Efficiently Quantifying and Mitigating Ripple Effects in Model Editing

Anonymous ACL submission

Abstract

Large Language Models have revolutionized numerous tasks with their remarkable efficacy. However, editing these models, crucial for rectifying outdated or erroneous information, often leads to a complex issue known as the ripple effect in the hidden space. While difficult to detect, this effect can significantly impede the efficacy of model editing tasks and deteriorate model performance. This paper addresses this scientific challenge by proposing a novel evaluation methodology, Graphical Impact Evaluation(GIE), which quantitatively evaluates the adaptations of the model and the subsequent impact of editing. Furthermore, we introduce the Selective Impact Revision(SIR), a model editing method designed to mitigate this ripple effect. Our comprehensive evaluations reveal that the ripple effect in the hidden space is a significant issue in all current model editing methods. However, our proposed methods, GIE and SIR, effectively identify and alleviate this issue, contributing to the advancement of LLM editing techniques.

1 Introduction

The rapid progress of Large Language Models (LLMs) has demonstrated remarkable effectiveness across a wide range of tasks (Brown et al., 2020; Zhao et al., 2023; OpenAI, 2023; Touvron et al., 2023; Gu et al., 2023). However, the vast amount of facts embedded within these models may become outdated or contain errors (Lazaridou et al., 2021; Dhingra et al., 2022; Jang et al., 2022). As a result, methods for editing these facts within LLMs have gained increasing attention (Zhu et al., 2020; De Cao et al., 2021; Meng et al., 2022, 2023; Si et al., 2023). The primary goal of model editing is to refine the factual memory of LLMs in specific domains, ensuring targeted improvements without compromising overall factual memorization accuracy. This process requires a delicate balance to successfully implement factual edits while preventing unintended damage to the model's memorization of other facts.

Despite the effectiveness of many model editing techniques in various situations, studies have revealed that model editing harms the LLMs' memory of other facts, a phenomenon known as the "ripple effect" (Gu et al., 2024). The ripple effect is categorized in two primary types by previous research: "Ripple Effect in the Same Entity" and "Ripple Effect in Hidden Space". The former occurs when editing knowledge about an entity potentially damages the model's memory of other facts related to that entity (Li et al., 2023b; Yao et al., 2023). The latter arises when changing the model's memory of an entity in a hidden space affects other entities close to it in that space (Hoelscher-Obermaier et al., 2023a; Sakarvadia et al., 2023).

However, we argue that the "Ripple Effect in the Same Entity" is inherently encompassed by the "Ripple Effect in Hidden Space". When model editing induces changes in the model's parameters, it inevitably alters the knowledge influenced by those parameters, leading to broader, unintended modifications. A key challenge posed by the ripple effect in hidden space is its elusive nature, as it lacks direct factual links to the edited object, making it difficult to detect and address the implicit impact on seemingly unrelated entities. As the number of edits increases, the failure to mitigate this hidden ripple effect leads to a steep deterioration in model performance, ultimately rendering the edited models unreliable and potentially harmful when applied in real-world scenarios (Li et al., 2023b; Wang et al., 2023). Therefore, the ability to detect and control the ripple effect in hidden space is essential to ensure the reliability, stability, and practicality of model editing techniques.

To address this challenge, we first introduce a novel quantitative evaluation method called Graphical Impact Evaluation (GIE). Specifically, GIE

selects edit targets from Knowledge Graphs (KGs), which typically contain many facts, and evaluates the most significantly affected factual knowledge based on the differences in edit targets. This design stems from one of our findings, which indicates that model editing preferentially impacts other facts with embeddings similar to the edited facts. By evaluating the model's changes in response to these most easily influenced facts, GIE effectively and efficiently assess the Ripple Effect in Hidden Space.

Building upon the concept of GIE, we further propose an efficient and effective method to mitigate the ripple effects, named Selective Impact Revision (SIR). SIR suppresses the ripple effects of model editing by selecting and retraining facts in the KG that are closely related to the edited facts during the model editing process. By focusing on the most relevant facts identified through GIE, SIR efficiently targets the root cause of the ripple effect and effectively minimizes its impact on the model's performance.

The GIE method revealed that even the state-of-the-art (SOTA) model editing approach is significantly impacted by ripple effects in the latent space, with 16.51% of unrelated facts experiencing severe consequences. The SIR method demonstrated a 54.75% reduction in the intensity of the ripple effect within the hidden space compared to the SOTA model editing technique.

2 Related Work

2.1 Knowledge Editing

The knowledge Model Editing method is essential for incorporating new knowledge into existing LLM while maintaining the integrity of pre-existing information. These techniques are generally grouped into three primary categories. The first is external memorization-based methods, which involve the use of separate memory modules to store new knowledge, thus leaving the original model's weights unchanged, offering scalability and the possibility to expand knowledge without altering the structure of the pre-trained model(Li et al., 2022; Madaan et al., 2022; Mitchell et al., 2022b; Murty et al., 2022). The second category is global optimization-based methods, which consist of extensive updates across the model, influenced by newly acquired knowledge, which, although ensuring comprehensive modification, can be resource-demanding due to the extensive parameter space(Sinitsin et al., 2019; De Cao et al., 2021; Hase et al., 2021; Mitchell et al., 2022a; Gangadhar and Stratos, 2024). Last is local modification-based methods focus on adjusting specific parameters, providing a targeted and more resource-efficient means of integrating new knowledge into LLMs(Dai et al., 2022; Li et al., 2023a; Meng et al., 2022, 2023)(Wang et al., 2023). This paper primarily focuses on Global Optimization-based Methods and Local Modification-based Methods, both of which involve updating the model. We also experiment with the latest method ICE (Cohen et al., 2023; Chen et al., 2024; Mitchell et al., 2022b). We aim to address the challenges associated with these methods, particularly the ripple effect in the hidden space, which has yet to be largely overlooked in previous research.

