PMET: Precise Model Editing in a Transformer

Xiaopeng Li, Shasha Li*, Shezheng Song, Jing Yang, Jun Ma, Jie Yu*

National University of Defense Technology
Changsha, Hunan 410073 China
xiaopeng.lyy@gmail.com, {shashali, ssz614, yj, majun}@nudt.edu.cn, yangjing2036@126.com

Abstract

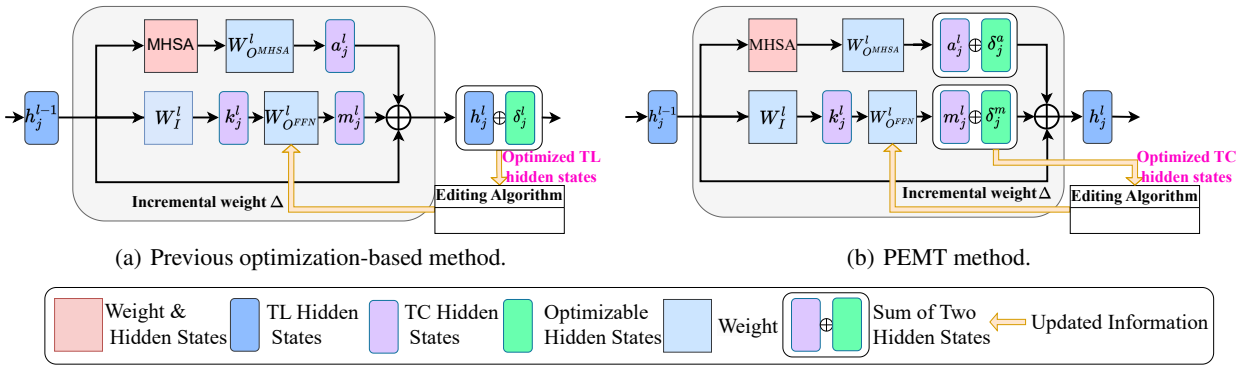
Model editing techniques modify a minor proportion of knowledge in Large Language Models (LLMs) at a relatively low cost, which have demonstrated notable success. Existing methods assume Transformer Layer (TL) hidden states are values of key-value memories of the Feed-Forward Network (FFN). They usually optimize the TL hidden states to memorize target knowledge and use it to update the weights of the FFN in LLMs. However, the information flow of TL hidden states comes from three parts: Multi-Head Self-Attention (MHSA), FFN, and residual connections. Existing methods neglect the fact that the TL hidden states contains information not specifically required for FFN. Consequently, the performance of model editing decreases. To achieve more precise model editing, we analyze hidden states of MHSA and FFN, finding that MHSA encodes certain general knowledge extraction patterns. This implies that MHSA weights do not require updating when new knowledge is introduced. Based on above findings, we introduce PMET, which simultaneously optimizes Transformer Component (TC, namely MHSA and FFN) hidden states, while only using the optimized TC hidden states of FFN to precisely update FFN weights. Our experiments demonstrate that PMET exhibits state-of-the-art performance on both the COUNTERFACT and zsRE datasets. Our ablation experiments substantiate the effectiveness of our enhancements, further reinforcing the finding that the MHSA encodes certain general knowledge extraction patterns and indicating its storage of a small amount of factual knowledge. Our code is available at https://github.com/xpq-tech/PMET.

Introduction

Large language models (LLMs), as an emerging form of knowledge base (Petroni et al. 2019; Heinzerling and Inui 2021; Cao et al. 2021), are extensively employed worldwide, primarily addressing queries through knowledge recall. Nonetheless, these models are often criticized for furnishing erroneous information (Ji et al. 2023; Zhao et al. 2023). The cost of fine-tuning or training from scratch to correct the minor proportion of erroneous knowledge is frequently deemed impractical. Fortunately, recent model editing techniques have demonstrated the ability to modify minor proportion of knowledge in LLMs with relatively low cost (Meng et al. 2022b; Mitchell et al. 2022a; Yao et al. 2023). Model editing aims to modify the internal knowledge of LLMs without resorting to vanilla training or fine-tuning. The success rates of model editing in edited-knowledge and in knowledge related to the edited-knowledge are assessed separately based on efficacy and generalization, while the preservation of irrelevant edited-knowledge is measured by specificity (also known as locality) (Mitchell et al. 2022a). For the purpose of uniformity, we collectively refer to efficacy and generalization as reliability. Additionally, two metrics are employed to evaluate the generative capacity of the post-edited model: fluency and consistency (Meng et al. 2022a). Model editing methods can be classified into two categories based on whether the original model weights are modified: *weight-preserved* and *weight-modified* methods (Yao et al. 2023). Weight-preserved methods often require additional content, whereas weight-modified methods directly edit model weights without the need for extra content, making them a more lightweight alternative.

The weight-modified approaches include learning-based (Mitchell et al. 2022a) and optimization-based methods (Meng et al. 2022b). Learning-based methods utilize gradient information to update the weights, but they suffer from poor knowledge generalization and are prone to overfitting. The optimization-based method ROME (Meng et al. 2022a) alleviates this by solving a optimization problem and updates FFN weights incrementally. The subsequent MEMIT (Meng et al. 2022b) further improves ROME, enabling mass editing in a single operation and demonstrating impressive editing performance. In detail, ROME and MEMIT view FFN as key-value memories (Geva et al. 2021), where the hidden states before and after passing through FFN weight W can be considered as keys k and values v, satisfying $Wk = v$. ROME and MEMIT extract TC (Transformer Component, namely MHSA and FFN) hidden states as keys and optimizes TL (Transformer Layer) hidden states as values, ultimately obtaining the desired weights by solving a least square problem. However, the information flow of TL hidden states comes from three parts: Multi-Head Self-Attention (MHSA), FFN, and residual connections. Both ROME and MEMIT uses optimized TL hidden states as the values (i.e., target knowledge representations) for updating FFN weights, which **overlooks the irrelevant information within the TL hidden states that is not required by FFN**,

*Corresponding Author.



(a) Previous optimization-based method. (b) PEMT method.

