Rethinking Residual Distribution in Locate-then-Edit Model Editing

Xiaopeng Li Shangwen Wang Shasha Li* Shezheng Song Bin Ji Jun Ma* Jie Yu*
National University of Defence Technology
`{xiaopengli,wangshangwen13,shashali,ssz614,jibin,majun,yj}@nudt.edu.cn`

Abstract

Model editing enables targeted updates to the knowledge of large language models (LLMs) with minimal retraining. Among existing approaches, locate-then-edit methods constitute a prominent paradigm: they first identify critical layers, then compute residuals at the final critical layer based on the target edit, and finally apply least-squares-based multi-layer updates via **residual distribution**. While empirically effective, we identify a counterintuitive failure mode: residual distribution, a core mechanism in these methods, introduces weight shift errors that undermine editing precision. Through theoretical and empirical analysis, we show that such errors increase with the distribution distance, batch size, and edit sequence length, ultimately leading to inaccurate or suboptimal edits. To address this, we propose the **B**oundary **L**ayer **U**pdat**E (BLUE)** strategy to enhance locate-then-edit methods. Sequential batch editing experiments on three LLMs and two datasets demonstrate that BLUE not only delivers an average performance improvement of 35.59%, significantly advancing the state of the art in model editing, but also enhances the preservation of LLMs' general capabilities. Our code is available at https://github.com/xpq-tech/BLUE.

1 Introduction

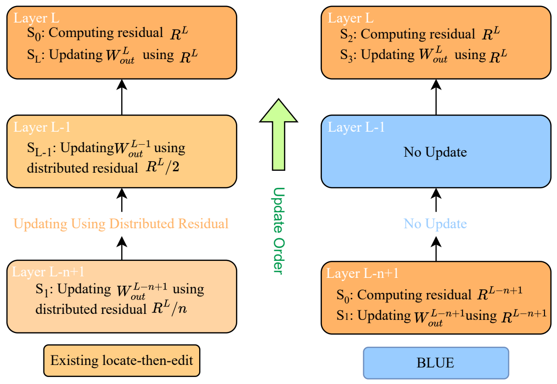
Large language models (LLMs) possess powerful comprehension and generation capabilities and have become foundational infrastructure for various AI applications. However, the knowledge encoded in the parameters of LLMs is limited to the training data and cannot be updated to reflect changes in world knowledge. Updating the parameters of LLMs through retraining to keep them in sync with world knowledge entails high computational costs [1]. Recently, model editing has garnered increasing attention as a promising technique for efficiently updating the parameterized knowledge in LLMs, which aims to correct erroneous or outdated knowledge within LLMs without compromising their other capabilities [2].

Locate-then-edit methods are a prominent family of model editing techniques. They treat the Feed-Forward Network (FFN) as a key-value memory [3] and update the key layers responsible for storing factual knowledge using a least-squares solution. Specifically, these methods first employ causal tracing analysis to identify multiple critical layers within LLMs that encode factual information. They then use optimization techniques to compute the residuals required to update the final critical layer. Finally, the residuals are distributed evenly from the first to the last critical layer, and the updates are applied using a least-squares solution [4, 5], a process referred to as **residual distribution**.

Although locate-then-edit methods have achieved remarkable performance on model editing tasks [6], **we identify a counterintuitive failure mode: residual distribution, a core mechanism in these methods, introduces weight shift errors that undermine editing precision.** Specifically, we

* Corresponding Authors.

first empirically demonstrate that the contribution of residual distribution to model editing diminishes as the distribution distance increases and that the distributed residual is not the optimal residual for editing. Subsequently, we theoretically find that the upper bound of weight update errors increases with: (a) the size of the editing batch, (b) the number of sequential edits, and (c) the residual distribution distance. These findings indicate that **residual distribution can actually negatively impact the model editing of the locate-then-edit approaches**.



Layer L
S_0: Computing residual R^L
S_L: Updating W_{out}^{L} using R^L
Layer L-1
S_{L-1}: Updating W_{out}^{L-1} using distributed residual $R^L/2$
Updating Using Distributed Residual
Layer L-n+1
S_1: Updating W_{out}^{L-n+1} using distributed residual R^L/n
Existing locate-then-edit
Update Order
Layer L
S_2: Computing residual R^L
S_3: Updating W_{out}^{L} using R^L
Layer L-1
No Update
No Update
Layer L-n+1
S_0: Computing residual R^{L-n+1}
S_1: Updating W_{out}^{L-n+1} using R^{L-n+1}
BLUE

Figure 1: Comparison of existing locate-then-edit methods and BLUE. The n critical layers are edited, where L denotes the last critical layer.

Therefore, we propose the **B**oundary **L**ayer **U**pdat**E** (**BLUE**) strategy, which enhances locate-then-edit methods by updating only the first and last critical layers through direct computation of residuals, without residual distribution. The comparison between BLUE and existing locate-then-edit methods is shown in Figure 1. We apply BLUE to enhance MEMIT [5], RECT [7], PRUNE [8] and AlphaEdit [6]. Results from 12 sequential editing experiments conducted on three LLMs and two datasets show that BLUE improves the performance of existing locate-then-edit methods by an average of **35.59**%. Our further analysis on downstream tasks and representation shift demonstrates that BLUE also enhances the ability of locate-then-edit methods to preserve general capabilities and mitigates representation shifts in the post-edit LLMs. Furthermore, we show that BLUE not only improves editing efficiency, but also strengthens the performance of locate-then-edit methods in long-form model editing [9, 10] scenarios. In summary, our contributions are as follows:

- Through empirical and theoretical analysis of residual distribution, we reveal that residual distribution of locate-then-editing methods actually leads to inaccurate model editing.
- We propose the BLUE strategy, which discards residual distribution and enhances existing locate-then-edit methods by updating only the first and last critical layers through direct residual computation.
- Experimental results show that locate-then-edit methods enhanced with BLUE outperform the original methods, better preserve LLMs' general capabilities, and further mitigate representation shifts in the post-edit LLMs.

2 Related Work

Model editing can be categorized into **parameter-preserving** and **parameter-modifying** approaches, depending on whether the original model parameters are altered.

Parameter-preserving model editing employs techniques like prompt engineering or attaching additional parameters [11–14]. A representative method for prompt engineering is IKE [15], which retrieves n contexts of the edited knowledge for a query to guide the model's response without altering its internal parameters. Methods that attach additional parameters include SERAC [16] and GRACE [17], which store new memories externally and internally, respectively, by introducing new parameter modules.

Parameter-modifying approaches achieve model editing by directly or indirectly adjusting model parameters [18, 10]. Direct methods, such as FT-L [19], perform constrained fine-tuning on a small number of layers to integrate new knowledge. Indirect methods can be divided into **meta-learning** and **locate-then-edit** methods. **Meta-learning** methods, like MEND [20], leverage a hypernetwork to transform edit-related representations and gradients into parameter updates. In contrast, **locate-then-edit** methods, such as ROME [4] and MEMIT [5], adopt a key-value memory perspective to identify and update single or multiple critical layers using least-squares optimization. Among these, locate-then-edit methods have gained popularity and inspired several variants, such as PMET [21], which focuses on precise editing, and AlphaEdit [6], which enhances the retention of original knowledge and strengthens sequential editing capability.

3 Background

3.1 Model Editing Problem

Model editing aims to efficiently update the knowledge of LLMs so that they remain in real-time sync with reality [22]. Factual knowledge changes rapidly, making its update in LLMs a pressing need. Factual knowledge can be represented as a triplet (s, r, o), where s is the subject, r is the relation, and o is the object. This knowledge can be transformed into a prompt $p_i + o$, where $p_i \in \mathcal{P}$ is an element of the set $\mathcal{P}$ that expresses the semantics of (s, r) in natural language. The element that most directly expresses the semantics of (s, r) is called p, while the rest elements are p_r. The goal of model editing is to redirect the object o in the triplet to a new object o^*, represented as $t = (s, r, o) \to o^*$. To evaluate whether the post-edit model is effective on the post-edit knowledge triplet and does not affect other triplets, assessments are made from three aspects: efficacy, generalization, and specificity. Efficacy evaluates whether the model's prediction on p is redirected to o^*. Generalization evaluates whether the model's prediction on p_r is redirected to o^*. Specificity evaluates whether the model maintains its original predictions on inputs outside the set $\mathcal{P}$. To evaluate the generative capability of the post-edit model, [4] also uses fluency and consistency as evaluation metrics. Fluency measures the degree of repetition in the text generated by the model after editing; higher repetition indicates lower fluency. Consistency evaluates the degree of alignment between the content generated by the post-edit model based on s and the reference text of the subject associated with the new object o^*. For more details, please refer to [4].

Model editing can be categorized according to whether the editing is sequential and the batch size into sequential editing, batch editing, and sequential batch editing [23]. Sequential editing refers to the continuous editing of a single piece of knowledge, while batch editing involves editing multiple pieces of knowledge at once. Sequential batch editing combines these two scenarios, involving the sequential editing of batch knowledge. This problem definition is highly relevant to the ever-changing nature of bulk knowledge in practice. Therefore, we directly present the problem of sequential batch editing. Suppose there is a sequence of n knowledge sets to be updated: $[\mathbb{T}_1, \mathbb{T}_2, \ldots, \mathbb{T}_n]$, where each knowledge set $\mathbb{T}_i = \{t_1, t_2, \ldots\}$. Sequential batch editing requires that after performing n sequential batch edits, the post-edit model can successfully predict all n knowledge sets without affecting knowledge outside these sets.

3.2 Locate-then-Edit Model Editing

The locate-then-edit model editing is one of the most popular series of model editing methods [1]. These approaches typically use causal tracing [4] to identify the critical layers $\mathcal{L}$ where knowledge is stored, and then compute weight shifts using least squares modeling to update the weights.

Specifically, they view the feed-forward network (FFN) as key-value memories [3]. Let $\boldsymbol{h}^{l-1}$ be the residual stream of the $l-1$ layer, and $\boldsymbol{a}^l$ be the output of the self-attention block of the $l \in \mathcal{L}$ layer. The key-value memories of the FFN can be represented as follows:

$$\underbrace{\boldsymbol{m}^l}_{\text{value}} = \boldsymbol{W}^l_{\text{out}} \underbrace{\sigma(\boldsymbol{W}^l_{\text{in}} \gamma(\boldsymbol{h}^{l-1} + \boldsymbol{a}^l))}_{\text{key} := \boldsymbol{k}}, \tag{1}$$

where $\boldsymbol{m}^l$ is the output of the FFN block, and $\boldsymbol{W}^l_{\text{in}}$ and $\boldsymbol{W}^l_{\text{out}}$ are the input and output mapping weights of the FFN block, respectively. σ and γ are activation functions. $\boldsymbol{W}^l_{\text{out}} := \boldsymbol{W}^l_0$ is viewed as a linear associative memory that associates keys and values:

$$\boldsymbol{K}^l_0 = [\boldsymbol{k}^l_1 | \boldsymbol{k}^l_2 | \ldots | \boldsymbol{k}^l_n], \boldsymbol{M}^l_0 = [\boldsymbol{m}^l_{0,1} | \boldsymbol{m}^l_{0,2} | \ldots | \boldsymbol{m}^l_{0,n}]. \tag{2}$$

Before editing, the linear associative memory satisfies:

$$\boldsymbol{W}^l_0 = \arg\min_{\boldsymbol{W}} \left\| \boldsymbol{W} \boldsymbol{K}^l_0 - \boldsymbol{M}^l_0 \right\|^2. \tag{3}$$

When new memories need to be inserted, a new group of keys $\boldsymbol{K}^l_1$ and values $\boldsymbol{M}^l_1$ will be updated into $\boldsymbol{W}^l_0$. Thus the new weight should satisfy:

$$\boldsymbol{W}^l_1 = \arg\min_{\boldsymbol{W}} \underbrace{\left\| \boldsymbol{W} \boldsymbol{K}^l_0 - \boldsymbol{M}^l_0 \right\|^2}_{\text{preserve old}} + \underbrace{\left\| \boldsymbol{W} \boldsymbol{K}^l_1 - \boldsymbol{M}^l_1 \right\|^2}_{\text{insert new}}. \tag{4}$$

Let $\boldsymbol{W}_1^l = \boldsymbol{W}_0^l + \Delta^l$ where Δ^l is weight shifts. By applying the normal equation to Eq. (4), its closed-form solution can be written as:

$$\Delta^l = \boldsymbol{R}^l {\boldsymbol{K}_1^l}^T \left(\boldsymbol{K}_0^l {\boldsymbol{K}_0^l}^T + \boldsymbol{K}_1^l {\boldsymbol{K}_1^l}^T \right)^{-1}, \tag{5}$$

where $\boldsymbol{R}^l = \left(\boldsymbol{M}_1^l - \boldsymbol{W}_0^l \boldsymbol{K}_1^l \right)$ is the residual of the new memories when evaluated on old weights $\boldsymbol{W}_0^l$ [5]. $\boldsymbol{K}_1^l$ and $\boldsymbol{K}_0^l$ are computed for each layer. In most locate-then-edit methods [5, 6], the residual of layer l is evenly distributed from the residual of last critical layer $L = \max(\mathcal{L})$:

$$\boldsymbol{R}^l = \frac{\boldsymbol{R}^L}{L - l + 1} = \frac{\boldsymbol{M}_1^L - \boldsymbol{W}_0^L \boldsymbol{K}_1^L}{L - l + 1}, \tag{6}$$

where $\boldsymbol{M}_1^L = [\boldsymbol{m}_1^L | \boldsymbol{m}_2^L | ... | \boldsymbol{m}_u^L]$ represents u entries of new memories. Each entry is computed using the following formula:

$$\boldsymbol{m}_i^L = \boldsymbol{h}_i^L + \boldsymbol{\delta}_i^L = \boldsymbol{W}_0^L \boldsymbol{k}_i^L + \boldsymbol{\delta}_i^L, \tag{7}$$

where $\boldsymbol{\delta}_i^L$ is a residual vector optimized by:

$$\mathbf{m}_i^L = \mathbf{h}_i^L + \arg\min_{\boldsymbol{\delta}_i^L} \frac{1}{P} \sum_{j=1}^{P} - \log \mathbb{P}_{\theta(\mathbf{h}_i^L += \boldsymbol{\delta}_i^L)} [o^* | x_j \oplus p] \tag{8}$$

The objective of the above equation is to optimize the learnable $\boldsymbol{\delta}_i^L$ to maximize the probability of the model predicting o^*. $x_j \oplus p$ represents the concatenation of the jth randomly generated P prefixes by the model with prompt p to enhance generalization. $\theta(\mathbf{h}_i^L + = \boldsymbol{\delta}_i^L)$ denotes adding $\boldsymbol{\delta}_i^L$ to $\mathbf{h}_i^L$.

