and locality [8, 12], with further studies exploring editing ripple effects [13], bidirectional generalization [14], and its impact on downstream tasks [15, 16, 17, 18]. Progress along with these attempts is

[†]Corresponding authors.

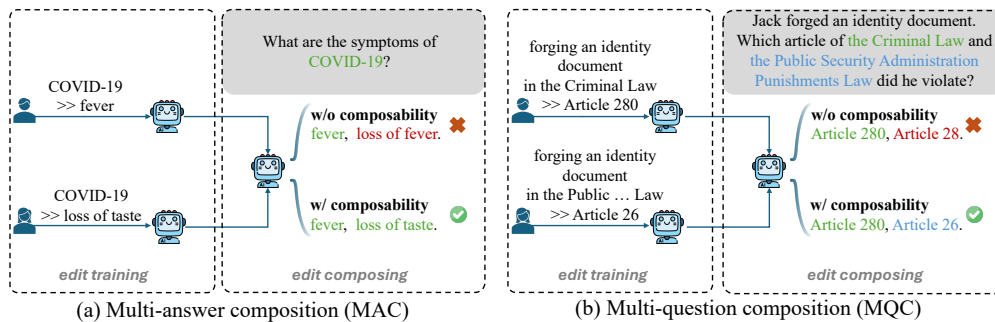


Figure 1: There are two stages in CME: *edit training* and *edit composing*. (a) An example for MAC: Multiple symptoms for a disease. (b) An example for MQC: Multiple laws involved in one crime.

unanimously evaluated under a *single-edit assumption*; that is, LLMs recall each atomic edit in isolation. For example, if an edit previously taught an LLM that COVID-19 causes “loss of taste”, evaluations only test whether this LLM can correctly recall this isolated fact, e.g., “I have loss of taste, do I have COVID-19?” Unfortunately, this evaluation paradigm diverges significantly from real-world demands. In many practical domains, such as medicine and law (as shown in Fig. 1), questions likely require access to multiple edits simultaneously. In Fig. 1(a), for instance, answering a multi-symptom diagnostic question requires composing two pieces of knowledge (multi-answer composition, MAC); if these facts were introduced through prior edits, LLMs must effectively leverage both. Similarly, in Fig. 1(b), questions may require reasoning over multiple pieces of edited knowledge to form a unified answer (multi-question composition, MQC). In this paper, we formally define and benchmark tasks that demand such compositional capability as Compositional Model Editing (CME) tasks. The success of CME hinges on two criteria: (1) in the *edit training* stage, individual edits must be performed in a way that guards their composability for downstream inference; (2) in the *edit composing* stage, question-related edits must be effectively merged.

As shown in Fig. 2, state-of-the-art editing methods suffer from *knowledge loss*, *incorrect preceding* and *knowledge sinking*, which comprehensively encompass all the failure cases we observed. MEND [9] and ROME [8] modify the entire matrix in the feed-forward network (FFN) through hypernetworks and closed-form solutions, respectively, which yields significant *knowledge loss* when composing different edits. AlphaEdit [19] exhibits a similar level of *knowledge loss*, likely because it prevents interference between edits by projecting each update into the null space of unrelated questions. However, in the MAC and MQC settings we target, where a single question depends on multiple edits, this orthogonality collapses and conflicts between edits still persist. WISE [20] alleviates *knowledge loss* by randomly masking parameter updates of each individual edit, though the likelihood of mask overlap grows significantly as the number of edits increases and thus composability degrades. GRACE [21], MELO [22], and T-patcher [23] replace hidden states or insert new key-value pairs in the FFN layer in a low-rank form. Such low storage overhead for each edit enables them to treat all edits as an external vector database attached to the original parameters. Despite reduced *knowledge loss*, all three methods suffer from the pronounced issue of *incorrect preceding*. Collectively, these limitations reveal a pressing gap, i.e., *how to train and compose different edits to improve edit composability?*

Our response to the question is the proposed A³E. Specifically, previous works [24, 25] have pointed out that the FFN layer in Transformer [26] is a key-value neural memory [27], where the down projection matrix stores robust and generalizable foundation knowledge pre-trained on large datasets. The composition of these foundation knowledge pieces forms multiple advanced knowledge needed in the generation process of LLMs. We hypothesize and empirically verify that leveraging the pre-trained foundation knowledge in the down projection matrix is sufficient for not only composing the new knowledge but also boosting composability. Thus, in order to *train edits with better composability*, we Adaptively combine the pre-trained foundation knowledge in the down projection matrix to reduce *knowledge loss*. By Adaptively regularizing the use of selected knowledge at the last answer token and non-label logits, we effectively alleviate *knowledge sinking* and *incorrect preceding*. In order to *compose edits to further enhance composability*, we utilize a vector database [21, 22] to filter out irrelevant edits in a question-wise way at inference and Adaptively merge relevant edits with proper combinations of pre-trained foundation knowledge. In summary, our contributions are four-fold.

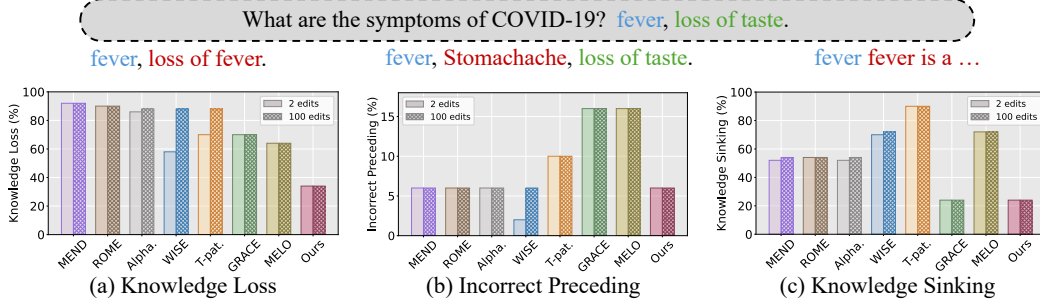


Figure 2: The illustration of *knowledge loss*, *incorrect preceding* and *knowledge sinking*. The text in blue, red, green represents the first correct answer, the wrong answer and the second correct answer. (a) *Knowledge loss*: Part of answers are not correctly generated. (b) *Incorrect preceding*: Wrong answers are generated before the correct answers. (c) *Knowledge sinking*: Context-irrelevant content are generated after prior answers; that is, the starting point for generating subsequent answer “loss of taste” is no longer the question itself (e.g., “What are the symptoms of COVID-19?”), but the question combined with the prior answers (e.g., “What are the symptoms of COVID-19? fever”).

- We are the first to define and benchmark the CME evaluation setup, paving the way for future model editors to support multi-edit reasoning that is crucial for real-world use cases.
- We conduct a systematic analysis of existing methods under CME, revealing three pivotal failure modes in composability, including *knowledge loss*, *incorrect preceding* and *knowledge sinking*.
- We develop a dual-stage framework, edit training (adaptive combining and regularizing) and edit composing (adaptive merging), that preserves the effectiveness of individual edits while enabling their synergistic integration during inference.
- Through extensive experiments on two datasets and four CME tasks, we improves the composability by at least 22.45%, without sacrificing the performance of non-compositional model editing.

2 Preliminaries

In this section, we first provide a formal definition of compositional model editing, and then demonstrate how to perform edit training and edit composing in FFN. Please refer to Tab. 3 for the definition of notations.

2.1 Compositional Model Editing (CME)

In model editing (ME), let $f_{\Theta} : \mathbb{X} \mapsto \mathbb{Y}$, parameterized by Θ , represents a model mapping the input x to the output $f_{\Theta}(x)$. Given a model Θ_0 before a single edit (x_e, y_e) , let $\mathcal{I}(x_e)$ denote the in-scope input with the same semantics as the edit question. Then the objective of ME is:

$$f_{\Theta_e} = \text{ME}(f_{\Theta_0}, x_e, y_e), \quad \text{s.t. } f_{\Theta_e}(x) = \begin{cases} y_e & \text{if } x \in \mathcal{I}(x_e), \\ f_{\Theta_0}(x) & \text{if } x \notin \mathcal{I}(x_e). \end{cases} \quad (1)$$

Compositional model editing (CME) contains two stages: *edit training* and *edit composing*. For the *edit training* stage, an edit is trained for (x_e, y_e) . For the *edit composing* stage, the model needs to compose the edit for (x_e, y_e) with other edits at inference time to answer a single question. Let $\mathcal{C}(\mathcal{X}_e)$ represent the question that contains the semantics of multiple edit questions $\mathcal{X}_e = \{x_e, x_e^1, x_e^2, \dots, x_e^{c-1}\}$ including x_e , where c is the composition number. Let $\mathcal{C}(\mathcal{Y}_e)$ represent the target output that contains corresponding multiple answers $\mathcal{Y}_e = \{y_e, y_e^1, y_e^2, \dots, y_e^{c-1}\}$ including y_e . The shared objective for the two stages becomes

$$f_{\Theta_e} = \text{CME}(f_{\Theta_0}, x_e, y_e), \quad \text{s.t. } f_{\Theta_e}(x) = \begin{cases} \mathcal{C}(\mathcal{Y}_e) & \text{if } x \in \mathcal{I}(\mathcal{C}(\mathcal{X}_e)), \\ f_{\Theta_0}(x) & \text{else} \end{cases} \quad (2)$$

It is worth noting that, first, the edits involved in edit composing are not necessarily all contained within $(\mathcal{X}_e, \mathcal{Y}_e)$, which introduces interference from edits useless to the current problem. Additionally, within our definition, all edits may appear asynchronously over time and be distributed spatially. Therefore, our definition is applicable to both lifelong model editing and federated model editing.

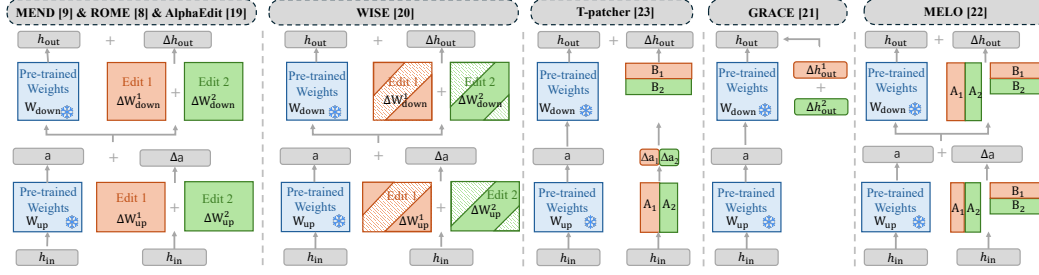


Figure 3: The visualization of different baselines, where MEND, ROME, AlphaEdit and WISE edit either $\mathbf{W}_{\text{down}}$ or both $\mathbf{W}_{\text{up}}$ and $\mathbf{W}_{\text{down}}$ with or without mask. In contrast, GRACE, T-patcher, and MELO edit the $\mathbf{W}_{\text{down}}$ or both $\mathbf{W}_{\text{up}}$ and $\mathbf{W}_{\text{down}}$ in different low-rank forms.

2.2 Edit Training and Edit Composing in FFN

Based on the results in Fig. 2, we categorize the factors influencing the composability of edits into **Composable** and **Context-preserving**. Whether the edits are composable refers to whether there is significant interference between the trained edits, which impacts *knowledge loss* and *incorrect preceding*. Context-preserving, on the other hand, refers to whether the edits affect the model's ability to follow the context to generate other answers, thereby influencing *knowledge sinking*. In this section, we discuss whether *edit training* stage of existing methods effectively have these properties and whether their *edit composing* preserves these properties as shown in Table 1.

Most existing model editing methods insert new knowledge by modifying the parameters or outputs of the FFN layer during the forward process of different tokens. The FFN operates as key-value neuron memories, where the values (rows) in the down projection layer, store pre-trained foundation knowledge obtained from vast amounts of data [24, 25]. During the forward process, it retrieves values from the down projection matrix $\mathbf{W}_{\text{down}} \in \mathbb{R}^{n \times m}$ by matching the keys in the up projection matrix $\mathbf{W}_{\text{up}} \in \mathbb{R}^{m \times n}$ with the input $h_{\text{in}} \in \mathbb{R}^{1 \times m}$:

$$h_{\text{out}} = \text{FFN}(h_{\text{in}}) = \mathbf{a}\mathbf{W}_{\text{down}} + \mathbf{b}_{\text{down}}, \quad \text{where } \mathbf{a} = \text{Act}(h_{\text{in}}\mathbf{W}_{\text{up}} + \mathbf{b}_{\text{up}}). \quad (3)$$

Here, Act is the activation function (e.g., Relu [28], Gelu [29], and SwiGLU [30]). $\mathbf{a}$ is the vector of activation values and serves as the weights to retrieve different pre-trained foundation knowledge in $\mathbf{W}_{\text{down}}$. $\mathbf{b}_{\text{down}} \in \mathbb{R}^{1 \times m}$ and $\mathbf{b}_{\text{up}} \in \mathbb{R}^{1 \times n}$ are bias vectors.

We summarize *edit training* and *edit composing* in FFN for existing methods in Fig. 3. Among them, GRACE only edits the forward process of the last input token while achieving context-preserving. The other methods modify the forward process of all tokens but lack context-preserving. MEND, ROME, AlphaEdit and WISE modify the entire $\mathbf{W}_{\text{down}}$, or $\mathbf{W}_{\text{up}}$ and $\mathbf{W}_{\text{down}}$ to insert new knowledge in *edit training*, thus different $\Delta\mathbf{W}_{\text{up}}$ and $\Delta\mathbf{W}_{\text{down}}$ are added together to utilize multiple pieces of knowledge in *edit composing*. AlphaEdit only edits $\mathbf{W}_{\text{down}}$ and projects $\Delta\mathbf{W}_{\text{down}}$ onto the null space of the preserved knowledge including other edits before applying it to $\mathbf{W}_{\text{down}}$ to avoid interference between edits. However, in the CME settings we focus on, where a single question depends on multiple edits, this orthogonality becomes ineffective and conflicts between edits still persist. WISE learns more composable edits by masking part of $\mathbf{W}_{\text{up}}$ and $\mathbf{W}_{\text{down}}$ in *edit training*, but the mask on the whole matrix with no carefully designed composing algorithm struggles to keep a large number of edits composable in *edit composing*. T-patcher concatenates multiple keys $\mathbf{A} \in \mathbb{R}^{m \times r}$ and corresponding values $\mathbf{B} \in \mathbb{R}^{r \times m}$ to $\mathbf{W}_{\text{up}}$ and $\mathbf{W}_{\text{down}}$ respectively. For GRACE, the output h_{out} are replaced with the sum of edit vectors $\Delta h_{\text{out}} \in \mathbb{R}^{1 \times m}$ for multiple pieces of knowledge. MELO trains a LoRA [31] ($\mathbf{A}, \mathbf{B}$) for each edit, where $\mathbf{A} \in \mathbb{R}^{n \times r}$ and $\mathbf{B} \in \mathbb{R}^{r \times m}$. The combination of different knowledge is achieved by concatenating $\mathbf{A}$ and $\mathbf{B}$ of different LoRAs, respectively. Thanks to the low storage overhead of their low-rank form, GRACE and MELO employ a vector database to store all edits and filter out useless ones during inference,

Table 1: Summary of model editing methods under composable and context-preserving in *edit training* and if *edit composing* preserves these properties.