Figure 1: Comparison between PMET and existing methods in a Transformer layer. (a) Existing optimization-based methods employ optimized TL hidden states to perform vague updates on FFN weights. (b) PMET simultaneously optimizes the TC hidden states of both MHSA and FFN, but only uses the optimized TC hidden states of FFN to perform precise updates on FFN weights.

resulting in imprecise updating of weights based on non-accurate target knowledge representations and compromising editing performance. To address this issue, we suggest optimizing the TC hidden states directly of FFN to memorize target knowledge for precise updates on FFN weights.

But unexpectedly, during the practical process of directly optimizing the TC hidden states of FFN, we encounter occasional optimization bottlenecks where the TC hidden states can not be aligned with target knowledge. We attribute this phenomenon to the limitations imposed by the parameter space of the TC hidden states of FFN. A plausible solution to address this is the supplementary optimization of TC hidden states within MHSA. However, this introduces a novel inquiry: whether MHSA, like FFN, possesses the capability to store factual knowledge (Geva et al. 2021), necessitating updates to its weights. In pursuit of this inquiry, we endeavor to analyze hidden states of MHSA and FFN to determine the role of MHSA in LLMs' knowledge recall. We then observe that the knowledge contained within MHSA undergoes more frequent changes compared to that within FFN. Combining previous findings of MHSA (Geva et al. 2023; Wang et al. 2022; Hassid et al. 2022) with our observation, we believe that MHSA works as a knowledge extractor and stores certain general knowledge extraction patterns. This suggests the potential for supplementary optimization of TC hidden states of the MHSA to expand the function space, **without** necessitating updates to its weights.

Based on the above finding, we propose PMET, which simultaneously optimizes TC hidden states of MHSA and FFN, utilizing solely the optimized TC hidden states of FFN as the target knowledge representations for updating FFN weights, enabling precise updates. The differences between PMET and existing methods are illustrated in Figure 1. Our experiments demonstrate that PMET exhibits state-of-the-art comprehensive performance in editing GPT-J (6B) (Wang and Komatsuzaki 2021) and GPT-NeoX (20B) (Black et al. 2022) on the zsRE and COUNTERFACT datasets. Specifically, in COUNTERFACT dataset, PMET shows a 3.3% average reliability enhancement over the state-of-the-art method, while in zsRE dataset, it achieves a 0.4% average improvement. Furthermore, our series of ablation experiments demonstrate that our enhancements are effective and PMET strikes a good balance among reliability, specificity, fluency, and consistency. To sum up, our main contributions are as follows:

- We reveal that the MHSA works as a knowledge extractor, encodes certain general knowledge extraction patterns, and stores a small amount of factual knowledge.
- We propose PMET, which leverages the general knowledge extraction patterns of MHSA and simultaneously optimizes the TC hidden states of MHSA and FFN to memorize target knowledge. However, PMET only uses the optimized TC hidden states of FFN to update FFN weights due to the unnecessary updates to MHSA weights.
- Our experiments with GPT-J (6B) on the zsRE and counterfact datasets highlight PMET's superiority across multiple dimensions. Additionally, editing GPT-NeoX (20B) on the COUNTERFACT dataset underscores PMET's superior reliability and consistency over exsiting methods.

Related Work

Model Editing

Model editing is an emerging field in recent years, mainly aimed at mitigating the high cost of model training. Model editing methods can be classified into two categories: weight-modified and weight-preserved(Mitchell et al. 2022b; Zheng et al. 2023; Hernandez, Li, and Andreas 2023). Weight-preserved methods typically achieve this preservation by introducing external models (Mitchell et al. 2022b), utilizing in-context learning (Zheng et al. 2023), or altering the LLMs' representation space (Hernandez, Li, and Andreas 2023). These approaches effectively safeguard non-target knowledge while modifying the target knowledge. However, as the number of knowledge modifications increases, the required additional content also grows substantially. In contrast, weight-modified methods

(Sinitsin et al. 2020; Mitchell et al. 2022a; Meng et al. 2022b,a) directly modify the model weights for editing, thereby avoiding the aforementioned content increasing issue. Initially, weight-modified methods use approaches like multi-loss fine-tune (Sinitsin et al. 2020) and constrained fine-tune (Zhu et al. 2020). Yet these methods often suffer from overfitting. To address this issue, researchers later proposed meta-learning methods (De Cao, Aziz, and Titov 2021; Mitchell et al. 2022a) and optimization-based methods (Meng et al. 2022a). Nevertheless, as the number of edited-knowledge increases, the efficacy and generalization of these methods deteriorate significantly. This challenge is tackled by MEMIT (Meng et al. 2022b), which further improves ROME and enable edit a large amount of knowledge in one go. The optimization-based methods are built upon the conclusion that FFN are key-value memories (Geva et al. 2021) and update FFN weights by optimizing the TL hidden states to memorize target knowledge. While these methods have achieved some success in model editing, they confuse the information flow among the MHSA, FFN, and residual connections, leading to a non-accurate updates on FFN weights. In contrast to these methods, PMET simultaneously optimizes the TC hidden states of MHSA and FFN to memorize target knowledge, while only use the optimized TC hidden states of FFN, facilitating precise updates of FFN weights.

Post-Hoc Explanation of Transformers

Post-hoc explanation is a broad field, and our focus is on understanding the roles of the two components, MHSA and FFN, in Transformers (Kovaleva et al. 2019; Wang et al. 2022; Hassid et al. 2022; Geva et al. 2022; Kobayashi et al. 2023; Hao et al. 2021; Geva et al. 2023). Currently, it is widely believed that FFN serves as the main carrier for storing factual knowledge(Geva et al. 2021; Meng et al. 2022a,b), and each layer of FFN contributes to knowledge recall(Geva et al. 2022, 2023). MHSA is primarily responsible for capturing the degree of association between different tokens, focusing on interactions between content (Hao et al. 2021; Kobayashi et al. 2023), and extracting attributes of subjects (Geva et al. 2023). Other studies have shown that MHSA contains different levels of redundant information (Wang et al. 2022; Hassid et al. 2022). These findings imply that MHSA may store certain general patterns used for knowledge extraction. However, they do not completely clarify this and fail to provide insights into whether MHSA stores factual knowledge. We endeavor to analyze the hidden states of MHSA and FFN to explore these.

