MicroEdit: Neuron-level Knowledge Disentanglement and Localization in Lifelong Model Editing

Anonymous ACL submission

Abstract

Large language models (LLMs) require continual knowledge updates to keep pace with the evolving world. While various model editing methods have been proposed, most face critical challenges in lifelong learning contexts due to two fundamental limitations: (1) *Edit Overshooting* - parameter updates intended for a specific fact spill over to unrelated regions, causing interference with previously retained knowledge; and (2) *Knowledge Entanglement* - polysemantic neurons' overlapping encoding of multiple concepts makes it difficult to isolate and edit a single fact. In this paper, we propose MicroEdit, a neuron-level editing method that performs minimal and controlled interventions within LLMs. By leveraging a sparse autoencoder (SAE), MicroEdit disentangles knowledge representations and activates only a minimal set of necessary neurons for precise parameter updates. This targeted design enables fine-grained control over the editing scope, effectively mitigating interference and preserving unrelated knowledge. Extensive experiments show that MicroEdit outperforms prior methods and robustly handles lifelong knowledge editing across QA and Hallucination settings on LLaMA[1] and Mistral[2]. Our code can be found at: `https://anonymous.4open.science/r/MicroEdit-200B`.

1 Introduction

Large language models (LLMs) accumulate substantial world knowledge during pretraining (Roberts et al., 2020). However, as real-world knowledge continually evolves, these models inevitably retain outdated or incorrect information, necessitating timely correction and updating. To address this limitation, lifelong knowledge editing (Hartvigsen et al., 2024) emerges as a strategic solution, aiming to enable continuous and dynamic knowledge updates over extended time horizons.

[1]meta-llama/Meta-Llama-3-8B

[2]mistralai/Mistral-7B-v0.1

Previous knowledge editing methods (Meng et al., 2022a; Mitchell et al., 2022a) are designed primarily for single or limited edits, lacking the capacity to support long-term, multi-round knowledge updates, which often leads to catastrophic forgetting or model collapse. Thus, these methods struggle to accommodate the evolving knowledge requirements in real world scenarios.

Through systematic evaluation experiments with LLMs, we observe that current editing methods suffer from two major issues under lifelong editing scenarios: (1) *Edit Overshooting*, where updates inadvertently modify parameters unrelated to the target knowledge, leading to degraded performance on unrelated tasks; (2) *Knowledge Entanglement*, a phenomenon induced by the polysemanticity of neurons, where semantically overlapping representations may lead to unintended modifications of non-target knowledge, even when edits are applied to the correct parameters. Section 2 presents a detailed quantified analysis to reveal the origins of these limitations and their effects on editing reliability and accuracy.

To address these issues, we propose a novel knowledge editing framework that enables controlled knowledge updates via neuron-level minimal editing within large language models. For *Edit Overshooting*, the Sparse Autoencoder (SAE) activates only a minimal subset of neurons for knowledge instance, naturally limiting the scope of reconstruction and parameter updates to target-specific regions and reducing interference with unrelated parameters. For *Knowledge Entanglement*, SAE adopts an overcomplete hidden layer, where sparsely activated neurons are encouraged to learn monosemantic representations. This reduces semantic overlap in the parameter space, effectively mitigating Knowledge Entanglement and enabling more precise, controlled edits. The contributions of this paper are summarized as follows:

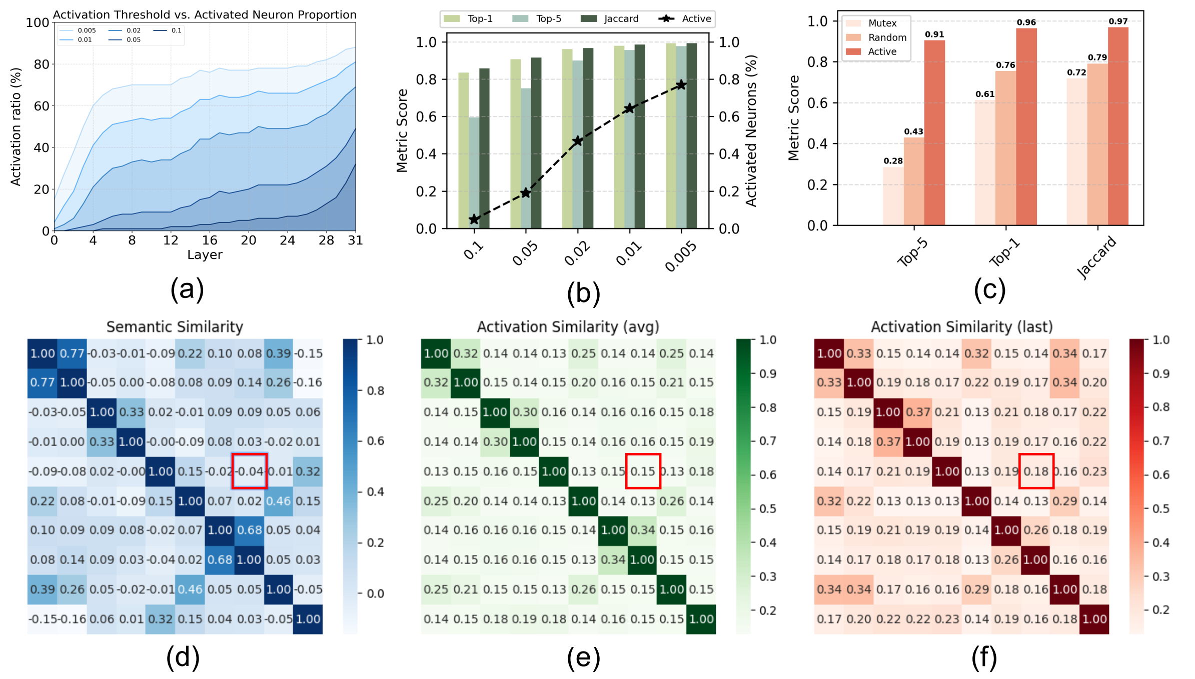

Figure 1: (a) The proportion of activated neurons across different layers of the LLM under varying activation thresholds; (b) The results when only neurons above a specific threshold are activated, we evaluate outputs using three metrics: Top-1 (exact match of the highest probable token), Top-5 (top-5 token predictions are strictly matched), and Jaccard similarity (set overlap of top-5 predictions); (c) Results of three activation strategies: activating neurons above the threshold (Active), randomly selected neurons (Random), and only activating neurons below the threshold (Mutex); (d) Visualization of semantic similarity among 10 randomly selected knowledge instances; (e-f) neuron activation similarities of these 10 knowledge instances computed by averaging across all tokens (e) or using only the last token (f).

- We identify two key limitation in current knowledge editing methods by quantified analysis: *Edit Overshooting* and *Knowledge Entanglement*, which hinder precise and reliable knowledge modification.
- We develop MicroEdit, an editing method that performs sparse neuron-level updates via a pretrained SAE to enhance reliability and precision in lifelong knowledge editing.
- Extensive experiments are conducted on lifelong knowledge editing across LLaMA and Mistral models. The results demonstrate the effectiveness of MicroEdit.