Method	Edit Training		Edit Composing
	Composable	Context-preserving	
MEND	✗	✗	✗
ROME	✗	✗	✗
AlphaEdit	✗	✗	✗
WISE	✓	✗	✗
T-patcher	✗	✗	✗
GRACE	✗	✓	✗
MELO	✗	✗	✗
A ³ E	✓	✓	✓

which achieves more composable *edit composing* with a large number of edits than others but the results contain more *incorrect preceding* and are far from ideal.

3 Rethinking the Composability in Model Editing

This section examines existing model editing methods to explore three key questions: *How to train composable edits?* *How to train context-preserving edits?* *How to compose edits to further enhance composability?* Our answers are **Finding 1**, **Finding 2** and **Finding 3**, respectively.

Experimental settings. We conducted analytical experiments using the Llama3-8B model [32] on the PEAK-CF dataset [33] with 50 2-edit composition samples. During inference, we generate 30 tokens for each question, which is enough to output all edited answers. We analyze the performance of existing model editing methods with the metric $SR-S = \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{t=1}^{|\mathcal{D}_{\text{test}}|} \mathbb{I}(\forall a \in \mathcal{Y}_t, \text{rank}(a) < |\mathcal{Y}_t|)$, which is the rate of question t in test dataset $\mathcal{D}_{\text{test}}$ that all edited answers $\mathcal{Y}_t$ are output before irrelevant content, where $\text{rank}(\cdot)$ represents the rank of an answer in *all answers in the output* and $|\mathcal{Y}_t|$ represents the number of edited answers. Please refer to Appendix H for an example.

Finding 1. *The combination of pre-trained foundation knowledge is enough for ME while boosting the composability.* As introduced in Sec. 1, $\mathbf{W}_{\text{down}}$ stores pre-trained foundation knowledge obtained from vast amounts of data. Our hypothesis is that the ability of LLMs to combine different advanced knowledge stems from these well-trained foundation knowledge elements. These elements boosts the composability with a more robust and generalizable knowledge representation from large-scale pre-training. We validate our hypothesis by the average performance of methods editing full matrix and in low-rank form when editing $\mathbf{W}_{\text{up}}$ or $\mathbf{W}_{\text{down}}$ at different layers. As shown in Fig. 4(a) and Fig. 4(b), editing $\mathbf{W}_{\text{up}}$ is sufficient to guarantee the performance compared with editing $\mathbf{W}_{\text{down}}$ as well as editing both $\mathbf{W}_{\text{up}}$ and $\mathbf{W}_{\text{down}}$. This means the pre-trained foundation knowledge in $\mathbf{W}_{\text{down}}$ is sufficient to express new knowledge. The generally higher $SR-S$ of editing $\mathbf{W}_{\text{up}}$ in different layers shown in Fig. 4(c) confirms our hypothesis. For a more comprehensive discussion about the sufficiency of editing $\mathbf{W}_{\text{up}}$ for higher $SR-S$ and less *knowledge loss*, please refer to Appendix I.

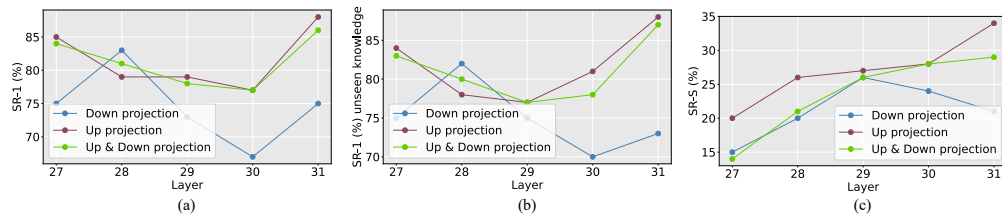


Figure 4: (a) Performance of non-compositional ME measured by $SR-I$ when editing $\mathbf{W}_{\text{down}}$, $\mathbf{W}_{\text{up}}$ or both of them. (b) Performance of non-compositional ME measured by $SR-I$ when editing $\mathbf{W}_{\text{down}}$, $\mathbf{W}_{\text{up}}$ or both of them with pseudo samples to simulate totally unseen knowledge (Appendix I). (c) Performance of CME measured by $SR-S$ when editing $\mathbf{W}_{\text{down}}$, $\mathbf{W}_{\text{up}}$ or both of them.

Finding 2. *Less change of the hidden state of the last answer token [la], more context-preserving.* We demonstrate in Fig. 5(a) the relationship between the context-preserving rate and the change of the hidden state at the edited FFN layer when inferring with [la] (e.g., “What are the symptoms of COVID-19? fever”). The results show an evident inverse relationship between these two factors in the base-lines, indicating that a smaller change in the hidden state of [la] can achieve better context-preserving. The finding also answers why GRACE achieves context-following, because it has no parametric interference on the forward process of [la] by editing the last input token only.

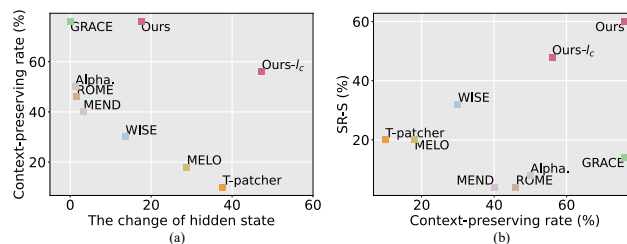


Figure 5: The relevance (a) between the change of hidden state at [la] and the context-preserving rate (b) between context-preserving and $SR-S$.

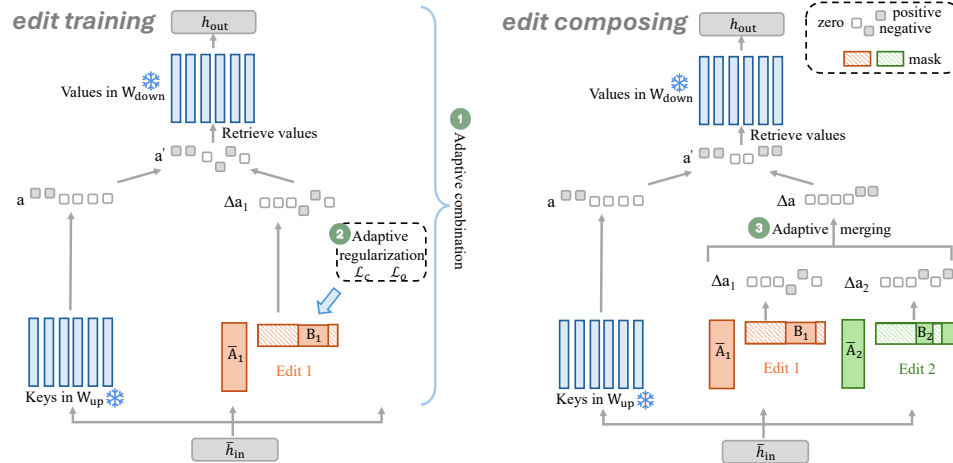


Figure 7: Our proposed A³E, including adaptive combination (Sec. 4.1), adaptive regularization (Sec. 4.1) and adaptive merging (Sec. 4.2).

Finding 3. *Setting the elements with different signs in different edits to zero (Adaptive Merging) during edit composing enhances the composability.* The elements with opposite signs between different edits may contain conflicting knowledge. In Fig. 6, inspired by [34], we validate that setting these opposing elements in different edits to zero better balances the knowledge forgetting in the edits and their negative interference on each other. As can be seen, the mean performance of methods editing the full matrix and in low-rank form with *Adaptive Merging* outperforms directly composing different edits.

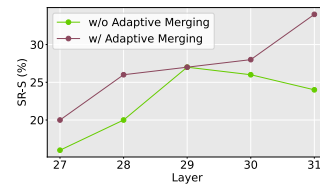


Figure 6: Performance of CME measured by SR-S with or without *Adaptive Merging*.

4 The Proposed Method

Although the three findings mentioned above contributes the composability of model editing, there still lacks of a unified framework to fully integrate their advantages. *How to train composable and context-preserving edits?* Based on Finding 1, we adaptively select the pre-trained foundation knowledge using a mask and in a low-rank form to utilize the vector database, namely **Adaptive Combination** of pre-trained foundation knowledge. We propose two **Adaptive Regularization** to regularize the use of pre-trained foundation knowledge inspired by Finding 2 while further alleviating *incorrect preceding* in low-rank form. *How to compose edits to further enhance composability?* We utilize a vector database to filter out useless edits following [21, 22] while proposing **Adaptive Merging** for different combinations of pre-trained foundation knowledge inspired by Finding 3. The method is shown in Fig. 7. Please refer to Appendix D for the vector database and Alg. 1-2 for the complete editing progress.

4.1 How to train composable and context-preserving edits?

Adaptive Combination. Inspired by Finding 1, we perform model editing only on W_{up} in a low-rank form similar to LoRA but incorporate an adaptive mask on B . The adaptive mask aims to only preserve the most related pre-trained foundation knowledge to achieve a better trade-off between the composability and learning ability. Specifically, an (A, B) pair is trained for each edit following [22]. The forward process of the edited FFN is calculated as:

$$h_{out} = \text{FFN}_{edit}(h_{in}, A, B, M) = aW_{down} + b_{down},$$

$$\text{where } a = \text{Act}(h_{in}W_{up} + \underbrace{\bar{h}_{in}\bar{A}(B \times M)}_{\Delta a} + b_{up}). \quad (4)$$

Algorithm 1 The Edit Training Stage

- 1: **Input** The initial LLM f_θ , the edit dataset $\mathcal{D}_{\text{edit}}$, the initialized vector database $\mathcal{S}_{\text{db}}$.
 - 2: **Output** The vector database $\mathcal{S}_{\text{db}}$
 - 3: **for** each edit $(x_e, y_e) \in \mathcal{D}_{\text{edit}}$ **do**
 - 4: Compute mask $\mathbf{M}_e$.
 - 5: Replace $h_{\text{out}} = \text{FFN}(h_{\text{in}})$ (Eq. (3)) with $h_{\text{out}} = \text{FFN}_{\text{edit}}(h_{\text{in}}, \mathbf{A}_e, \mathbf{B}_e, \mathbf{M}_e)$ (Eq. (4)) at selected layers.
 - 6: Update $(\mathbf{A}_e, \mathbf{B}_e)$ with $\mathcal{L}$ (Eq. (7)).
 - 7: Save $\mathbf{K}_e$ into $\mathcal{S}_{\text{db}}$ as the key of vector database and save $(\mathbf{A}_e, \mathbf{B}_e)$ as the corresponding value.
 - 8: **end for**
-

$\mathbf{A} \in \mathbb{R}^{m \times r}$, $\mathbf{B} \in \mathbb{R}^{r \times n}$ are the components of LoRA while $\bar{h}_{\text{in}}$ and $\bar{\mathbf{A}}$ are normalized h_{in} and $\mathbf{A}$ to avoid the weights of using different $\mathbf{B}$ have significant differences and lead to *knowledge loss*. $\mathbf{M} \in \mathbb{R}^{r \times n}$ is the mask to select top k related values in $\mathbf{W}_{\text{down}}$, where the correlation is calculated by the dot product between values in $\mathbf{W}_{\text{down}}$ and the answer tokens' embedding:

Adaptive Regularization. Based on Finding 2, we design a regularization $\mathcal{L}_c$ to restrict the additional use of pre-trained foundation knowledge by the last answer token [la], thereby enabling the context-preserving to generate the next answer. Specifically:

$$\mathcal{L}_c = \|\bar{h}_{\text{in}}^{[\text{la}]} \bar{\mathbf{A}}\|_2, \quad (5)$$

where $\bar{h}_{\text{in}}^{[\text{la}]}$ represents the normalized input hidden state of the FFN layer at [la]. $\mathcal{L}_c$ achieves context-preserving by encouraging a low $\Delta \mathbf{a}$ in Eq. (4).

In order to further alleviate *incorrect preceding* in low-rank form, we propose another regularization $\mathcal{L}_o$ to restrict the logits of tokens $h_{\text{logit}}^{[\text{a}]}$ except labels by:

$$\mathcal{L}_o = \sum_{[\text{a}]} \sum_{i \in \mathcal{W}_{[\text{a}]}} [h_{\text{logit}}^{[\text{a}]}]_i, \quad (6)$$

where [a] represents the positions of an edit sample with labels, $\mathcal{W}_{[\text{a}]}$ represents the set of all token IDs except the label of the position [a]. During the *edit training* stage, we optimize $(\mathbf{A}, \mathbf{B})$ with the loss:

$$\mathcal{L} = \mathcal{L}_e + \alpha \mathcal{L}_c + \beta \mathcal{L}_o, \quad (7)$$

where $\mathcal{L}_e$ are cross-entropy loss, α and β are hyperparameters.

4.2 How to compose edits?

Adaptive Merging. During the inference time, after filtering out useless edits by the vector database $\mathcal{S}_{\text{db}}$ (Appendix D), we need to *compose different combinations of pre-trained foundation knowledge* and different edits may conflict in the utilization of pre-trained foundation knowledge. Specifically, some edits aim to leverage the j -th value in $\mathbf{W}_{\text{down}}$ to compose the target answer, so $[\Delta \mathbf{a}]_j$ tends to be positive. In contrast, other edits may want to erase the knowledge in the j -th row to compose the target answer, so $[\Delta \mathbf{a}]_j$ tends to be negative. Inspired by Finding 3, when such conflicts occur, we set $[\Delta \mathbf{a}]_j$ of all used edits to 0 to avoid the overly strong impact on each other. For simplicity, we represent this merging process of different $\Delta \mathbf{a}$ as $\hat{\sum}$ and the $\Delta \mathbf{a}$ of preserved useful edits by vector database as $\mathcal{S}$. Thus the forward process during inference changes to:

$$\begin{aligned} h_{\text{out}} &= \text{FFN}_{\text{test}}(h_{\text{in}}, \mathcal{S}) = \mathbf{a} \mathbf{W}_{\text{down}} + \mathbf{b}_{\text{down}}, \\ \text{where } \mathbf{a} &= \text{Act}(h_{\text{in}} \mathbf{W}_{\text{up}} + \hat{\sum}_{\Delta \mathbf{a} \in \mathcal{S}} \Delta \mathbf{a} + \mathbf{b}_{\text{up}}). \end{aligned} \quad (8)$$

5 Experiment

5.1 Evaluation Benchmarks

Dataset. We utilize the PEAK-CF and PEAK-T datasets [33], which contain a large number of questions with multiple answers. We keep the questions for which Llama3-8B [32] and Mistral-7B

Algorithm 2 The Edit Composing Stage

- 1: **Input** The initial LLM f_θ , the vector database $\mathcal{S}_{\text{db}}$, the test dataset $\mathcal{D}_{\text{test}}$.
 - 2: **Output** The answers of the queries in $\mathcal{D}_{\text{test}}$.
 - 3: **for** each query $\mathcal{X}_t \in \mathcal{D}_{\text{test}}$ **do**
 - 4: Retrieve edits and calculate the $\Delta\mathbf{a}$ set and $\mathcal{S}_t$ related to $\mathcal{X}_t$ with its representation h_{db} (Eq. (9)).
 - 5: Generate the output with replacing $h_{\text{out}} = \text{FFN}(h_{\text{in}})$ (Eq. (3)) with $h_{\text{out}} = \text{FFN}_{\text{test}}(h_{\text{in}}, \mathcal{S}_t)$ (Eq. (8))
 - 6: **end for**
-

[35] models have more than four unknown answers, with 1,949 instances in PEAK-CF and 922 instances in PEAK-T. We directly use the rephrase instances and locality instances from PEAK-CF and PEAK-T to test the generalization and locality. Please refer to Appendix G for the specific construction process.