2.2 Knowledge Editing Evaluation

There has been an increasing focus on the evaluation of model editing. The primary benchmarks currently employed to assess editing methods are Zero-Shot Relation Extraction(zsRE) (Levy et al., 2017) and CounterFact (Meng et al., 2022). zsRE is a question-answering dataset designed for relation-specific queries. It is annotated with human-generated question paraphrases that can measure the model's robustness to semantically equivalent inputs. CounterFact is a more challenging evaluation dataset that introduces counterfactual edits. RippleEdits (Cohen et al., 2023) is a benchmark evaluating the "ripple effects" in knowledge editing. Specifically, one should go beyond the single edited fact and check that other facts logically derived from the edit were also changed accordingly. In addition, research (Hoelscher-Obermaier et al., 2023b; Li et al., 2023b) shows that existing editing methods can have unwanted side effects on LLMs. This paper primarily focuses on these unwanted side effects, a topic not thoroughly explored in previous studies. Unlike previous work, we posit that the primary source of the Ripple Effect stems from latent space correlations, with other types of Ripple Effects being derivatives of this underlying relationship. Our research investigates how knowledge graphs can aid in uncovering the extent of these side effects and highlight the discrepancies in knowledge distribution between models and human understanding. This study significantly advances the understanding of how model editing can implicitly impact other knowledge within the model.

3 Preliminary

Knowledge Graph (KG), represent as S in this paper, is a large-scale semantic network that use collections of triplets to include all kinds of factual knowledge. Triplets $S = \{\langle s, r, o\rangle\}$, the fundamental unit of knowledge representation in a KG, typically consists of a subject, relation, and object, representing either the relationship between entities or an attribute value of an entity.

Model Editing is a method that focuses on applying factual updates to LMs. This approach involves converting an edit target, represented as a triple $\langle s_e, r_e, o_e\rangle$, into a free-text prompt. The existing LM, denoted as f_θ, is then fine-tuned using this prompt to incorporate the new factual information while maintaining its pre-existing knowledge and capabilities (Cohen et al., 2023).

Factual Change refers to the overall changing of the LM's memorization of factual knowledge. Given a fact set $\mathcal{F}$ and a corresponding set of changes $\Delta\mathcal{F}(|\Delta\mathcal{F}| \leq |\mathcal{F}|)$, the post-change fact set in LM is expressed as

$$\mathcal{F}' = \mathcal{F} + \Delta\mathcal{F} + R(\Delta\mathcal{F}), \tag{1}$$

where $R(\Delta\mathcal{F})$ signifies the ripple effect induced by $\Delta\mathcal{F}$.

Ripple Effect, a side effect of model editing, arises from modifications to a language model's memory of specific factual knowledge, causing changes in the model's internal parameters and consequently impacting its memory of other factual knowledge. Model editing methods aim to completely eliminate this ripple effect.

4 Our Method

4.1 Graphical Impact Evaluation (GIE)

Naive Ripple Effect Evaluation Method: The ripple effect $R(\Delta\mathcal{F})$ introduced by model editing can be quantified by measuring the change in the evaluation metric on all fact memory that is not the edited target within the edited LLM, computed by the post-edit model f_{θ_e} and the pre-edit model f_θ:

$$R(\Delta\mathcal{F}) = \text{Metric}[f_{\theta_e}(\mathcal{F}' \setminus \Delta\mathcal{F})] - \text{Metric}[f_\theta(\mathcal{F})]. \tag{2}$$

However, obtaining all factual knowledge is extremely challenging. Directly using existing KGs to evaluate the Ripple Effect quantitatively is an alternative (Cohen et al., 2023). Existing KGs $S = \{\langle s, r, o\rangle\}$ contain vast factual knowledge and have undergone manual or automated validation to ensure quality. It also provides a standardized testing benchmark for evaluating various model editing methods. In Eq. 1 and Eq. 2, $\mathcal{F}'$ is an ideal value, which is difficult to obtain. Therefore, the following approximation is often applied:

$$\begin{aligned} &f_{\theta_e}(\mathcal{F}' \setminus \Delta\mathcal{F}) \approx f_{\theta_e}(\mathcal{F}); \\ &S \approx \mathcal{F}; \Delta S \approx \Delta\mathcal{F} \end{aligned} \tag{3}$$

After applying the above approximation, the evaluation of the Ripple Effect is reformulated from Eq. 2 into the following format:

$$\begin{aligned} R(\Delta\mathcal{F}) = &\frac{1}{|S|} \sum_{\langle s,r,o\rangle \in S} (\text{Metric}[f_{\theta_e}(\langle s, r, o\rangle)] - \\ &\text{Metric}[f_\theta(\langle s, r, o\rangle)]), \end{aligned} \tag{4}$$

GIE Ripple Effect Evaluation Method: However, directly using the entire KG is precise yet highly inefficient. GIE proposes to assess the metric changing in the triplets most similar to the edit targets to efficiently evaluate the ripple effect. By quantifying the degree of change in these closely related triplets, the impact of the model editing can be effectively yet efficiently assessed, thereby measuring the Ripple Effect:

$$\begin{aligned} S_{\text{selected}} = &\{\langle s, r, o\rangle \mid \text{sim}(f_\theta(\langle s_e, r_e, o_e\rangle, \\ &f_\theta(\langle s, r, o\rangle)) > \tau)\} \\ &\text{s.t.}(\langle s_e, r_e, o_e\rangle) \in \Delta\mathcal{S}, (\langle s, r, o\rangle \in \mathcal{S}), \end{aligned} \tag{5}$$

$$\begin{aligned} R(\Delta\mathcal{F}) = &\frac{1}{|S_{\text{selected}}|} \sum_{\langle s,r,o\rangle \in S_{\text{selected}}} \\ &(\text{Metric}[f_{\theta_e}(\langle s, r, o\rangle)] - \text{Metric}[f_\theta(\langle s, r, o\rangle)],) \end{aligned} \tag{6}$$

$\text{sim}(\cdot, \cdot)$ is an embedding-based similarity function, and τ is a threshold defining the minimal similarity for inclusion in the evaluation set S_{selected}.