Fang et al. [6] extends locate-then-edit methods to the sequential batch editing scenario. They cache the keys $\boldsymbol{K}_p$ of previously edited knowledge and incorporate $\boldsymbol{K}_p$ into the least squares optimization, ultimately deriving the following closed-form solution for sequential batch editing:

$$\Delta_{\text{seq}}^l = \boldsymbol{R}^l {\boldsymbol{K}_1^l}^T \left(\boldsymbol{K}_p^l {\boldsymbol{K}_p^l}^T + \boldsymbol{K}_0^l {\boldsymbol{K}_0^l}^T + \boldsymbol{K}_1^l {\boldsymbol{K}_1^l}^T \right)^{-1} \tag{9}$$

4 Rethinking the Residual Distribution of Locate-then-Edit Model Editing

In this section, we analyze the residual distribution both empirically (Section 4.1) and theoretically (Section 4.2). Based on these insights, we further propose a novel strategy to enhance locate-then-edit model editing (Section 4.3). We focus on the classic locate-then-edit model editing method, MEMIT [5]. Experiments are conducted on three LLMs: Llama3-8B-Instruct [24], GPT-J (6B) [25], and GPT2-XL, using the CounterFact dataset [4]. Unless otherwise specified, we use the first 200 samples from the CounterFact dataset. The critical layers analyzed for each model are: Llama3-8B: $\{4, 5, 6, 7, 8\}$, GPT-J (6B): $\{3, 4, 5, 6, 7, 8\}$ and GPT2-XL: $\{13, 14, 15, 16, 17\}$.

4.1 Analyzing Residual Distribution in Locate-then-Edit Model Editing

4.1.1 How Does the Distributed Residual Contribute to the Editing Object?

To measure the contribution of the distributed residuals to the editing object, we first define a **contribution score**:

$$s = \mathbb{P}_{\theta^*}(o^* | p) - \mathbb{P}_{\theta}(o^* | p) \tag{10}$$

where $\mathbb{P}_{\theta^*}(o^* | p)$ represents the probability of the post-edit model θ^* regarding the edited knowledge $t = (s, r, o) \to o^*$. The rationale behind this is that the probability of the pre-edit model θ assigning to o^* on knowledge t is often low, while the model editing aims for the post-edit model to assign the highest probability to o^*.

From Equ. (1) and Equ. (4), we can know that the essence of locate-then-edit is that the post-edit model can activate the new memory $\boldsymbol{m}_i^l = \boldsymbol{m}_{0,i}^l + \boldsymbol{\delta}_i^L / (L - l + 1)$ in the FFN block at layer l using the key $\boldsymbol{k}_i^l$ corresponding to p, where $\boldsymbol{m}_{0,i}^l$ represents the original memory of the model. Therefore, we directly replace the output of the FFN block at layer l with the new memory $\boldsymbol{m}_i^l$ to eliminate the potential impact of activation failure. For comparison, we also directly compute residuals for each layer, following the same process as $\boldsymbol{m}_i^L$. By using the distributed residual for *simulated editing*, we can accurately measure the contribution of new memories while avoiding direct edits to the model.

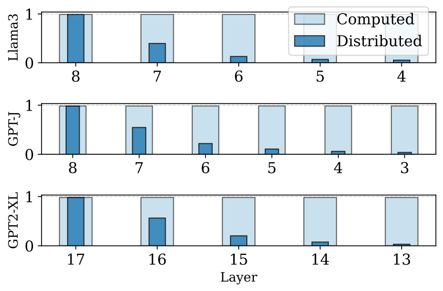


Figure 2: The average contribution score of different *simulated editing* layers.

We perform *simulated editing* in each critical layer. The average contribution scores are shown in Figure 2. For the distributed residuals, it can be observed that only the last critical layer achieves a contribution score close to 1.0. The contribution scores of the other layers were all below 0.7, showing a decreasing trend layer by layer. For the first critical layer, the contribution score is below 0.1 in three LLMs. This indicates that **the farther the residuals are distributed, the lower their contribution to the editing object. Even distribute through just one layer can lead to a significant drop in the contribution score.** In contrast, for the computed residuals, their contribution scores in each layer consistently approach 1.0.

4.1.2 Is the Distributed Residual the Optimal Residual for Editing?

From Section 4.1.1, we observe that directly computing $\boldsymbol{m}_i^l$ for each layer achieves high contribution scores. Therefore, we assume that the directly computed $\boldsymbol{m}_i^l$ represents the optimal memory for editing. To verify whether residual distribution is optimal, we first compare the similarity between residual distribution and the directly computed $\boldsymbol{m}_i^l$, and then evaluate their performance in model editing.

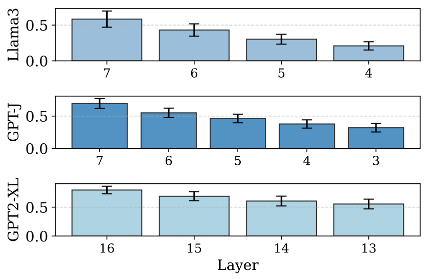


Figure 3: The variation in cosine similarity between the distributed and the directly computed memory across different layers.

Similarity Analyzing. The variation in cosine similarity between the distributed and the directly computed $\boldsymbol{m}_i^l$ is shown in Figure 3. It shows that the cosine similarity between the distributed and the directly computed $\boldsymbol{m}_i^l$ exhibits a layer-by-layer decreasing trend, indicating that **the further residuals are distributed, the farther $\boldsymbol{m}_i^l$ deviates from the optimal memory.** To further investigate how residual distribution affects model editing performance, we next perform single-layer model editing using both the distributed residuals and the directly computed residuals.

Post-edit LLM Performance. We update single layers of the model using distributed residuals and computed residuals, respectively. The results for Llama3 under the batch editing setting are shown in Figure 4, while results for GPT-J and GPT2-XL are presented in Figure 9 of Appendix C. Specificity and Fluency remain comparable across different cases, with outcomes closely matching the original model. This indicates that small-batch edits effectively retain the model's original state. However, significant differences arise in Efficacy and Generalization between the two methods. For Efficacy, models edited with computed residuals outperform those with distributed residuals by over 3× on average, while for Generalization, the improvement exceeds 2×. In terms of Consistency, computed residuals achieve an average improvement of more than 10%. **These findings indicate that distributed residuals introduce significant information loss during model editing, leading to a higher likelihood of editing failures.**

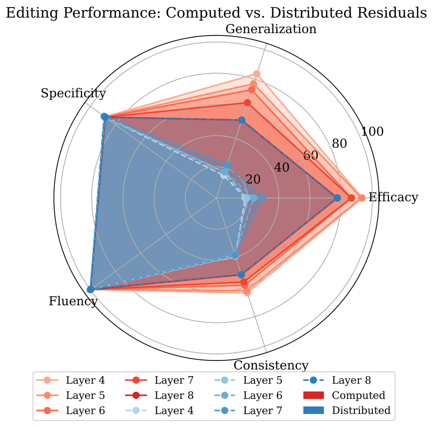


Figure 4: Performance variations when editing different single layers of the model using computed and distributed residuals separately. Fluency and Consistency are normalized.

4.2 Theoretical Analysis of Residual Distribution in Locate-then-Edit Methods

Theorem 4.1. *In the locate-then-edit model editing, when using residual distribution, the upper bound for the weight shift error between the exact weight shift Δ^{l^*} and the actual weight shift Δ^l is* *given by*

$$\|\Delta^{l^*} - \Delta^l\|_2 \leq \left(\|\boldsymbol{R}^{l^*} - \boldsymbol{R}^L\|_2 + (L-l)\|\boldsymbol{R}^L\|_2\right)\|\boldsymbol{Q}\|_2, \tag{11}$$

where $\boldsymbol{R}^{l^*}$ *denotes the exact residual, and* $\boldsymbol{Q} = \boldsymbol{K}_1^{l\,T}\left(\boldsymbol{K}_0^l\boldsymbol{K}_0^{l\,T} + \boldsymbol{K}_1^l\boldsymbol{K}_1^{l\,T}\right)^{-1}$.

The proof of the above theorem is presented in Appendix A. We also discuss the rationale for using the upper bound instead of the lower bound in Remark 4.2. In Equ. (11), $\|\boldsymbol{R}^l\|_2$ and $\|\boldsymbol{Q}\|_2$ increase with the number of new memories (i.e., the size of the editing batch [5]). When the number of new memories is fixed, the upper bound increases with $\|\boldsymbol{R}^{l^*} - \boldsymbol{R}^L\|_2$ and $L-l$. $L-l$ increases as the residual distributes further, while it is unclear how $\|\boldsymbol{R}^{l^*} - \boldsymbol{R}^L\|_2$ changes. To explore this, we assume the computed residual is the exact residual and analyze how $\|\boldsymbol{R}^{l^*} - \boldsymbol{R}^L\|_2$ changes across layers. In Figure 5, we show how $\|\boldsymbol{R}^{l^*} - \boldsymbol{R}^L\|_2$ changes across layers. It shows that $\|\boldsymbol{R}^{l^*} - \boldsymbol{R}^L\|_2$ increases as the residual distribution extends farther. Therefore, we can conclude that **weight shift error increases with both the distance of residual distribution and the size of the editing batch**.

Remark 4.2. *The lower bound of the weight shift error can more effectively reflect the variation in error, but according to* $||\mathbf{R}^{l^*}\mathbf{Q} - \mathbf{R}^l\mathbf{Q}||_2 \geq \sigma_{\min}(\mathbf{Q})||\mathbf{R}^{l^*} - \mathbf{R}^l||_2$ *we see that the lower bound of the error shift is very small or even zero, where* $\sigma_{\min}(\mathbf{Q})$ *is the smallest singular value of* $\mathbf{Q}$. *This occurs because we cannot guarantee that* $\mathbf{Q}$ *is nondegenerate. Even if* $\mathbf{Q}$ *is nondegenerate,* $\sigma_{\min}(\mathbf{Q})$ *would still be very small, making the result insignificant. Additionally, although a growing upper bound does not necessarily imply an increase in error, our upper bound determines the worst-case scenario of the error, providing essential insight into selecting the layers to update.*

Considering the closed-form solution of locate-then-edit model editing in sequential batch editing, the following lemma can be derived.

Lemma 4.3. *In sequential batch editing, when using residual distribution, the upper bound of the weight shift error between the exact weight shift* Δ^{l^*} *and the actual weight shift* Δ^l *for locate-then-edit methods is given by*

$$\|\Delta^{l^*} - \Delta^l\|_2 \leq \left(\|\boldsymbol{R}^{l^*} - \boldsymbol{R}^L\|_2 + (L-l)\|\boldsymbol{R}^l\|_2\right)\|\boldsymbol{Q}'\|_2, \tag{12}$$

where $\boldsymbol{R}^{l^*}$ *denotes the exact residual, and* $\boldsymbol{Q}' = \boldsymbol{K}_1^{l\,T}\left(\boldsymbol{K}_p^l\boldsymbol{K}_p^{l\,T} + \boldsymbol{K}_0^l\boldsymbol{K}_0^{l\,T} + \boldsymbol{K}_1^l\boldsymbol{K}_1^{l\,T}\right)^{-1}$.