Evaluation metrics. Inspired by [33], we evaluate baselines and our methods with $SR\text{-}S = \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{t=1}^{|\mathcal{D}_{\text{test}}|} \mathbb{I}(\forall a \in \mathcal{Y}_t, \text{rank}(a) < |\mathcal{Y}_t|)$, which represents the success rate with all edited answers in the output without *incorrect preceding*. In Appendix H, we provide examples about $SR\text{-}S$ while discussing its effectiveness for evaluation and the results of its variants; $SR\text{-}I = \frac{1}{c|\mathcal{D}_{\text{test}}|} \sum_{i=1}^{|\mathcal{D}_{\text{test}}|} \sum_{a \in \mathcal{Y}_t} \mathbb{I}(a \in \hat{\mathcal{Y}}_t)$, which represents the success rate that edited answers $a \in \mathcal{Y}_t$ are in the output answer set $\hat{\mathcal{Y}}_t$. c represents the composition number, which means c pieces of edited knowledge are needed for a question; $GSR = \frac{1}{|\mathcal{D}_{\text{rep}}|} \sum_{t=1}^{|\mathcal{D}_{\text{rep}}|} \mathbb{I}(\forall a \in \mathcal{Y}_t, \text{rank}(a) < |\mathcal{Y}_t|)$, but the questions are changed into rephrased ones with the same semantic to evaluate the generalization; $LSR = \frac{1}{|\mathcal{D}_{\text{loc}}|} \sum_{t=1}^{|\mathcal{D}_{\text{loc}}|} \text{Rouge-L}(\hat{\mathcal{Y}}_t^{\text{loc}}, \mathcal{Y}_t^{\text{loc}})$, which represents the f1 score of Rouge-L [36] between the edited outputs $\hat{\mathcal{Y}}_t^{\text{loc}}$ for locality questions and origin output $\mathcal{Y}_t^{\text{loc}}$. Please refer to Appendix K for examples of the input.

Baselines. We compared our method with FT, ROME, KE, MEND, AlphaEdit, WISE, T-patcher, GRACE, and MELO. Please refer to Appendix G for more details.

5.2 Experimental Results

Different datasets and CME tasks. To evaluate the effectiveness of the proposed A³E, we conduct large-scale experiments with all instances of the PEAK-CF and PEAK-T datasets, covering four CME tasks: independent multi-answer composition (IMAC), independent multi-question composition (IMQC), multi-answer composition (MAC) and multi-question composition (MQC). In independent scenarios, we manually filter out useless edits for each question to directly verify the effectiveness of adaptive combination, regularization and merging without the effect of vector database. In MAC and MQC, the vector database is used to evaluate in real-world scenarios with a large number of edits asynchronously and distributedly. We set the composition number to 2, which means two pieces of edited knowledge are needed for a question. The results shown in Table 2 indicate that A³E consistently performs better on four CME tasks across two datasets. For IMAC on PEAK-CF, A³E surpasses all baselines by at least 22.45% in $SR\text{-}S$, at least 13.63% in $SR\text{-}I$ and at least 18.36% in GSR . Even on challenging MAC, A³E consistently achieves the best editing performance and generalization while maintaining similar level locality. The lead of A³E on PEAK-T is even more obvious, with at least 67.16% and 406.73% in $SR\text{-}S$ for IMQC and MQC respectively.

Different number of edits. To evaluate the performance of A³E throughout the lifelong usage process, we compare the performance among A³E, GRACE and MELO, which have relatively strong MAC and MQC capability. The results of MAC on PEAK-CF, as shown in Fig. 8(a), indicate that A³E maintains the most outstanding performance until 3898 edits. Please refer to Appendix. J.1 for more results.

Different composition number. So far, we have conducted experiments with a composition number of 2. To demonstrate that A³E still has superiority when it needs to simultaneously utilize more edited knowledge, we compared the performance of A³E, GRACE, and MELO with composition numbers ranging from 1 to 4, each containing 50 (50×1), 100 (50×2), 150 (50×3) and 200 (50×4)

Table 2: Performance measured by metrics in Sec. 5.1 in 1) IMAC (3898 edits) and MAC (3898 edits) on PEAK-CF. 2) IMAC (1984 edits) and MAC (1984 edits) on PEAK-T. 3) IMQC (1948 edits) and MQC (1948 edits) on PEAK-CF. 4) IMQC (992 edits) and MQC (992 edits) on PEAK-T.

Method	PEAK-CF							PEAK-T						
	IMAC			MAC				IMAC			MAC			
	SR-S $\uparrow$	SR-I $\uparrow$	GSR $\uparrow$	SR-S $\uparrow$	SR-I $\uparrow$	GSR $\uparrow$	LSR $\uparrow$	SR-S $\uparrow$	SR-I $\uparrow$	GSR $\uparrow$	SR-S $\uparrow$	SR-I $\uparrow$	GSR $\uparrow$	LSR $\uparrow$
FT	0.31	12.01	0.36	0.00	0.00	0.00	5.25	0.10	1.92	0.10	0.00	0.00	0.00	3.79
ROME	0.92	16.30	0.82	0.00	0.62	0.00	12.25	0.30	18.24	0.60	0.00	0.00	0.00	6.22
KE	0.92	10.31	0.72	0.00	0.00	0.00	10.51	0.10	5.04	0.10	0.00	0.00	0.00	4.56
MEND	2.36	23.09	1.90	0.00	0.56	0.00	12.35	1.01	12.40	0.81	0.00	0.00	0.00	5.41
AlphaEdit	5.44	31.01	4.57	6.98	33.24	6.16	15.17	2.02	21.78	2.12	0.50	16.28	0.41	8.45
WISE	<u>39.99</u>	<u>65.20</u>	<u>38.60</u>	0.10	5.65	0.10	15.21	<u>23.08</u>	54.64	<u>21.37</u>	0.20	2.92	0.20	8.40
T-patcher	26.64	52.54	12.83	0.00	0.59	0.00	1.01	20.77	43.6	18.15	0.00	0.00	0.00	0.67
GRACE	15.20	57.44	14.53	12.99	<u>50.16</u>	9.75	27.19	6.55	41.93	6.05	5.65	<u>33.07</u>	5.04	<u>37.54</u>
MELO	19.46	44.33	18.47	<u>18.22</u>	38.51	<u>16.68</u>	24.13	9.48	28.94	9.07	<u>6.96</u>	22.94	<u>6.56</u>	35.98
A ³ E	48.97	75.49	45.69	43.38	70.72	39.48	<u>25.76</u>	41.63	<u>51.62</u>	36.90	38.00	49.60	35.89	40.97

Method	PEAK-CF							PEAK-T						
	IMQC			MQC				IMQC			MQC			
	SR-S $\uparrow$	SR-I $\uparrow$	GSR $\uparrow$	SR-S $\uparrow$	SR-I $\uparrow$	GSR $\uparrow$	LSR $\uparrow$	SR-S $\uparrow$	SR-I $\uparrow$	GSR $\uparrow$	SR-S $\uparrow$	SR-I $\uparrow$	GSR $\uparrow$	LSR $\uparrow$
FT	5.64	21.65	4.82	0.00	0.00	0.00	2.09	0.00	3.23	0.40	0.00	0.00	0.00	2.12
ROME	23.49	47.08	19.38	0.00	0.41	0.00	4.52	19.15	21.92	14.52	0.00	0.00	0.00	3.63
KE	1.03	15.40	0.77	0.00	0.00	0.00	3.92	4.23	8.06	3.13	0.00	0.00	0.00	2.98
MEND	5.23	22.38	5.05	0.00	0.00	0.00	3.77	14.11	17.54	8.77	0.00	0.00	0.00	4.02
AlphaEdit	7.69	28.57	5.64	5.33	24.92	4.21	6.92	16.33	18.44	11.09	<u>5.65</u>	25.11	<u>5.04</u>	4.11
WISE	<u>46.56</u>	<u>70.11</u>	<u>44.51</u>	0.0	3.23	0.0	7.28	<u>27.62</u>	<u>32.77</u>	24.60	0.20	3.33	0.20	4.40
T-patcher	33.23	60.41	31.30	0.00	0.00	0.00	0.45	26.81	31.68	<u>25.20</u>	0.00	0.00	0.00	0.32
GRACE	6.56	39.02	5.23	3.59	34.61	2.87	38.71	6.05	32.06	5.54	4.84	<u>30.55</u>	3.63	<u>38.59</u>
MELO	20.31	44.16	18.47	<u>9.13</u>	<u>36.16</u>	<u>8.21</u>	<u>43.22</u>	6.25	23.49	5.85	2.82	19.86	2.02	36.25
A ³ E	52.51	73.49	52.41	26.46	53.44	20.31	45.73	46.17	68.85	38.31	28.63	53.73	21.77	41.37

edits, respectively. As shown in Fig. 8(b), although the *SR-S* gradually decreases with increasing composition number, A³E always maintains a significantly better performance. Please refer to Appendix J.1 for more results.

Non-compositional ME. A stronger composability should not come at the expense of worse non-compositional ME performance. Therefore, we compare the average non-compositional ME performance of A³E and baselines. The results, as shown in Fig. 8(c), demonstrate that our method is on par with baselines in *SR-I*, *GSR* and *LSR*. We provide non-compositional results of other datasets in Appendix J.6.

Different backbones and prompts. To investigate the impact of different backbones (Llama3-8B and Mistral-7B) and few-shot prompts (please refer to Appendix K) on the proposed method, we conduct experiments in MAC setting with 3898 edits.

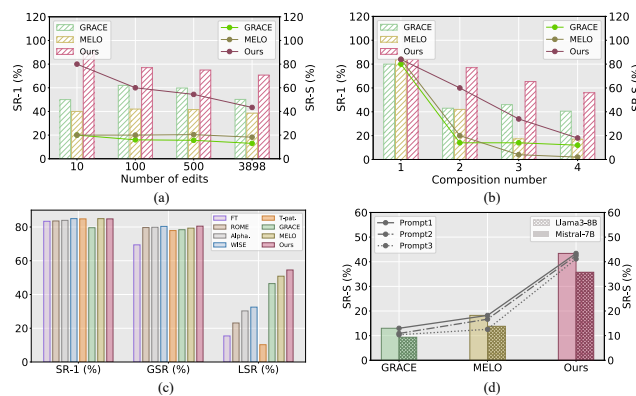


Figure 8: (a) Performance change of A³E and baselines in MAC until 3898 edits on PEAK-CF. (b) The effect of composition number from 1 to 4 in MAC on PEAK-CF. (c) The comparison between A³E and baselines in non-compositional ME on PEAK-CF. (d) The effect of different backbones and different prompts in MAC on PEAK-CF.

The results in Fig. 8(d) show that: (1) A³E has a leading SR-S when using different backbones and different prompts. (2) A³E and baselines generally have better editing performance on Llama3-8B. (3) When using different few-shot prompts to combine multiple questions, there is a certain performance difference in A³E. This may be due to the fact that, although we manually ensure that few-shot prompts do not contain answers and content obviously related to answers, there may still be some implicit associations between them and the answers. However, this does not affect the significant improvement of A³E on the composability of model editing. Please refer to Appendix. J.3 and Appendix. J.4 for more comprehensive results. We also provide more discussions on failure cases in Appendix J.7 and generalization in Appendix J.8.

5.3 Ablation Study

As shown in Fig. 9(a), we conduct experiments in MAC with 3898 edits to evaluate seven ablation versions removing the proposed $\mathcal{L}_o$, $\mathcal{L}_c$, adaptive combination, adaptive merging, norm and editing only $\mathbf{W}_{up}$, respectively. All the components contribute to the composability of A³E. Please refer to Appendix. J.5 for more results. We further demonstrate three advantages of the components.

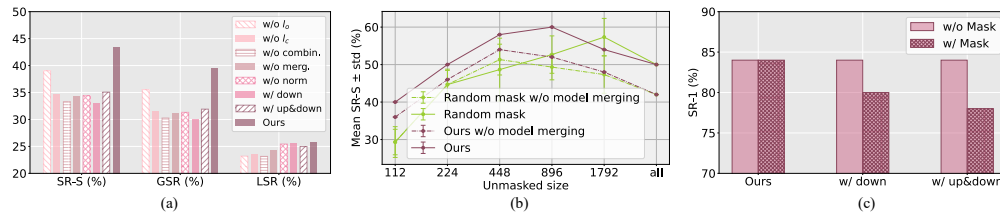


Figure 9: (a) Performance of A³E and seven ablation versions in MAC on PEAK-CF. (b) The robustness of edited knowledge when randomly masking $\mathbf{B}$ in $\mathbf{W}_{up}$ editing, $\mathbf{W}_{down}$ editing and $\mathbf{W}_{up}\&\mathbf{W}_{down}$ editing in non-compositional ME on PEAK-CF. (c) Performance improved by adaptive merging with random masks from different seeds or adaptive combination in MAC on PEAK-CF.

Advantage 1. *Adaptive combination achieves performance beyond the trade-off between composability and learning ability.* As can be seen from Fig. 9(b), although adding a random mask can achieve good composability, the larger standard deviation indicates that it is significantly affected by the random seed. Meanwhile, as the unmask size increases, the learning ability gradually improves with the increasing overlap between edits. We overcome this trade-off by using adaptive combination to select more useful pre-trained foundation knowledge.