Methodology

Preliminaries

Language Modeling We focus on autoregressive, decoder-only LLMs denoted as $\mathcal{F}_\theta$. These models transform the input sequence $\boldsymbol{x}$ into z tokens $x_1, ..., x_z$ and then input them into L layers of Transformer decoders to obtain the probabilities of the next token x_{z+1} as follows:

$$\begin{aligned}\mathcal{F}_\theta(x_1, ..., x_z) &= \text{softmax}\left(W_{\text{E}}\gamma\left(h_z^{L-1} + a_z^L + m_z^L\right)\right) \\ &= \mathbb{P}\left(x_{z+1}|x_1, ..., x_z\right)\end{aligned} \tag{1}$$

Here, W_E and γ represent the embedding matrix and layernorm, respectively, and a_z^L and m_z^L are the TC hidden states of the MHSA and FFN of the L-th layer, respectively. Note that the MHSA and FFN in (1) are parallel (Wang and Komatsuzaki 2021; Black et al. 2022). The general forms of the MHSA and FFN at the l-th layer and the j-th token x_j^l are given by:

$$\begin{aligned}a_j^l &= W_{O^{\text{MHSA}}}^l \text{MHSA}^l\left(\gamma\left(h_1^{l-1}, h_2^{l-1}, ..., h_j^{l-1}\right)\right), \\ m_j^l &= W_{O^{\text{FFN}}}^l \sigma\left(W_I^l \gamma\left(h_j^{l-1}\right)\right)\end{aligned} \tag{2}$$

Here, $W_{O^{\text{MHSA}}}^l$ and $W_{O^{\text{FFN}}}^l$ are the output weights of the MHSA and FFN at the l-th layer, respectively, and σ represents the non-linear activation function. We have omitted bias terms for simplicity.

Model Editing Problem Previous researches on model editing have been limited to defining the problem solely based on editing the triples (i.e., subject-relation-object) themselves (Meng et al. 2022a; Mitchell et al. 2022b), overlooking the knowledge contained within the triples. Consequently, the edited models are unable to reason based on the edited knowledge (Cohen et al. 2023). In this paper, we redefine the model editing problem from a subject-centric perspective, where the edited knowledge is associated with the subject, aiming to enable the edited models to reason based on the subject.

Let $\mathcal{F}_\theta$ be an LLM that has learned N pieces of knowledge related to subject S, represented by the set:

$$K^S = \left\{\left\{\boldsymbol{x}_i^S, \boldsymbol{y}_i^S\right\}, i \in \{0, 1, 2, ..., N\}\right\} \tag{3}$$

Here, $\boldsymbol{x}_i^S$ and $\boldsymbol{y}_i^S$ represent the knowledge clue sequence and the knowledge point sequence, respectively, for the i-th piece of knowledge. For example, for subject 'Shakespeare,' a knowledge clue about the subject could be: "Shakespeare is a," and the knowledge point about the knowledge clue is "playwright." The LLM $\mathcal{F}_\theta$ satisfies: $\mathcal{F}_\theta\left(\boldsymbol{x}_i^S\right) = \boldsymbol{y}_i^S, \forall i \in \{0, 1, 2, ..., N\}$. The objective of model editing is to modify N' pieces of knowledge in the LLM related to subject S to the target knowledge:

$$K^{S_t} = \left\{\left\{\boldsymbol{x}_i^{S_t}, \boldsymbol{y}_i^{S_t}\right\}, i \in \{0, 1, 2, ..., N'\}\right\} \tag{4}$$

while keeping the $M (M \gg N)$ pieces of knowledge in the set

$$K^{\neg S_t} = \left\{\left\{\boldsymbol{x}_j^{\neg S_t}, \boldsymbol{y}_j^{\neg S_t}\right\}, j \in \{0, 1, 2, ..., M\}\right\} \tag{5}$$

that are unrelated to the N' pieces of model learned knowledge. Hence, the edited LLM $\mathcal{F}_{\theta^*}$ should satisfy:

$$\begin{aligned}&\mathcal{F}_{\theta^*}\left(\boldsymbol{x}_i^{S_t}\right) = \boldsymbol{y}_i^{S_t} \wedge \mathcal{F}_{\theta^*}\left(\boldsymbol{x}_j^{\neg S_t}\right) = \boldsymbol{y}_j^{\neg S_t}, \\ &\forall i \in \{0, 1, 2, ..., N'\}, j \in \{0, 1, 2, ..., M\}.\end{aligned} \tag{6}$$

The evaluation metrics for model editing can be found in Appendix B (Appendix will be found in (Li et al. 2023)).

Investigating the Role of MHSA and FFN in LLMs' Knowledge Recall

Inspired by Geva et al. (Geva et al. 2023), who analyzed critical subject information by mapping intermediate topic representations to vocabulary tokens, we compare the differences of hidden states h_z^{l-1} and a_z^l of the last token before and after (i.e., input and output) flowing through the l-th layer MHSA, both in the vector space and the vocabulary space. Similarly, we perform the same analysis on the hidden states h_z^{l-1} and m_z^l of the last token before and after flowing through the l-th layer FFN.

We use 1209 factual statements from (Meng et al. 2022a) as knowledge queries to explore the knowledge contained within GPT-J (6B). The last token position of each query aggregates the information of the entire query; thus, we consider the hidden state of the last token of each query as a representation of the key knowledge related to that query from the LLMs. We hypothesis ***a positive correlation between the similarity of hidden states and the consistency of knowledge*** (Liang et al. 2020). To measure the similarity, we calculate the cosine similarity of hidden states and the Jaccard similarity (Murphy 1996) of the mapping to vocabulary tokens. Specifically, we extract the hidden states of the last token before and after each layer of MHSA and FFN, compute their cosine similarity, and obtain the top-k tokens in the vocabulary. Subsequently, we calculate the Jaccard similarity between the top-k tokens of the intermediate states before and after the process. The Jaccard similarity is defined as follows:

$$J_k\left(T\left(h_1\right), T\left(h_2\right)\right) = \frac{\left|T\left(h_1\right) \cap T\left(h_2\right)\right|}{\left|T\left(h_1\right) \cup T\left(h_2\right)\right|} \tag{7}$$

where $T(h_1)$ and $T(h_2)$ represent the top-k mappings of the hidden states h_1 and h_2 on the vocabulary, respectively. We set $k = 50$ in our experiments.