2 Empirical Insights into Editing Limits

We conduct an empirical analysis and identify two key factors that limit the effectiveness of current methods in lifelong knowledge editing scenarios.

Inefficient Parameter Updates: *Edit Overshooting*. Prior work has shown that not all neurons contribute equally to the computation of specific knowledge during inference (Geva et al., 2021; Dai et al., 2022). We investigate this with three empirical studies (Figure 1(a–c)). We measure neuron activation rates across layers under varying thresholds in Figure 1(a). Early layers exhibit sparse activation, which increases in deeper layers, with output layers being most active. Neurons with activation above 0.1 are rare in lower and middle layers. In Figure 1(b), we compares original outputs with those obtained by masking low-activation neurons. We find that retaining only the top 60% activated neurons preserves performance, indicating that inference relies on a subset of critical neurons. Further experimental results in Figure 1(c) confirm that only highly activated neurons are essential for representing the target knowledge. These results suggest that updating only the parameters associated with highly activated neurons is sufficient for editing. However, existing methods often ignore this sparsity and perform overly broad updates, affecting irrelevant parameters, a phenomenon we refer to as ***Edit Overshooting***. The key challenge lies in identifying and updating only the parameters most relevant to the target knowledge.

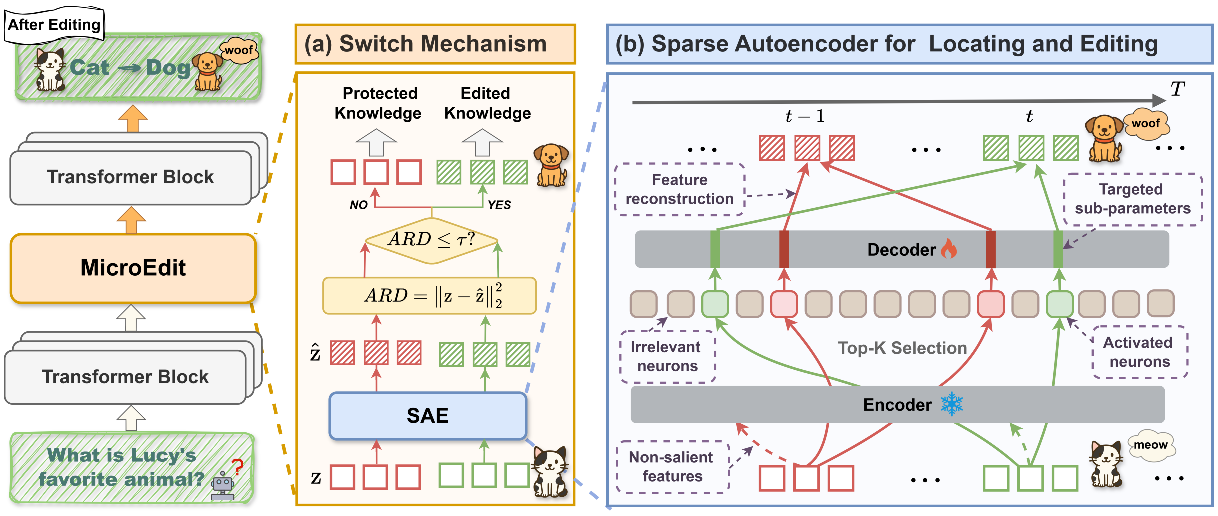

Figure 2: Overview of MicroEdit. (a) illustrates the Switch Mechanism, which distinguishes between in-scope (Green Square) and out-of-scope (Red Square) knowledge for inference. (b) shows the knowledge modification process using the SAE. The model's original intermediate features are mapped to a sparse set of neurons via the encoder. These sparse activations are then decoded through the corresponding sub-parameters of the decoder to reconstruct a targeted representation, enabling the model to modify the original output (e.g., from Cat to Dog).

Neuron-Level Semantic Coupling: *Knowledge Entanglement*. In addition to the phenomenon discussed above, deep neural networks neurons are polysemantic, meaning that individual neurons often encode multiple, semantically unrelated pieces of information (Bricken et al., 2023; Elhage et al., 2022). To empirically study this issue, we conducted an activation similarity analysis across different knowledge statements (Figure 1 (d-f)): In Figure 1(d), we shows the semantic similarity between 10 randomly selected knowledge statements. Figure 1(e) and (f) report neuron activation similarities computed by averaging across all tokens or using only the last token. Interestingly, even for semantically unrelated knowledge pairs, such as knowledge 5 *"Bananas contain high amounts of potassium"* and knowledge 8 *"The Sun revolves around the Earth in geocentric models"*, we observe up to 18% overlap in activated neurons. This indicates that modifying the parameters associated with one piece of knowledge may unintentionally impact others encoded by the same neurons. We refer to this phenomenon as ***Knowledge Entanglement***, which highlights the risk of knowledge interference when updating shared neural representations. Such representational overlap poses a fundamental challenge to editing precision and motivates the need for more disentangled update mechanisms. The central challenge is to isolate more disentangled features that uniquely represent the target knowledge without disrupting others.

3 Preliminary

3.1 Lifelong Knowledge Editing

Lifelong Knowledge Editing refers to the process of incrementally transforming an initial model f_{θ_0} into an edited model f_{θ_T} through multiple rounds of knowledge updating, enabling continual incorporation of new information while preserving existing knowledge and ensuring behavioral stability of the model. We denote the model update process as $f_{\theta_0} \rightarrow f_{\theta_T}$, where θ_0 represents the initial parameters of the language model, and θ_T denotes the parameters after T rounds of knowledge editing. Each editing objective is defined as a pair (x_e, y_e) in edit dataset $\mathcal{D}_{\text{edit}}$, where x_e is the knowledge prompt and y_e is the desired output that the initial model fails to produce correctly. Accordingly, the editing objective can be formally defined as:

$$f_{\theta'}(x_i) = \begin{cases} y_i & \text{if } x_i \in \mathcal{D}_{\text{edit}} \\ f_{\theta_0}(x_i) & \text{if } x_i \notin \mathcal{D}_{\text{edit}} \end{cases} \tag{1}$$

To evaluate the effectiveness of model editing methods, we adopt the following three metrics:

Reliability measures the average accuracy of the model on the edited samples after the T-th edit, indicating whether the model successfully incorporates the intended knowledge change.

$$\mathbb{E}_{(x_e,y_e)\sim\{(x_e^t,y_e^t)\}_{t=0}^{T}} \mathbb{1}\left\{argmax_y f_{\theta_T}(y \mid x_e) = y_e\right\} \tag{2}$$

Generalization measures the average accuracy on an extended dataset $\mathcal{R}(x, y)$ related to the edited knowledge, reflecting the model's ability to generalize the edit to semantically similar contexts.