Advantage 2. *$\mathcal{L}_c$ makes better context-preserving.* See Fig. 5(a), by incorporating $\mathcal{L}_c$, our method significantly reduces the change in the hidden state at the last answer token compared to our methods without $\mathcal{L}_c$, thus improving the context-preserving capability and achieving a level comparable to GRACE. Meanwhile, as shown in Fig. 5(b), our stronger context-preserving capability successfully translates into a higher SR-S.

Advantage 3. *Adaptive merging and adaptive combination mutually reinforce each other.* In Fig. 9(c), we compare the impact of test-time random masks on $\mathbf{B}$ for editing $\mathbf{W}_{down}$ and $\mathbf{W}_{up}$ in the non-compositional ME scenario, where the mask size is set to the mean size of the parts that require adaptive merging in the CME scenario. It shows that editing $\mathbf{W}_{up}$ is less likely to suffer from forgetting caused by the presence of adaptive merging.

6 Conclusion

In this paper, we first define and benchmark the CME task, which aims to evaluate the ability of model editing methods to simultaneously utilize multiple edited knowledge. By revisiting existing model editing methods, we point out their lack of composability manifests as *knowledge loss*, *incorrect preceding* and *knowledge sinking*. To mitigate them, we propose A³E to train more composable and context-preserving edits and compose edits with enhanced composability. Extensive experiments under various scenarios, backbones, and prompts demonstrate the effectiveness of the proposed A³E.

Acknowledgements

We thank all the anonymous reviewers for their constructive suggestions on improving this paper. This paper is partially supported by Hong Kong Innovation and Technology Fund (ITF) grant MHP/034/22 and National Natural Science Foundation of China (NSFC) grant 62441236.

References

- [1] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.
- [2] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [4] Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*, 2020.
- [5] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.
- [6] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, 2023.
- [7] Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. *arXiv preprint arXiv:2104.08164*, 2021.
- [8] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.
- [9] Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. Fast model editing at scale. *arXiv preprint arXiv:2110.11309*, 2021.
- [10] Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. Pmet: Precise model editing in a transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18564–18572, 2024.
- [11] Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, et al. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*, 2024.
- [12] Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. *arXiv preprint arXiv:2305.13172*, 2023.
- [13] Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics*, 12:283–298, 2024.
- [14] Jun-Yu Ma, Jia-Chen Gu, Zhen-Hua Ling, Quan Liu, and Cong Liu. Untying the reversal curse via bidirectional language model editing. *arXiv preprint arXiv:2310.10322*, 2023.
- [15] Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. Model editing can hurt general abilities of large language models. *arXiv preprint arXiv:2401.04700*, 2024.

- [16] Akshat Gupta, Anurag Rao, and Gopala Anumanchipalli. Model editing at scale leads to gradual and catastrophic forgetting. *arXiv preprint arXiv:2401.07453*, 2024.
- [17] Zihao Lin, Mohammad Beigi, Hongxuan Li, Yufan Zhou, Yuxiang Zhang, Qifan Wang, Weng-peng Yin, and Lifu Huang. Navigating the dual facets: A comprehensive evaluation of sequential memory editing in large language models. *arXiv preprint arXiv:2402.11122*, 2024.
- [18] Jun-Yu Ma, Hong Wang, Hao-Xiang Xu, Zhen-Hua Ling, and Jia-Chen Gu. Perturbation-restrained sequential model editing. *arXiv preprint arXiv:2405.16821*, 2024.
- [19] Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Xiang Wang, Xiangnan He, and Tat-seng Chua. Alphaedit: Null-space constrained knowledge editing for language models. *arXiv preprint arXiv:2410.02355*, 2024.
- [20] Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. *arXiv preprint arXiv:2405.14768*, 2024.
- [21] Tom Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with grace: Lifelong model editing with discrete key-value adaptors. *Advances in Neural Information Processing Systems*, 36, 2024.
- [22] Lang Yu, Qin Chen, Jie Zhou, and Liang He. Melo: Enhancing model editing with neuron-indexed dynamic lora. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19449–19457, 2024.
- [23] Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. Transformer-patcher: One mistake worth one neuron. *arXiv preprint arXiv:2301.09785*, 2023.
- [24] Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*, 2020.
- [25] Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. Knowledge neurons in pretrained transformers. *arXiv preprint arXiv:2104.08696*, 2021.
- [26] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.
- [27] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. *Advances in neural information processing systems*, 28, 2015.
- [28] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.
- [29] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.
- [30] Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.
- [31] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.
- [32] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.
- [33] Jun-Yu Ma, Zhen-Hua Ling, Ningyu Zhang, and Jia-Chen Gu. Neighboring perturbations of knowledge editing on large language models. *arXiv preprint arXiv:2401.17623*, 2024.
- [34] Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36, 2024.

- [35] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- [36] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.
- [37] Chenmian Tan, Ge Zhang, and Jie Fu. Massive editing for large language models via meta learning. *arXiv preprint arXiv:2311.04661*, 2023.
- [38] Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*, 2022.
- [39] Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. Memory-based model editing at scale. In *International Conference on Machine Learning*, pages 15817–15831. PMLR, 2022.
- [40] Zexuan Zhong, Zhengxuan Wu, Christopher D Manning, Christopher Potts, and Danqi Chen. Mquake: Assessing knowledge editing in language models via multi-hop questions. *arXiv preprint arXiv:2305.14795*, 2023.
- [41] Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. The role of permutation invariance in linear mode connectivity of neural networks. *arXiv preprint arXiv:2110.06296*, 2021.
- [42] Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? *Advances in neural information processing systems*, 33:512–523, 2020.
- [43] Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. *arXiv preprint arXiv:2212.09849*, 2022.
- [44] Michael S Matena and Colin A Raffel. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716, 2022.
- [45] Ronald A Fisher. On the mathematical foundations of theoretical statistics. *Philosophical transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character*, 222(594-604):309–368, 1922.
- [46] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.
- [47] Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. Task arithmetic in the tangent space: Improved editing of pre-trained models. *Advances in Neural Information Processing Systems*, 36, 2024.
- [48] Shachar Don-Yehiya, Elad Venezian, Colin Raffel, Noam Slonim, Yoav Katz, and Leshem Choshen. Cold fusion: Collaborative descent for distributed multitask finetuning. *arXiv preprint arXiv:2212.01378*, 2022.
- [49] Protection Regulation. Eu general data protection regulation. *available at, available at: [https://eur-lex.europa.eu/eli/reg/2016/679/oj/\(accessed 10 September 2023\)](https://eur-lex.europa.eu/eli/reg/2016/679/oj/(accessed 10 September 2023))*, 2016.
- [50] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. *arXiv preprint arXiv:1706.04115*, 2017.
- [51] Yiwei Wang, Muhao Chen, Nanyun Peng, and Kai-Wei Chang. Deepedit: Knowledge editing as decoding with constraints. *arXiv preprint arXiv:2401.10471*, 2024.
- [52] Ce Zheng, Lei Li, Qingxiu Dong, Yuxuan Fan, Zhiyong Wu, Jingjing Xu, and Baobao Chang. Can we edit factual knowledge by in-context learning? *arXiv preprint arXiv:2305.12740*, 2023.

- [53] Weizhi Fei, Xueyan Niu, Guoqing Xie, Yanhua Zhang, Bo Bai, Lei Deng, and Wei Han. Retrieval meets reasoning: Dynamic in-context editing for long-text understanding. *arXiv preprint arXiv:2406.12331*, 2024.
- [54] Keyuan Cheng, Gang Lin, Haoyang Fei, Lu Yu, Muhammad Asif Ali, Lijie Hu, Di Wang, et al. Multi-hop question answering under temporal knowledge editing. *arXiv preprint arXiv:2404.00492*, 2024.
- [55] Yucheng Shi, Qiaoyu Tan, Xuansheng Wu, Shaochen Zhong, Kaixiong Zhou, and Ninghao Liu. Retrieval-enhanced knowledge editing in language models for multi-hop question answering. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 2056–2066, 2024.
- [56] Xiaoshuai Song, Zhengyang Wang, Keqing He, Guanting Dong, Yutao Mou, Jinxu Zhao, and Weiran Xu. Assessing and post-processing black box large language models for knowledge editing. In *Proceedings of the ACM on Web Conference 2025*, pages 1716–1732, 2025.
- [57] Abhika Mishra, Akari Asai, Vidhisha Balachandran, Yizhong Wang, Graham Neubig, Yulia Tsvetkov, and Hannaneh Hajishirzi. Fine-grained hallucination detection and editing for language models. *arXiv preprint arXiv:2401.06855*, 2024.
- [58] Zihao Zhao, Yuchen Yang, Yijiang Li, and Yinzhi Cao. Ripplecot: Amplifying ripple effect of knowledge editing in language models via chain-of-thought in-context learning. *arXiv preprint arXiv:2410.03122*, 2024.
- [59] Changyue Wang, Weihang Su, Qingyao Ai, Yujia Zhou, and Yiqun Liu. Decoupling reasoning and knowledge injection for in-context knowledge editing. *arXiv preprint arXiv:2506.00536*, 2025.
- [60] Junda Zhu, Lingyong Yan, Haibo Shi, Dawei Yin, and Lei Sha. Atm: Adversarial tuning multi-agent system makes a robust retrieval-augmented generator. *arXiv preprint arXiv:2405.18111*, 2024.
- [61] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. *arXiv preprint arXiv:1907.02189*, 2019.
- [62] Hong Huang, Lan Zhang, Chaoyue Sun, Ruogu Fang, Xiaoyong Yuan, and Dapeng Wu. Distributed pruning towards tiny neural networks in federated learning. In *2023 IEEE 43rd International Conference on Distributed Computing Systems (ICDCS)*, pages 190–201. IEEE, 2023.
- [63] Hong Huang, Weiming Zhuang, Chen Chen, and Lingjuan Lyu. Fedmef: Towards memory-efficient federated dynamic pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27548–27557, 2024.
- [64] Hong Huang, Hai Yang, Yuan Chen, Jiaxun Ye, and Dapeng Wu. Fedrts: Federated robust pruning via combinatorial thompson sampling. *arXiv preprint arXiv:2501.19122*, 2025.
- [65] Hong Huang and Dapeng Wu. Quaff: Quantized parameter-efficient fine-tuning under outlier spatial stability hypothesis. *arXiv preprint arXiv:2505.14742*, 2025.
- [66] Junda Zhu, Jun Ai, Yujun Li, Yichun Yin, Yasheng Wang, Lifeng Shang, and Qun Liu. RidgeloRA: Matrix ridge enhanced low-rank adaptation of large language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.
- [67] Yichen Wu, Hongming Piao, Long-Kai Huang, Renzhen Wang, Wanhua Li, Hanspeter Pfister, Deyu Meng, Kede Ma, and Ying Wei. Sd-lora: Scalable decoupled low-rank adaptation for class incremental learning. *arXiv preprint arXiv:2501.13198*, 2025.
- [68] Hongming Piao, Yichen Wu, Dapeng Wu, and Ying Wei. Federated continual learning via prompt-based dual knowledge transfer. In *Forty-first International Conference on Machine Learning*, 2024.

Appendices Contents

A	Notation	16
B	Related Work	16
C	Compositional vs. Collect & Edit	17
D	The Design of Vector Database	18
E	Handling Continuous Editing Situations	18
F	Computational and Memory Overhead	18
G	Experiment Details	19
H	Discussions of $SR-S$	20
I	The Sufficiency of Editing W_{up}	21
I.1	For higher $SR-S$	21
I.2	For less <i>knowledge loss</i>	21
J	Additional Experimental Results	21
J.1	The effect of the composition number and the number of edits	21
J.2	The effect of hyperparameters	22
J.3	The effect of different backbones	22
J.4	The effect of different prompts	23
J.5	Ablation study	25
J.6	Non-compositional ME evaluation on classic datasets for ME	25
J.7	Failure cases analysis	25
J.8	Generalization	26
K	Description of Dataset and Few-shot Prompts	28
L	Limitations	28
M	Broader Impact	29

A Notation

Table 3: The definition of the notations.

Notation	Definition
$\mathbf{m}$	Input & output dimension of feed-forward network (FFN)
$\mathbf{n}$	Hidden dimension of FFN
$\mathbf{d}$	Vocabulary size
$\mathbf{W}_{\text{up}}$	Parameters of the first layer of FFN; $\mathbf{W}_{\text{up}} \in \mathbb{R}^{m \times n}$
$\mathbf{W}_{\text{down}}$	Parameters of the second layer of FFN; $\mathbf{W}_{\text{down}} \in \mathbb{R}^{n \times m}$
$\mathbf{b}_{\text{up}}$	Bias vector of the first layer of FFN; $\mathbf{b}_{\text{up}} \in \mathbb{R}^{1 \times n}$
$\mathbf{b}_{\text{down}}$	Bias vector of the second layer of FFN; $\mathbf{b}_{\text{down}} \in \mathbb{R}^{1 \times m}$
h_{in}	Input hidden state of FFN; $h_{\text{in}} \in \mathbb{R}^{1 \times m}$
h_{out}	Output hidden state of FFN; $h_{\text{out}} \in \mathbb{R}^{1 \times m}$
$\mathbf{a}$	Activation vector after the first layer of FFN; $\mathbf{a} \in \mathbb{R}^{1 \times n}$
$(\mathbf{A}, \mathbf{B})$	Two learnable low-rank matrices $\mathbf{A} \in \mathbb{R}^{n \times r} (\mathbb{R}^{m \times r})$ and $\mathbf{B} \in \mathbb{R}^{r \times m} (\mathbb{R}^{r \times n})$ to constrain the parameter updates to lie in a low-rank subspace following [1,2]
$\mathbf{M}$	The mask to select pre-trained foundation knowledge in $\mathbf{W}_{\text{down}}$; $\mathbf{M} \in \mathbb{R}^{r \times n}$
h_{logit}	Output logits of LLM
$\mathcal{W}$	The set of all token IDs
$\mathcal{S}$	The preserved edits for inference by the vector database
a	Generated answer of LLM
$\hat{\mathcal{Y}}$	The set of generated answers of LLM
$\mathcal{Y}$	The set of edited answers

B Related Work

Model Editing. Model editing has become a popular technique to correct LLMs without the costly re-training. Existing model editing methods are mainly divided into constrained fine-tuning, meta-learning, locating-and-editing, and memory-based methods. Meta-learning methods such as MEND [9] and KE [7] train a hypernetwork to generate model updates, with MALMEN [37] addressing the cancellation effects. Locating-and-editing methods like ROME [8] identify factual associations in the feed-forward layer and update them. Building on this, MEMIT [38] extends it to a batch edit setting, while AlphaEdit [19] projects perturbation onto the null space of the preserved knowledge to achieve better lifelong model editing. To avoid conflicts among numerous edits and the impact on original knowledge, memory-based methods such as SERAC [39], T-patcher [23], GRACE [21], MELO [22] and WISE [20] construct extra working memory in different knowledge preservation forms, from which useless knowledge is filtered out during inference.