The average changes in cosine and Jaccard similarities of the last tokens from 1209 factual statements across all layers and components of GPT-J are shown in Figure 2. *In the first 15 layers of GPT-J, both the MHSA and FFN exhibit relatively frequent changes in their hidden states. However, after the 15th layer, the intermediate states of the FFN undergo a slower rate of change, gradually stabilizing in a specific direction. While the hidden states of the MHSA continue to undergo frequent changes, and their directions remain uncertain throughout the knowledge extraction of GPT-J.* Considering our hypothesis regarding the relationship between hidden states and knowledge, this phenomenon suggests that the knowledge contained in FFN's hidden states tends to become consistent after a certain period, while the knowledge contained in MHSA's hidden states undergoes frequent changes throughout knowledge recall of GPT-J. We attribute this observation to the fact that **the MHSA continuously extracts various types of knowledge, while the FFN primarily extracts its own knowledge** (Geva et al. 2021; Meng et al. 2022a). Furthermore, considering previous findings regarding the extraction of attributes from the MHSA with observed redundancies (Geva et al. 2023; Wang et al. 2022; Hassid et al. 2022), we believe that the MHSA works as a knowledge extractor and stores certain general knowledge extraction patterns. Thus **we suggest that when introducing new knowledge, there is no need to update the MHSA weights.**

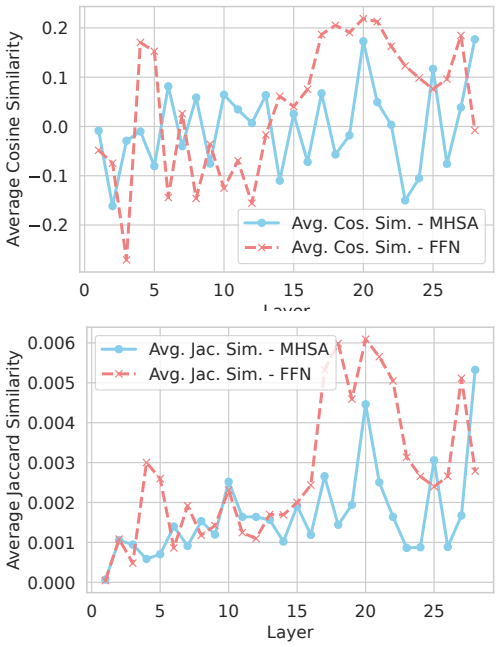


Figure 2: The changes in the average cosine similarity and average Jaccard similarity of the hidden states before and after MHSA and FFN.

PMET Method

PMET first computes the target knowledge representations in the last critical layers of FFN by simultaneously optimizing the TC hidden states of both MHSA and FFN. Secondly, PMET only updates FFN weights in the critical layers through target knowledge representations. Overall, PMET optimizes an objective function to obtain target weights (Meng et al. 2022b):

$$W_1 \triangleq \underset{W}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \left\| W k_i - v_i \right\|^2 + \sum_{i=n+1}^{n+u} \left\| W k_i - v_i \right\|^2 \right). \tag{8}$$

Here, $k_i \triangleq k_i^l$ and $v_i \triangleq v_i^l$ represent the sets of keys and values, respectively, encoding the subject-related knowledge in the l-th layer. $\sum_{i=1}^{n} \left\| W k_i - v_i \right\|^2$ indicates that we want to retain n pieces of knowledge, while $\sum_{i=n+1}^{n+u} \left\| W k_i - v_i \right\|^2$ indicates that we want to modify $u \gg 1$ pieces of knowledge. We represent the keys and values as matrices stacked horizontally: $[k_1 \mid k_2 \mid \cdots \mid k_n] \triangleq K$ and $[v_1 \mid v_2 \mid \cdots \mid v_n] \triangleq V$, and we consider the target weight W_1 as the sum of the original weight W_0 and the incremental weight Δ, i.e., $W_1 = W_0 + \Delta$. Based on the derivation from MEMIT (Meng et al. 2022b), the formal expression for the incremental weight is:

$$\Delta = R K_1^T (C_0 + K_1 K_1^T)^{-1}, \tag{9}$$

where $R \triangleq V_1 - W_0 K_1$ represents the residual between the

values V_1 (namely target knowledge representations) corresponding to the keys K_1 of the target knowledge and the model original knowledge W_0K_1. $C_0 \triangleq K_0K_0^T = \lambda\mathbb{E}_k\left[kk^T\right]$ is an estimate of the set of previously memorized keys obtained through sampling. Here, λ is a hyperparameter which balances the degree of model modification and preservation.

To explain clearly, let's consider modifying the N' knowledge instances K^S related to the subject S in LLMs to the target knowledge K^{S_t}. Assuming that the set of previously memorized keys C_0 has already been obtained through sampling, and knowledge clues $\boldsymbol{x}_i^S$ have been inputed into the original model to obtain W_0K_1, we then need the sets of keys and values for the target knowledge, denoted as K_1 and V_1, respectively. Similar to MEMIT, we calculate the target knowledge set of the last critical layer $L = \max(\mathcal{R})$. Throughout this paper, we mainly use h_i^L, m_i^L, a_i^L, and δ_i to represent the hidden states of the last tokens of the subject S in the knowledge clues $\boldsymbol{x}_i^S$.