$$\mathbb{E}_{(x'_e, y_e)\sim\{\mathcal{R}(x_e^t, y_e^t)\}_{t=0}^{T}} \mathbb{1}\left\{argmax_y f_{\theta_T}(y \mid x'_e) = y_e\right\} \quad (3)$$

Locality measures the relative accuracy change on unrelated data $\mathcal{O}(x, y)$ before and after the T-th edit, assessing whether the edit introduces undesired side effects on the model's original behavior.

$$\mathbb{E}_{(x', y')\sim\{\mathcal{O}(x_e^t, y_e^t)\}_{t=0}^{T}} \mathbb{1}\left\{argmax_y f_{\theta_T}(y \mid x') = y'\right\} \quad (4)$$

3.2 Sparse Autoencoder

Sparse Autoencoders (SAEs) are neural architectures that learn efficient and structured representations by encouraging sparsity in the hidden layer. This is typically achieved by applying a sparse activation function $\phi(\cdot)$ to limit the number or magnitude of active neurons for each input. $W_{enc} \in \mathbb{R}^{n\times d}$ and b_{enc} is the weight and bias of SAE encoder. $W_{dec} \in \mathbb{R}^{n\times d}$ and b_{dec} is the weight and bias of SAE decoder. Given an input vector $\mathrm{z} \in \mathbb{R}^d$, the encoder maps it to a hidden representation:

$$\hat{h} = f_{enc}(\mathrm{z}) = \phi(W_{enc}\mathrm{z} + b_{enc}) \quad (5)$$

The decoder reconstructs the featrue from the sparse code:

$$\hat{\mathrm{z}} = f_{dec}(\hat{h}) = W_{dec}\hat{h} + b_{dec} \quad (6)$$

The imposed sparsity promotes more interpretable and disentangled representations, making sparse autoencoders (SAEs) valuable for tasks such as model interpretability, concept discovery, and controllable model editing.

4 Methodology

To address the challenges of *Edit Overshooting* and *tKnowledge Entanglement* in lifelong knowledge editing, we propose MicroEdit, a neuron-level editing framework consisting of two components: a Sparse Autoencoder (SAE) for precise localization and a Switch module for scope control.The overall framework is illustrated in Figure 2.

4.1 SAE for Knowledge Editing

Recent work (Geva et al., 2021; Meng et al., 2022a) reveals that Transformer FFNs function as key-value memories and are effective targets for knowledge editing. Similarly, SAE can be seen as a key-value structure, with W_{enc} as the key and W_{dec} as the value. As shown in Figure 2(b), we freeze the parameters of the LLM and the SAE encoder,updating only the SAE decoder during editing. We attach the SAE to the l-th layer of the LLM and extract its residual input $\mathbf{z}$ as the SAE input for each prompt. Top-k activation is adopted in the SAE to enforce sparsity by retaining only the k most activated neurons. The inference process of the encoder is as follows:

$$\hat{h} = Topk(W_{enc}\mathrm{z} + b_{enc}) \quad (7)$$

Unlike conventional autoencoding objectives that aim to reconstruct the original input $\mathbf{z}$, our goal is to steer the model's behavior toward generating the desired output y. In this setting, the reconstruction $\hat{\mathbf{z}}$ serves as an intervention that alters the model's internal representations to produce the target output. To ensure that model behavior changes only in a controlled and localized manner, our method confines representation modifications to a minimal subspace. Specifically, due to the Top-k sparse activation, the modified residual $\hat{\mathbf{z}}$ is reconstructed solely from a limited set of decoder vectors:

$$\hat{\mathbf{z}} = \sum_{i\in\mathcal{I}_k} \hat{h}_i \cdot W_{dec}[i] + b_{dec} \quad (8)$$

$\mathcal{I}_k \subset \{1, ..., n\}$ denotes the position indices of the Top-k activated neurons within the SAE. As a result, the update to the model's internal representation is constrained within the subspace spanned by the selected decoder weights:

$$\triangle\mathbf{z} = \hat{\mathbf{z}} - \mathbf{z} \in \mathrm{Span}(W_{dec}[\mathcal{I}_k]) \quad (9)$$

This subspace constraint naturally limits the propagation of edits, reduces interference with unrelated knowledge, and contributes to the stability of successive edits. Accordingly, the loss function for the editing process at round t is defined as follows:

$$\mathcal{L}_{edit} = -\log P(y_e^t | \hat{\mathbf{z}}^{(t)}(x_e^t, \theta_{dec}^{(t-1)}), \theta_{LM}) \quad (10)$$

where $\theta_{dec}^{(t-1)}$ denotes the trainable parameters of the decoder after $t-1$ rounds of editing, θ_{LM} denotes the frozen parameters of the language model.

4.2 Switch Mechanism

The original SAE is trained to reconstruct $\hat{\mathbf{z}}$ to approximate the original hidden states $\mathbf{z}$. But reconstruction is not fully accurate, forcing all knowledge to pass through the SAE during inference

Table 1: Main results for ZsRE. **Bold** is the best result, underline denotes the second-best. T : Num Edits.

Method	QA															
	$T = 1$				$T = 10$				$T = 100$				$T = 1000$			
	Rel.	Gen.	Loc.	Avg.	Rel.	Gen.	Loc.	Avg.	Rel.	Gen.	Loc.	Avg.	Rel.	Gen.	Loc.	Avg.
LLaMA-3-8B																
FT	**1.00**	**0.99**	0.02	0.67	0.69	0.64	0.01	0.45	0.64	0.57	0.02	0.41	0.62	0.56	0.02	0.40
FT-EWC	**1.00**	**0.99**	0.02	0.67	0.69	0.63	0.01	0.44	0.65	0.58	0.02	0.42	0.62	0.56	0.02	0.40
ROME	0.99	0.97	0.96	0.97	0.42	0.42	0.19	0.34	0.07	0.07	0.01	0.05	0.02	0.02	0.01	0.02
MEMIT	0.88	0.69	**1.00**	0.86	0.77	0.69	0.98	0.81	0.76	0.70	0.89	0.78	0.00	0.00	0.00	0.00
MEND	0.98	0.97	0.99	**0.98**	0.00	0.01	0.11	0.04	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
GRACE	0.90	0.28	**1.00**	0.73	0.90	0.28	**1.00**	0.73	**0.90**	0.28	**1.00**	0.78	**0.90**	0.39	**1.00**	0.76
WISE	0.95	0.95	0.84	0.91	0.86	**0.80**	0.99	0.88	0.71	0.64	**1.00**	0.78	0.61	0.57	**1.00**	0.73
MicroEdit	**1.00**	0.92	0.95	0.96	**0.93**	0.78	0.95	**0.89**	0.89	**0.71**	0.97	**0.86**	0.87	**0.65**	**1.00**	**0.84**
Mistral-7B																
FT	**1.00**	**0.99**	0.01	0.67	0.77	0.72	0.02	0.50	0.74	0.67	0.05	0.49	0.70	0.65	0.08	0.48
FT-EWC	**1.00**	**0.99**	0.01	0.67	0.77	0.72	0.01	0.50	0.74	0.67	0.05	0.49	0.70	0.65	0.08	0.48
ROME	0.87	0.83	0.99	0.90	0.44	0.42	0.41	0.42	0.07	0.07	0.01	0.05	0.01	0.01	0.00	0.01
MEMIT	0.88	0.85	**1.00**	0.91	0.23	0.22	0.23	0.23	0.03	0.03	0.01	0.02	0.04	0.04	0.02	0.03
MEND	0.99	0.96	**1.00**	**0.98**	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
GRACE	0.78	0.36	**1.00**	0.71	0.77	0.36	**1.00**	0.71	0.79	0.36	**1.00**	0.72	**0.78**	0.36	**1.00**	0.71
WISE	0.99	0.97	0.99	**0.98**	0.85	**0.81**	0.99	0.88	0.78	**0.73**	**1.00**	0.84	0.66	0.63	**1.00**	0.76
MicroEdit	0.95	0.68	**1.00**	0.88	**0.91**	0.78	**1.00**	**0.89**	**0.86**	**0.73**	**1.00**	**0.86**	**0.80**	**0.68**	0.98	**0.82**