Model Editing Evaluation. From single edit to batch edit and then to lifelong edit, the evaluation of model editing methods is becoming increasingly close to real-world applications and more challenging in terms of the ability to handle interference between different edits and with the original knowledge. In addition to the classic assessments of reliability [9, 10], generalization [10, 11], and locality [8, 12], Cohen et al. [13] proposed evaluating the ripple effect of model editing, which means a series of knowledge that needs to be changed along with the new knowledge. Ma et al. [14] evaluated the knowledge inserted into the model bidirectionally, while these works [15, 16, 17, 18] assessed the impact of model editing methods on general downstream tasks. Unlike previous works, we are the first to point out that model editing methods should possess composability, namely enabling the model to use multiple edited knowledge simultaneously. On the one hand, our work extends locality to scenarios where multiple edits are needed for the same input (multi-question composition & multi-answer composition), which needs to avoid interference between useful edits. On the other hand, our work expands generalization in two ways. From the input perspective, we explore how to make an edit generalize to scenarios where only part of the input requires the edit (multi-question composition). From the output perspective, we explore how to make an edit generalize to scenarios when there is some content that corresponds to multiple other edits in the output (multi-question composition & multi-answer composition). This work takes model editing a step further towards more complete answers and more complex questions. It is noteworthy that a seemingly related

work [40] considers the success of a single edit in a multi-hop question with reasoning, instead of compositional question/answers that require multiple edits in our focus.

Model Merging. Model merging aims to merge the weights of different models while retaining their respective knowledge. However, due to permutation symmetry [41] and disjoint optimization trajectories [42], simply averaging two models can lead to a catastrophic drop in their performance. RegMean [43] proposed a closed-form solution by solving a local linear regression problem for each linear layer, but it requires additional data transmission and inference overhead. Fisher merging [44] weighs the parameters in each model with the Fisher Information Matrix [45, 46], but suffers from memory overhead and conflicts during the merging of multiple models. Recently, since different fine-tuned models initialized from the same pre-trained model effectively share a part of the optimization trajectory, model merging methods based on pre-trained models, such as TIES-Merging [34], task arithmetic in tangent space [47], and CoID Fusion [48], have achieved leading results. Our method draws inspiration from these works but differs in that our proposed compositional model editing aims to enable the model to utilize model merging to use multiple edited knowledge simultaneously during the inference.

C Compositional vs. Collect & Edit

compositional model editing addresses a different scenario where: during the edit training for “fever” and the edit training for “loss of taste”, **they are unaware of each other**. This means we cannot collect all of the knowledge to be edited and then directly compose them at text level. We will explain why this scenario is meaningful below.

As stated in Sec. 2.1 (cf. Lines 97–99) and illustrated in Fig. 1, we believe that it is necessary to **support model editing in a federated manner**. Specifically, when an open-source model is deployed globally, numerous distinct errors may arise simultaneously across different deployed regions. We want these deployers can benefit from the edits of each other.

- In such cases, similar to the assumption that raw data cannot be shared in federated learning due to regulations like GDPR [49], **different deployers are only able to share trained edits**. In such cases, when a deployer receives trained edits from other deployers, it does not know the questions of the edits. **It is impossible to get complete answers during edit training** because:
 1. The deployer has to iterate through all previous edit questions to determine which previous edits need to be composed with the newly received edit to form a complete answer.
 2. However, storing all previous edit questions is **impossible in lifelong model editing**.
- When exchanging raw data is allowed, **local deployers still have to perform immediate model editing to mitigate potential adverse effects without waiting to receive errors discovered by other deployers**. In this scenario, assuming that each of n deployers updates an answer to the same question and exchanges these n different answers with each other.
 - Continuously injecting complete knowledge would require n **times edit training for each deployer at most**.
 - Exchanging independently trained edits would only require 1 **time edit training for each deployer**.

Training edits independently while ensuring their composability supports model editing in a federated manner **much more efficiently**. This is especially important in the field of model editing, which emphasizes **immediate correction of errors**.

Meanwhile, for **multi-question composition**, compositional model editing is clearly meaningful because,

- Edits for any two different questions may need to be composed during testing.
- Different errors to be edited may be discovered at different time.

Therefore, it is impossible to exhaustively cover all possible composition cases for complete knowledge during the edit training process.

D The Design of Vector Database

To enhance the *edit composing* stage, we adapt the vector database in [21, 22] in a simple but effective way. We retrieve useful edited knowledge with a threshold γ to ensure locality and a top- p selection to balance the completeness of knowledge and interference from useless edits:

$$\mathcal{S} = \{(\mathbf{A}, \mathbf{B}) | (\mathbf{K}, \mathbf{A}, \mathbf{B}) \in \mathcal{S}_{\text{db}} \text{ and } h_{\text{db}} \mathbf{K}^\top > \gamma \text{ and } h_{\text{db}} \mathbf{K}^\top \in \text{top-}p(h_{\text{db}} \mathbf{K}^\top)\}, \quad (9)$$

where $\mathcal{S}$ represents the retrieved edits from the vector database $\mathcal{S}_{\text{db}}$ with $\mathbf{K} \in \mathbb{R}^{1 \times m}$ as the key and $(\mathbf{A}, \mathbf{B})$ as the value. $\mathbf{K}$ is saved from a selected hidden state of the edit question. $h_{\text{db}} \in \mathbb{R}^{1 \times m}$ represents the selected hidden state of the test question, which are hyperparameters. We use h_{db} to retrieve edits by matching $\mathbf{K}$. Because of the mask $\mathbf{M}$, we only need to save the unmasked $\mathbf{B}$ and the unmasked position. Please refer to Alg. 1 and Alg. 2 for the construction and utilization of $\mathcal{S}_{\text{db}}$. Please also refer to Alg. 2 for the complete test procedure.

E Handling Continuous Editing Situations

When adding an edit for “loss of taste,” if we recognize that the existing edit for “fever” is outdated, **our method can easily remove the outdated edit** with the following locate-and-delete algorithm:

1. Iterate and mask each edit $(\mathbf{A}, \mathbf{B})$ in the retrieved part $\mathcal{S}$ of the vector database during the generation process $\rightarrow$ locate the edit that leads to the outdated answer.
2. Delete the located edit.

We consider a three-answer composition scenario with 3×50 successful edits to test the effectiveness of the locate-and-delete algorithm above, where each question receives two correct edits and one outdated edit. **The accuracy of deleting the outdated edits is 100%**. We encourage the construction of outdated edits feedback and avoidance mechanisms when deploying A³E.

F Computational and Memory Overhead

In Tab. 4 and Tab. 5, we provide a more detailed analysis of the computational and memory overhead compared to baselines with Llama3-8B and Llama3-70B from four aspects: **storage overhead**, **trainable parameters**, **extra FLOPs for inference (infer.)**, **extra FLOPs for training (train.)**. We only provide the extra FLOPs for training of T-patcher, GRACE, MELO and A³E because other methods that update the whole matrix or train a hypernetwork have significantly larger training overhead. Specifically,

- **Storage overhead** refers to the additional parameters that need to be stored compared to the pre-trained model for 1000 edits.
- **Trainable parameters** denote the parameters that are updated during training for each edit.
- **Infer.** denotes extra FLOPs compared to the pre-trained model for the inference of u tokens with v input tokens.
- **Train.** denotes extra FLOPs compared to the vanilla forward-backward progress with cross-entropy loss for w training steps of an edit with z -token answer.

For **Llama3-8B**,

We also provide the extra FLOPs of the proposed adaptive combination, adaptive regularization and adaptive merging for inference and training:

- Adaptive merging of A³E: $1.64u \times 10^{-5}$ GFLOPs
- Adaptive combination of A³E: $0.03z$ GFLOPs
- Adaptive regularization of A³E: $1.28wz \times 10^{-4} + 1.64w \times 10^{-5}$ GFLOPs

In A³E, w is set to 50 and $z < u$ for a instance. The extra FLOPs for training and inference are very small compared to the generation of u tokens with v input tokens with an 8B model, which is about $16u + 16v$ GFLOPs.

Table 4: Computational and Memory Overhead in Llama3-8B.

Method	Storage Overhead	Trainable Parameters	infer. (GFLOPs)	train. (GFLOPs)
FT	0	0.54GB	0	-
ROME	1.64GB	0.54GB	0	-
KE	0.72GB	0.72GB	0	-
MEND	1.14GB	1.14GB	0	-
AlphaEdit	1.64GB	0.54GB	0	-
WISE	0.11GB	0.11GB	$0.27v + 0.27u$	-
T-patcher	0.07GB	32.77KB	$0.02u$	$0.01wz$
GRACE	0.07GB	32.77KB	0.02	0
MELO	0.13GB	98.30KB	0.02	0
A ³ E	0.07GB	39.94KB	$0.02 + 4.92u \times 10^{-5}$	$0.03z + 1.28wz \times 10^{-4} + 1.64w \times 10^{-5}$

Table 5: Computational Overhead in Llama3-70B.

Method	Trainable Parameters	infer. (GFLOPs)	train. (GFLOPs)
FT	1.88GB	0	-
ROME	1.88GB	0	-
KE	2.52GB	0	-
MEND	2.09GB	0	-
AlphaEdit	1.88GB	0	-
WISE	0.38GB	$0.94v + 0.94u$	-
T-patcher	65.54KB	$0.03u$	$0.02wz$
GRACE	65.54KB	0.03	0
MELO	180.22KB	0.03	0
A ³ E	72.71KB	$0.03 + 1.19u \times 10^{-4}$	$0.09z + 1.28wz \times 10^{-4} + 2.87w \times 10^{-5}$

For Llama3-70B,

We also provide the extra FLOPs of the proposed adaptive combination, adaptive regularization and adaptive merging for inference and training:

- Adaptive merging of A³E: $1.19u \times 10^{-4}$ GFLOPs
- Adaptive combination of A³E: $0.09z$ GFLOPs
- Adaptive regularization of A³E: $1.28wz \times 10^{-4} + 2.87w \times 10^{-5}$ GFLOPs

In A³E, w is set to 50 and $z < u$ for a instance. The extra FLOPs for training and inference are very small compared to the generation of u tokens with v input tokens with an 70B model, which is about $140u + 140v$ GFLOPs.

G Experiment Details

For the baselines, we follow the reproduction code and hyperparameters as described in [11]. For A³E, we set the unmasked size k to 896 for PEAK-CF and to 448 for PEAK-T. We set the loss weight α to 8 and β to 1 to balance their utilities. We store the output of the 5-th down projection layer at the last subject token as $\mathbf{K}$ and h_{db} . We set the edited FFN layer to 31, the learning rate to 0.01, and train each edit for 50 epochs. To better evaluate the priority of different answers within the model, for all baselines and A³E, we assign a penalty of 10 to the answers that have already been generated. How to achieve effective compositional model editing under both instruction-tuned models and conventional inference settings remains an area requiring urgent exploration. For GRACE, MELO and A³E, we use the vector database in Appendix D with the same $p = 4$ and $\gamma = 0.5$. The experiments are conducted on a server with 8 NVIDIA RTX 5880 Ada GPUs.

For the benchmark construction with PEAK-CF as an example: (1) The original PEAK-CF is a model editing dataset containing questions with multiple answers. (2) We queried Llama3-8B and Mistral-7B, selecting questions from PEAK-CF where both models missed four or more answers, along with their corresponding missing answers in PEAK-CF. (3) For multi-answer composition, the retained questions and their randomly selected c missing answers form a test instance, where c is the composition number. (See Tab. 17 for an example) (4) For multi-question composition, we randomly selected c retained questions without repetition and one randomly chosen missing answer per question to form a test instance. (See Tab. 19 for an example)

H Discussions of *SR-S*

Fig. 10 shows a correct example and a false example of *SR-S*. *SR-S* aims to accurately measure the interference between different edits to evaluate the composability without considering the case of outputting unrelated answers after all correct answers. This is to mitigate the confounding effect of conventional model editing, where “outputting unrelated answers after correct answers” is also a problem, and warrant fairness in evaluating composability. We report the metric *SR-AS* when “outputting unrelated answers after all correct answers” is considered as an error in Table 6. A³E maintains a clearly superior performance.

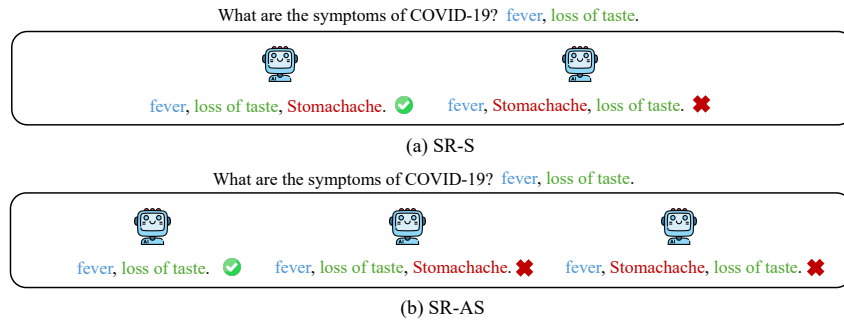


Figure 10: The illustration of (a) *SR-S* and (b) *SR-AS*. The text in blue, red, green represents the first correct answer, the wrong answer and the second correct answer, respectively.

Table 6: Performance measured by *SR-S* and *SR-AS* in 1) IMAC (3898 edits) and MAC (3898 edits) on PEAK-CF. 2) IMAC (1984 edits) and MAC (1984 edits) on PEAK-T. 3) IMQC (1948 edits) and MQC (1948 edits) on PEAK-CF. 4) IMQC (992 edits) and MQC (992 edits) on PEAK-T.