Unlike ROME and MEMIT, which add optimizable parameters δ_i to the TL hidden states h_i^L at the L-th layer (as shown in Figure 1 (a)) and obtain the optimized TL hidden states $v_i = h_i^L + \delta_i$ through gradient descent, PMET adds optimizable parameters δ_i^a and δ_i^m to the TC hidden states a_i^L and m_i^L of the components (i.e., MHSA and FFN) at the L-th layer, respectively. Then, PMET only retains the optimized TC hidden states of FFN to update FFN weights, denoted as $v_i^m = m_i^L + \delta_i^m = \underset{v_i^m}{\operatorname{argmin}} \mathcal{L}(v_i^m)$ (as shown in Figure 1 (b)). $\mathcal{L}(v_i^m)$ is defined as follows:

$$\begin{aligned}\mathcal{L}(v_i^m) = \mu * D_{\mathrm{KL}}\left(\mathbb{P}_{\mathcal{F}_\theta^\dagger}\left[\boldsymbol{y} \mid p'\right] \| \mathbb{P}_{\mathcal{F}_\theta}\left[\boldsymbol{y} \mid p'\right]\right) + \\ \varphi * \frac{1}{P}\sum_{j=1}^{P} -\log \mathbb{P}_{\mathcal{F}_\theta^\dagger}\left[\boldsymbol{y}_i^{S_t} \mid \mathrm{pref}_j \oplus p(\boldsymbol{x_i})\right],\end{aligned} \tag{10}$$

where φ and μ are hyperparameters used to balance reliability and specificity. $\mathcal{F}_\theta^\dagger \triangleq \mathcal{F}_\theta\left(a_i^L += \delta_i^a, m_i^L += \delta_i^m\right)$ represents the optimizable parameters δ_i^a and δ_i^m are added to the TC hidden states of MHSA and FFN at the L-th layer of the model $\mathcal{F}_\theta$, respectively. $\mathrm{pref}_j \oplus p(\boldsymbol{x_i})$ and p' are, as in (Meng et al. 2022a,b), prefixes used to enhance the generalization of the target knowledge and the prompt template used for calculating the KL divergence: '$\{S\}$ is a'. After calculating the values of all the target knowledge that need to be changed, they can be stacked into a matrix V_1.

To edit multiple layers of the model, we need to spread the residual R to all critical layers. MEMIT spreads updates evenly over the range of critical layers $\mathcal{R}$ as $R^l = \frac{V_1 - W_0K_1}{L-l+1}$. In contrast, PMET adopts a square root spread to convey more precise information to critical layers:

$$R^l = \frac{V_1 - W_0K_1}{\sqrt{L-l+1}}. \tag{11}$$

Now that we have C_0 and R^l, the next step is to obtain keys K_1. Keys are related to specific weights to be edited and represent the hidden states before entering those specific weights. Similar to (Meng et al. 2022a,b), the keys k_i^l at the l-th layer are defined as follows:

$$k_i^l = \frac{1}{P}\sum_{j=1}^{P} \mathrm{prev}(W, \mathrm{pref}_j \oplus S), \tag{12}$$

where $\mathrm{prev}(W, \mathrm{pref}_j \oplus S)$ represents the hidden state of the input $\mathrm{pref}_j \oplus S$ before flowing through the weight W. If one wants to edit $W_{O^{\mathrm{FFN}}}^l$ in (2), then $\mathrm{prev}(W_{O^{\mathrm{FFN}}}^l, x) = \sigma\left(W_I^l\gamma\left(h_j^{l-1}(x)\right)\right)$.

With this, PMET follows the same algorithm steps as MEMIT to update FFN weights.

Experiments

Baselines and Datasets

Our experiments are conducted on GPT-J (6B) and GPT-NeoX (20B). Our baseline methods include the improved Constrained Fine-Tuning (FT+W) (Zhu et al. 2020; Meng et al. 2022b), the learning-based method MEND (Mitchell et al. 2022a), and the optimization-based methods ROME (Meng et al. 2022a) and MEMIT (Meng et al. 2022b). For the datasets, we performed counterfactual update experiments on two datasets: Zero-Shot Relation Extraction (zsRE) (Levy et al. 2017) and COUNTERFACT (Meng et al. 2022a). More details about datasets can be found in Appendix C.

Editing Experiments

The score is the harmonic mean of efficacy, generalization, and specificity, representing the balance between reliability (i.e., efficacy and generalization) and specificity of the editing method(Meng et al. 2022a). Note that in our experiments, we updates counterfactual information, so we evaluated specificity based on factual information, while when testing for efficacy and generalization, we used counterfactual information as the standard. As a result, the unedited LLMs performed poorly in terms of efficacy and generalization but exhibited good performance in terms of specificity. Implementation details are presented in Appendix D.

Editing Knowledge in Counterfact As mentioned in (Meng et al. 2022a), we also conducted 17 counterfactual experiments by sampling 17 integers $n_i = \exp\left(\ln(10000) * \frac{i}{16}\right)$ from a log-scale curve for editing. The performance of PMET and other existing methods on GPT-J (6B) in these 17 edits is shown in Figure 3. With the exception of being slightly inferior to MEMIT in terms of specificity, PMET outperforms all baselines in all other metrics.

Table 1 shows the results of all methods on 10K counterfactual edits. The results show that PMET outperforms existing methods in score, efficacy, fluency, and consistency, but is slightly inferior to MEMIT in specificity, and like MEMIT, it is far behind the meta-learning based method MEND. In the trade-off between editing reliability and specificity, both PMET and MEMIT tend to prioritize reliability, while MEND leans towards specificity. While sacrificing some specificity for improved reliability is acceptable until better methods are available, we hope to find a compromise in the future.