may distort non-target information and compromise model stability. Motivated by the SAE's capability in anomaly detection, we introduce a switch mechanism to distinguish between in-scope knowledge and out-of-scope knowledge as illustrated in Figure 2(a).

Specifically, the switch mechanism computes SAE output $\hat{\mathbf{z}}$ for all inputs, but substitutes $\hat{\mathbf{z}}$ for the original hidden state $\mathbf{z}$ only when the input is in-scope. Otherwise, $\mathbf{z}$ is retained and propagated unchanged. This design ensures that unrelated knowledge remains unaffected during the editing process. To support this mechanism, we introduce the Average Reconstruction Distance (ARD) as follows:

$$\mathcal{L}_{rec} = \text{ARD}(x_{edit}) = \frac{1}{\alpha \cdot S} \sum_{s=1}^{S} \|\hat{\mathbf{z}}_s - \mathbf{z}_s\|_2^2 \tag{11}$$

where S is the length of the edited tokens, D is the feature dimension of each token, α is the scaling factor. We apply a threshold τ on ARD to separate editable targets from high-ARD local knowledge. Thus, the output of MicroEdit during inference is:

$$MicroEdit(\mathbf{z}) = \begin{cases} \hat{\mathbf{z}} & \text{if } ARD \leq \tau \\ \mathbf{z} & \text{if } ARD > \tau \end{cases} \tag{12}$$

At the same time, we also use ARD as an auxiliary loss to enlarge the gap between edited and unrelated knowledge. However, during editing training, we observe that jointly optimizing for editing accuracy and ARD deviation is challenging. To address this, we adopt a two-stage training strategy: (1) First, optimize the reconstruction loss $\mathcal{L}_{rec}$ to maximize deviation from the global ARD reference; (2) Then, optimize the combined objective $\mathcal{L}_{rec} + \mathcal{L}_{edit}$, encouraging both ARD deviation and accurate editing.

To ensure localized updates during knowledge editing, we apply an explicit gradient mask to the W_{dec}. The mask matrix is defined as:

$$M^{(t)}[i] = \begin{cases} 1, & \text{if } i \in \mathcal{I}_k^{(t)} \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

Let $G^{(t)} = \bigtriangledown_{W_{dec}} \mathcal{L}^t$ be the full gradient, and $M^{(t)} \in 0, 1^{n \times d}$ is the broadcasted row mask. The decoder update becomes:

$$W_{dec}^{(t)} = W_{dec}^{(t-1)} - \eta \cdot (M^{(t)} \odot G^{(t)}) \tag{14}$$

This ensures that only the activated neurons influence the decoder during each editing round. The pseudo-code of our method is provided in Algorithms 1 and 2 in appendix C.2.

5 Experiment

5.1 Experimental Setup

Datasets and Evaluation Metrics. Following (Hartvigsen et al., 2024), we select ZsRE (Levy

Table 2: Main results for SelfCheckGPT. **Bold** is the best result, underline denotes the second-best. - denotes out-of-range values without comparative significance. T : Num Edits.

Hallucination																
	LLaMA-3-8B								Mistral-7B							
	$T=1$		$T=10$		$T=100$		$T=600$		$T=1$		$T=10$		$T=100$		$T=600$	
Method	Rel.	Loc.	Rel.	Loc.	Rel.	Loc.	Rel.	Loc.	Rel.	Loc.	Rel.	Loc.	Rel.	Loc.	Rel.	Loc.
FT	**1.01**	0.06	4.93	0.08	3.40	0.16	4.01	0.22	**1.02**	0.05	3.01	0.10	2.20	0.18	2.56	0.24
FT-EWC	**1.01**	0.06	4.90	0.08	3.41	0.16	4.01	0.22	**1.02**	0.05	3.06	0.10	2.21	0.18	2.56	0.24
ROME	1.93	0.97	8.76	0.65	-	0.02	-	0.02	1.89	0.99	2.67	0.91	-	0.02	-	0.02
MEMIT	4.37	**1.00**	2.69	0.99	-	0.01	-	0.00	1.60	**1.00**	11.32	0.93	-	0.06	-	0.06
MEND	4.17	0.98	-	0.01	-	0.00	-	0.00	3.39	0.99	-	0.01	-	0.00	-	0.00
GRACE	1.03	**1.00**	9.18	**1.00**	9.92	1.00	9.94	**1.00**	1.15	**1.00**	7.23	**1.00**	11.09	**1.00**	9.24	**1.00**
WISE	1.47	0.88	1.22	0.92	1.41	**1.00**	3.70	1.00	1.29	0.98	1.47	0.95	2.48	0.94	6.22	0.94
MicroEdit	1.10	0.99	**1.04**	**1.00**	**1.10**	**1.00**	**2.26**	**1.00**	1.22	**1.00**	**1.04**	**1.00**	**1.24**	**1.00**	**2.20**	**1.00**

et al., 2017) as a closed-book question-answering dataset to evaluate knowledge editing capabilities, and SelfCheckGPT (Potsawee et al., 2023) as a benchmark for hallucination correction. For zsRE, we assess MicroEdit using three metrics: Reliability, Generalization, and Locality. For the hallucination dataset, the Generalization is not applicable, we instead use Perplexity (PPL) to evaluate Reliability, while still computing Locality to assess the model performance after editing.

Models and Baselines. We evaluate our proposed method, **MicroEdit**, on two widely used large language models, **LLaMA-3-8B** and **Mistral-7B**. Meanwhile, we employ the corresponding SAEs: sae-llama-3-8b-32x from EleutherAI[3] and mistral-7b-res-wg from SAE Lens (Joseph et al., 2024). Comparisons are made against a broad range of baselines, including direct parameter modification methods such as standard fine-tuning (FT), continual learning-based fine-tuning with Elastic Weight Consolidation (FT-EWC), Locate-then-edit approaches (ROME, MEMIT), and meta-learning based method (MEND). We further compare against parameter-preserving methods, including hidden state modification (GRACE), and side MLP memory augmentation (WISE). More details about baselines can be found in Appendix B.