Method	PEAK-CF				PEAK-T			
	IMAC		MAC		IMAC		MAC	
	SR-S↑	SR-AS↑	SR-S↑	SR-AS↑	SR-S↑	SR-AS↑	SR-S↑	SR-AS↑
FT	0.31	0.26	0.00	0.00	0.10	0.00	0.00	0.00
ROME	0.92	0.62	0.00	0.00	0.30	0.05	0.00	0.00
KE	0.92	0.41	0.00	0.00	0.10	0.00	0.00	0.00
MEND	2.36	1.95	0.00	0.00	1.01	0.05	0.00	0.00
AlphaEdit	5.44	4.67	6.98	5.23	2.02	1.64	0.50	0.30
WISE	<u>39.99</u>	<u>30.12</u>	0.10	0.00	<u>23.08</u>	<u>17.97</u>	0.20	0.10
T-patcher	26.64	18.98	0.00	0.00	20.77	16.74	0.00	0.00
GRACE	15.20	11.90	12.99	10.31	6.55	5.13	5.65	4.31
MELO	19.46	15.03	<u>18.22</u>	<u>14.73</u>	9.48	7.60	<u>6.96</u>	<u>5.54</u>
A ³ E	48.97	40.69	43.38	35.92	41.63	33.68	38.00	30.90

Method	PEAK-CF				PEAK-T			
	IMQC		MQC		IMQC		MQC	
	SR-S↑	SR-AS↑	SR-S↑	SR-AS↑	SR-S↑	SR-AS↑	SR-S↑	SR-AS↑
FT	5.64	3.70	0.00	0.00	0.00	0.00	0.00	0.00
ROME	23.49	18.99	0.00	0.00	19.15	15.12	0.00	0.00
KE	1.03	0.81	0.00	0.00	4.23	3.53	0.00	0.00
MEND	5.23	3.59	0.00	0.00	14.11	10.28	0.00	0.00
AlphaEdit	7.69	6.16	5.33	4.31	16.33	12.70	<u>5.65</u>	<u>4.54</u>
WISE	<u>46.56</u>	<u>37.68</u>	0.00	0.00	<u>27.62</u>	<u>21.67</u>	0.20	0.00
T-patcher	33.23	26.39	0.00	0.00	26.81	20.26	0.00	0.00
GRACE	6.56	4.93	3.59	2.77	6.05	4.54	4.84	3.93
MELO	20.31	15.61	<u>9.13</u>	<u>7.39</u>	6.25	5.04	2.82	2.28
A ³ E	52.51	43.02	26.46	22.07	46.17	37.40	28.63	24.19

I The Sufficiency of Editing $\mathbf{W}_{up}$

I.1 For higher *SR-S*

From an intuitive perspective, new information does not appear out of thin air, it will be related to past information to some extent.

From an empirical perspective, we have shown in Fig. 4(a) that using existing columns to form new knowledge (edit $\mathbf{W}_{up}$) can achieve results comparable to or even slightly better than editing $\mathbf{W}_{down}$ alone, as well as editing both $\mathbf{W}_{up}$ and $\mathbf{W}_{down}$ simultaneously. In Fig. 4(b), we conduct the experiment above with fake relation in the input (e.g. share border with -> share boodee with) and fake answer (e.g. Syria -> Sykie) to simulate unseen knowledge. It can be seen that the performance does not change a lot. These experiments indicate that editing $\mathbf{W}_{up}$ is enough to guarantee the performance.

From a mathematical perspective, $\mathbf{W}_{down}$ is often of full rank, which to some extent guarantees its expressive power in its output space.

I.2 For less *knowledge loss*

From an intuitive perspective, the distinction between pretrained knowledge and edited knowledge from existing model editing methods lies in the fact that pretrained knowledge is the combination of pretrained foundation knowledge in $\mathbf{W}_{down}$, whereas existing model editing methods directly manipulate the output of $\mathbf{W}_{down}$. Meanwhile, we know that pretrained models can leverage pretrained knowledge to answer questions requiring multiple answers, but when using multiple edited knowledge, they exhibit significant knowledge loss. Therefore, we intuitively hypothesize that knowledge loss may partly stem from the fact that edited knowledge is not the combination of pretrained foundation knowledge.

From an empirical perspective, we observe that editing $\mathbf{W}_{up}$ (i.e., combining pretrained foundation knowledge into edited knowledge) results in less min knowledge loss across 27-31 layers and across baselines compared to editing $\mathbf{W}_{down}$ or editing both $\mathbf{W}_{up}$ and $\mathbf{W}_{down}$ in Tab. 7 while leading to higher *SR-S* and similar non-compositional performance (Fig. 4).

Table 7: Min knowledge loss across 27-31 layers and across baselines when editing $\mathbf{W}_{up}$, $\mathbf{W}_{down}$, $\mathbf{W}_{up}$ and $\mathbf{W}_{down}$.

Method	Min knowledge loss (%)
$\mathbf{W}_{up}$	60.6
$\mathbf{W}_{down}$	71.4
$\mathbf{W}_{up}$ and $\mathbf{W}_{down}$	69.7

J Additional Experimental Results

J.1 The effect of the composition number and the number of edits

In Fig. 11, we provide complete experimental results for the effect of different number of edits and the composition number in MAC and MQC on PEAK-CF and PEAK-T. In PEAK-CF dataset, we also conducted additional experiments by sampling 50 questions with 10+ unknown answers each and evaluated the performance with a composition number of 10.

Table 8: Experimental results on PEAK-CF dataset with composition number 10.

Method	ROME	AlphaEdit	WISE	T-patcher	GRACE	MELO	A ³ E
SR-1	2.40	4.60	9.40	6.60	8.60	12.20	17.20

As shown in Tab. 8, the experimental results demonstrate that when a large number of edits need to be applied simultaneously, A³E still significantly outperforms the baselines on the *SR-I* metric, which means A³E retains more edited knowledge with large composition numbers.

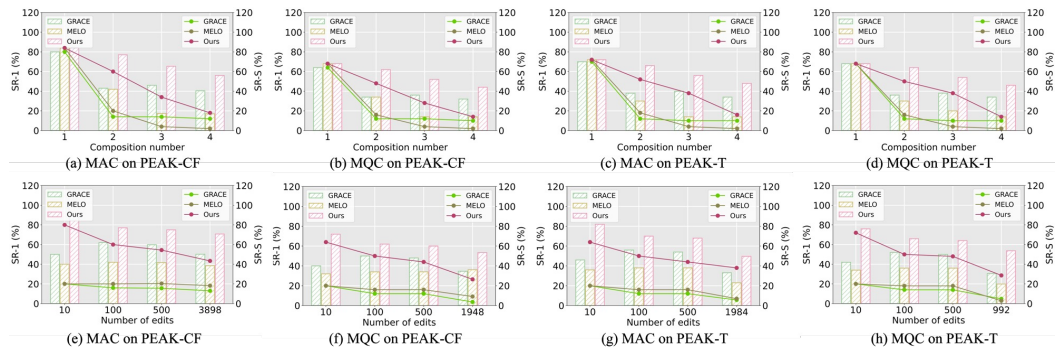


Figure 11: (a)-(d) Performance with different composition number measured by $SR-S$ and $SR-I$ in MAC and MQC on PEAK-CF and PEAK-T. (e)-(h) Performance change with different number of edits measured by $SR-S$ and $SR-I$ in MAC and MQC on PEAK-CF and PEAK-T.

J.2 The effect of hyperparameters

The hyperparameters of A^3E are not sensitive and are easy to adjust. First, as shown in Fig. 12(a)-12(d), compared with the performance improvement of A^3E , its performance is not sensitive to the selection of α , β , γ , and p . Second, the adjustment of hyperparameters is straightforward for two reasons: 1) As shown in Fig. 12(a)-12(e), all hyperparameters exhibit similar behavior across different datasets (i.e., PEAK-CF and PEAK-T), indicating that a one-time hyperparameter search is almost sufficient, and repeated re-tuning can be avoided. 2) As shown in Fig. 12(a)-12(b), there exist correlations among certain hyperparameters, e.g., setting $\alpha = 8\beta$ often leads to the best performance. This enables joint tuning, thereby reducing the number of hyperparameters that need to be adjusted.

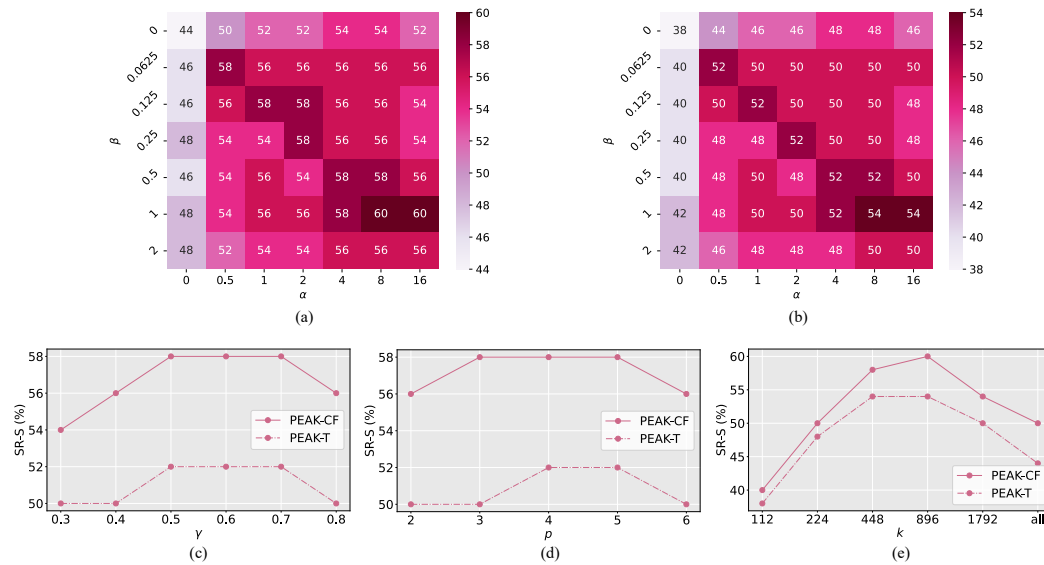


Figure 12: The effect of (a) α , β on the performance measured by $SR-S$ in MAC on PEAK-CF, (b) α , β on the performance measured by $SR-S$ in MAC on PEAK-T. The effect of (c) γ , (d) p and (e) k on the performance measured by $SR-S$ in MAC on PEAK-CF and PEAK-T.

J.3 The effect of different backbones

Table 9 shows the performance with backbone Mistral-7B in four scenarios on two datasets.

Table 9: Performance with Mistral-7B measured by metrics in Sec. 5.1 in 1) IMAC (3898 edits) and MAC (3898 edits) on PEAK-CF. 2) IMAC (1984 edits) and MAC (1984 edits) on PEAK-T. 3) IMQC (1948 edits) and MQC (1948 edits) on PEAK-CF. 4) IMQC (992 edits) and MQC (992 edits) on PEAK-T.

METHOD	PEAK-CF								PEAK-T							
	IMAC			MAC				LSR↑	IMAC			MAC				LSR↑
	SR-S↑	SR-I↑	GSR↑	SR-S↑	SR-I↑	GSR↑	LSR↑		SR-S↑	SR-I↑	GSR↑	SR-S↑	SR-I↑	GSR↑	LSR↑	
FT	0.26	10.19	0.31	0.00	0.00	0.00	4.93		0.00	2.03	0.32	0.00	0.00	0.00	2.10	
ROME	0.77	13.83	0.71	0.00	0.52	0.00	11.51		14.44	17.61	11.83	0.00	0.00	0.00	3.59	
KE	0.76	8.74	0.73	0.00	0.00	0.00	9.88		3.19	5.08	2.55	0.00	0.00	0.00	3.85	
MEND	1.97	19.58	1.66	0.00	0.00	0.00	11.61		10.64	11.06	7.14	0.00	0.00	0.00	3.98	
ALPHAEDIT	4.54	26.30	3.99	5.75	27.99	5.20	15.06		12.31	11.63	9.03	<u>6.27</u>	19.41	<u>6.95</u>	4.15	
WISE	<u>33.91</u>	<u>58.70</u>	<u>32.68</u>	0.10	4.57	0.10	14.98		19.25	<u>40.93</u>	17.84	0.00	1.21	0.00	8.02	
T-PATCHER	22.22	44.56	11.19	0.00	0.59	0.00	0.95		<u>20.21</u>	31.68	<u>20.53</u>	0.00	0.00	0.00	0.32	
GRACE	12.68	48.72	12.68	10.70	<u>42.24</u>	8.22	<u>23.55</u>		4.56	20.22	4.51	5.37	<u>23.61</u>	5.00	<u>40.14</u>	
MELO	15.24	35.25	14.52	<u>13.80</u>	32.99	<u>13.23</u>	23.11		7.86	23.99	7.60	5.75	18.95	5.24	34.76	
A ³ E	40.84	64.03	39.86	35.73	59.55	33.30	24.21		34.80	43.43	31.21	31.75	41.53	30.04	40.88	

METHOD	PEAK-CF								PEAK-T							
	IMQC			MQC				LSR↑	IMQC			MQC				LSR↑
	SR-S↑	SR-I↑	GSR↑	SR-S↑	SR-I↑	GSR↑	LSR↑		SR-S↑	SR-I↑	GSR↑	SR-S↑	SR-I↑	GSR↑	LSR↑	
FT	0.28	9.79	0.31	0.00	0.00	0.00	5.24		0.00	2.71	0.00	0.00	0.00	0.00	2.06	
ROME	0.82	13.28	0.79	0.00	0.40	0.00	12.55		15.89	18.35	12.15	0.00	0.00	0.00	3.53	
KE	0.82	8.40	0.69	0.00	0.00	0.00	12.49		3.51	6.75	2.62	0.00	0.00	0.00	2.90	
MEND	2.12	18.82	1.82	0.00	0.36	0.00	12.73		11.71	14.69	7.34	0.00	0.00	0.00	3.92	
ALPHAEDIT	4.88	25.26	4.38	3.60	21.03	2.82	15.70		13.55	15.44	9.28	<u>4.74</u>	21.01	<u>4.44</u>	4.00	
WISE	<u>38.96</u>	<u>58.73</u>	<u>37.27</u>	0.00	2.71	0.00	7.32		<u>23.39</u>	<u>27.42</u>	20.56	0.00	2.82	0.00	4.20	
T-PATCHER	23.90	42.80	12.28	0.00	0.37	0.00	1.90		22.25	26.53	<u>21.09</u>	0.00	0.00	0.00	0.34	
GRACE	13.64	46.80	13.91	6.69	<u>31.74</u>	4.46	<u>44.85</u>		5.02	26.85	4.64	4.06	<u>25.56</u>	3.20	<u>37.57</u>	
MELO	16.94	36.96	15.40	<u>7.60</u>	30.29	<u>6.88</u>	42.10		5.44	19.76	4.84	2.42	16.73	1.81	36.20	
A ³ E	43.94	61.50	43.74	22.33	44.76	18.06	45.33		38.31	57.66	32.06	23.99	44.96	19.15	40.27	

J.4 The effect of different prompts

Table 10-11 shows the performance of A³E and state-of-the-art baselines WISE, GRACE, MELO with different prompts in four scenarios on two datasets.

Table 10: Performance with prompt 2 measured by metrics in Sec. 5.1 in 1) IMAC (3898 edits) and MAC (3898 edits) on PEAK-CF. 2) IMAC (1984 edits) and MAC (1984 edits) on PEAK-T. 3) IMQC (1948 edits) and MQC (1948 edits) on PEAK-CF. 4) IMQC (992 edits) and MQC (992 edits) on PEAK-T.