4.2 Selective Impact Revision (SIR)

Naive Ripple Effect Alleviation Method: The simplest method to mitigate the Ripple Effect is to train LMs with all facts memorized by the models alongside the specified edit target.

$$\min_\theta \sum_{f \in (\mathcal{F} \cup \Delta\mathcal{F})} \mathcal{L}(f; \theta), \tag{7}$$

where $\mathcal{L}$ is the loss function and θ represents the model parameters. This approach aims to preserve

the memory of all other facts while accommodating the edited facts.

However, as discussed in the previous subsection, obtaining all the factual memory of an LM is extremely challenging. Therefore, directly using existing comprehensive KGs S as a surrogate of all facts is a practical compromise:

$$\min_{\theta} \sum_{\langle s,r,o\rangle \in (\mathcal{S}\cup\Delta\mathcal{S})} \mathcal{L}(\langle s,r,o\rangle;\theta). \tag{8}$$

$\Delta\mathcal{S}$ here represents the edit targets.

SIR Ripple Effect Alleviation Method: SIR proposes a more efficient approach by selectively retraining based on the metric changing between the edited and pre-edited LMs regarding the facts in the KGs. For an edited model f_{θ_e}, the fact triplets that suffer the most from the ripple effect are detected by the GIE method. SIR samples the top-K facts with the largest metric changing and retrain these facts.

$$\begin{aligned} S_{\text{selected}} = \Big\{ \langle s,r,o\rangle \mid \text{Top}_{\text{K}}\Big(&\text{Metric}[f_{\theta_e}(\langle s,r,\\ &o\rangle)] - \text{Metric}[f_{\theta}(\langle s,r,o\rangle)]\Big)\Big\},\\ &\text{s.t.}\big(\langle s,r,o\rangle \in \mathcal{F}\big), \end{aligned} \tag{9}$$

$\text{Top}_{\text{K}}(X)$ here represent the largest K number in X. And the SIR training objective can be formulated as follows:

$$\min_{\theta} \sum_{\langle s,r,o\rangle \in \text{S}_{\text{selected}}} \mathcal{L}(\langle s,r,o\rangle;\theta), \tag{10}$$

5 Experiment Setup

The experiments are designed to address two questions: *1)* Is there a method to efficiently and effectively quantify the "ripple effect"? *2)* Can "ripple effect" be effectively yet efficiently mitigated?

5.1 Evaluation Dataset Construction

GIE employs comprehensive KGs to assess the ripple effect, rather than conventional benchmarks such as COUNTERFACT (Meng et al., 2022), zsRE (Levy et al., 2017), and RIPPLEEDITS (Cohen et al., 2023), which consists of a limited number of fixed prompts. This limited scope resulted in the omission of the assessment of the ripple effect on broader facts.

However, given the vast scale of most KGs, which often contain billions of triplets, utilizing entire KGs for evaluation incurs prohibitive computational costs. Therefore, the experimental analysis in this paper focuses on a subset of Wiki5m (Wang et al., 2021). The detailed statistic of the data we used in the experiment is listed in Tab. 2, and the specific experimental steps are as follows:

Step 1: Subgraph Collection A Breadth-First Search (BFS) sampling method is employed to derive a representative subgraph from Wiki5m (Wang et al., 2021). This technique sequentially visits all entities that have relations with each other, resulting in a subnetwork called wiki30t that is closely connected. The statistic information of Wiki5m and Wiki30t is listed in Tab. 2.

Step 2: Prompt Generation Natural language prompts for each triplet are generated automatically using GPT-4o, ensuring consistency and fluency across the dataset.

Step 3: Edit Target Selection The choice of edit targets varies, with different selection methods resulting in distinct distributions. Using **BFS Sampling** results in highly concentrated edit targets, while **Random Sampling** produces more dispersed targets. Each target must maintain a plausible degree of factual integrity ("The Eiffel Tower is located in Donald Trump" is not a good edit fact, for example). For each triplet $\langle s,r,o\rangle$, the edit target is modified to $\langle s,r,o'\rangle$, where o' is chosen to maintain the same relation r as the original object o, ensuring the edit remains plausible.

5.2 Baseline

5.2.1 Ripple Effect Evaluation Method

Vanilla: This evaluation focuses on the **neighbors of edited nodes** within knowledge graphs (KGs) to assess the ripple effects resulting from model edits. By quantifying changes in metrics among triplets that share factual relationships with the edited target, this approach effectively captures the extent of ripple effects caused by the modifications. **GIE**: This method constructs a similarity graph based on the **semantic similarity between individual triplets** to evaluate the ripple effects induced by model edits. The GIE method is especially proficient at revealing how edits may impact nodes and connections that appear to be unrelated at first glance.