$\|\boldsymbol{K}_p^l\boldsymbol{K}_p^{l\,T}\|_2$ increases with the number of sequential edits, and thus Section 4.3 indicates that **the weight shift error also increases with the number of sequential edits.**

4.3 BLUE: Boundary Layer UpdatE for Improving Locate-then-Edit Model Editing

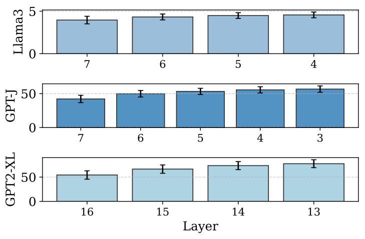


Figure 5: Variation of $\|\boldsymbol{R}^{l^*} - \boldsymbol{R}^L\|_2$ across layers.

From the previous analysis, we know that the residual distribution of the locate-then-edit model editing is inherently inaccurate, and this inaccuracy increases as the residuals are distributed farther. So, *how can we enable locate-then-edit model editing to perform multi-layer updates while mitigating the negative impact of the residual distribution?* A straightforward method is to compute residuals separately for each layer. However, this reduces the efficiency of the locate-then-edit approach, and it remains unclear whether it is necessary to compute residuals and perform updates for all critical layers individually.

Table 1: Average optimization steps.

Model	**Layer**: Steps
GPT2-XL	**[13-17]**: [16.37, 8.43, 1.71, 0.32, 0.10]
GPT-J (6B)	**[3-8]**: [10.47, 1.68, 0.11, 0.0, 0.0, 0.0]
Llama3 (8B)	**[4-8]**: [25.0, 11.10, 0.63, 0.0, 0.0]

Therefore, we conduct experiments where residuals are computed and updates performed for all critical layers. Computation is stopped when the loss falls below 0.05. We update the critical layers sequentially in the order of increasing layers and record the number of optimization steps required to compute residuals for each layer. The results are shown in Table 1. It can be observed that after completing the first layer update, the number of residual computation steps in subsequent layers decreased significantly across all LLMs. In GPT2-XL, the decrease is 48.5%; in GPT-J, 84.0%; and in Llama3, 55.6%. Furthermore, after updating the first two layers, the optimization steps in the third layer for GPT2-XL, GPT-J and Llama3 drop below 2.0, indicating that **only two layers of weights needed to be updated to achieve the editing goal.**

The above observations indicate that when calculating residuals separately for all layers, updating just two layers is sufficient to achieve the editing object. The new question is: *which two layers should be updated to achieve optimal editing performance?* According to Theorem 4.1, the farther the distribution of residuals, the greater the upper bound of the weight shift error. The last critical layer is exactly the layer where the residual is computed. As a result, the first layer updated by the current method is the most affected by the residual distribution. To mitigate this effect, **we choose the first critical layer as the first layer for updates. For the second layer, we choose the last critical layer** for the following reasons: 1) The selection of the first critical key layer is based on the premise that the residual is computed at the last critical layer; and 2) BLUE is an optimization strategy, and we aim to preserve the mechanism used in existing locate-then-edit model editing methods, which compute residuals at the last key layer, to ensure broader applicability.

Therefore, we propose a **Boundary Layer UpdatE (BLUE)** strategy to boost the locate-then-edit methods. BLUE updates only the boundary layers of the critical layers by directly computing residuals of them, specifically the first critical layer and the last critical layer. This not only reduces the number of layers to be updated, but we also demonstrate in Section 5 that it performs better and better preserves LLMs' general capabilities. BLUE is suitable for locate-then-edit methods that perform multi-layer updates using even residual distribution: MEMIT [5], RECT [7], PRUNE [8] and AlphaEdit [6].

5 Experiments

In our experiments, we demonstrate that BLUE can enhance the performance of current locate-then-edit methods (Section 5.2), improve the retention of the original LLMs' capabilities (Section 5.3), and alleviate the hidden state shifts introduced by locate-then-edit approaches (Section 5.4). Additionally, in Appendix 5.5, we show that BLUE also boosts locate-then-edit methods in long-form model editing. In Appendix 5.6, we present ablation studies demonstrating that selecting the first and last critical layers for editing in BLUE is the optimal choice. Appendix I illustrates the efficiency improvements introduced by BLUE. In Appendix J, we empirically validate Theorem 4.1 and Lemma 4.3, demonstrating that the weight shift error increases with both the batch size and the number of sequential edits. Finally, we present the results of BLUE applied to the square root residual distribution method, PMET, in Appendix K.

5.1 Experimental Setup

Datasets & LLMs. Our experiments are conducted on two datasets: CounterFact [4] and zsRE [26]. We select three LLMs as the editing subjects: GPT2-XL [27], GPT-J (6B) [25], Llama3 (8B) [28]. We also conduct experiments with Llama2 (13B) to further validate the effectiveness of BLUE in Appendix G.

Baselines. BLUE is a facilitation strategy designed for locate-then-edit model editing that performs multi-layer updates, which has been proven in prior research to achieve the best editing performance [6]. Therefore, our baselines only consider locate-then-edit model editing methods. The locate-then-edit methods we consider are: MEMIT [5], PRUNE [8], RECT [29], and AlphaEdit [6]. We present the experimental details in the Appendix D.

5.2 Enhancing Editing Performance with BLUE

We first verify whether BLUE can enhance locate-then-edit model editing. Sequential batch editing better aligns with real-world batch knowledge updates, and we follow [6] by using sequential batch editing experiments to validate the capabilities of BLUE. We randomly sample 2,000 samples from the dataset and perform sequential batch editing with a batch size of 100. The results of the sequential batch editing are shown in Table 2. We use red to highlight the results enhanced by BLUE. The

Table 2: Comparison of BLUE enhanced locate-then-model editing methods with original locate-then-model editing methods on the sequential model editing task. We color all results that are actually enhanced by BLUE in red. We directly adopt the baseline results from [6] to avoid the energy consumption caused by redundant computation.

Method	Model	Counterfact					ZsRE		
		Efficacy↑	Generalization↑	Specificity↑	Fluency↑	Consistency↑	Efficacy↑	Generalization↑	Specificity↑
Pre-edited		$7.85_{\pm0.26}$	$10.58_{\pm0.26}$	$89.48_{\pm0.18}$	$635.23_{\pm0.11}$	$24.14_{\pm0.08}$	$36.99_{\pm0.30}$	$36.34_{\pm0.30}$	$31.89_{\pm0.22}$
MEMIT	Llama3	$65.65_{\pm0.47}$	$64.65_{\pm0.42}$	$51.56_{\pm0.38}$	$437.43_{\pm1.67}$	$6.58_{\pm0.11}$	$34.62_{\pm0.36}$	$31.28_{\pm0.34}$	$18.49_{\pm0.19}$
PRUNE	Llama3	$68.25_{\pm0.46}$	$64.75_{\pm0.41}$	$49.82_{\pm0.36}$	$418.03_{\pm1.52}$	$5.90_{\pm0.10}$	$24.77_{\pm0.27}$	$23.87_{\pm0.27}$	$20.69_{\pm0.23}$
RECT	Llama3	$66.05_{\pm0.47}$	$63.62_{\pm0.43}$	$61.41_{\pm0.37}$	$526.62_{\pm0.44}$	$20.54_{\pm0.09}$	$86.05_{\pm0.23}$	$80.54_{\pm0.27}$	$31.67_{\pm0.22}$
AlphaEdit	Llama3	$98.90_{\pm0.10}$	$94.22_{\pm0.19}$	$67.88_{\pm0.29}$	$622.49_{\pm0.16}$	$32.40_{\pm0.11}$	$94.47_{\pm0.13}$	$91.13_{\pm0.19}$	$32.55_{\pm0.22}$
MEMIT$_{\text{BLUE}}$	Llama3	$99.57_{\pm0.24}$	$94.13_{\pm0.77}$	$83.77_{\pm0.77}$	$626.26_{\pm0.51}$	$32.29_{\pm0.38}$	$95.94_{\pm0.38}$	$90.98_{\pm0.69}$	$32.41_{\pm0.81}$
PRUNE$_{\text{BLUE}}$	Llama3	$96.73_{\pm0.64}$	$89.68_{\pm0.85}$	$57.79_{\pm1.14}$	$627.39_{\pm0.37}$	$33.39_{\pm0.36}$	$86.55_{\pm0.86}$	$82.22_{\pm1.00}$	$31.04_{\pm0.80}$
RECT$_{\text{BLUE}}$	Llama3	$98.77_{\pm0.40}$	$93.40_{\pm0.74}$	$79.34_{\pm0.86}$	$619.07_{\pm0.62}$	$30.62_{\pm0.37}$	$94.37_{\pm0.49}$	$89.49_{\pm0.76}$	$32.76_{\pm0.81}$
AlphaEdit$_{\text{BLUE}}$	Llama3	$99.93_{\pm0.09}$	$97.25_{\pm0.48}$	$75.24_{\pm0.98}$	$624.90_{\pm0.49}$	$33.79_{\pm0.38}$	$95.77_{\pm0.39}$	$91.73_{\pm0.65}$	$31.96_{\pm0.80}$
Pre-edited		$16.22_{\pm0.31}$	$18.56_{\pm0.45}$	$83.11_{\pm0.13}$	$621.81_{\pm0.67}$	$29.74_{\pm0.51}$	$26.32_{\pm0.37}$	$25.79_{\pm0.25}$	$27.42_{\pm0.53}$
MEMIT	GPT-J	$98.55_{\pm0.11}$	$95.50_{\pm0.16}$	$63.64_{\pm0.31}$	$546.28_{\pm0.88}$	$34.89_{\pm0.15}$	$94.91_{\pm0.16}$	$90.22_{\pm0.23}$	$30.39_{\pm0.27}$
PRUNE	GPT-J	$86.15_{\pm0.34}$	$86.85_{\pm0.29}$	$53.87_{\pm0.35}$	$427.14_{\pm0.53}$	$14.78_{\pm0.11}$	$0.15_{\pm0.02}$	$0.15_{\pm0.02}$	$0.00_{\pm0.00}$
RECT	GPT-J	$98.80_{\pm0.10}$	$86.58_{\pm0.28}$	$72.22_{\pm0.28}$	$617.31_{\pm0.19}$	$41.39_{\pm0.12}$	$96.38_{\pm0.14}$	$91.21_{\pm0.21}$	$27.79_{\pm0.26}$
AlphaEdit	GPT-J	$99.75_{\pm0.08}$	$96.38_{\pm0.23}$	$75.48_{\pm0.21}$	$618.50_{\pm0.17}$	$42.08_{\pm0.15}$	$99.79_{\pm0.14}$	$96.00_{\pm0.22}$	$28.29_{\pm0.25}$
MEMIT$_{\text{BLUE}}$	GPT-J	$99.70_{\pm0.30}$	$96.90_{\pm0.50}$	$74.61_{\pm0.95}$	$620.89_{\pm0.73}$	$40.82_{\pm0.44}$	$99.58_{\pm0.18}$	$94.77_{\pm0.67}$	$28.36_{\pm0.94}$
PRUNE$_{\text{BLUE}}$	GPT-J	$97.77_{\pm0.53}$	$97.28_{\pm0.48}$	$57.12_{\pm1.00}$	$608.73_{\pm0.89}$	$36.62_{\pm0.42}$	$60.51_{\pm1.35}$	$58.57_{\pm1.35}$	$22.77_{\pm0.87}$
RECT$_{\text{BLUE}}$	GPT-J	$98.70_{\pm0.41}$	$91.18_{\pm0.84}$	$74.78_{\pm0.94}$	$620.52_{\pm0.65}$	$39.79_{\pm0.43}$	$97.93_{\pm0.38}$	$93.86_{\pm0.69}$	$26.32_{\pm0.91}$
AlphaEdit$_{\text{BLUE}}$	GPT-J	$99.77_{\pm0.17}$	$97.13_{\pm0.48}$	$75.23_{\pm0.95}$	$621.07_{\pm0.62}$	$41.34_{\pm0.44}$	$99.63_{\pm0.16}$	$95.96_{\pm0.59}$	$28.67_{\pm0.94}$
Pre-edited		$22.23_{\pm0.73}$	$24.34_{\pm0.62}$	$78.53_{\pm0.33}$	$626.64_{\pm0.31}$	$31.88_{\pm0.20}$	$22.19_{\pm0.24}$	$31.30_{\pm0.27}$	$24.15_{\pm0.32}$
MEMIT	GPT2-XL	$94.70_{\pm0.22}$	$85.82_{\pm0.28}$	$60.50_{\pm0.32}$	$477.26_{\pm0.54}$	$22.72_{\pm0.15}$	$79.17_{\pm0.32}$	$71.44_{\pm0.36}$	$26.42_{\pm0.25}$
PRUNE	GPT2-XL	$82.05_{\pm0.38}$	$78.55_{\pm0.34}$	$53.02_{\pm0.35}$	$530.47_{\pm0.39}$	$15.93_{\pm0.11}$	$21.62_{\pm0.30}$	$19.27_{\pm0.28}$	$13.19_{\pm0.18}$
RECT	GPT2-XL	$92.15_{\pm0.26}$	$81.15_{\pm0.33}$	$65.13_{\pm0.31}$	$480.83_{\pm0.62}$	$21.05_{\pm0.16}$	$81.02_{\pm0.31}$	$73.08_{\pm0.35}$	$24.85_{\pm0.25}$
AlphaEdit	GPT2-XL	$99.50_{\pm0.24}$	$93.95_{\pm0.34}$	$66.39_{\pm0.31}$	$597.88_{\pm0.18}$	$39.38_{\pm0.15}$	$94.81_{\pm0.30}$	$86.11_{\pm0.29}$	$25.88_{\pm0.21}$
MEMIT$_{\text{BLUE}}$	GPT2-XL	$98.27_{\pm0.47}$	$88.67_{\pm0.93}$	$67.13_{\pm1.01}$	$587.19_{\pm1.52}$	$35.64_{\pm0.46}$	$93.62_{\pm0.70}$	$85.34_{\pm1.04}$	$26.55_{\pm0.93}$
PRUNE$_{\text{BLUE}}$	GPT2-XL	$88.19_{\pm1.12}$	$80.48_{\pm1.10}$	$52.23_{\pm1.05}$	$594.08_{\pm1.18}$	$20.28_{\pm0.44}$	$47.94_{\pm1.33}$	$45.03_{\pm1.32}$	$16.72_{\pm0.75}$
RECT$_{\text{BLUE}}$	GPT2-XL	$95.67_{\pm0.73}$	$80.97_{\pm1.15}$	$66.88_{\pm1.00}$	$567.09_{\pm2.09}$	$30.30_{\pm0.53}$	$83.48_{\pm1.06}$	$75.24_{\pm1.24}$	$25.25_{\pm0.89}$
AlphaEdit$_{\text{BLUE}}$	GPT2-XL	$99.40_{\pm0.28}$	$96.00_{\pm0.60}$	$76.63_{\pm0.93}$	$621.92_{\pm0.56}$	$40.98_{\pm0.43}$	$96.88_{\pm0.50}$	$89.58_{\pm0.91}$	$25.93_{\pm0.92}$