Method	PEAK-CF								PEAK-T							
	IMAC				MAC				IMAC				MAC			
	SR-S↑	SR-1↑	GSR↑	SR-S↑	SR-1↑	GSR↑	LSR↑	SR-S↑	SR-1↑	GSR↑	SR-S↑	SR-1↑	GSR↑	LSR↑		
FT	0.26	9.97	0.30	0.00	0.00	0.00	5.36	0.09	1.59	0.09	0.00	0.00	0.00	3.15		
ROME	0.76	13.54	0.68	0.00	0.51	0.00	12.17	0.50	15.15	0.25	0.00	0.00	0.00	6.17		
KE	0.75	8.48	0.59	0.00	0.00	0.00	10.65	0.09	4.18	0.09	0.00	0.00	0.00	4.79		
MEND	1.92	18.80	1.55	0.00	0.45	0.00	12.06	0.82	10.00	0.66	0.00	0.00	0.00	5.37		
AlphaEdit	4.60	26.21	3.86	5.90	28.10	5.21	15.82	1.71	18.41	1.80	0.42	13.76	0.34	8.14		
WISE	<u>32.81</u>	<u>53.54</u>	<u>31.67</u>	0.13	4.59	0.12	15.46	<u>18.88</u>	<u>44.78</u>	<u>17.72</u>	0.27	2.51	0.27	8.04		
T-patcher	22.43	44.30	10.77	0.05	0.44	0.04	1.81	17.46	36.71	15.49	0.00	0.00	0.00	0.70		
GRACE	12.66	48.04	12.10	10.88	<u>41.96</u>	8.21	22.73	5.39	34.98	5.22	4.62	<u>27.81</u>	4.32	<u>37.57</u>		
MELO	17.75	40.52	16.84	<u>16.68</u>	<u>35.20</u>	<u>15.31</u>	<u>25.05</u>	8.58	26.35	8.45	<u>6.26</u>	21.11	<u>6.11</u>	<u>35.07</u>		
A ³ E	47.88	73.86	44.68	42.48	69.20	38.70	25.88	40.66	50.40	36.25	37.13	48.68	35.26	40.03		

Method	PEAK-CF								PEAK-T							
	IMQC				MQC				IMQC				MQC			
	SR-S↑	SR-1↑	GSR↑	SR-S↑	SR-1↑	GSR↑	LSR↑	SR-S↑	SR-1↑	GSR↑	SR-S↑	SR-1↑	GSR↑	LSR↑		
FT	4.68	17.98	4.00	0.00	0.00	0.00	2.73	0.01	2.68	0.34	0.00	0.00	0.00	2.76		
ROME	19.51	39.10	16.09	0.00	0.34	0.00	4.75	15.92	18.20	12.07	0.00	0.00	0.00	3.02		
KE	0.85	12.79	0.64	0.00	0.00	0.00	3.25	3.49	6.63	2.58	0.00	0.00	0.00	2.45		
MEND	4.21	18.05	4.07	0.00	0.00	0.00	3.04	11.5	14.28	7.15	0.00	0.00	0.00	4.28		
AlphaEdit	6.50	24.15	4.76	4.51	21.06	3.56	6.85	13.81	15.58	9.38	<u>4.77</u>	21.23	<u>4.26</u>	4.48		
WISE	<u>38.14</u>	<u>57.51</u>	<u>36.46</u>	0.09	2.55	0.08	7.90	<u>22.51</u>	<u>26.66</u>	20.52	0.06	2.96	0.06	4.88		
T-patcher	27.95	50.93	26.32	0.09	0.11	0.08	0.29	22.45	26.47	<u>21.58</u>	0.00	0.22	0.00	0.53		
GRACE	5.35	32.55	4.25	3.10	28.88	2.48	38.33	4.87	26.57	4.94	3.83	<u>25.81</u>	3.24	<u>38.58</u>		
MELO	18.45	40.30	16.78	8.45	<u>32.99</u>	<u>7.60</u>	<u>39.48</u>	5.65	21.22	5.53	2.36	18.40	2.05	36.45		
A ³ E	51.28	71.83	51.19	26.01	52.22	19.97	44.69	45.02	67.14	37.81	27.82	52.84	21.52	40.77		

Table 11: Performance with prompt 3 measured by metrics in Sec. 5.1 in 1) IMAC (3898 edits) and MAC (3898 edits) on PEAK-CF. 2) IMAC (1984 edits) and MAC (1984 edits) on PEAK-T. 3) IMQC (1948 edits) and MQC (1948 edits) on PEAK-CF. 4) IMQC (992 edits) and MQC (992 edits) on PEAK-T.

Method	PEAK-CF								PEAK-T							
	IMAC			MAC					IMAC			MAC				
	SR-S↑	SR-1↑	GSR↑	SR-S↑	SR-1↑	GSR↑	LSR↑	SR-S↑	SR-1↑	GSR↑	SR-S↑	SR-1↑	GSR↑	LSR↑		
FT	0.28	9.33	0.24	0.00	0.00	0.00	5.08	0.08	1.49	0.08	0.00	0.00	0.00	3.95		
ROME	0.72	12.78	0.64	0.00	0.48	0.00	12.61	0.47	14.31	0.24	0.00	0.00	0.00	6.88		
KE	0.72	8.09	0.56	0.00	0.00	0.00	10.24	0.10	3.95	0.08	0.00	0.00	0.00	4.58		
MEND	1.83	17.92	1.47	0.00	0.43	0.00	12.58	0.80	9.81	0.65	0.00	0.00	0.00	5.28		
AlphaEdit	4.34	24.78	3.65	5.58	26.56	4.92	15.12	1.62	17.40	1.62	0.40	13.01	0.33	8.75		
WISE	<u>31.58</u>	<u>51.53</u>	<u>30.48</u>	0.13	4.42	0.12	15.99	<u>18.17</u>	<u>43.10</u>	<u>17.06</u>	0.26	2.42	0.26	8.78		
T-patcher	20.97	41.43	10.07	0.05	0.41	0.04	1.75	16.34	34.36	14.51	0.00	0.00	0.00	0.66		
GRACE	12.05	45.75	11.52	10.41	<u>39.95</u>	7.82	<u>27.64</u>	5.13	33.30	4.97	4.40	<u>26.49</u>	4.12	<u>37.07</u>		
MELO	13.36	30.52	12.68	<u>12.57</u>	<u>26.51</u>	<u>11.55</u>	<u>24.03</u>	6.44	19.82	6.41	<u>4.69</u>	<u>15.94</u>	<u>4.63</u>	<u>34.95</u>		
A ³ E	46.38	71.54	43.28	41.15	67.03	37.49	25.39	39.25	48.64	35.03	35.80	46.99	34.02	40.86		

Method	PEAK-CF								PEAK-T							
	IMQC			MQC					IMQC			MQC				
	SR-S↑	SR-1↑	GSR↑	SR-S↑	SR-1↑	GSR↑	LSR↑	SR-S↑	SR-1↑	GSR↑	SR-S↑	SR-1↑	GSR↑	LSR↑		
FT	4.38	16.81	3.74	0.00	0.00	0.00	2.62	0.01	2.50	0.32	0.00	0.00	0.00	2.65		
ROME	18.42	36.93	15.20	0.00	0.32	0.00	4.54	15.03	17.19	11.40	0.00	0.00	0.00	3.85		
KE	0.80	12.08	0.60	0.00	0.00	0.00	3.07	3.33	6.32	2.46	0.00	0.00	0.00	2.34		
MEND	4.13	17.71	3.99	0.00	0.00	0.00	3.98	10.96	13.61	6.81	0.00	0.00	0.00	4.12		
AlphaEdit	6.14	22.83	4.50	4.26	19.91	3.36	6.53	13.06	14.73	8.87	<u>4.51</u>	20.07	<u>4.02</u>	4.29		
WISE	<u>36.71</u>	<u>55.35</u>	<u>35.09</u>	0.09	2.45	0.08	7.67	<u>21.66</u>	<u>25.65</u>	19.77	0.36	2.86	0.20	4.74		
T-patcher	26.16	47.68	24.64	0.00	0.00	0.00	0.27	20.98	24.74	<u>20.20</u>	0.00	0.00	0.00	0.51		
GRACE	5.09	30.99	4.04	2.96	<u>27.49</u>	2.37	38.78	4.63	25.28	4.72	3.64	<u>24.59</u>	3.10	<u>38.04</u>		
MELO	13.87	30.34	12.61	<u>6.39</u>	24.84	<u>5.74</u>	<u>39.73</u>	4.26	16.13	4.11	1.89	13.76	1.45	37.08		
A ³ E	49.49	69.33	49.40	25.10	50.40	19.28	45.13	43.60	65.03	36.64	26.94	51.19	20.85	41.51		

J.5 Ablation study

In Table 12, we provide the comprehensive results of ablation study in four scenarios on two datasets.

Table 12: Ablation study measured by metrics in Sec. 5.1 in 1) IMAC (3898 edits) and MAC (3898 edits) on PEAK-CF. 2) IMAC (1984 edits) and MAC (1984 edits) on PEAK-T. 3) IMQC (1948 edits) and MQC (1948 edits) on PEAK-CF. 4) IMQC (992 edits) and MQC (992 edits) on PEAK-T.

Method	PEAK-CF								PEAK-T							
	IMAC				MAC				IMAC				MAC			
	SR-S $\uparrow$	SR-1 $\uparrow$	GSR $\uparrow$		SR-S $\uparrow$	SR-1 $\uparrow$	GSR $\uparrow$	LSR $\uparrow$	SR-S $\uparrow$	SR-1 $\uparrow$	GSR $\uparrow$		SR-S $\uparrow$	SR-1 $\uparrow$	GSR $\uparrow$	LSR $\uparrow$
A ³ E w/o l_o	44.07	67.94	41.12		39.04	63.65	35.53	23.19	37.47	46.46	33.21		34.20	44.64	32.30	40.97
A ³ E w/o l_c	39.18	60.39	36.55		34.70	56.58	31.59	23.61	33.31	41.30	29.52		30.40	39.68	28.71	40.77
A ³ E w/o combin.	37.54	57.88	35.03		33.26	54.22	30.27	23.15	32.13	39.83	28.47		29.32	38.28	27.70	40.61
A ³ E w/o merg.	38.60	59.51	36.02		34.20	55.75	31.12	24.31	32.47	40.26	28.78		29.64	38.69	28.00	40.95
A ³ E w/o norm	38.85	59.89	36.25		34.41	56.10	31.32	25.44	31.85	39.49	28.23		29.07	37.95	27.46	41.34
A ³ E w/ down	37.30	57.50	34.80		33.04	53.86	30.07	25.62	31.22	38.71	27.67		28.50	37.20	26.92	40.72
A ³ E w/ up&down	39.58	61.02	36.93		35.07	57.16	31.91	25.76	32.54	40.35	28.84		29.70	38.77	28.06	40.02
A ³ E	48.97	75.49	45.69		43.38	70.72	39.48	25.76	41.63	51.62	36.90		38.00	49.60	35.89	40.97

Method	PEAK-CF								PEAK-T							
	IMQC				MQC				IMQC				MQC			
	SR-S $\uparrow$	SR-1 $\uparrow$	GSR $\uparrow$		SR-S $\uparrow$	SR-1 $\uparrow$	GSR $\uparrow$	LSR $\uparrow$	SR-S $\uparrow$	SR-1 $\uparrow$	GSR $\uparrow$		SR-S $\uparrow$	SR-1 $\uparrow$	GSR $\uparrow$	LSR $\uparrow$
A ³ E w/o l_o	47.26	66.14	47.17		23.81	48.09	18.28	45.66	41.56	61.96	34.47		25.76	48.35	19.59	39.23
A ³ E w/o l_c	42.01	58.79	41.93		21.17	42.75	16.24	45.59	36.94	55.07	35.65		22.90	42.98	17.42	39.10
A ³ E w/o combin.	40.52	56.71	40.45		20.42	41.23	15.67	44.29	35.39	52.78	29.38		21.95	41.19	16.68	39.17
A ³ E w/o merg.	40.96	57.32	40.88		20.64	41.68	15.84	44.67	36.40	54.28	30.20		22.57	42.36	17.17	39.61
A ³ E w/o norm	40.17	56.22	40.10		20.24	40.88	15.53	44.99	36.63	54.62	30.39		22.71	42.63	17.28	40.82
A ³ E w/ down	39.38	55.12	39.31		19.84	40.08	15.23	44.30	35.17	52.44	29.18		21.80	40.93	16.59	38.51
A ³ E w/ up&down	41.04	57.45	40.97		20.68	41.77	15.87	44.75	37.32	55.65	30.97		23.14	43.44	17.60	38.33
A ³ E	52.51	73.49	52.41		26.46	53.44	20.31	45.73	46.17	68.85	38.31		28.63	53.73	21.77	41.37

J.6 Non-compositional ME evaluation on classic datasets for ME

As shown in Tab. 13, the non-compositional performance of A³E is on par with baselines.

Table 13: Non-compositional results on classic ME dataset ZsRE [50] and Counterfact [8] following [11] on Llama3-8B.

Method	ZsRE			Counterfact		
	SR-1 $\uparrow$	GSR $\uparrow$	LSR $\uparrow$	SR-1 $\uparrow$	GSR $\uparrow$	LSR $\uparrow$
FT	47.95	52.32	73.08	51.53	48.14	63.95
ROME	99.53	97.52	95.57	100.00	92.77	77.40
MEND	92.24	90.70	96.23	79.10	58.76	91.19
AlphaEdit	98.00	96.54	95.57	99.55	95.03	79.21
WISE	93.39	92.53	100.00	98.59	93.50	99.32
T-patcher	95.28	94.24	92.62	95.03	93.79	89.38
GRACE	96.69	94.00	100.00	99.77	94.12	100.00
MELO	95.28	94.08	100.00	98.53	92.88	100.00
A ³ E	97.00	96.77	100.00	99.77	93.79	100.00

J.7 Failure cases analysis

We conducted **failure case analysis for overlapping, contradictory, and capacity-exceeding edits** as follows:

- For **overlapping and contradictory cases**, we conducted statistical analysis on the PEAK-CF dataset under the IMAC scenario with 50 cases of two-edit composition, measuring:
 - The overlapping selection of pre-trained foundation knowledge, calculated by the overlapping length between non-zero Δa (Sec. 4.1) regions of two edits in adaptive merging (Sec. 4.2)
 - The contradictory utilization of pre-trained foundation knowledge, calculated by the length of regions with different signs between Δa (Sec. 4.1) of two edits in adaptive merging (Sec. 4.2)

The table below presents how the *SR-S* metric, which is inversely proportional to the proportion of failure cases, varies across different overlapping and contradictory ranges. As the overlapping length and contradictory length increase, the *SR-S* of A³E gradually decreases, but it remains significantly higher than the baselines.

Table 14: *SR-S* across different overlapping ranges.

Overlapping	[0, 500)	[500, 1000)	[1000, 1500)	[1500, +∞)
SR-S	75.00	66.67	45.45	50.00

Table 15: *SR-S* across different contradictory ranges.