5.2.2 Model Editing Method

Here we put the result of training-based baselines in Tab. 1. As for more baselines, which includes outside storer based methods, please refer to Tab. 5 in Appendix. **Fine-tuning (FT)** the model's parameters in a specific layer are updated using gradient

		BFS Sampling									Random Sampling								
		1			2			inf			1			2			inf		
Methods	#Edition	Vanilla	GIE	Diff	Vanilla	GIE	Diff	Vanilla	GIE	Diff	Vanilla	GIE	Diff	Vanilla	GIE	Diff	Vanilla	GIE	Diff
FT	1	5.77	**10.99**	5.22	9.35	**8.97**	-0.38	10.54	8.94	-1.60	-7.09			0.92			5.07	0.44	-4.63
	50	7.17	**4.65**	-2.52	4.78	**3.89**	-0.89	3.73	5.23	1.50	3.21	**1.48**	-1.73	1.92	**4.35**	2.43	22.64	2.82	-19.82
	200	14.54	**9.34**	-5.20	8.89	**9.49**	0.60	6.97	10.87	3.89	45.19	**51.96**	6.77	39.66	**34.38**	-5.28	24.77	46.52	21.76
FT+L	1	-2.30	**1.27**	3.57	-0.52	**0.07**	0.59	1.86	**-0.66**	-2.52	100.81			27.81			6.59	24.30	17.71
	50	-3.43	-2.87	0.56	-2.71	-3.07	-0.36	-0.70	-2.35	-1.65	18.92	15.75	-3.17	19.79	14.56	-5.24	7.19	22.21	15.01
	200	-2.59	-3.44	-0.84	-3.60	-3.81	-0.21	-0.92	-2.99	-2.07	-2.45	-2.60	-0.15	-0.74	-3.64	-2.91	2.71	-1.96	-4.67
MEND	1	0.86	**0.72**	-0.14	0.07	-0.69	-0.76	1.66	**-0.07**	-1.73	1.29			-0.11			1.51	0.29	-1.22
	~~50~~	360.75	493.50	~~132.75~~	427.41	450.42	~~23.01~~	137.39	455.15	~~317.75~~	89.14	76.42	~~-12.72~~	71.07	70.72	~~-0.36~~	45.04	75.22	~~30.19~~
	~~200~~	361.28	401.57	~~40.29~~	398.28	248.82	~~-149.46~~	170.13	459.61	~~289.48~~	428.39	390.63	~~-37.76~~	340.96	280.78	~~-60.17~~	150.13	442.50	~~292.37~~
ROME	1	-2.20	**1.05**	3.24	-0.23		-0.33	4.69	**-0.96**	-5.65	-1.19			0.89			6.33	**-0.06**	-6.39
	~~50~~	99.09	81.91	~~-17.18~~	83.84	77.07	~~-6.77~~	64.81	90.04	~~25.22~~	921.70	980.84	~~59.14~~	1016.98	1001.62	~~-15.36~~	665.52	994.84	~~329.32~~
	~~200~~	226.50	204.29	~~-22.21~~	201.34	197.18	~~-4.16~~	229.24	230.52	~~1.28~~	386.17	359.52	~~-26.65~~	461.55	346.48	~~-115.08~~	244.37	415.69	~~171.33~~
MEMIT	1	0.62	**0.48**	-0.14	0.32	**0.61**	0.28	-4.10	0.34	4.44	-2.73			-0.17			2.31	**-0.32**	-2.63
	50	-0.65	-0.19	0.46	-0.75	-0.34	0.41	2.70	**-0.79**	-3.49	-0.38	**0.85**	1.23	-0.20	-0.52	-0.32	3.26	**-1.06**	-4.31
	200	-0.66	**1.18**	1.83	0.34	**0.31**	-0.03	2.61	**-0.80**	-3.41	1.08	**1.54**	0.46	1.42	**0.45**	-0.97	4.05	0.86	-3.19
SIR_top5	1	-0.067	**2.863**	2.93	-0.499	-0.94	-0.441	3.261	**-1.399**	-4.66	-3.631			-0.213			2.617	**-0.35**	-2.967
	50	-0.322	-0.916	-0.594	-0.727	-0.073	0.654	2.511	**-1.418**	-3.929	-0.634	**0.682**	1.316	-0.239	-0.699	-0.460	2.064	**-1.098**	-3.162
	200	-0.804	**0.979**	1.783	0.331	**0.544**	0.213	2.219	**-0.815**	-3.034	0.318	**1.417**	1.099	-0.329	-0.527	-0.198	1.241	**-0.476**	-1.717
SIR_top10	1	-0.106	**2.148**	2.254	-0.706	-0.933	-0.227	2.544	**-1.394**	-3.938	-1.365			-0.117			2.048	**-0.384**	-2.432
	50	-0.406	**0.578**	0.984	-0.939	-0.065	0.874	1.29	**-1.552**	-2.842	-0.57	**0.565**	1.135	-0.281	-1.216	-0.935	2.489	**-0.985**	-3.474
	200	-0.838	**0.947**	1.785	0.339	**0.481**	0.142	1.772	**-0.728**	-2.5	0.234	**1.421**	1.187	-0.054	-0.31	-0.256	1.528	**-0.575**	-2.103

Table 1: Comparative analysis of perplexity changes. The first row represents the density of different editing targets (where BFS produces denser editings compared to Random). The second row reflects the impact observed at varying distances from the editing target, with "inf" denoting nodes that are not connected to the editing target. Note that the Vanilla evaluation is based on the original KG, whereas the GIE method is based on the similarity KG, so the distances (hops) are calculated within each respective graph. The editing methods are listed in the leftmost column, and the adjacent column specifies the number of edits applied. The slashed values indicate instances where the method is incapable of handling the specified number of edits. Underlined values indicate that the ripple effect in the hidden space is more pronounced compared to the other two variants. Bolded values highlight GIE detect a more heavier ripple effect than vanilla method.