results indicate that **the BLUE strategy effectively enhances the performance of a range of locate-then-edit methods in sequential batch editing tasks**. 89.58% of the results (86 out of 96) were enhanced. After using BLUE, the editing performance of different editing methods is enhanced across various LLMs, as shown in Table 4 of Appendix B. It can be observed that the BLUE strategy significantly improves the performance of PRUNE and noticeably enhances the editing performance of locate-then-edit methods on Llama3 and GPT2-XL. For other cases, such as AlphaEdit on GPT-J, the improvements are minimal due to its already strong baseline performance.

5.3 Boosting General Capability Retention via BLUE

Model editing should not affect other aspects of LLMs. In addition to using specificity and fluency for evaluation, this goal can also be achieved by assessing changes in the general capabilities of the models after editing. Following the work of [6], we evaluate the

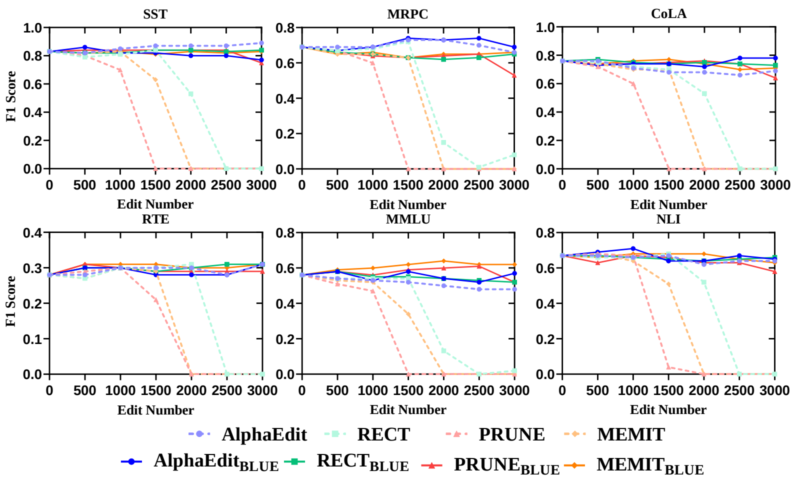
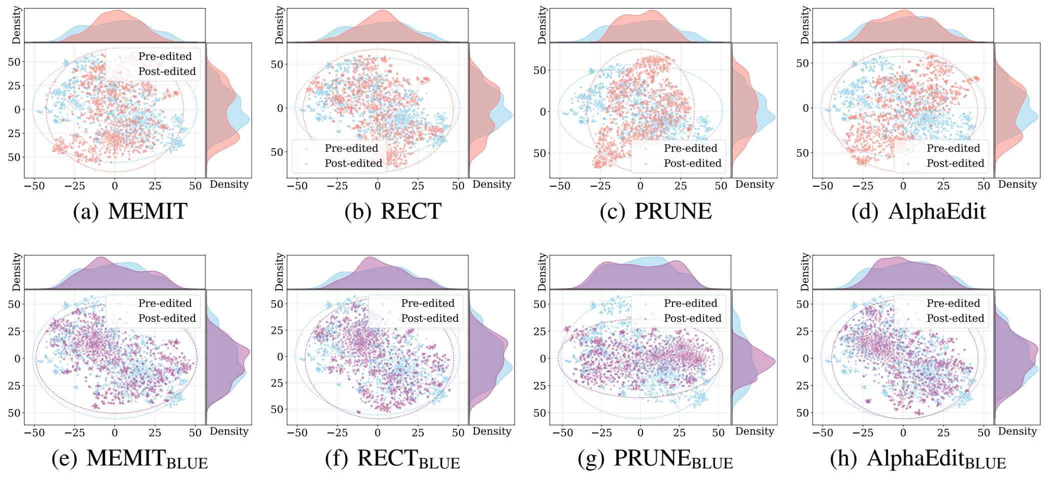


Figure 6: F1 scores of the post-edited Llama3 (8B) on six tasks.

general capabilities of LLMs before and after editing using six natural language tasks from the General Language Understanding Evaluation (GLUE) benchmark [30]. Specifically, we achieve this through the following six evaluation tasks: SST (The Stanford Sentiment Treebank) [31], MRPC (Microsoft Research Paraphrase Corpus) [32], MMLU (Massive Multi-task Language Understanding) [33], RTE (Recognizing Textual Entailment) [34], CoLA (Corpus of Linguistic Acceptability) [35], and NLI (Natural Language Inference) [36].

We conduct a total of 3,000 sequential edits on Llama3 (8B), with a batch size of 100 for each edit. Every 500 steps, we evaluate the performance of the post-edited LLMs on these six tasks. The results are shown in Figure 6. After 3,000 edits, the general capabilities of models edited by RECT, PRUNE, and MEMIT are almost entirely lost. In contrast, models edited by the BLUE-enhanced versions of these methods maintain their general capabilities well. Notably, AlphaEdit inherently demonstrates strong general capability retention, and AlphaEdit$_{\text{BLUE}}$ does not compromise this ability. These results indicate that **BLUE enhances the general capability retention of locate-then-edit methods**.



(a) MEMIT (b) RECT (c) PRUNE (d) AlphaEdit

(e) MEMIT$_{\text{BLUE}}$ (f) RECT$_{\text{BLUE}}$ (g) PRUNE$_{\text{BLUE}}$ (h) AlphaEdit$_{\text{BLUE}}$

Figure 7: The distribution of hidden states in pre-edited and post-edited Llama3 (8B).

5.4 Mitigating Hidden States Shifts with BLUE

The existing locate-then-edit methods often result in shifts in the hidden states of the model after editing [6]. In this part, we verify whether BLUE can alleviate this phenomenon. Specifically, we extract the hidden states of 1,000 randomly selected factual prompts from LLMs before and after editing. These hidden states are then reduced to two dimensions using t-SNE. The post-editing LLMs mentioned here are the models described in Section 5.2. We then visualize the hidden states, and the results are shown in Figure 7. It can be observed that the shifts in hidden states corresponding to the locate-then-edit method enhanced by BLUE are weaker than those of the original method. This demonstrates that **BLUE can mitigate hidden states shifts caused by locate-then-edit methods**. The results on GPT-J (6B) and GPT2-XL can be found in Appendix H.

5.5 BLUE Boosts Long-Form Performance of Locate-Then-Edit Approaches

We conduct long-form model editing experiments on editing Llama3 (8B) using UnKEBench [10], a representative work in long-form evaluation benchmarks. We adopt MEMIT and AlphaEdit, both enhanced using the editing paradigm proposed in AnyEdit [9], as baselines, and follow the same experimental setup as used in AnyEdit. As illustrated in Table. 3, the results demonstrate that BLUE **can also improve the performance of locate-then-edit methods in long-form evaluation scenarios**. We also conduct a case study on a long-form editing task to demonstrate in detail the effectiveness of BLUE in Appendix E.

Table 3: Comparison of Editing Methods on UnKEBench. The left side of '/' represents the LLM's edited output for original questions, while the right side represents the edited output for paraphrase questions.

Metric	BLEU	ROUGE-1	ROUGE-2	ROUGE-L	BERT Score	ROUGE-L (SubQ)
MEMIT	77.65 / 66.17	90.78 / 81.60	87.12 / 73.94	90.43 / 80.81	95.69 / 92.32	47.36
AlphaEdit	62.82 / 51.78	80.02 / 69.08	71.62 / 56.80	79.10 / 67.68	91.67 / 87.41	42.72
MEMIT$_{\text{BLUE}}$	**82.34 / 74.76**	**90.79 / 85.36**	86.89 / **78.94**	90.33 / **84.68**	**96.50 / 94.72**	**52.15**
AlphaEdit$_{\text{BLUE}}$	**68.64 / 61.49**	**84.90 / 78.54**	**77.98 / 68.40**	**84.08 / 77.37**	**93.39 / 90.97**	**49.49**

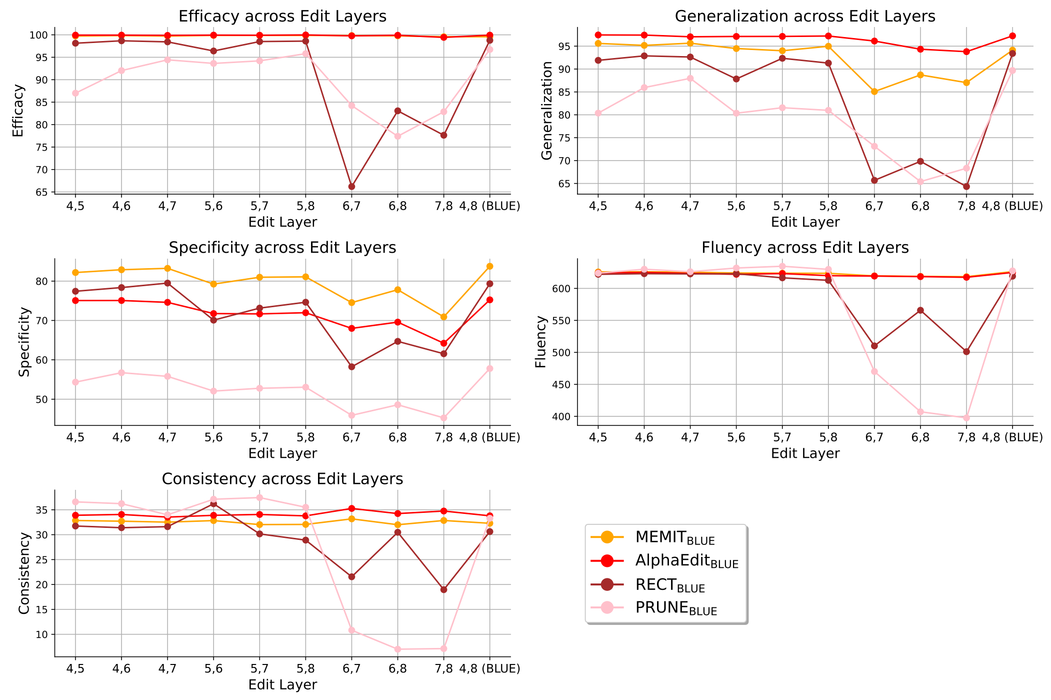


Figure 8: Layer Ablation of BLUE.