Contradictory	[0, 200)	[200, 400)	[400, 600)	[600, +∞)
SR-S	66.67	68.75	57.14	46.67

Table 16: *SR-S* comparison with baselines.

Baselines	MEND	ROME	AlphaEdit	WISE	T-patcher	GRACE	MELO
SR-S	4.00	4.00	8.00	32.00	20.00	14.00	20.00

- For **capacity-exceeding cases**, we found that the failure cases of baselines and A³E in both compositional model editing and non-compositional model editing scenarios often involve numerical data in answers. For example,
 - **Edit 1:** Rheinmetall, which has designed **12 cm leFH 08**
 - **Test 1:** Rheinmetall, which has designed **12 cm leFH 8** (failure)
 - **Edit 2:** Rheinmetall, which has designed **35.5 cm Haubitze M1**
 - **Test 2:** Rheinmetall, which has designed **35.5 cm Haubitze M1** (success)
 - **Composition test:** Rheinmetall, which has designed **12 cm leFH, 12 cm Haubitze M8** (failure)

This may stem from the model’s inherent inability to accurately distinguish numerical data. Therefore, addressing such inherent limitations of the model is a valuable direction for future work in model editing.

We believe these findings can guide future research to further improve edit composability.

J.8 Generalization

The potential generalization limitations are as follows:

- **Noisy scenarios.** In Fig. 8(d), we tested the impact of different few-shot prompts on A³E’s performance. These varied few-shot prompts not only controlled the model’s output but also simulated different noise present in the context. The results show that while A³E’s performance is affected, it still maintains a leading performance compared to the baselines.
- **Noisy open-domain scenarios.** We humbly acknowledge that in open-domain scenarios, noisy contexts may still impact the retrieval accuracy of the vector database. Since A³E primarily focuses on edit training and edit composing, we will explore
 - optimizing the hidden state positions for retrieval based on the real data distribution in application scenarios
 - employing more powerful embedding models
 - training a question-rewriting module to better align with the vector database

for optimal performance in the future work.

- **Adversarial scenarios.** We conducted additional experiments using 2×50 successful edits from the PEAK-CF dataset under the IMAC setting by providing the tested question with incorrect answers in the few-shot prompts. The results demonstrate that A³E consistently (100%) outputs the edited answers across all test cases, proving its robustness against contextual interference.

Table 17: An example of multi-answer composition containing two pieces of edited knowledge, where the blue text indicates the question and the rephrased question.

Question	Bertrand Russell is the author of
Answer	Power: A New Social Analysis, A History of Western Philosophy
Edit 1 x_e	Bertrand Russell is the author of
Edit 1 y_e	Power: A New Social Analysis
Edit 2 x_e	Bertrand Russell is the author of
Edit 2 y_e	A History of Western Philosophy
Test $\mathcal{X}_t$	Tim Dorsey, who has written the Cadillac Beach, Nuclear Jellyfish, Triggerfish Twist, Hammerhead Ranch Motel, The Big Bamboo, Orange Crush (novel), Hurricane Punch, The Stingray Shuffle, Atomic Lobster, Torpedo Juice (novel), Florida Roadkill.\n Jerusalem, which is the partner town of NYC, New York, Praha, New York City, United States, Rio de Janeiro, NY, New York, NY, Prague, New York City, Tehran, Buenos Aires, Moscow, Manhattan.\n Pushkin is the author of The Fountain of Bakhchisaray, Eugene Onegin, The Tale of the Fisherman and the Fish, Poltava (poem), The Tale of the Golden Cockerel, Dubrovsky (novel), The Belkin Tales, Onegin, The Stone Guest (play), The Bronze Horseman (poem), The Queen of Spades (story), The Tale of the Priest and of His Workman Balda, The Gypsies, The Blizzard, Tatiana Larina, The Tale of the Dead Princess and the Seven Knights.\n WWE is the owner of WWE Classics on Demand, FCW Florida Heavyweight Championship, WWE Studios, WWE Films, WWE Network, NXT, FCW, WWE Classics On Demand, FCW Southern Heavyweight Championship, WCW, World Championship Wrestling, WCW, Inc., Florida Championship Wrestling, NXT Wrestling, Universal Wrestling Corporation, WWE NXT.\n Bertrand Russell is the author of
Test $\mathcal{Y}_t$	Power: A New Social Analysis, A History of Western Philosophy
Rephrased test	Tim Dorsey, who has written the Cadillac Beach, Nuclear Jellyfish, Triggerfish Twist, Hammerhead Ranch Motel, The Big Bamboo, Orange Crush (novel), Hurricane Punch, The Stingray Shuffle, Atomic Lobster, Torpedo Juice (novel), Florida Roadkill.\n Jerusalem, which is the partner town of NYC, New York, Praha, New York City, United States, Rio de Janeiro, NY, New York, NY, Prague, New York City, Tehran, Buenos Aires, Moscow, Manhattan.\n Pushkin is the author of The Fountain of Bakhchisaray, Eugene Onegin, The Tale of the Fisherman and the Fish, Poltava (poem), The Tale of the Golden Cockerel, Dubrovsky (novel), The Belkin Tales, Onegin, The Stone Guest (play), The Bronze Horseman (poem), The Queen of Spades (story), The Tale of the Priest and of His Workman Balda, The Gypsies, The Blizzard, Tatiana Larina, The Tale of the Dead Princess and the Seven Knights.\n WWE is the owner of WWE Classics on Demand, FCW Florida Heavyweight Championship, WWE Studios, WWE Films, WWE Network, NXT, FCW, WWE Classics On Demand, FCW Southern Heavyweight Championship, WCW, World Championship Wrestling, WCW, Inc., Florida Championship Wrestling, NXT Wrestling, Universal Wrestling Corporation, WWE NXT.\n Bertrand Russell is the writer of
Locality test	Philosophy, which is played by

- **Larger models and black-box models.** A³E is applicable to larger LLMs. Due to limitations in computational resources, we provide experimental results for GRACE, MELO, and A³E under the MAC scenario using Llama3-70B and the PEAK-CF dataset. As shown in Table 18, A³E maintains the leading performance.

Table 18: MAC and MQC performance on PEAK-CF with Llama3-70B.

Method	MAC				MQC			
	SR-S $\uparrow$	SR-1 $\uparrow$	GSR $\uparrow$	LSR $\uparrow$	SR-S $\uparrow$	SR-1 $\uparrow$	GSR $\uparrow$	LSR $\uparrow$
GRACE	16.32	42.48	11.24	24.02	3.59	37.28	3.62	38.16
MELO	28.12	48.64	26.58	23.18	12.13	39.27	11.33	41.41
A ³ E	50.69	69.47	47.10	25.10	33.12	55.65	32.98	44.26

However, A³E cannot be applied to black-box models. In the field of knowledge editing for large language models, one research direction focuses on editing white-box models, i.e., model editing [22, 21, 20, 19]. The access to model weights is a prerequisite for applying these editing techniques. Another research direction addresses knowledge editing for extremely large-scale models where model editing may be prohibitively expensive or black-box models by performing edits at the text level, including [51, 52, 13, 40, 53, 54, 55, 56, 57, 58, 59, 60]. A³E follows the first

direction to edit white-box models and focuses on compositional model editing because text-level knowledge editing is not applicable for the following reason:

- As stated in Sec. 2.1 and illustrated in Fig. 1, we aim to support model editing in a federated manner [61, 62, 63, 64]. Specifically, when a model is deployed globally, numerous distinct errors may arise simultaneously across different deployed regions. We want these deployers can benefit from the edits of each other.
- In such cases, similar to the assumption that raw data cannot be shared in federated learning due to regulations like GDPR [49], different deployers are only able to share trained edits rather than the texts.

In a centralized scenario, A³E can also be implemented by the owner of black-box models for immediate error correction with much more composable edits. We recommend users to choose methods from either direction based on their needs, and we will explore how to perform compositional knowledge editing at the text level for black-box models in our future work.

K Description of Dataset and Few-shot Prompts

In Table 17 and 19, we provide examples of the simultaneous use of two pieces of knowledge in the MAC and MQC scenarios, respectively. In Table 20, we provide another two prompts to test the effect of different prompts.

Table 19: An example of multi-question composition containing two edited pieces of knowledge, where the blue text indicates the question and the rephrased question.

Edit 1 x_e	Carmarthenshire shares border with
Edit 1 y_e	Ceredigion
Edit 2 x_e	Turkey shares border with
Edit 2 y_e	Syria
Test $\mathcal{X}_t$	The answers of the questions "Daimler has made the", "Saxony is adjacent to" are Mercedes-Benz, Hamburg respectively.\n The answers of the questions "Karlheinz Stockhausen is the composer of", "Daimler is the parent organization of" are Originale, Car2go respectively.\n The answers of the questions "Katherine Roberts is the writer of", "Florida International University has the employer" are The Colossus Crisis, Carlos Alvarez respectively.\n The answers of the questions "contraception has a subclass of", "Kering has subsidiary" are Intrauterine device, Volcom respectively.\n The answers of the questions "Carmarthenshire shares border with", "Turkey shares border with" are
Test $\mathcal{Y}_t$	Ceredigion, Syria
Rephrased test	The answers of the questions "Daimler has made the", "Saxony is adjacent to" are Mercedes-Benz, Hamburg respectively.\n The answers of the questions "Karlheinz Stockhausen is the composer of", "Daimler is the parent organization of" are Originale, Car2go respectively.\n The answers of the questions "Katherine Roberts is the writer of", "Florida International University has the employer" are The Colossus Crisis, Carlos Alvarez respectively.\n The answers of the questions "contraception has a subclass of", "Kering has subsidiary" are Intrauterine device, Volcom respectively.\n The answers of the questions "Carmarthenshire is adjacent to", "Turkey is adjacent to" are
Locality test	Vasif Talibov is a citizen of

L Limitations

Although the proposed method significantly enhances the composability of model editing, it still falls short of achieving fully accurate composition. Future research will continue to explore the factors that influence composability and further improve the effectiveness of compositional model editing. The performance of instruct models under penalty-free decoding condition also needs to be explored. Additionally, as this paper primarily focuses on how to train and compose edits with composability, it does not delve deeply into the organization of vector databases. Future work will aim to continuously

Table 20: Different few-shot prompts.

Prompt 1 $\mathcal{X}_t$	Tim Dorsey, who has written the Cadillac Beach, Nuclear Jellyfish, Triggerfish Twist, Hammerhead Ranch Motel, The Big Bamboo, Orange Crush (novel), Hurricane Punch, The Stingray Shuffle, Atomic Lobster, Torpedo Juice (novel), Florida Roadkill.\n Jerusalem, which is the partner town of NYC, New York, Praha, New York City, United States, Rio de Janeiro, NY, New York, NY, Prague, New York City, Tehran, Buenos Aires, Moscow, Manhattan.\n Pushkin is the author of The Fountain of Bakhchisaray, Eugene Onegin, The Tale of the Fisherman and the Fish, Poltava (poem), The Tale of the Golden Cockerel, Dubrovsky (novel), The Belkin Tales, Onegin, The Stone Guest (play), The Bronze Horseman (poem), The Queen of Spades (story), The Tale of the Priest and of His Workman Balda, The Gypsies, The Blizzard, Tatiana Larina, The Tale of the Dead Princess and the Seven Knights.\n WWE is the owner of WWE Classics on Demand, FCW Florida Heavyweight Championship, WWE Studios, WWE Films, WWE Network, NXT, FCW, WWE Classics On Demand, FCW Southern Heavyweight Championship, WCW, World Championship Wrestling, WCW, Inc., Florida Championship Wrestling, NXT Wrestling, Universal Wrestling Corporation, WWE NXT.\n
Prompt 2 $\mathcal{X}_t$	Saxony is adjacent to Hamburg, Nordsachsen, Liberec Region, Sachsen, Saxony-Anhalt, Bavaria, Sachsen-Anhalt, Thuringen, North Rhine-Westphalia, Thuringia, Saxony Anhalt, Lower Saxony, Brandenburg.\n Florida International University has the employer Carlos Alvarez, Les Standiford, Elizabeth Price Foley, Stanley Fish, Leonard Strickman, Barbara Walsh, Jerry Markham.\n Contraception has a subclass of Intrauterine device, withdrawal method, Emergency contraception, Coitus interruptus, Condom, intrauterine device, morning-after pill.\n Kering has subsidiary Volcom, Saint Laurent Paris, Gucci Group, Christopher Kane, Puma, Boucheron, Gucci, Saint Laurent, Bottega Veneta, Balenciaga, Sergio Rossi.\n
Prompt 3 $\mathcal{X}_t$	Westchester County is adjacent to Bronx County, The Bronx, south Bronx, Putnam County, Rockland, Bronx.\n Margery Allingham is the writer of The Case of the Late Pig, Hide My Eyes, The Mind Readers, Cargo of Eagles, The China Governess, Sweet Danger, Coroner's Pidgin.\n The position Philippine President is held by Fidel Ramos, Corazon Aquino, Emilio Aguinaldo, Macapagal, Joseph Ejercito Estrada, Roxas, Quezon, Benigno Aquino III, Diosdado Macapagal, Ferdinand Marcos, Manuel Quezon.\n cytokine has a subclass of Monokine, Interferon, lymphokine, interleukins, Chemokine.\n

optimize the structure of vector databases. The effectiveness of other PEFT methods [65, 66, 67, 68] need to be explore as well.

M Broader Impact

This paper presents work whose goal is to advance the field of model editing. Model editing is a cutting-edge concept that holds immense significance in a variety of critical and rapidly evolving fields, including but not limited to medical or legal fields. In these scenarios, mistakes of models are found after deployment, thus a timely and effective correction is needed. Our research has taken a step towards more available model editing by allowing multiple edits to be used simultaneously, leading to safer and more ethical applications across a broad spectrum of industries.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: The abstract and introduction accurately reflect the paper's contributions and scope.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: Limitations are discussed in Appendix L.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[NA\]](#)

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Implementation details are provided in Appendix G.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Release the codes upon acceptance of the paper.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Details are provided in Appendix G.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Model editing on a specific example is determinative when no additional randomly initialized parameters are introduced. In this case, there is no associated error bar. We provide results in different datasets, models and prompts.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [\[Yes\]](#)

Justification: Computer resources are provided in Appendix G.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [\[Yes\]](#)

Justification: The authors conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [\[Yes\]](#)

Justification: In Appendix M.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [\[Yes\]](#)

Justification: For the pre-trained models and the datasets used in this paper, we check them manually to ensure the safety.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [\[Yes\]](#)

Justification: The authors cite the original paper that produced the pre-trained models and datasets used in this paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: The paper does not release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [\[Yes\]](#)

Justification: We conduct the experiments of model editing on LLMs.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.