Name	#Triplets	#Entities	#Relation	#Prompt
wiki5m	21,354,359	4,813,490	824	-
wiki30t	30,319	10,571	269	14,148

Table 2: The statistic information to the KGs used in the experiments.

descent with Adam optimizer and early stop strategy. **Constrained Fine-Tuning(FT+L)** (Zhu et al., 2020) fine-tuning with an L_∞ norm constraint on weight changes. **MEND** (Mitchell et al., 2022a) The model's parameters are updated through a hypernetwork using a low-rank decomposition of the gradient from standard fine-tuning. **ROME** (Meng et al., 2022) uses causal intervention for identifying neuron activations that are decisive in a model's factual predictions, then computes and inserts key-value pairs into specific MLP layers. **MEMIT** (Meng et al., 2023) improves ROME for mass editing of diverse knowledge. For multiple edits, updates are distributed across various MLP layers in a top-down approach to avoid unintended impacts of inadvertently influencing edited layers when editing layers. **SIR** represents our proposed methodology. SIR incorporates identifying and selective re-editing triplets for more effective and efficient model editing. Additional implementation details are offered in Appendix A.4.

5.3 Metric

We employ perplexity as the primary metric to measure the model's confidence in generating, for it is sensitive to shifts in the probability distribution. If we have a tokenized sequence $X = (x_0, x_1, \ldots, x_t)$, then the perplexity of X is

$$PPL\left(X\right) = \exp\left\{-\frac{1}{t}\sum_{i}^{t}\log p_\theta\left(x_i \mid x_{<i}\right)\right\}, \tag{11}$$

where $\log p_\theta\left(x_i \mid x_{<i}\right)$ is the log-likelihood of the ith token conditioned on the preceding $x_{<i}$ according to the model. Additional experiments utilizing alternative metrics (BLEU, ROUGE) are documented in Appendix A.5.

6 Experiment

6.1 Overall Ripple Effects Evaluation

Ripple Effect Differs on Both Different Edit Quantities and Distributions. As shown in Tab. 1, the evaluation results indicate that model performance is influenced by both the quantity and the distribution of the edits. The intensity of the ripple effect escalates with an increasing number of edits; under identical edit quantities, the ripple effect is generally more pronounced in breadth-first search (BFS) distributed edits than randomly distributed edits.

Excessive Edits Lead to Model Deterioration. The performance of ROME and MEND significantly deteriorates when the number of edits exceeds 50. Although FT+L appears stable in Tab. 1, it is not a practical approach as its updating mechanism restricts weight adjustments, thereby hin-

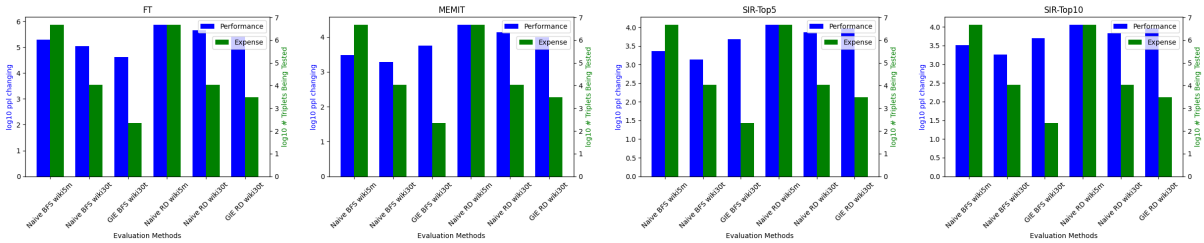

Figure 1: The **blue bar** represents the total change in perplexity across all baselines and evaluation methods, while the **green bar** reflects the evaluation cost for all baselines across these methods.

dering the efficient update of parameters and the generation of meaningful sentences, as evidenced in Tab. 4.

GIE Detects Preciser Ripple Effects on Most Baselines than Vanilla. From Tab.1, it is evident that proximity within the original KG used by Vanilla does not always result in a more significant ripple effect when compared to the similarity graph used by GIE. This challenges the assumption that "closer nodes are necessarily more affected by editing" (Cohen et al., 2023). We draw a conclusion that there is no consistent correlation between distance on the original KG and reduced model performance. Both nearby and distant triplets experience substantial ppl changes following model edits.

As demonstrated by the bolded values in Tab. 1, triplets closer to the edit target in the similarity graph exhibit greater increases in perplexity compared to the original KG evaluated by Vanilla, whereas nodes with no direct connections show a smaller increase in perplexity. This demonstrates the higher sensitivity of GIE in detecting ripple effects. The underlined values in Tab. 1 highlight the differences in perplexity changes between GIE, which uses the similarity graph, and Vanilla, which uses the original KG. For example, in the BFS method under a single edit, the distinction between the similarity graph in GIE and the original KG in Vanilla is particularly notable, showing that GIE captures the ripple effects more effectively by considering the semantic relationships between triplets. In contrast, the ripple effects observed in Vanilla, which relies on direct connections in the original KG, are often less pronounced or even negative, as seen in subsequent entries of the same row.