5.2 Main Results

To demonstrate the effectiveness of MicroEdit, we conduct experiments on two foundational language models under both QA and Hallucination settings. The results are shown in Table 1 and Table 2. Across nearly all configurations, MicroEdit consistently achieves strong performance, especially in long-term sequential editing scenarios. In contrast, fine-tuning-based methods tend to overfit rapidly. The locate-then-edit approach suffers from significant *Edit Overshooting* problem during sequential edits, often leading to model collapse. MEND fails entirely after multiple rounds of editing. GRACE performs well in terms of reliability and locality, but demonstrates limited generalization capability. While WISE accounts for knowledge storage constraints, it is still affected by *Knowledge Entanglement*, leading to performance drop as the number of edits T increases.

Table 3: Scaling to 3K and 5K edits. Bold is the best result. ZsRE. LLaMA-3-8B.

Method	$T=3000$				$T=5000$			
	Rel.	Gen.	Loc.	Avg.	Rel.	Gen.	Loc.	Avg.
GRACE	**0.89**	0.28	1.00	0.72	**0.90**	0.27	1.00	0.66
WISE	0.47	0.44	1.00	0.64	0.42	0.40	1.0	0.61
MicroEdit	0.83	**0.59**	**1.00**	**0.81**	0.80	**0.54**	**1.00**	**0.78**

In addition, we scale the experiment to 3K and 5K edits. As shown in Table 3, MicroEdit outperforms the strongest baselines GRACE and WISE, with a 12% improvement over the second-best method at 5K edits. GRACE maintains high reliability but suffers from poor generalization, while WISE shows performance degradation as the number of edits increases. Only MicroEdit achieves stable and balanced performance under extremely long sequential editing, demonstrating its effectiveness in liflong knowledge editing scenarios.

5.3 Ablation Study

We conduct a series of ablation studies to evaluate the impact of key components as shown in Table 4.

[3]EleutherAI/sae-llama-3-8b-32x

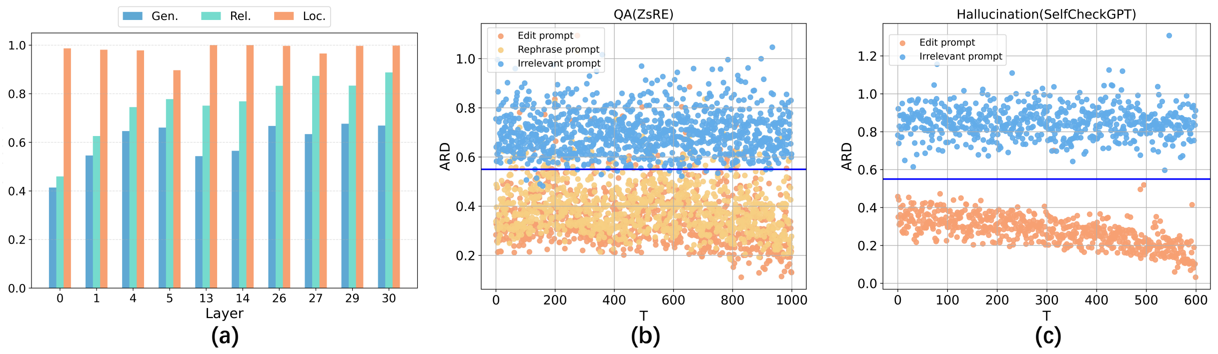

Figure 3: (a) Performance of MicroEdit across different layers of LLaMA-3-8B; (b–c) The ARD of ZsRE and SelfCheckGPT data during the inference stage. LLaMA-3-8B.

Table 4: Results of ablation study using 1K edits. Bold font indicates the best result. ZsRE. LLaMA-3-8B.

	Rel.	Gen.	Loc.	Avg.
MicroEdit	**0.87**	**0.65**	**1.00**	**0.84**
- TopK activation	0.27	0.27	1.00	0.51
- Shunting Mechanism	0.85	0.64	0.67	0.72
- Distance Regularization	0.50	0.43	1.00	0.64

(1) Ablating the Top-k sparse activation leads to a substantial performance drop, rendering the editing module nearly ineffective. This degradation arises from two key factors. First, the lack of sparsity amplifies *Edit Overshooting*, resulting in pronounced forgetting of unrelated knowledge. Second, without sparsity constraints, ARD becomes difficult to regulate, undermining the effectiveness of the switch mechanism and further weakening the editing capability.

(2) The absence of the switch mechanism leads to a notable degradation in Locality, as existing SAE cannot reliably reconstruct the full range of original representations. However, model retains around 67% performance, showing a degree of robustness. This indicates that the switch mechanism can effectively isolate non-target knowledge during editing, enhancing both precision and stability.

(3) Without the Distance Regularization editing step, the model struggles to keep ARD below the global average while optimizing toward the editing target, leading to uncontrolled ARD increases, which hinder the separation of in-scope and out-of-scope knowledge, ultimately degrading editing performance.

5.4 Detailed Analysis and Discussion

Effect of Editing Layer on SAE Behavior. To examine the effect of layer depth on MicroEdit, we conduct editing experiments at early, middle, and late layers. The results in Figure 3(a) show that editing at early layers leads to suboptimal performance, as these layers mainly capture low-level linguistic patterns, which makes it difficult to precisely localize target knowledge. Performance improves in the middle layers and reaches its peak in the later layers. We attribute this to the hierarchical structure of language models, where semantic representations become increasingly refined in deeper layers. Late layers tend to encode factual knowledge more explicitly, enabling more targeted and stable edits. Editing at these layers aligns better with the model's representational structure, allowing MicroEdit to more precisely locate and modify neurons associated with the target knowledge, leading to more effective and robust outcomes.

Effect of k in Top-k Activation. We further analyze how the choice of k affects editing performance. We use the SAE trained on the LLaMA-3-8B model by EleutherAI, which is roughly following the recipe detailed in (Gao et al., 2024). Accordingly, this SAE is trained with $k = 192$ activated neurons. To explore the influence of this hyperparameter, we conduct experiments by varying the value of k. The results in Figure 4 show that the original setting of $k = 192$ yields the best performance. When k is too small, the activated

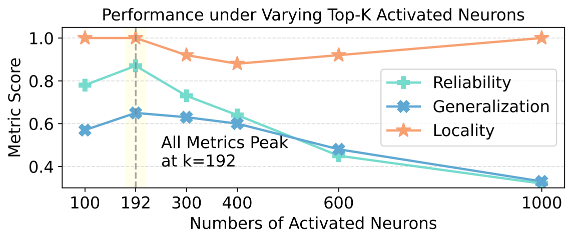

Figure 4: Model performance under different Top-k settings.

neurons fail to adequately cover all relevant knowledge features, resulting in incomplete representation. Conversely, when k is too large, neuron sharing across different knowledge instances becomes more frequent, leading to interference between injected knowledge and a decline in performance. In light of these findings, we retain the original SAE setting and use $k = 192$ in our experiments.