Then, we applied PMET to conduct 10K edits on GPT-NeoX (20B) on the COUNTERFACT dataset, and the results are shown in the lower part of Table 1. Similarly, PMET outperforms MEMIT in terms of reliability and consistency, but lags behind in specificity. These might be because PMET

Editor	Score	Efficacy	Generalization	Specificity	Fluency	Consistency
GPT-J (6B)	22.4	15.2 (0.7)	17.7 (0.6)	83.5 (0.5)	622.4 (0.3)	29.4 (0.2)
FT-W	67.6	99.4 (0.1)	77.0 (0.7)	46.9 (0.6)	293.9 (2.4)	15.9 (0.3)
MEND	23.1	15.7 (0.7)	18.5 (0.7)	**83.0** (0.5)	618.4 (0.3)	31.1 (0.2)
ROME	50.3	50.2 (1.0)	50.4 (0.8)	50.2 (0.6)	589.6 (0.5)	3.3 (0.0)
MEMIT	85.8	98.9 (0.2)	88.6 (0.5)	73.7 (0.5)	619.9 (0.3)	40.1 (0.2)
PMET	**86.2**	**99.5** (0.1)	**92.8** (0.4)	71.4 (0.5)	**620.0** (0.3)	**40.6** (0.2)
GPT-NeoX (20B)	23.7	16.8 (1.9)	18.3 (1.7)	81.6 (1.3)	620.4 (0.6)	29.3 (0.5)
MEMIT	82.0	97.2 (0.8)	82.2 (1.6)	**70.8** (1.4)	**606.4** (1.0)	36.9 (0.6)
PMET	**84.3**	**98.4** (0.2)	**89.4** (0.5)	70.3 (0.5)	598.1 (0.6)	**38.9** (0.2)

Table 1: 10,000 counterfact edits on GPT-J (6B) and GPT-NeoX (20B). Within parentheses is the 95% confidence interval.

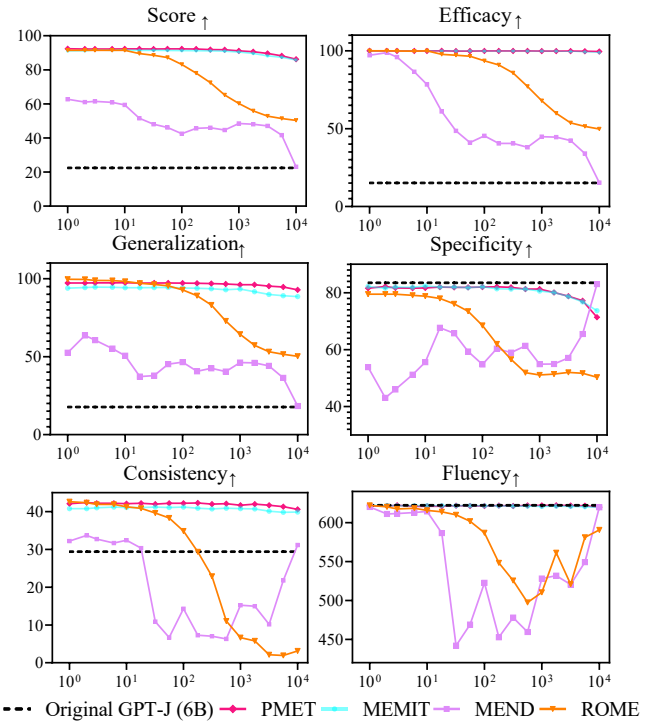


Figure 3: The editing performance of PMET and baselines varies with the number of edits (X-axis).

employs square root propagation (11), resulting in greater changes to the model and hence more damage to specificity . We further investigate this in the following ablation experiments. Nevertheless, these results demonstrate that PMET achieves the most significant updates to target knowledge compared to existing methods.

Editing 10K Knowledge in ZsRE The results of editing 10K knowledge on the zsRE dataset are presented in Table 2. The results demonstrate that PMET outperforms existing methods in all three metrics: efficacy, generalization, and specificity. It is worth noting that the original GPT-J (6B) model has a specificity score of only 27.0, and therefore, the specificity of the edited models is also lower than this value.

Editor	Efficacy	Generalization	Specificity
GPT-J	26.4 (±0.6)	25.8 (±0.5)	27.0 (±0.5)
FT-W	69.6 (±0.6)	64.8 (±0.6)	24.1 (±0.5)
MEND	19.4 (±0.5)	18.6 (±0.5)	22.4 (±0.5)
ROME	21.0 (±0.7)	19.6 (±0.7)	0.9 (±0.1)
MEMIT	96.7 (±0.3)	89.7 (±0.5)	26.6 (±0.5)
PMET	**96.9**(±0.3)	**90.6** (±0.2)	**26.7** (±0.2)

Table 2: 10,000 zsRE Edits on GPT-J (6B).

Ablation Study

We conduct three sets of ablation experiments and demonstrate that: 1) PMET simultaneously optimizing the TC hidden states of MHSA and FFN can result in enhanced reliability; 2) The updating of MHSA weights contributes marginally to the improved generalization of editing while also inflicting greater damage to specificity; and 3) square root spreads in PMET enhances reliability but leads to larger changes in the model, ultimately affecting specificity. All the ablation experiments were conducted on COUNTERFACT using GPT-J (6B), with parameters consistent with the previous experiments in COUNTERFACT.

We first conduct experiments where PMET only optimizes TC hidden states of FFN (i.e., δ_i^a is removed) for 1K, 5623, and 10K counterfactual edits. The experimental results are shown in Table 3 under "w/o δ_i^a". This shows that simultaneously optimizing TC hidden states of FFN and MHSA can result in better reliability compared to only optimizing TC hidden states of FFN.

Next, we update the weights of both MHSA and FFN using the optimized TC hidden states. This means we updated $W^l_{O^{\text{MHSA}}}$ and $W^l_{O^{\text{FFN}}}$ simultaneously in (2), and additionally computed the keys for (12) as $\text{prev}(W^l_{O^{\text{MHSA}}}, x) = \text{MHSA}^l\left(h_1^{l-1}(x), h_2^{l-1}(x), ..., h_j^{l-1}(x)\right)$. The results are shown in Table 3 under "w/ $W^l_{O^{\text{MHSA}}}$". The results indicate that additionally updating MHSA weights can slightly improve editing generalization, but at the same time, it worsens the specificity. This might be due to the fact that MHSA weights store certain general knowledge extraction patterns along with a small amount of factual knowledge. While up-