5.6 Ablation Study

To verify that the first and last critical layers selected by BLUE are optimal for editing, we perform 30 sequential edits with a batch size of 100 on Llama3 (8B). As shown in Figure 8, the optimal editing layers for all four BLUE-enhanced methods are consistently layers 4 and 8—the ones selected by BLUE. **This provides empirical evidence that the layers chosen by BLUE are indeed optimal for model editing.**

6 Conclusion

This paper rethinks the role of residual distribution in locate-then-edit model editing. Through empirical and theoretical analyses, we show that residual distribution is not an optimal choice, as it leads to increasing errors in weight updates with larger batch sizes, more sequential edits, and greater distribution distances. Based on these findings, we propose the BLUE strategy, which improves locate-then-edit methods by updating only the first and last critical layers of the model. Sequential batch editing experiments on three LLMs and two datasets demonstrate that BLUE effectively enhances editing performance. Further experiments and analyses indicate that BLUE also improves the retention of LLMs' original general capabilities and mitigates shifts in hidden states after editing. Moreover, BLUE yields gains in both time and memory efficiency and strengthens locate-then-edit methods in long-form model editing tasks.

Acknowledgements

This work was supported by the National Key Research and Development Program of China under Grant 2024YFB4506200, the Science and Technology Innovation Program of Hunan Province under Grant 2024RC1048, the National Key Laboratory Foundation Project under Grant 2024-KJWPDL-14, and the National Natural Science Foundation of China under Grant 62402506.

References

[1] Wang, S., Y. Zhu, H. Liu, et al. Knowledge editing for large language models: A survey. ACM Comput. Surv., 57(3), 2024.

[2] Yao, Y., P. Wang, B. Tian, et al. Editing large language models: Problems, methods, and opportunities. In H. Bouamor, J. Pino, K. Bali, eds., Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 10222–10240. Association for Computational Linguistics, Singapore, 2023.

[3] Geva, M., R. Schuster, J. Berant, et al. Transformer feed-forward layers are key-value memories. In M.-F. Moens, X. Huang, L. Specia, S. W.-t. Yih, eds., Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 5484–5495. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021.

[4] Meng, K., D. Bau, A. Andonian, et al. Locating and editing factual associations in gpt. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, A. Oh, eds., Advances in Neural Information Processing Systems, vol. 35, pages 17359–17372. Curran Associates, Inc., 2022.

[5] Meng, K., A. S. Sharma, A. J. Andonian, et al. Mass-editing memory in a transformer. In The Eleventh International Conference on Learning Representations.

[6] Fang, J., H. Jiang, K. Wang, et al. Alphaedit: Null-space constrained knowledge editing for language models. In The Thirteenth International Conference on Learning Representations.

[7] Gu, J.-C., H.-X. Xu, J.-Y. Ma, et al. Model editing harms general abilities of large language models: Regularization to the rescue. In Y. Al-Onaizan, M. Bansal, Y.-N. Chen, eds., Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 16801–16819. Association for Computational Linguistics, Miami, Florida, USA, 2024.

[8] Ma, J.-Y., H. Wang, H.-X. Xu, et al. Perturbation-restrained sequential model editing. In The Thirteenth International Conference on Learning Representations.

[9] Jiang, H., J. Fang, N. Zhang, et al. Anyedit: Edit any knowledge encoded in language models. In Forty-second International Conference on Machine Learning.

[10] Deng, J., Z. Wei, L. Pang, et al. Everything is editable: Extend knowledge editing to unstructured data in large language models. In The Thirteenth International Conference on Learning Representations.

[11] Zhong, Z., Z. Wu, C. D. Manning, et al. Mquake: Assessing knowledge editing in language models via multi-hop questions. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 15686–15702. 2023.

[12] Li, X., S. Li, S. Song, et al. Swea: Updating factual knowledge in large language models via subject word embedding altering. In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 39, pages 24494–24502. 2025.

[13] Huang, Z., Y. Shen, X. Zhang, et al. Transformer-patcher: One mistake worth one neuron. In The Eleventh International Conference on Learning Representations.

[14] Wang, P., Z. Li, N. Zhang, et al. Wise: rethinking the knowledge memory for lifelong model editing of large language models. In Proceedings of the 38th International Conference on Neural Information Processing Systems, NIPS '24. Curran Associates Inc., Red Hook, NY, USA, 2025.

[15] Zheng, C., L. Li, Q. Dong, et al. Can we edit factual knowledge by in-context learning? In H. Bouamor, J. Pino, K. Bali, eds., Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 4862–4876. Association for Computational Linguistics, Singapore, 2023.

[16] Mitchell, E., C. Lin, A. Bosselut, et al. Memory-based model editing at scale. In International Conference on Machine Learning, pages 15817–15831. PMLR, 2022.

[17] Hartvigsen, T., S. Sankaranarayanan, H. Palangi, et al. Aging with grace: Lifelong model editing with discrete key-value adaptors. Advances in Neural Information Processing Systems, 36, 2024.

[18] Tan, C., G. Zhang, J. Fu. Massive editing for large language models via meta learning. In The Twelfth International Conference on Learning Representations.

[19] Zhu, C., A. S. Rawat, M. Zaheer, et al. Modifying memories in transformer models. arXiv preprint arXiv:2012.00363, 2020.

[20] Mitchell, E., C. Lin, A. Bosselut, et al. Fast model editing at scale. In International Conference on Learning Representations. 2022.

[21] Li, X., S. Li, S. Song, et al. Pmet: Precise model editing in a transformer. Proceedings of the AAAI Conference on Artificial Intelligence, 38(17):18564–18572, 2024.

[22] Zhang, N., Y. Yao, B. Tian, et al. A comprehensive study of knowledge editing for large language models. arXiv preprint arXiv:2401.01286, 2024.

[23] Mazzia, V., A. Pedrani, A. Caciolai, et al. A survey on knowledge editing of neural networks. arXiv preprint arXiv:2310.19704, 2023.

[24] Meta, A. Introducing meta llama 3: The most capable openly available llm to date. Meta AI, 2024.

[25] Wang, B., A. Komatsuzaki. Gpt-j-6b: A 6 billion parameter autoregressive language model, 2021.

[26] Levy, O., M. Seo, E. Choi, et al. Zero-shot relation extraction via reading comprehension. arXiv preprint arXiv:1706.04115, 2017.

[27] Radford, A., J. Wu, R. Child, et al. Language models are unsupervised multitask learners. OpenAI blog, 1(8):9, 2019.

[28] AI@Meta. Llama 3 model card. 2024.

[29] Gu, J.-C., H.-X. Xu, J.-Y. Ma, et al. Model editing harms general abilities of large language models: Regularization to the rescue. In Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 16801–16819. 2024.

[30] Wang, A. Glue: A multi-task benchmark and analysis platform for natural language understanding. arXiv preprint arXiv:1804.07461, 2018.

[31] Socher, R., A. Perelygin, J. Wu, et al. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 conference on empirical methods in natural language processing, pages 1631–1642. 2013.

[32] Dolan, B., C. Brockett. Automatically constructing a corpus of sentential paraphrases. In Third international workshop on paraphrasing (IWP2005). 2005.

[33] Hendrycks, D., C. Burns, S. Basart, et al. Measuring massive multitask language understanding. arXiv preprint arXiv:2009.03300, 2020.

[34] Bentivogli, L., P. Clark, I. Dagan, et al. The fifth pascal recognizing textual entailment challenge. TAC, 7(8):1, 2009.

[35] Warstadt, A. Neural network acceptability judgments. arXiv preprint arXiv:1805.12471, 2019.

[36] Williams, A., N. Nangia, S. R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. arXiv preprint arXiv:1704.05426, 2017.

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 main claims made in 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: The paper discusses the limitations of the work 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: [Yes]

Justification: The paper provides the full set of assumptions and a complete (and correct) proof for Theorem 4.1.

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: Yes, we provide a zip file with the code needed to reproduce our experiments, as well as a README with instructions.

   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: [Yes]

Justification: We share our code. The datasets will be automatically downloaded from open-source resources when the code is executed.

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, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We share our code and hyperparameters.

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: We report error bars in Figures 2 3 5 and 96% CI in the main results (Table. 2).

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: We discuss computational resources used for all experiments in Appendix D.

   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 research conducted in the paper conforms, 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: Please see 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: [NA]

    Justification: The paper poses no such risks.

    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: We cite the papers that introduced the models and data used in our work.

    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: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    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.

A Proof of Theorem 4.1

Proof. Let ${\boldsymbol{K}_1^l}^T \left(\boldsymbol{K}_0^l {\boldsymbol{K}_0^l}^T + \boldsymbol{K}_1^l {\boldsymbol{K}_1^l}^T\right)^{-1} := \boldsymbol{Q}$, then the weight shifts error is:

$$\begin{aligned} \|\Delta^{l^*} - \Delta^l\|_2 &= \|\boldsymbol{R}^{l^*}\boldsymbol{Q} - \boldsymbol{R}^l\boldsymbol{Q}\|_2 && (13)\\ &\leq \|\boldsymbol{R}^{l^*} - \boldsymbol{R}^l\|_2\|\boldsymbol{Q}\|_2 && (14)\\ &\leq \|\boldsymbol{R}^{l^*} - \frac{\boldsymbol{R}^L}{L-l+1}\|_2\|\boldsymbol{Q}\|_2 && (15)\end{aligned}$$

According to Section 4.1.2, the directly computed $\boldsymbol{m}_i^l$ represents the optimal memory for editing, and thus we have $\boldsymbol{R}^L = \boldsymbol{R}^{L^*}$. Then, Equ. (15) can be written as:

$$\begin{aligned} &\|\boldsymbol{R}^{l^*} - \frac{\boldsymbol{R}^{L^*}}{L-l+1}\|_2\|\boldsymbol{Q}\|_2 && (16)\\ =&\|\boldsymbol{R}^{l^*} - \boldsymbol{R}^{L^*} + \boldsymbol{R}^{L^*} - \frac{\boldsymbol{R}^{L^*}}{L-l+1}\|_2\|\boldsymbol{Q}\|_2 && (17)\\ \leq&\left(\|\boldsymbol{R}^{l^*} - \boldsymbol{R}^{L^*}\|_2 + \|\boldsymbol{R}^{L^*} - \frac{\boldsymbol{R}^{L^*}}{L-l+1}\|_2\right)\|\boldsymbol{Q}\|_2 && (18)\\ \leq&\left(\|\boldsymbol{R}^{l^*} - \boldsymbol{R}^{L^*}\|_2 + \frac{L-l}{L-l+1}\|\boldsymbol{R}^{L^*}\|_2\right)\|\boldsymbol{Q}\|_2 && (19)\\ \leq&\left(\|\boldsymbol{R}^{l^*} - \boldsymbol{R}^{L}\|_2 + (L-l)\|\boldsymbol{R}^{L}\|_2\right)\|\boldsymbol{Q}\|_2 && (20)\end{aligned}$$

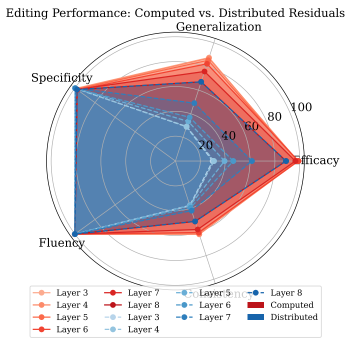
□

B BLUE's Performance Improvement in Locate-then-Edit Model Editing Across LLMs

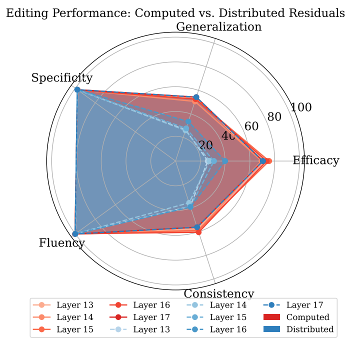
We summarize the enhancement degree of each method by BLUE across different models, as shown in the table below.

Table 4: The average performance improvement of BLUE on different locate-then-edit model editing across various LLMs. The abnormal results of PRUNE editing GPT-J on the ZsRE dataset are excluded in our statistics.

Model	MEMIT	PRUNE	RECT	AlphaEdit
Llama3	129.61%	144.52%	27.05%	2.50 %
GPT-J	6.73%	44.36%	0.58%	0.03%
GPT2-XL	17.02%	41.24%	9.47%	4.00%



(a) GPT-J (6B)



(b) GPT-XL

Figure 9: Performance variations when editing different single layers of the model using computed and distributed residuals separately. For better visualization, Fluency and Consistency were normalized.

C Supplementary Results of Post-edit LLM Performance

We show the supplementary results of post-edit LLM performance in Figure 9. The results also indicate that distributed residuals introduce significant information loss during model editing, leading to a higher likelihood of editing failures.

D Experiment Details

All our experiments are conducted on A800 GPUs. The baseline methods used for comparison in the experiments are kept in their original settings, with PRUNE following the reproduction settings of [6]. For baseline methods enhanced by BLUE, all configurations remain consistent with the original baselines, except for AlphaEdit$_{\text{BLUE}}$. For AlphaEdit$_{\text{BLUE}}$, we set the α values for Llama3 (8B), GPT-J (6B), and GPT2-XL to 1, 95, and 80, respectively, to ensure the invertibility of matrices during the editing process, thereby achieving better editing performance. For a clearer understanding of the baselines, please refer to [6].