Effect of ARD Threshold on Edit-Locality Separation. We analyze the impact of the ARD threshold on editing performance. During inference, ARD is used to distinguish between in-scope and out-of-scope knowledge, as shown in Figure 3(b–c). In the hallucination experiment Figure 3(c), a threshold of 0.55 effectively separates in-scope from out-of-scope knowledge. In the QA setting Figure 3(b), although the the separation is less distinct with some overlap, the model still achieves robust editing performance, demonstrating strong resilience and generalization. We attribute the difference to dataset characteristics: the QA data has simple structure and higher semantic overlap, leading to limited activation variation and less clear ARD separation. In contrast, the Hallucination data exhibits greater semantic diversity, promoting more stable and separable representations. More details are in Appendix A.1.

6 Related Works

6.1 Knowledge Editing

Knowledge editing aims to modify models precisely while maintaining their performance and stability. ROME and MEMIT (Meng et al., 2022a,b) insert factual knowledge into specific layers identified through causal tracing. While WilKE (Hu et al., 2024) selects specific layers for each fact. To isolate edit effects, AlphaEdit (Fang et al., 2025) applies a null-space constraint, confining changes to the target knowledge. Another strategy exemplified by MEND is to train a small editing network to generate parameter updates (Mitchell et al., 2022a; Tan et al., 2024). In contrast, other methods preserve original model parameters by using external memory or small trainable modules. SERAC and Grace (Mitchell et al., 2022b; Hartvigsen et al., 2024) keep new information in an explicit memory or local codebook, avoiding any change to base weights. T-Patcher and MELO (Huang et al., 2023; Yu et al., 2024) attach lightweight modules (extra neurons or dynamic LoRA blocks, MELO (Yu et al., 2024), inspired by DyLoRA (Valipour et al., 2023), implements edits via LoRA without altering the model's core parameters. REMEDI (Hernandez et al., 2024) maps natural language statements to fact encodings within a model's internal representations. WISE (Wang et al., 2024) use a side memory to modify and store knowledge.

6.2 Sparse Autoencoder

Sparse Autoencoder is an autoencoder that learns interpretable representations by enforcing sparse activations in the hidden layer. k-Sparse Autoencoders are proposed by (Makhzani and Frey, 2013). By retaining only the top k most highly activated neurons, they effectively improves classification performance and is well-suited for large-scale problems. Recently, Sparse Autoencoders have been widely used for interpreting LLMs (Shu et al., 2025). By reconstructing internal activations of the model into more monosemantic representations, SAEs help clarify previously uninterpretable neuron behaviors caused by polysemanticity, improving model transparency (Cunningham et al., 2023; Bricken et al., 2023; Templeton et al., 2024). In addition, numerous open-source SAE toolkits pretrained on various LLMs have been released (He et al., 2024; Lieberum et al., 2024; Joseph et al., 2024). Research on SAEs continues to advance. OpenAI has further explored existing SAE frameworks and discovered clean scaling laws with respect to autoencoder size and sparsitys (Gao et al., 2024). (O'Neill and Bui, 2024) leverages SAEs to identify interpretable computational circuits within LLMs. (Paulo et al., 2024) proposes a pipeline for generating and evaluating SAE feature interpretations.

7 Conclusion

In this work, we identify two core challenges in lifelong model editing: *Edit Overshooting* caused by excessive updates to irrelevant parameters and *Knowledge Entanglement* which stems from the inherent polysemanticity of neurons in LLMs. To alleviate these issues, we propose MicroEdit, which adopts Top-k sparse activation to restrict parameter updates and leverages the structural properties of an overcomplete sparse autoencoder to encourage more monosemantic neuron representations, thereby reducing representation overlap and unintended interference. Extensive experiments demonstrate that MicroEdit achieves competitive performance across different scenarios and LLM models.

8 Limitation

MicroEdit relies on a pre-trained Sparse Autoencoder (SAE) as a pluggable component to extract and modify internal representations. While pre-trained SAEs are available for most mainstream models, applying MicroEdit to less common or customized models often requires training an SAE from scratch, introducing additional time and computational costs.

References

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nicholas L Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Alex Tamkin, Karina Nguyen, Brayden McLean, and 5 others. 2023. Towards monosemanticity: Decomposing language models with dictionary learning. Technical report, Anthropic.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. 2023. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*.

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

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022. Toy models of superposition. Technical report, Anthropic, Harvard.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. 2025. Alphaedit: Null-space constrained model editing for language models. In *The Thirteenth International Conference on Learning Representations*.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. 2024. Scaling and evaluating sparse autoencoders. *arXiv preprint arXiv:2406.04093*.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495.

Tom Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. 2024. Aging with grace: Lifelong model editing with discrete key-value adaptors. *Advances in Neural Information Processing Systems*, 36.

Zhengfu He, Wentao Shu, Xuyang Ge, Lingjie Chen, Junxuan Wang, Yunhua Zhou, Frances Liu, Qipeng Guo, Xuanjing Huang, Zuxuan Wu, and 1 others. 2024. Llama scope: Extracting millions of features from llama-3.1-8b with sparse autoencoders. *arXiv preprint arXiv:2410.20526*.

Evan Hernandez, Belinda Z. Li, and Jacob Andreas. 2024. Inspecting and editing knowledge representations in language models. In *First Conference on Language Modeling*.

Chenhui Hu, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. 2024. WilKE: Wise-layer knowledge editor for lifelong knowledge editing. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3476–3503, Bangkok, Thailand. Association for Computational Linguistics.

Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. 2023. Transformer-patcher: One mistake worth one neuron. In *The Eleventh International Conference on Learning Representations*.

Bloom Joseph, Tigges Curt, and Chanin David. 2024. Saelens. `https://github.com/jbloomAus/SAELens`.

Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. 2024. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. *arXiv preprint arXiv:2408.05147*.

Alireza Makhzani and Brendan Frey. 2013. K-sparse autoencoders. *arXiv preprint arXiv:1312.5663*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022a. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. 2022b. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. 2022a. Fast model editing at scale. In *International Conference on Learning Representations*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. 2022b. Memory-based model editing at scale. In *International Conference on Machine Learning*, pages 15817–15831. PMLR.

Charles O'Neill and Thang Bui. 2024. Sparse autoencoders enable scalable and reliable circuit identification in language models. *arXiv preprint arXiv:2405.12522*.

Gonçalo Paulo, Alex Mallen, Caden Juang, and Nora Belrose. 2024. Automatically interpreting millions of features in large language models. *arXiv preprint arXiv:2410.13928*.