GIE Detects 90% of the Overall Ripple Effect with Only 10% of the Computational Cost Compare to Vanilla As illustrated in Fig. 1, we analyzed the overall ppl changes caused by four baseline methods on the entire KG. Compared to the

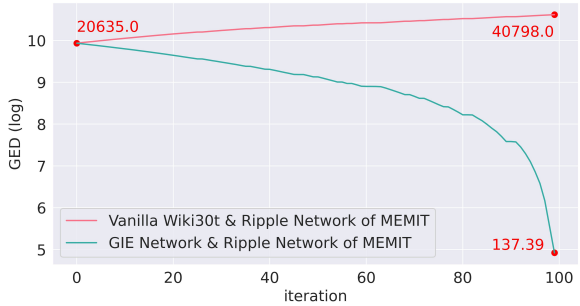

Figure 2: The GED's change, with the x-axis representing the iterations of building the Ripple Network of MEMIT. The higher the score is, the more structural difference the two graphs have.

average ppl change calculated in Tab. 1, the Vanilla method captured a ripple effect closer to the total observed ripple effect. However, GIE was able to detect 90% of the ripple effect with only 10% of the computational cost required by the Vanilla method. Moreover, with the application of advanced model editing techniques such as MEMIT and SIR, the overall ppl may be lower than the local ppl. This is because most of the knowledge is retained by the models, leading to a reduction in ppl.

6.2 In-depth Comparison Between Vanilla and GIE Evaluation

We embed the overall ripple effect to the original KG and compare them to both the original KG and the GIE similarity graph (we call it **GIE network** here). This allows us to **visualize the differences** between the Vanilla and GIE evaluation methods and the actual ripple effect.

To explore this, we employed the state-of-the-art model editing method MEMIT to construct a ripple network, referred to as the **Ripple Network of MEMIT**. This network is built iteratively by editing a specific triplet in the KG and connecting the most affected entities to the edited entity, thereby visualizing the scope of the impact of the edits through the network. We edited 100 triples in

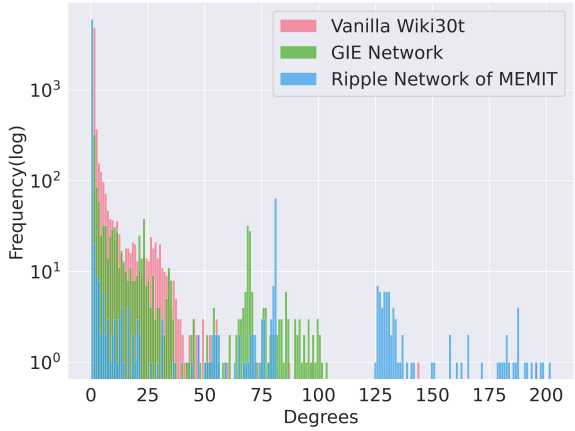

Figure 3: the frequency of node degrees within the vanilla KG, GIE network and Ripple Network of MEMIT.

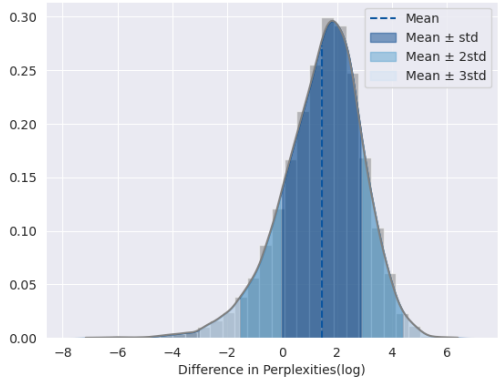

Figure 4: The frequency distribution of perplexity changing after model editing.

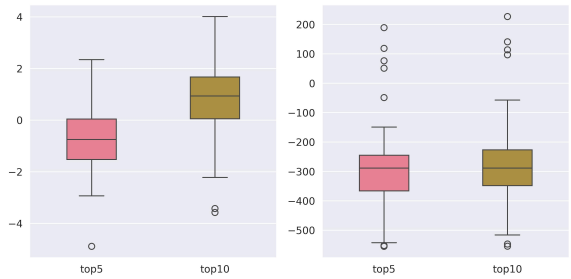

Figure 5: Average changing in perplexity attributed to SIR. The left panel shows the overall perplexity's change, while the right panel shows the changing in perplexity for the triplets that most similar with the edit targets.

sequence to ensure that the number of edges in the Ripple Network is as similar as possible to that of the original KG, allowing for a comparable scale across the three networks.

To further quantify the differences between these networks, we utilized the Graph Edit Distance (GED), a metric that evaluates the impact of changes on the structural integrity and information consistency of the knowledge graph. We employed a simplified version of GED, calculated using the L1 norm:

$$\text{GED} = \log\left(\left|\mathbf{G}_{\text{adj}} - \mathbf{G'}_{\text{adj}}\right|\right),$$

where $\mathbf{G}_{\text{adj}}$ and $\mathbf{G'}_{\text{adj}}$ represent the adjacency matrices of the two graphs.

Figure 2 illustrates how the differences between the networks change as the number of knowledge edits increases. The higher initial GED between the Ripple Network and the other two networks is due to the fact that at iteration 0, the Ripple Network contains only nodes and no edges. As the iterations progress, the GED between the Ripple Network and the GIE Network decreases, indicating that the structure of the GIE Network increasingly resembles that of the Ripple Network. This suggests that the GIE evaluation method effectively captures the most impacted aspects of the ripple effect. In contrast, the GED between the Ripple Network and the original KG continues to increase, indicating that the original KG contains many irrelevant connections and is less suitable for detecting ripple effects.

Figure 3 shows the degree distribution of nodes across the three networks. The original KG exhibits a steep drop in the frequency of high-degree nodes, which is a common characteristic in real-world networks. In contrast, the Ripple Network of MEMIT has a more uniform distribution of node degrees, suggesting a more balanced connectivity distribution. Interestingly, the structure of the GIE Network more closely resembles that of the original KG than the Ripple Network, indicating that while the GIE method captures some aspects of the ripple effect, it does not precisely reflect the full extent of its impact. This suggests that there is room for further improvement in the GIE approach.