E Case Study on Long-form Model Editing Task

To demonstrate in detail the improvements of BLUE in long-form model editing, we present a case study of editing Llama-3 (8B) in Table 5. It can be seen that the method enhanced by BLUE generates content that is noticeably closer to the target output compared to the original method. This provides additional evidence for the effectiveness of BLUE in long-form model editing tasks.

F Batch Model Editing

In addition to sequential batch editing, large-scale batch editing is also an important aspect of evaluating the performance of model editing methods. Therefore, we conducted 10,000 batch edits for both the baseline and the BLUE-enhanced methods, with the results shown in Table 6. The results in the table indicate that while the improvement in large-scale batch editing after applying the BLUE enhancement to the baseline is not as significant as in sequential batch editing, the baselines enhanced by BLUE still demonstrate overall stronger performance. Specifically, 70.83% of the metrics (68 out of 96) are improved. Note that although the baselines enhanced by BLUE performed better in terms of efficacy and generalization, they show worse results in specificity. This suggests that while the BLUE-enhanced model editing methods strengthen the knowledge being edited, it also affects other unrelated knowledge. Achieving optimal performance across all three metrics simultaneously remains a major challenge in model editing [14]. This is particularly true for locate-then-edit methods, as BLUE serves as an enhancement to existing editing methods without altering their original modeling. Therefore, addressing this issue may require future work on improving the original modeling of

Table 5: Examples of different editing methods in long-form editing scenarios.

Type	**Text**
Input	What is George Rankin's occupation?
Paraphrase	What does George Rankin do for a living?
Target Output	George Rankin has been actively involved in politics for over a decade. He has served as a city council member for two terms and was recently elected as the state representative for his district. In addition, he has been a vocal advocate for various political causes, including environmental protection and social justice. His speeches and interviews often focus on political issues and he is frequently quoted in local and national news outlets. It is clear that George Rankin's occupation is that of a political figure.
AlphaEdit	George Rankin has been involved in politics for over 20 years. He has served as a member of the state legislature for 10 years and has been a mayor for 5 years. He has also been a member of various political committees and has run for several political offices, including governor and U.S. Senate.
MEMIT	George Rankin has been involved in politics for over 20 years. He has served as a member of the state legislature for 10 years and has been a mayor for 5 years. He has also run for governor and senator, but has not held those offices.
AlphaEdit$_{BLUE}$	George Rankin has been actively involved in politics for over a decade. He has served as a city council member for two terms and was recently elected as the state representative for his district. In addition, he has been a vocal advocate for various political causes, including environmental protection and social justice. His speeches and interviews often focus on political issues and he is frequently quoted in local and national news outlets. It is clear that George Rankin has been involved in politics for a long time and continues to be an active and influential figure in the political scene.
MEMIT$_{BLUE}$	George Rankin has been actively involved in politics for over a decade. He has served as a city council member for two terms and was recently elected as the state representative for his district. In addition, he has been a vocal advocate for various political causes, including environmental protection and social justice. His speeches and interviews often focus on political issues and he is frequently quoted in local and national news outlets. It is clear that George Rankin's involvement in politics is a significant aspect of his public persona and identity.

editing methods. Nevertheless, the overall superior performance of the BLUE-enhanced methods also demonstrates that BLUE can improve the editing performance of locate-then-edit approaches in batch editing.

G Results on LLaMA-2-13B

To demonstrate the effectiveness of BLUE on larger models, we selecte AlphaEdit and MEMIT to perform sequential editing experiments on layers 30–34 of LLaMA-2-13B, following the same setup as in the main experiments. The results are shown in Table 7. The results demonstrate that the BLUE-enhanced methods achieve better overall editing performance than the baseline methods on both the CounterFact and ZsRE datasets. We bolded all the enhanced results. On the CounterFact dataset, all metrics except for Specificity show improvements over the baseline. On the ZsRE dataset, all metrics of AlphaEdit show improvements, while MEMIT exhibits improvements in all metrics except for Specificity. This indicates that BLUE is also effective on LLaMA-2-13B, further validating the effectiveness of our approach.

H Hidden States Shifts in GPT-J (6B) and GPT2-XL

We present the hidden state shifts before and after model editing for GPT-J (6B) and GPT2-XL in Figs. 10 and 11, respectively. Similar results to those on Llama3 (8B) are observed for GPT-J (6B) and GPT-2 XL. The BLUE-enhanced baselines have a smaller overall impact on the model's hidden

Table 6: Comparison of BLUE enhanced locate-then-model editing methods with original locate-then-model editing methods on the batch model editing task. *Eff.*, *Gen.*, *Spe.*, *Flu.* and *Consis.* denote Efficacy, Generalization, Specificity, Fluency and Consistency, respectively. We color all results that are actually enhanced by BLUE in red.

Method	Model	Counterfact					ZsRE		
		Eff.↑	Gen.↑	Spe.↑	Flu.↑	Consis.↑	Eff.↑	Gen.↑	Spe.↑
Pre-edited		7.02$_{\pm0.50}$	9.44$_{\pm0.49}$	89.73$_{\pm0.36}$	630.00$_{\pm0.22}$	24.21$_{\pm0.17}$	35.67$_{\pm0.58}$	34.81$_{\pm0.58}$	31.83$_{\pm0.44}$
MEMIT	Llama3	93.53$_{\pm0.48}$	74.12$_{\pm0.75}$	84.18$_{\pm0.40}$	626.24$_{\pm0.26}$	29.71$_{\pm0.20}$	86.57$_{\pm0.48}$	82.58$_{\pm0.54}$	32.47$_{\pm0.44}$
PRUNE		93.64$_{\pm0.48}$	84.44$_{\pm0.57}$	60.00$_{\pm0.57}$	625.11$_{\pm0.26}$	36.83$_{\pm0.22}$	13.29$_{\pm0.34}$	13.75$_{\pm0.52}$	15.34$_{\pm0.54}$
RECT		58.07$_{\pm0.97}$	39.88$_{\pm0.86}$	88.15$_{\pm0.37}$	628.71$_{\pm0.23}$	26.11$_{\pm0.18}$	70.35$_{\pm0.65}$	65.04$_{\pm0.67}$	32.45$_{\pm0.44}$
AlphaEdit		88.89$_{\pm0.62}$	69.91$_{\pm0.79}$	83.98$_{\pm0.41}$	625.78$_{\pm0.25}$	28.66$_{\pm0.19}$	84.07$_{\pm0.52}$	80.15$_{\pm0.57}$	32.50$_{\pm0.44}$
MEMIT$_{\text{BLUE}}$		99.28$_{\pm0.17}$	93.83$_{\pm0.40}$	79.34$_{\pm0.46}$	626.09$_{\pm0.26}$	33.04$_{\pm0.21}$	95.38$_{\pm0.23}$	92.62$_{\pm0.33}$	31.68$_{\pm0.44}$
PRUNE$_{\text{BLUE}}$		99.37$_{\pm0.16}$	94.30$_{\pm0.35}$	60.17$_{\pm0.57}$	623.54$_{\pm0.24}$	36.64$_{\pm0.21}$	86.83$_{\pm0.44}$	83.55$_{\pm0.50}$	28.23$_{\pm0.42}$
RECT$_{\text{BLUE}}$		94.02$_{\pm0.46}$	79.25$_{\pm0.69}$	85.45$_{\pm0.39}$	627.57$_{\pm0.24}$	30.34$_{\pm0.19}$	85.27$_{\pm0.48}$	77.73$_{\pm0.58}$	31.83$_{\pm0.44}$
AlphaEdit$_{\text{BLUE}}$		98.62$_{\pm0.23}$	90.76$_{\pm0.48}$	79.61$_{\pm0.45}$	625.04$_{\pm0.27}$	32.46$_{\pm0.21}$	94.14$_{\pm0.27}$	90.64$_{\pm0.38}$	31.52$_{\pm0.44}$
Pre-edited		15.20$_{\pm0.70}$	17.70$_{\pm0.60}$	83.50$_{\pm0.50}$	622.40$_{\pm0.30}$	29.40$_{\pm0.20}$	26.40$_{\pm0.60}$	25.80$_{\pm0.50}$	27.00$_{\pm0.50}$
MEMIT	GPT-J	98.72$_{\pm0.22}$	87.14$_{\pm0.56}$	74.05$_{\pm0.52}$	620.68$_{\pm0.30}$	39.72$_{\pm0.24}$	95.90$_{\pm0.30}$	89.06$_{\pm0.49}$	26.30$_{\pm0.50}$
PRUNE		91.54$_{\pm0.55}$	90.00$_{\pm0.49}$	57.49$_{\pm0.60}$	562.52$_{\pm0.58}$	37.34$_{\pm0.20}$	29.98$_{\pm0.67}$	26.91$_{\pm0.65}$	16.75$_{\pm0.40}$
RECT		88.13$_{\pm0.63}$	63.40$_{\pm0.83}$	79.31$_{\pm0.50}$	622.54$_{\pm0.27}$	35.62$_{\pm0.22}$	70.46$_{\pm0.70}$	61.90$_{\pm0.73}$	26.64$_{\pm0.50}$
AlphaEdit		99.26$_{\pm0.17}$	86.70$_{\pm0.56}$	69.65$_{\pm0.53}$	587.89$_{\pm0.49}$	39.51$_{\pm0.23}$	93.09$_{\pm0.36}$	82.64$_{\pm0.59}$	22.78$_{\pm0.47}$
MEMIT$_{\text{BLUE}}$		99.58$_{\pm0.13}$	97.40$_{\pm0.25}$	64.92$_{\pm0.55}$	615.97$_{\pm0.36}$	40.83$_{\pm0.25}$	98.18$_{\pm0.18}$	93.61$_{\pm0.38}$	25.91$_{\pm0.49}$
PRUNE$_{\text{BLUE}}$		99.36$_{\pm0.16}$	98.06$_{\pm0.22}$	56.82$_{\pm0.55}$	608.78$_{\pm0.33}$	41.76$_{\pm0.22}$	74.83$_{\pm0.65}$	71.24$_{\pm0.69}$	20.07$_{\pm0.46}$
RECT$_{\text{BLUE}}$		97.86$_{\pm0.28}$	88.41$_{\pm0.54}$	74.91$_{\pm0.52}$	621.45$_{\pm0.30}$	38.79$_{\pm0.23}$	90.20$_{\pm0.46}$	81.00$_{\pm0.61}$	27.31$_{\pm0.51}$
AlphaEdit$_{\text{BLUE}}$		99.39$_{\pm0.15}$	95.52$_{\pm0.35}$	69.28$_{\pm0.54}$	619.90$_{\pm0.32}$	41.25$_{\pm0.24}$	98.26$_{\pm0.19}$	96.63$_{\pm0.39}$	26.59$_{\pm0.50}$
Pre-edited		21.82$_{\pm0.81}$	24.16$_{\pm0.72}$	78.32$_{\pm0.55}$	626.78$_{\pm0.23}$	31.37$_{\pm0.20}$	22.17$_{\pm0.52}$	21.28$_{\pm0.51}$	24.2$_{\pm0.48}$
MEMIT	GPT2-XL	79.64$_{\pm0.79}$	65.86$_{\pm0.83}$	70.01$_{\pm0.56}$	625.67$_{\pm0.27}$	36.17$_{\pm0.22}$	62.46$_{\pm0.75}$	57.59$_{\pm0.77}$	25.86$_{\pm0.50}$
PRUNE		85.27$_{\pm0.69}$	78.30$_{\pm0.70}$	57.73$_{\pm0.62}$	604.09$_{\pm0.39}$	35.66$_{\pm0.21}$	42.71$_{\pm0.76}$	40.14$_{\pm0.75}$	19.01$_{\pm0.44}$
RECT		61.92$_{\pm0.95}$	48.68$_{\pm0.87}$	74.69$_{\pm0.54}$	625.87$_{\pm0.25}$	33.99$_{\pm0.21}$	49.37$_{\pm0.76}$	45.30$_{\pm0.74}$	25.64$_{\pm0.49}$
AlphaEdit		93.24$_{\pm0.49}$	76.28$_{\pm0.71}$	64.54$_{\pm0.57}$	604.70$_{\pm0.38}$	38.62$_{\pm0.23}$	61.26$_{\pm0.74}$	54.82$_{\pm0.76}$	20.83$_{\pm0.45}$
MEMIT$_{\text{BLUE}}$		87.54$_{\pm0.65}$	78.14$_{\pm0.71}$	65.37$_{\pm0.54}$	615.34$_{\pm0.43}$	37.10$_{\pm0.22}$	71.93$_{\pm0.72}$	67.51$_{\pm0.75}$	23.44$_{\pm0.49}$
PRUNE$_{\text{BLUE}}$		95.70$_{\pm0.40}$	90.18$_{\pm0.48}$	53.81$_{\pm0.57}$	596.30$_{\pm0.56}$	37.02$_{\pm0.24}$	51.06$_{\pm0.77}$	47.82$_{\pm0.76}$	14.74$_{\pm0.39}$
RECT$_{\text{BLUE}}$		70.93$_{\pm0.89}$	58.73$_{\pm0.85}$	72.09$_{\pm0.53}$	621.52$_{\pm0.34}$	34.78$_{\pm0.21}$	58.88$_{\pm0.77}$	54.29$_{\pm0.77}$	24.55$_{\pm0.49}$
AlphaEdit$_{\text{BLUE}}$		94.10$_{\pm0.46}$	81.20$_{\pm0.65}$	64.53$_{\pm0.56}$	620.79$_{\pm0.31}$	38.81$_{\pm0.21}$	76.68$_{\pm0.65}$	70.16$_{\pm0.73}$	23.00$_{\pm0.47}$