Manakul Potsawee, Liusie Adian, and Gales Mark. 2023. SelfcheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

Dong Shu, Xuansheng Wu, Haiyan Zhao, Daking Rai, Ziyu Yao, Ninghao Liu, and Mengnan Du. 2025. A survey on sparse autoencoders: Interpreting the internal mechanisms of large language models. *arXiv preprint arXiv:2503.05613*.

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

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, Alex Tamkin, Esin Durmus, Tristan Hume, Francesco Mosconi, C. Daniel Freeman, and 7 others. 2024. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. Technical report, Anthropic.

Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. 2023. Dylora: Parameter-efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, page 3274–3287.

Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. 2024. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. *Advances in Neural Information Processing Systems*, 37:53764–53797.

Lang Yu, Qin Chen, Jie Zhou, and Liang He. 2024. Melo: Enhancing model editing with neuron-indexed dynamic lora. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19449–19457.

A Datasets Details

A.1 ZsRE

ZsRE is a classic zero-shot question-answering dataset widely used in the model editing literature. Each record in the dataset contains a prompt-answer pair (x_{edit}, y_{edit}) that requires editing, a paraphrased prompt x_{gen} for evaluating generalization, and a locality pair (x_{loc}, y_{loc}) for measuring locality preservation. We adopt the same train/test split as (Mitchell et al., 2022a), consisting of 163,196 training examples and 19,086 test examples. Notably, MEND is the only method that requires fitting a hyper network on the training set; other methods discard the training set and directly perform edits and evaluations on the test set. For our experiments, We further extend the experiments to 3k and 5k records.

A.2 SelfCheckGPT

Following GRACE, we adopt the SelfCheckGPT dataset to evaluate the ability of knowledge editing methods to alleviate hallucinations. This dataset consists of factually incorrect sentences generated by GPT-3, each paired with a corrected version derived from Wikipedia. Compared to the ZsRE dataset, SelfCheckGPT features longer, more complex sentences that more closely resemble real-world knowledge editing scenarios. Each instance is formatted as either (x_{edit}, y_{edit}) and (x_{loc}, y_{loc}), aligning with the standard editing and locality evaluation settings. Unlike GRACE, which conducts experiments on GPT2-XL(1.5B), we perform experiments on larger scale LLMs such as LLaMA(8B) and Mistral(7B). Due to memory constraints similar to those in (Wang et al., 2024), we follow the same dataset split strategy, using a train/test split of 306/600 samples. Except for MEND, which requires training on the training set, all other methods perform edits exclusively on the test set. Table7 provides illustrative examples from the two datasets, and Table6 reports the corresponding dataset statistics.

Table 5: Datasets examples

Dataset	Type	Text
ZsRE	x_{edit}, y_{edit}	What university did Watts Humphrey attend? **University of Michigan**
	x_{gen}, y_{edit}	What university did Watts Humphrey take part in? **University of Michigan**
	x_{edit}, y_{edit}	Who played desmond doss father in hacksaw ridge? **Hugo Weaving**
SelfCheckGPT	x_{edit}, y_{edit}	This is a Wikipedia passage about heinz christian pander. Heinz Christian Pander (1794 - 1865) was a German anatomist and embryologist who was born in Riga, Latvia. He studied medicine at the University of Dorpat and later at the University of Berlin.**In 1820, he took part in a scientific expedition to Bokhara as a naturalist.**
	x_{loc}, y_{loc}	Tired and restlessly, drifting in and out of sleep. Hearing crashing and banging, thinking the roof will cave in. Not alert enough to quite know what.**it was, I yelled loudly for whoever was making those noises at such an hour to stop. They heard and listened, I'm guessing**

A.3 Analysis of the Dataset

We conduct a visualization analysis of the Edit Prompts and Locality Prompts from the ZSRE and SelfCheckGPT datasets as Figure 5 and 6. The results show that representations in SelfCheckGPT are more clearly separated with lower semantic redundancy, making it more suitable for stable and low-interference long-term editing. In contrast, ZSRE exhibits certain semantic overlaps, increasing the risk of knowledge interference and forgetting, thus posing greater challenges for long-term editing. Nevertheless, our method achieves strong performance on both datasets, demonstrating its robustness and effectiveness.

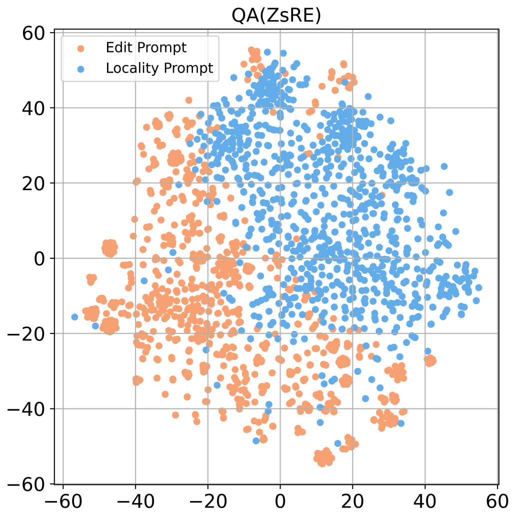

Figure 5: t-SNE visualization of semantic representations in the ZSRE dataset

B Implementation of Baselines

FT Following (Hartvigsen et al., 2024) We use Fine-tuning updates model parameters by minimizing the task-specific loss on the new data, allowing the model to adapt to the edited knowledge.

FT-EWC Elastic Weight Consolidation (EWC) addresses the problem of catastrophic forgetting by introducing a regularization term derived from the Fisher information matrix, which quantifies the importance of each parameter based on previous training (Hartvigsen et al., 2024).

ROME (Meng et al., 2022a) utilizes causal tracing techniques to localize factual knowledge within MLP layers and performs parameter editing by solving a constrained least-squares problem. Under the assumption that MLP layers serve as the primary knowledge storage mechanism (Geva et al.,

Table 6: Datasets statistics

SETTING	EDITING DATA	T	Pre-edit(LLaMA/Mistral)
QA	ZsRE	1000	0.27/0.36 ACC
Hallucination	SelfCheckGPT	600	15.83/11.06 PPL

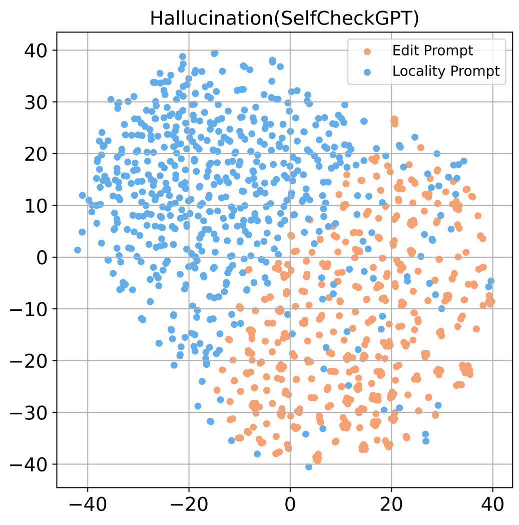

Figure 6: t-SNE visualization of semantic representations in the SelfCheckGPT dataset

2021), ROME incrementally modifies the model by injecting residual terms derived from a Lagrangian formulation.