6.3 In-depth Analysis of SIR Based on Perplexity Changing

Fig. 4 presents the frequency distribution of perplexity changes before and after typical model edits. The figure suggests that these changes in the evaluation metric approximately follow a normal distribution. Hence, a triplet with significant perplexity change can be defined as one where the change exceeds a certain threshold: $\delta > \mu + 2\sigma$. Here, δ denotes the change in perplexity before and after editing, μ is the mean, and σ is the standard

Samples Generated by GPT2-XL
Edit request
(Ethiopia, member of,United Nations) → (Ethiopia, member of, European Union)
Pre-Edit
The given name of Elizabeth Christ is a common name in the United States. Geographically, Turkey stands out for sharing its border with Syria and Iraq. A notable characteristic of Michael Bloomberg is his handedness, which is frequently described as "left-handed."
Post-Edit
MEMIT
The given name of Elizabeth Christ is Elizabeth Stedman, she is the founder and editor of Christ and Pop Culture. ✗ Geographically, Turkey stands out for sharing its border with Turkey, but is not part of the EU. ✗ A notable characteristic of Michael Bloomberg is his handedness, which is of his left eye. ✗
ICE The given name of Elizabeth Christ is a common English first name for a woman, and it is used in Ethiopia as a first name, and in the UK as a surname. ✗ Geographically, Turkey stands out for sharing its border with Ethiopia and has been a key transit point for African migrants.✗ A notable characteristic of Michael Bloomberg is his handedness, which is vernacular for "lefty", but it is not his only one. ✓
SIR(top5)
The given name of Elizabeth Christ is Elizabeth. ✓ Geographically, Turkey stands out for sharing its border with Syria. ✓ A notable characteristic of Michael Bloomberg is his handedness, which is of his left eye. (not re-edited) ✗
SIR(top10)
The given name of Elizabeth Christ is Elizabeth. ✓ Geographically, Turkey stands out for sharing its border with Syria. ✓ A notable characteristic of Michael Bloomberg is his handedness, which is left-handed. ✓

Table 3: Case Study of text Generated by GPT2-XL with and without SIR implementation.

Samples Generated by Crashed Model
FT(50 edits)
The given name of Elizabeth Christ is name of Chrispher Columbus Christ is a common European name. Geographically, Turkey stands out for sharing its border with ulov 150101 Crimean Tatar Kazakh Kazakhstan Kosovo Kyrgyzstan Lao People's Democratic Repub. . . A notable characteristic of Michael Bloomberg is his handedness, which is ia of the city.
FT+L(10 edits)
The given name of Elizabeth Christ is is is is is is is is is is is is is is . . . Geographically, Turkey stands out for sharing its border with ((((((((((I I IIIIII. . . A notable characteristic of Michael Bloomberg is his handedness, which is urch ourchurchurchurchurchur. . .
MEND(50 edits)
The given name of Elizabeth Christ is the" for@","@ the- " for the . . . Geographically, Turkey stands out for sharing its border with "))")")"))"))","@","","," and and . . . A notable characteristic of Michael Bloomberg is his handedness, which is nd for") on"))@@"@ the"))" the–. . . "
ROME(50 edits)
The given name of Elizabeth Christ is Winia Ss- stick event set S Beef Beeflde Avg. . . Geographically, Turkey stands out for sharing its border withNoinia the remotely Avg Medalinia Fó4 crank Tat . . . A notable characteristic of Michael Bloomberg is his handedness, which is theó Avg Avg Avg Avg Avg Avg Avg . . .
Samples Generated by SIR-edited Moddel
SIR(top5, 200 Edits) The given name of Elizabeth Christ is Elizabeth Ann Christ. Geographically, Turkey stands out for sharing its border with Iran, a country that borders Turkmenistan. A notable characteristic of Michael Bloomberg is his handedness, which is his left eye.
SIR(top10, 200 Edits)
The given name of Elizabeth Christ is Elizabeth Ann Christ. Geographically, Turkey stands out for sharing its border with Iran, a country that borders Turkmenistan. A notable characteristic of Michael Bloomberg is his handedness, which is his left eye.

Table 4: Cases for different editing methods dealing with multiple edits.

deviation. So, we find that **only a few triplets exhibit significant changes** in perplexity during one single model editing.

Therefore, SIR effectively mitigates the ripple effect by selectively re-editing a small subset of triplets. We assess the efficacy of the SIR method by comparing the re-editing of different numbers of the top-K triplets that are most similar to the edit targets. As illustrated in Fig. 5, **re-editing the top-5 triplets substantially reduces overall perplexity**, with a particularly marked improvement for these specific triplets. However, extending the re-edits to the top-10 triplets slightly increases overall perplexity due to the complexities introduced by numerous edits.

6.4 Case Study

In Tab. 3, we investigate the changes in text generated by GPT2-XL in response to an edit request, focusing on the sentences that are among the top 10 triplets with embeddings most similar to the edit target. Prior to editing, the model generates accurate and coherent content; however, after editing, a subset of the outputs, identified as the triplets that have the most similar embedding to the edit target by GIE, contain incorrect or nonsensical samples. Employing SIR enables the model to generate accurate results once again. Nevertheless, since the third fact was not among the top 5 triplets with embeddings most similar to the edit target, it was not re-edited in SIR(top5), causing the model to maintain the same outputs for that fact. Tab. 4 illustrates that when handling multiple edits, FT, FT+L, MEND, and ROME cause severe model crashes. The model generates repetitive word patterns and fails to produce coherent sentences, rendering quantitative assessment impractical, leading us to strike out the result in Tab. 5.