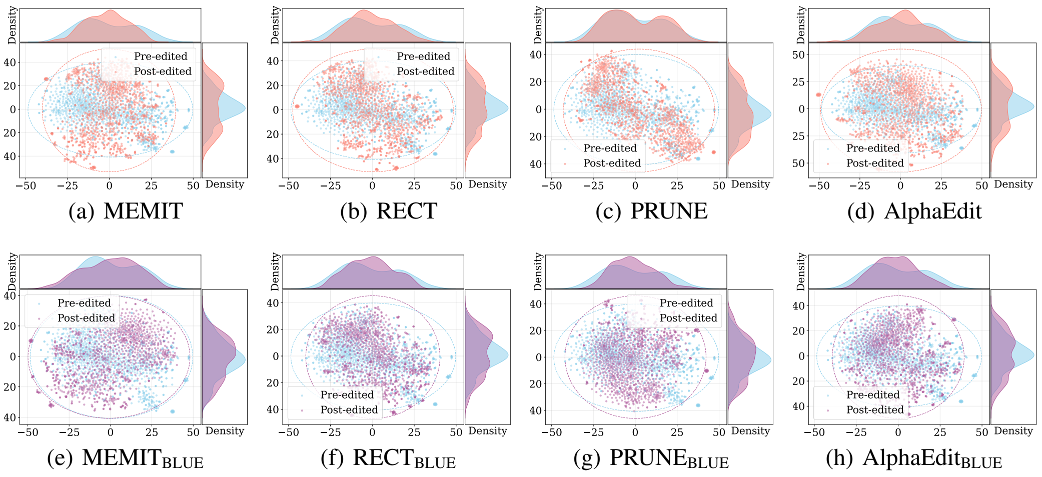
Table 7: Comparison of different editing methods on CounterFact and ZsRE datasets when editing LLaMA-2-13B.

Method	CounterFact Dataset					ZsRE Dataset		
	Efficacy	Generalization	Specificity	Fluency	Consistency	Efficacy	Generalization	Specificity
AlphaEdit	52.30$_{\pm2.19}$	48.95$_{\pm1.73}$	**50.53**$_{\pm2.21}$	408.92$_{\pm2.36}$	0.22$_{\pm0.01}$	52.31$_{\pm1.44}$	37.68$_{\pm1.49}$	9.11$_{\pm0.60}$
AlphaEdit$_{\text{BLUE}}$	**78.75**$_{\pm1.79}$	**69.53**$_{\pm1.81}$	44.27$_{\pm1.76}$	**513.49**$_{\pm3.09}$	**22.10**$_{\pm0.57}$	**53.01**$_{\pm2.32}$	**41.32**$_{\pm1.52}$	**9.18**$_{\pm0.60}$
MEMIT	79.20$_{\pm1.77}$	65.94$_{\pm1.79}$	**41.53**$_{\pm1.75}$	378.10$_{\pm4.01}$	14.44$_{\pm0.59}$	50.11$_{\pm1.46}$	34.70$_{\pm1.47}$	**9.71**$_{\pm0.61}$
MEMIT$_{\text{BLUE}}$	**79.60**$_{\pm1.78}$	**66.00**$_{\pm1.81}$	41.28$_{\pm1.65}$	**384.23**$_{\pm3.50}$	**14.65**$_{\pm0.59}$	**51.07**$_{\pm1.45}$	**37.13**$_{\pm1.52}$	9.07$_{\pm0.60}$

states compared to the original baselines. This indicates that BLUE can mitigate the hidden state shifts caused by locate-then-edit methods, suggesting that **the BLUE-enhanced baselines introduce fewer side effects to the original model.**

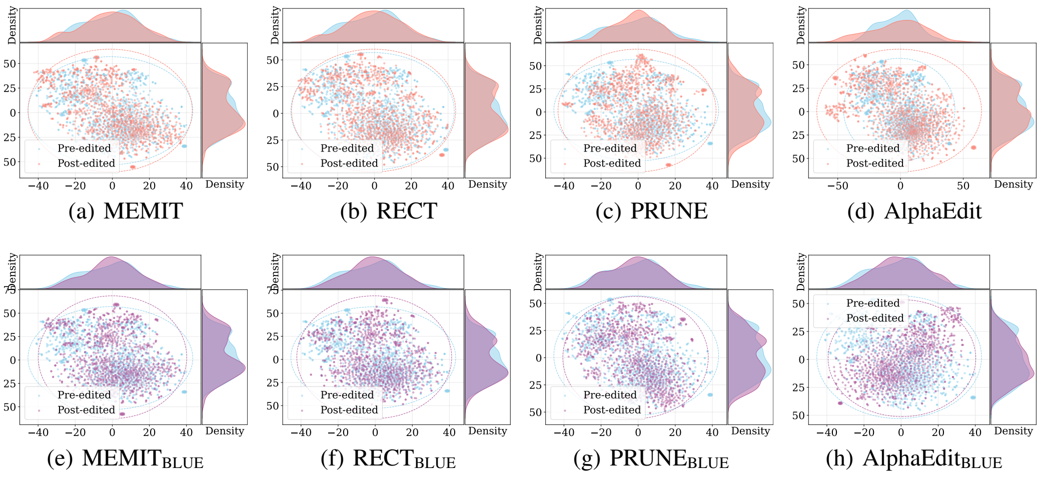
I Efficiency Analysis

To evaluate the time and peak memory required for a single edit, we perform 300 sequential edits with a batch size of 1 and record the editing time and peak memory for each edit. The average peak memory consumption and editing time per instance for each method are reported in Table 8 and Figure 12. In terms of peak memory, BLUE achieves higher memory efficiency than the original methods. This is


(a) MEMIT (b) RECT (c) PRUNE (d) AlphaEdit

(e) $\text{MEMIT}_{\text{BLUE}}$ (f) $\text{RECT}_{\text{BLUE}}$ (g) $\text{PRUNE}_{\text{BLUE}}$ (h) $\text{AlphaEdit}_{\text{BLUE}}$

Figure 10: The distribution of hidden states in pre-edited and post-edited GPT-J (6B).


(a) MEMIT (b) RECT (c) PRUNE (d) AlphaEdit

(e) $\text{MEMIT}_{\text{BLUE}}$ (f) $\text{RECT}_{\text{BLUE}}$ (g) $\text{PRUNE}_{\text{BLUE}}$ (h) $\text{AlphaEdit}_{\text{BLUE}}$

Figure 11: The distribution of hidden states in pre-edited and post-edited GPT2-XL.

intuitive: by discarding residual distributions and instead computing residuals only for the first and last key layers, BLUE reduces the memory overhead associated with residual distribution. Regarding editing time, with the exception of PRUNEBLUE on Llama-3 (8B)—which incurs a slightly higher editing time than its original counterpart—BLUE-enhanced methods consistently achieve lower average editing times across different models and datasets. These findings demonstrate that **BLUE not only improves the editing performance of locate-then-edit methods, but also enhances their editing efficiency.**

J Experimental validation of the theorem and lemma

In this section, we experimentally validate the correctness of the theorem and lemma presented in Section 4.2. In this section, we choose llama3 (8B) as the target model for editing, and use MEMIT and AlphaEdit as the editing methods.

J.1 Experimental verification of Theorem 4.1

According to Theorem 4.1, the weight shift error increases with the batch editing size. To verify this, we conduct batch editing experiments with 500, 1000, 5000, and 10000 edits on the CounterFact and ZsRE datasets. The results are shown in Figure 13. It can be clearly observed that on both datasets,

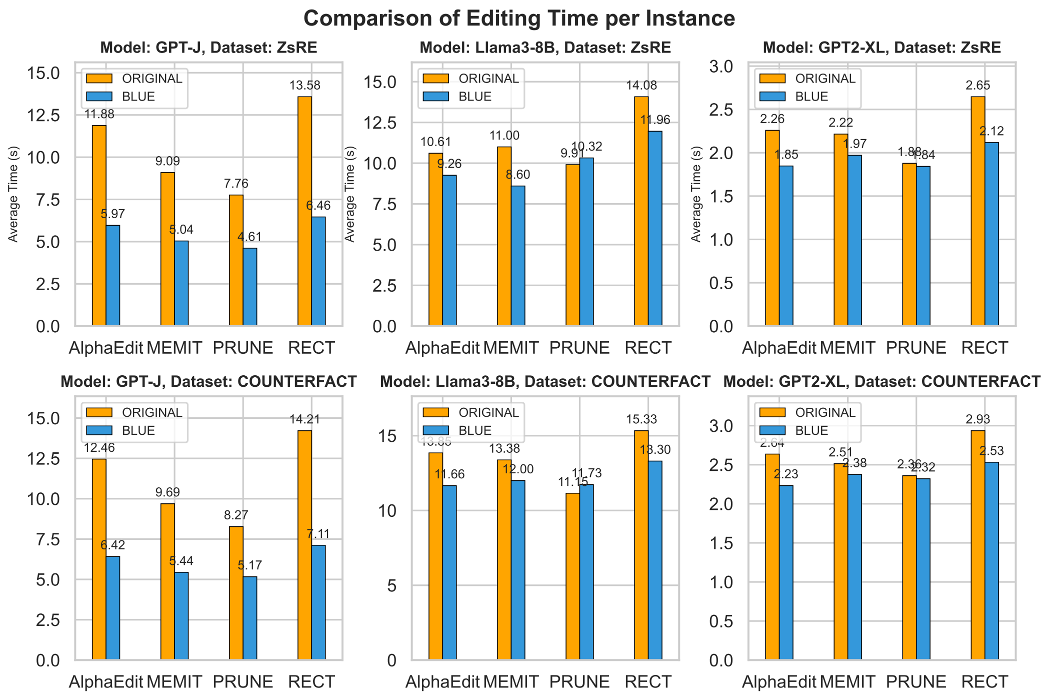


Figure 12: Average Editing Time per Instance for Different Methods

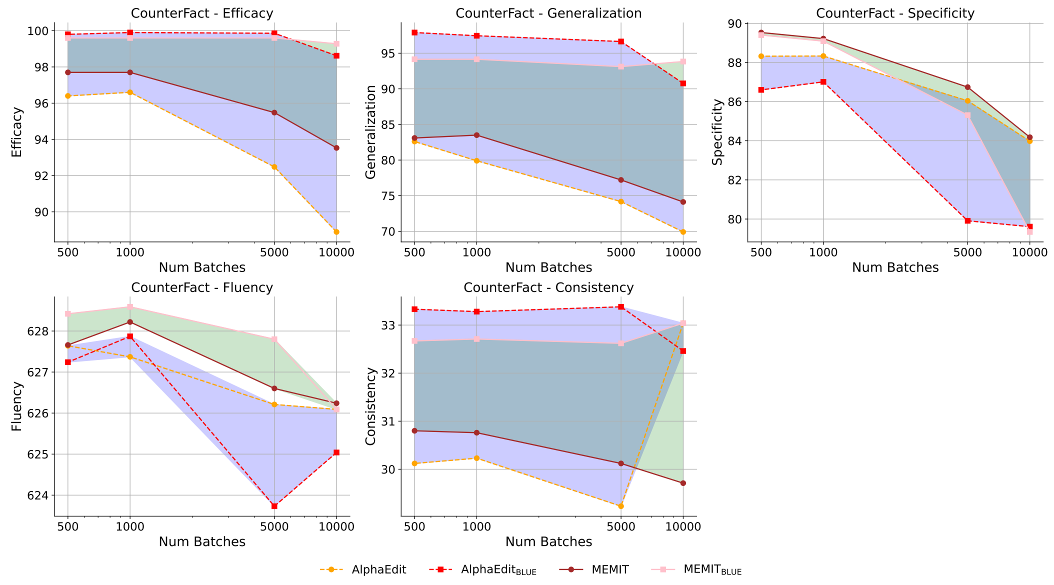
Table 8: Peak memory usage of both BLUE and locate-then-edit

Method	CounterFact			ZsRE		
	Llama-3	GPT-J	GPT-2XL	Llama-3	GPT-J	GPT-2XL
AlphaEdit	36.09	30.43	7.14	36.09	30.43	7.43
AlphaEdit$_{BLUE}$	35.10	28.92	7.24	35.43	28.92	7.49
MEMIT	37.07	31.68	7.34	37.07	31.68	7.34
MEMIT$_{BLUE}$	36.42	30.67	7.30	36.42	30.67	7.38
RECT	37.07	31.68	7.34	37.07	31.68	7.34
RECT$_{BLUE}$	36.42	30.67	7.22	36.42	30.67	7.37
PRUNE	38.17	33.18	7.53	38.17	33.18	7.53
PRUNE$_{BLUE}$	36.85	30.67	7.30	36.85	30.67	7.46

the overall performance gap between BLUE-enhanced methods and non-enhanced methods increases with the batch size, especially in terms of efficacy and generalization. Although not all metrics show a strictly increasing performance gap with larger batch sizes—for example, on the Consistency metric of the CounterFact dataset, the performance gap for 5000 batch edits is greater than that for 10000 edits—the overall trend still empirically supports the correctness of Theorem 4.1.