MEMIT (Meng et al., 2022b) is also built on the assumption that the FFN functions as a knowledge key-value store, and it modifies parameters of specific layers via least squares fitting. Unlike ROME, which updates only a single layer, MEMIT performs joint updates across multiple layers, enabling batch editing of a large number of facts. In sequential editing scenarios, MEMIT requires real-time correction whenever the model produces an error, often involving multiple rounds of operations on the original model.

MEND MEND (Mitchell et al., 2022a) updates the model by introducing a hyper-network that adjusts the gradients generated during standard fine-tuning. Specifically, it approximates the original gradients with a rank-1 decomposition and uses this representation to derive new update directions, which are then applied to specific layers of the target model.

GRACE GRACE (Hartvigsen et al., 2024) introduces a dynamic discrete key-value memory that evolves via key addition, expansion, and splitting. At inference, it retrieves the nearest key and selectively replaces the corresponding hidden activation to enable precise knowledge updates.

WISE WISE (Wang et al., 2024) introduces a dual-memory architecture for lifelong knowledge editing in large language models, where immutable main memory preserves pretrained knowledge and editable side memory stores newly edited facts. A routing mechanism dynamically selects between memories during inference, while a knowledge sharding strategy allocates edits into disentangled subspaces to prevent interference. This design effectively balances reliability, generalization, and locality, overcoming the trade-offs faced by prior editing methods.

C Experimental Details

C.1 Hyperparameters

In our experiments, we freeze both the parameters of the base language model and the SAE encoder, and train only the decoder of the SAE. For all models, batch size is 1, the learning rate is set to 5e-5 with the Adam (Kingma, 2014) optimizer and the α is 600. For LLaMA-3-8B, we use the sae-llama-3-8b-32x model from EleutherAI[4] , with an intermediate expansion dimension of 32. We set $k = 192$, a sparsity threshold of 0.55 for $T \geq 100$, and 0.5 for $T < 100$. The SAE is inserted at layer 30. For Mistral-7B, we use the 7b-res-wg model from SAE Lens (Joseph et al., 2024), with an intermediate expansion of 16. We set $k = 400$, a threshold of 0.15 for $T \geq 100$, and 0.1 for $T < 100$. The SAE is inserted at layer 24.

C.2 Pseudo Code of WISE

The pseudo-code of the MicroEdit editing stage is in Algorithm 1, and the inference stage is Algorithm 2.

C.3 Case Study

We present representative successful editing cases from ZsRE and SelfCheckGPT datasets in Table 7.

[4]EleutherAI/sae-llama-3-8b-32x

Table 7: Case study

State		Text
Prompt		What language is Garowe Principles written in?
Label		**Dutch**
Before Edit	☹	Garowe
After Edit	☺	**Dutch**
Prompt		This is a Wikipedia passage about carole gist. Carole Gist (born April 28, 1969) is an American beauty pageant titleholder from Detroit, Michigan who was crowned Miss USA 1990. She was the first African-American woman to win the Miss USA title. Gist represented the United States at the Miss Universe 1990 pageant held in Los Angeles, California, where she placed first runner-up to Mona Grudt of Norway. Gist was the first African-American woman to place in the Miss Universe pageant.
Label		**She was also the first contestant from Michigan to win Miss USA, and broke the five-year streak of winners from Texas.**
Before Edit	☹	Gist was born in Detroit, Michigan. She is the daughter of a Detroit police officer and a homemaker.
After Edit	☺	**She was also the first contestant from Michigan to win Miss USA, and broke the five-year streak of winners from Texas.**

Algorithm 1 Editing Stages

```
Input: Edit data $\mathcal{D}_{\text{edit}}$; Edit round $T$; LLM $f(\cdot)$; the designated layer $l$ in LLM; SAE encoder $f_{enc}$ and decoder $f_{dec}$; The trainable parameter $W_{dec}^{(t-1)}$ at round $t$; threshold $\tau$
Output: Trained decoder parameters $\theta_{\text{dec}}$
for each record $(x_e^t, y_e^t)$ from $\mathcal{D}_{\text{edit}}$, $t \in [T]$ do,
    $\mathbf{z} \leftarrow f^{\leq l}(x_e^t)$
    $\hat{h} \leftarrow f_{\text{enc}}(\mathbf{z}) = Topk(W_{\text{enc}}\mathbf{z} + \mathbf{b}_{\text{enc}})$    ▷ Eq. 7
    $\hat{\mathbf{z}} \leftarrow f_{\text{dec}}(\hat{h}) = W_{\text{dec}}^{(t-1)}\hat{h} + \mathbf{b}_{\text{dec}}$    ▷ Eq. 8
    Stage 1: Distance Regularization
    $\mathcal{L} \leftarrow \mathcal{L}_{rec}(\mathbf{z}, \hat{\mathbf{z}})$    ▷ Eq. 11
    $W_{\text{dec}}^{(t')} \leftarrow W_{dec}^{(t-1)} - \eta \cdot (M^{(t)} \odot \nabla_{W_{dec}}\mathcal{L})$    ▷ Eq. 14
    Stage 2: Reconstruction Editing
    $\mathcal{L} \leftarrow \mathcal{L}_{rec}(\mathbf{z}, \hat{\mathbf{z}}) + \mathcal{L}_{edit}(f^{>l}(\hat{\mathbf{z}}), y_e^t)$    ▷ Eq. 10
    $W_{\text{dec}}^{(t)} \leftarrow W_{dec}^{(t')} - \eta \cdot (M^{(t)} \odot \nabla_{W_{dec}}\mathcal{L})$    ▷ Eq. 14
end for
return the final SAE $W_{dec}^{T}$
```

Algorithm 2 Inference Stage

```
Input: Test data $\mathcal{D}_{\text{test}}$; LLM $f(\cdot)$;the designated layer $l$ in LLM; SAE encoder $f_{enc}$ and decoder $f_{dec}$; threshold $\tau$
Output: LLM output
for each query $x \in \mathcal{D}_{\text{test}}$ do
    $\mathbf{z} \leftarrow f^{\leq l}(x)$
    $\hat{h} \leftarrow f_{\text{enc}}(\mathbf{z}) = Topk(W_{\text{enc}}\mathbf{z} + \mathbf{b}_{\text{enc}})$    ▷ Eq. 7
    $\hat{\mathbf{z}} \leftarrow f_{\text{dec}}(\hat{h}) = W_{\text{dec}}\hat{h} + \mathbf{b}_{\text{dec}}$    ▷ Eq. 8
    if ARD$(\mathbf{z}, \hat{\mathbf{z}}) \leq \tau$ then    ▷ Eq. 13
        Edited Output$\leftarrow f^{>l}(\hat{\mathbf{z}})$
    else
        Original Output $\leftarrow f^{>l}(\mathbf{z})$
    end if
end for
```