Edits	Editor	Score	Efficacy	Generalization	Specificity	Fluency	Consistency
	GPT-J	22.4	15.2	17.7	83.5	622.4	29.4
1K	PMET	91.1	99.8	96.1	80.1	622.2	41.7
	w/o δ_i^a	90.8 (↓0.3)	99.6 (↓0.2)	96.6 (↑0.5)	79.1 (↓1.0)	622.4 (↑0.2)	42.2 (↑0.5)
	w/ $W^l_{O^{\text{MHSA}}}$	90.8 (↓0.3)	99.8 (→)	96.8 (↑0.7)	78.9 (↓1.2)	622.0 (↓0.2)	42.1 (↑0.4)
	Even spread	88.9 (↓2.2)	99.6 (↓0.2)	86.7 (↓9.9)	82.2 (↑ 3.1)	622.3 (↓0.1)	39.1 (↓3.1)
5623	PMET	88.0	99.7	94.5	74.2	621.7	41.3
	w/o δ_i^a	87.0 (↓1.0)	99.3 (↓0.4)	95.0 (↑0.5)	72.0 (↓2.2)	622.3 (↑0.6)	41.6 (↑0.3)
	w/ $W^l_{O^{\text{MHSA}}}$	86.7 (↓1.3)	99.6 (↓0.1)	96.0 (↑1.5)	70.8 (↓3.4)	621.3 (↓0.4)	41.6 (↑0.3)
	Even spread	85.8 (↓2.2)	98.2 (↓1.5)	82.2 (↓12.3)	79.4 (↑5.2)	621.8 (↑0.1)	38.1 (↓3.2)
10K	PMET	86.2	99.5	92.8	71.4	620.0	40.6
	w/o δ_i^a	85.0 (↓1.2)	98.9 (↓0.6)	89.0 (↓3.8)	71.6 (↑0.2)	621.2 (↑1.2)	40.0 (↓0.6)
	w/ $W^l_{O^{\text{MHSA}}}$	84.9 (↓1.3)	99.5 (→)	93.5 (↑0.7)	68.6 (↓2.8)	619.0 (↓1.0)	40.5 (↓0.1)
	Even spread	83.3 (↓2.9)	96.7 (↓2.8)	78.4(↓ 14.4)	77.3 (↑5.9)	621.8 (↑1.8)	37.4 (↓3.2)

Table 3: The results of the ablation experiments. w/o δ_i^a represents only optimizing the TC hidden state δ_i^m of FFN. w/ $W^l_{O^{\text{MHSA}}}$ represents simultaneously updating the weights of both MHSA and FFN. Even spread represents evenly spreading the residual R to the critical layers $\mathcal{R}$.

dating MHSA weights strengthens the extraction patterns of the knowledge similar to edited-knowledge, it may also impair the patterns of extracting other unrelated knowledge, making it more likely to harm specificity.

Finally, we evenly spread the residual R to the critical layers $\mathcal{R}$, and the results are shown in Table 3 under "Even spread". The results indicate that even spreading leads to better model retention (i.e., specificity and fluency), but efficacy, generalization, and consistency are much worse compared to square root spreading. This suggests that using even spreading in PMET may cause significant loss of update information, reducing the update reliability while preserving more model's original knowledge. While using square root spreading mitigates the loss of update information, improves reliability, but leads to larger changes in the model, causing more side effects to the specificity and fluency.

We further analyze the relationship between the editing performance and the norms of the incremental weight Δ in Appendix A. In summary, PMET strikes a good balance between reliability and specificity which becomes more pronounced as the number of edited knowledge increases.

Conclusion

We reveal that MHSA works as a knowledge extractor and encodes certain general knowledge extraction patterns. Based on this finding, we propose PMET, which simultaneously optimizes the TC hidden states of both MHSA and FFN while only uses the optimized TC hidden states of FFN to perform precise updates of FFN weights. Our experiments on zsRE and COUNTERFACT demonstrate the state-of-the-art performance of PMET. Furthermore, our ablation experiments show that our enhancements are effective, PMET strikes a good balance between different metrics, and MHSA stores a small amount of factual knowledge. Our findings contribute additional insights for a better comprehension of the roles played by MHSA and FFN, and our approach takes a step forward in terms of model editing techniques.

Limitations

Unlike knowledge graphs that explicitly store information in symbolic form (Liang et al. 2023b,a), LLMs implicitly store substantial knowledge in parameterized form. The partial opacity of LLMs' internal mechanisms poses challenges for direct weight modification in model editing. While approaches like PMET and MEMIT have shown promising results in some evaluations, their effectiveness does not necessarily indicate true internalization of edited knowledge by LLMs. Consequently, models edited by PMET and MEMIT cannot reason using the edited knowledge (e.g., after editing the knowledge "The Prime Minister of the UK is {Theresa}," to "{Rishi Sunak}" the edited model might generate statements like "The Prime Minister of England is {British}, not {Indian}"). Additionally, although this paper defines the problem of knowledge editing centered around subjects, benchmark and dataset construction have not strictly adhered to this definition but instead have been adapted to existing evaluation methods. In the future, we aim to devise more sophisticated editing methods and evaluation metrics (e.g., such as MQuAKE by Zhong et al. (2023) and RIPPLEEDITS by Cohen et al. (2023)) to advance model editing.

Ethical Statement

The original intention of our research into model editing techniques is to rectify errors and outdated knowledge in LLMs, enabling them to better serve our needs. However, these techniques also have the potential for misuse, allowing LLMs to generate false, toxic, and harmful content. Therefore, we emphasize the importance of not placing excessive trust in the generated content until LLMs are well-regulated.

Acknowledgments

This work was partly supported by the Hunan Provincial Natural Science Foundation Projects (No.2022JJ30668 and No. 2022JJ30046).

References

Black, S.; Biderman, S.; Hallahan, E.; Anthony, Q.; Gao, L.; Golding, L.; He, H.; Leahy, C.; McDonell, K.; Phang, J.; Pieler, M.; Prashanth, U. S.; Purohit, S.; Reynolds, L.; Tow, J.; Wang, B.; and Weinbach, S. 2022. GPT-NeoX-20B: An Open-Source Autoregressive Language Model. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, 95–136. virtual+Dublin: Association for Computational Linguistics.

Cao, B.; Lin, H.; Han, X.; Sun, L.; Yan, L.; Liao, M.; Xue, T.; and Xu, J. 2021. Knowledgeable or Educated Guess? Revisiting Language Models as Knowledge Bases. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 1860–1874. Online: Association for Computational Linguistics.