J.2 Experimental verification of Lemma 4.3

According to Lemma 4.3, the weight shift error increases with the number of sequential edits. To verify this, we conduct sequential editing experiments with a batch size of 1, using 100, 500, 1000, and 5000 edits. As shown in Figure 14, similar to the batch editing results, the overall trend on both datasets shows that the performance gap between the BLUE-enhanced methods and the original methods increases with the number of sequential edits. For instance, on the CounterFact dataset, the performance gap in efficacy and generalization between AlphaEdit and AlphaEdit$_{\text{BLUE}}$ increases significantly when the number of edits reaches 5000. Although specificity and consistency do not exhibit this trend, the patterns observed in efficacy and generalization still support the conclusion that weight shift error increases with the number of sequential edits, as efficacy and generalization directly reflect the impact of the editing methods.


(a) CounterFact

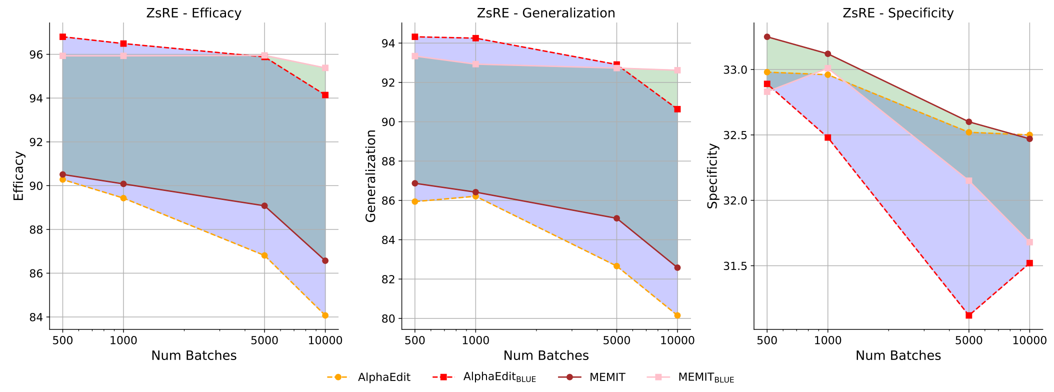
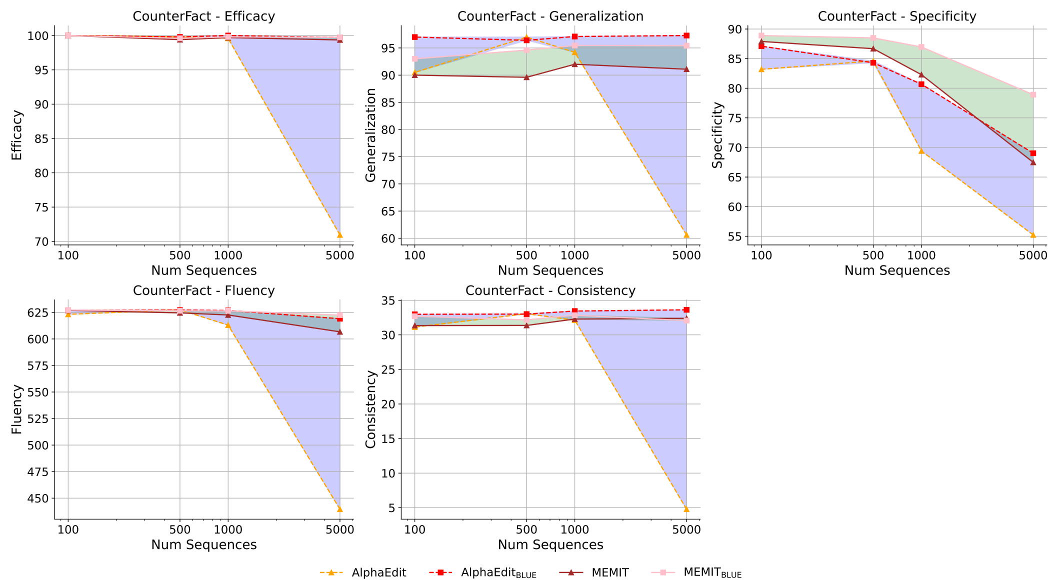

(b) ZsRE

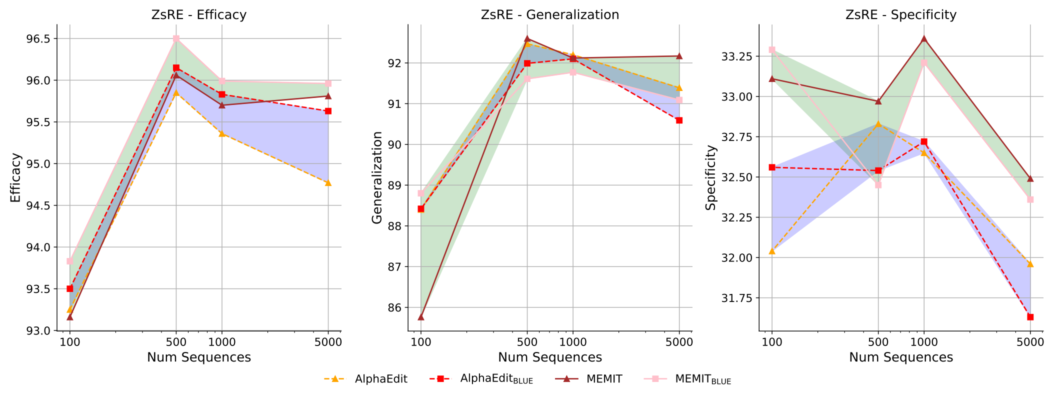
Figure 13: Performance Variation of Model Editing with Batch Edits on CounterFact and ZsRE Datasets

K BLUE in the Locate-then-edit Method with Square Root Residual Distribution

Some locate-then-edit methods (e.g., PMET [21]) use a square root residual distribution instead of an even spread. They claim that the square root residual distribution can mitigate information loss during residual distribution. Since BLUE is designed for locate-then-edit methods with even residual distribution, we do not consider such methods as baselines. Nevertheless, we attempt to enhance PMET with BLUE. The results of sequential batch editing are shown in Table 9. $\text{PMET}_{\text{BLUE}}$ exhibits a significant performance improvement when editing Llama3 on sequential model editing task, while its performance gains in other scenarios are relatively limited. We speculate that this may be because PMET's use of square root distribution retains more editing information compared to even distribution, leading to the limited improvement of BLUE. Additionally, PMET incorporates a self-attention module during editing optimization but only edits the FFN weights when updating model parameters. This might result in BLUE's two-layer update being insufficient to fully integrate the editing information into the model weights. Nevertheless, BLUE demonstrates notable improvements



(a) CounterFact



(b) ZsRE

Figure 14: Performance Variation of Model Editing with Sequential Edits on CounterFact and ZsRE Datasets

in the locate-then-edit approaches with residual even distribution, indicating that it remains applicable to most locate-then-edit methods.

L Limitations

As a general strategy applicable to locate-then-edit methods, BLUE enhances various aspects of the locate-then-edit paradigm. Extensive evidence demonstrates that BLUE improves the editing performance of locate-then-edit methods, preserves the original model's capabilities post-editing, and alleviates the shift in hidden representations introduced by editing. Moreover, with updates applied to only two layers, it also improves editing efficiency. However, the improvement of BLUE in the locate-then-edit method for editing reasoning knowledge (e.g., MQuAKE [11]) remains to be verified, and we leave it as future work..

Table 9: Comparison of PMET$_{\text{BLUE}}$ with original PMET on the sequential batch and batch model editing task. We color all results that are actually enhanced by BLUE in red.

Method	Model	Counterfact					ZsRE		
		Efficacy↑	Generalization↑	Specificity↑	Fluency↑	Consistency↑	Efficacy↑	Generalization↑	Specificity↑
Sequential Model Editing Task									
Pre-edited	Llama3	7.85±0.26	10.58±0.26	89.48±0.18	635.23±0.11	24.14±0.08	36.99±0.30	36.34±0.30	31.89±0.22
PMET	Llama3	99.47±0.26	90.78±0.84	76.07±0.92	619.62±0.66	32.45±0.41	94.97±0.45	89.98±0.75	32.95±0.81
PMET$_{\text{BLUE}}$	Llama3	99.57±0.24	94.13±0.69	83.77±0.77	626.26±0.51	32.29±0.38	96.07±0.36	91.73±0.66	32.66±0.81
Pre-edited	GPT-J	16.22±0.31	18.56±0.45	83.11±0.13	621.81±0.67	29.74±0.51	26.32±0.37	25.79±0.25	27.42±0.53
PMET	GPT-J	99.73±0.18	93.93±0.70	72.32±0.96	618.82±0.62	41.77±0.45	99.07±0.26	96.10±0.56	28.79±0.93
PMET$_{\text{BLUE}}$	GPT-J	99.57±0.24	92.82±0.76	77.61±0.92	620.02±0.59	39.19±0.43	99.16±0.25	87.37±1.01	28.13±0.93
Pre-edited	GPT2-XL	22.23±0.73	24.34±0.62	78.53±0.33	626.64±0.31	31.88±0.20	22.19±0.24	31.30±0.27	24.15±0.32
PMET	GPT2-XL	95.80±0.72	87.27±1.00	62.66±1.06	542.47±2.49	31.56±0.54	93.22±0.69	87.06±0.96	25.58±0.91
PMET$_{\text{BLUE}}$	GPT2-XL	95.30±0.76	85.57±1.08	67.93±0.99	603.03±1.37	37.20±0.42	89.32±0.91	80.53±1.19	26.74±0.92
Batch Model Editing Task									
Pre-edited	Llama3	7.02±0.50	9.44±0.49	89.73±0.36	630.00±0.22	24.21±0.17	35.67±0.58	34.81±0.58	31.83±0.44
PMET	Llama3	97.02±0.33	86.22±0.58	77.72±0.48	624.68±0.28	31.84±0.21	83.49±0.53	80.73±0.56	31.94±0.43
PMET$_{\text{BLUE}}$	Llama3	93.64±0.48	81.52±0.67	84.63±0.40	627.81±0.24	30.62±0.20	85.92±0.50	82.83±0.54	32.23±0.44
Pre-edited	GPT-J	15.20±0.70	17.70±0.60	83.50±0.50	622.40±0.30	29.40±0.20	26.40±0.60	25.80±0.50	27.00±0.50
PMET	GPT-J	99.57±0.13	92.48±0.44	71.41±0.52	620.31±0.31	40.79±0.24	89.24±0.46	82.69±0.59	25.51±0.49
PMET$_{\text{BLUE}}$	GPT-J	97.65±0.59	87.24±1.13	73.32±1.06	616.85±0.65	38.14±0.46	80.77±1.24	67.58±1.45	28.00±1.01
Pre-edited	GPT2-XL	21.82±0.81	24.16±0.72	78.32±0.55	626.78±0.23	31.37±0.20	22.17±0.52	21.28±0.51	24.20±0.48
PMET	GPT2-XL	81.14±0.77	70.45±0.79	66.42±0.56	622.16±0.32	37.09±0.22	60.25±0.78	56.29±0.78	23.95±0.49
PMET$_{\text{BLUE}}$	GPT2-XL	67.09±0.92	54.48±0.87	73.48±0.54	626.67±0.25	35.05±0.21	57.15±0.77	51.99±0.77	25.01±0.48

M Impact Statements

The model editing studied in this paper aims to efficiently update outdated or incorrect knowledge in large language models (LLMs). While the original intention behind such techniques is beneficial, there is also potential for misuse, such as injecting false or malicious content into LLMs. Therefore, we caution readers not to place blind trust in the content generated by LLMs.