Cohen, R.; Biran, E.; Yoran, O.; Globerson, A.; and Geva, M. 2023. Evaluating the Ripple Effects of Knowledge Editing in Language Models. *arXiv preprint arXiv:2307.12976*.

De Cao, N.; Aziz, W.; and Titov, I. 2021. Editing Factual Knowledge in Language Models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 6491–6506. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics.

Geva, M.; Bastings, J.; Filippova, K.; and Globerson, A. 2023. Dissecting Recall of Factual Associations in Auto-Regressive Language Models. arXiv:2304.14767.

Geva, M.; Caciularu, A.; Wang, K.; and Goldberg, Y. 2022. Transformer Feed-Forward Layers Build Predictions by Promoting Concepts in the Vocabulary Space. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 30–45. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics.

Geva, M.; Schuster, R.; Berant, J.; and Levy, O. 2021. Transformer Feed-Forward Layers Are Key-Value Memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 5484–5495. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics.

Hao, Y.; Dong, L.; Wei, F.; and Xu, K. 2021. Self-Attention Attribution: Interpreting Information Interactions Inside Transformer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14): 12963–12971.

Hassid, M.; Peng, H.; Rotem, D.; Kasai, J.; Montero, I.; Smith, N. A.; and Schwartz, R. 2022. How much does attention actually attend? Questioning the Importance of Attention in Pretrained Transformers. arXiv:2211.03495.

Heinzerling, B.; and Inui, K. 2021. Language Models as Knowledge Bases: On Entity Representations, Storage Capacity, and Paraphrased Queries. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 1772–1791. Online: Association for Computational Linguistics.

Hernandez, E.; Li, B. Z.; and Andreas, J. 2023. Inspecting and Editing Knowledge Representations in Language Models. arXiv:2304.00740.

Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y. J.; Madotto, A.; and Fung, P. 2023. Survey of Hallucination in Natural Language Generation. *ACM Computing Surveys*, 55(12): 1–38.

Kobayashi, G.; Kuribayashi, T.; Yokoi, S.; and Inui, K. 2023. Feed-Forward Blocks Control Contextualization in Masked Language Models. arXiv:2302.00456.

Kovaleva, O.; Romanov, A.; Rogers, A.; and Rumshisky, A. 2019. Revealing the Dark Secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 4365–4374. Hong Kong, China: Association for Computational Linguistics.

Levy, O.; Seo, M.; Choi, E.; and Zettlemoyer, L. 2017. Zero-Shot Relation Extraction via Reading Comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, 333–342. Vancouver, Canada: Association for Computational Linguistics.

Li, X.; Li, S.; Song, S.; Yang, J.; Ma, J.; and Yu, J. 2023. PMET: Precise Model Editing in a Transformer. arXiv:2308.08742.

Liang, K.; Liu, Y.; Zhou, S.; Tu, W.; Wen, Y.; Yang, X.; Dong, X.; and Liu, X. 2023a. Knowledge Graph Contrastive Learning Based on Relation-Symmetrical Structure. *IEEE Transactions on Knowledge and Data Engineering*, 1–12.

Liang, K.; Meng, L.; Liu, M.; Liu, Y.; Tu, W.; Wang, S.; Zhou, S.; and Liu, X. 2023b. Learn from relational correlations and periodic events for temporal knowledge graph reasoning. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1559–1568.

Liang, R.; Li, T.; Li, L.; Wang, J.; and Zhang, Q. 2020. Knowledge Consistency between Neural Networks and Beyond. arXiv:1908.01581.

Meng, K.; Bau, D.; Andonian, A.; and Belinkov, Y. 2022a. Locating and Editing Factual Associations in GPT. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 17359–17372. Curran Associates, Inc.

Meng, K.; Sharma, A. S.; Andonian, A.; Belinkov, Y.; and Bau, D. 2022b. Mass-Editing Memory in a Transformer. arXiv:2210.07229.

Mitchell, E.; Lin, C.; Bosselut, A.; Finn, C.; and Manning, C. D. 2022a. Fast Model Editing at Scale. In *International Conference on Learning Representations*.

Mitchell, E.; Lin, C.; Bosselut, A.; Manning, C. D.; and Finn, C. 2022b. Memory-Based Model Editing at Scale. arXiv:2206.06520.

Murphy, A. H. 1996. The Finley affair: A signal event in the history of forecast verification. *Weather and forecasting*, 11(1): 3–20.

Petroni, F.; Rocktäschel, T.; Riedel, S.; Lewis, P.; Bakhtin, A.; Wu, Y.; and Miller, A. 2019. Language Models as Knowledge Bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2463–2473. Hong Kong, China: Association for Computational Linguistics.

Sinitsin, A.; Plokhotnyuk, V.; Pyrkin, D.; Popov, S.; and Babenko, A. 2020. Editable Neural Networks. arXiv:2004.00345.

Wang, B.; and Komatsuzaki, A. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax. Accessed: 2023-12-21.

Wang, K.; Variengien, A.; Conmy, A.; Shlegeris, B.; and Steinhardt, J. 2022. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. arXiv:2211.00593.

Yao, Y.; Wang, P.; Tian, B.; Cheng, S.; Li, Z.; Deng, S.; Chen, H.; and Zhang, N. 2023. Editing Large Language Models: Problems, Methods, and Opportunities. arXiv:2305.13172.

Zhao, W. X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; Du, Y.; Yang, C.; Chen, Y.; Chen, Z.; Jiang, J.; Ren, R.; Li, Y.; Tang, X.; Liu, Z.; Liu, P.; Nie, J.-Y.; and Wen, J.-R. 2023. A Survey of Large Language Models. arXiv:2303.18223.

Zheng, C.; Li, L.; Dong, Q.; Fan, Y.; Wu, Z.; Xu, J.; and Chang, B. 2023. Can We Edit Factual Knowledge by In-Context Learning? arXiv:2305.12740.

Zhu, C.; Rawat, A. S.; Zaheer, M.; Bhojanapalli, S.; Li, D.; Yu, F.; and Kumar, S. 2020. Modifying Memories in Transformer Models. arXiv:2012.00363.