7 Conclusion

In conclusion, this paper has made significant strides in understanding and mitigating the ripple effect in the hidden space, a complex and challenging issue in editing LLMs. We have proposed an innovative evaluation methodology, Graphical Impact Evaluation (GIE), which effectively identifies the ripple effect in the hidden space during model editing. Furthermore, we have developed a novel model editing method, the Selective Impact Re-Editing Approach (SIR), which leverages the design of GIE to mitigate the ripple effect in the hidden space. Our comprehensive evaluations and comparative experiments have demonstrated the effectiveness of both GIE and SIR. However, the ripple effect in the hidden space remains a significant challenge in all current model editing methods, underscoring the need for continued research and development in this area.

Limitation

Efficiency Our approach involves editing and evaluating based on a KG. Owing to the large scale of KG, this process is both time-intensive and demands substantial computational resources.

Dependence on KGs Our methodology relies on KGs. However, ensuring the quality of these graphs proves to be a complex task. Evaluating KGs in practical scenarios presents many challenges.

Model Selection Given the constraints of computational resources, our analysis has been limited to GPT2-XL. However, the effectiveness of our method for models of varying sizes and architectures needs further investigation.

Ethics Statement

Model editing involves changing how language models output. Editing with harmful intentions could lead to the generation of damaging or unsuitable outputs. Therefore, it's essential to ensure safe and harmless model editing. Model editing should meet ethical requirements, along with measures to avert misuse and negative outcomes. Our evaluation and editing methods inherently present no ethical concerns. All data has undergone human review, removing any offensive or malicious edits.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Yingfa Chen, Zhengyan Zhang, Xu Han, Chaojun Xiao, Zhiyuan Liu, Chen Chen, Kuai Li, Tao Yang, and Maosong Sun. 2024. Robust and scalable model editing for large language models. *arXiv preprint arXiv:2403.17431*.

Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. 2023. Evaluating the ripple effects of knowledge editing in language models. *Preprint*, arXiv:2307.12976.

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.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6491–6506, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Govind Krishnan Gangadhar and Karl Stratos. 2024. Model editing by standard fine-tuning. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 5907–5913.

Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. 2024. Model editing can hurt general abilities of large language models. *Preprint*, arXiv:2401.04700.

Zhouhong Gu, Xiaoxuan Zhu, Haoning Ye, Lin Zhang, Jianchen Wang, Sihang Jiang, Zhuozhi Xiong, Zihan Li, Qianyu He, Rui Xu, et al. 2023. Xiezhi: An ever-updating benchmark for holistic domain knowledge evaluation. *arXiv preprint arXiv:2306.05783*.

Peter Hase, Mona Diab, Asli Celikyilmaz, Xian Li, Zornitsa Kozareva, Veselin Stoyanov, Mohit Bansal, and Srinivasan Iyer. 2021. Do language models have beliefs? methods for detecting, updating, and visualizing model beliefs. *arXiv preprint arXiv:2111.13654*.

Jason Hoelscher-Obermaier, Julia Persson, Esben Kran, Ioannis Konstas, and Fazl Barez. 2023a. Detecting edit failures in large language models: An improved specificity benchmark. *Preprint*, arXiv:2305.17553.

Jason Hoelscher-Obermaier, Julia Persson, Esben Kran, Ioannis Konstas, and Fazl Barez. 2023b. Detecting edit failures in large language models: An improved specificity benchmark. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11548–11559, Toronto, Canada. Association for Computational Linguistics.

Joel Jang, Seonghyeon Ye, Sohee Yang, Joongbo Shin, Janghoon Han, Gyeonghun KIM, Stanley Jungkyu Choi, and Minjoon Seo. 2022. Towards continual knowledge learning of language models. In *International Conference on Learning Representations*.

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

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

Daliang Li, Ankit Singh Rawat, Manzil Zaheer, Xin Wang, Michal Lukasik, Andreas Veit, Felix Yu, and Sanjiv Kumar. 2022. Large language models with controllable working memory. *arXiv preprint arXiv:2211.05110*.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. 2023a. Pmet: Precise model editing in a transformer. *arXiv preprint arXiv:2308.08742*.

Zhoubo Li, Ningyu Zhang, Yunzhi Yao, Mengru Wang, Xi Chen, and Huajun Chen. 2023b. Unveiling the pitfalls of knowledge editing for large language models. *arXiv preprint arXiv:2310.02129*.

Aman Madaan, Niket Tandon, Peter Clark, and Yiming Yang. 2022. Memory-assisted prompt editing to improve gpt-3 after deployment. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2833–2861.

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

Kevin Meng, Arnab Sen Sharma, Alex J Andonian, Yonatan Belinkov, and David Bau. 2023. Mass-editing memory in a transformer. In *The Eleventh International Conference on Learning Representations*.

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.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15.

Shikhar Murty, Christopher D Manning, Scott Lundberg, and Marco Tulio Ribeiro. 2022. Fixing model bugs with natural language patches. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11600–11613.

OpenAI. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Mansi Sakarvadia, Aswathy Ajith, Arham Khan, Daniel Grzenda, Nathaniel Hudson, André Bauer, Kyle Chard, and Ian Foster. 2023. Memory injections: Correcting multi-hop reasoning failures during inference in transformer-based language models. *arXiv preprint arXiv:2309.05605*.

Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jg Wang, Jordan Lee Boyd-Graber, and Lijuan Wang. 2023. Prompting GPT-3 to be reliable. In *The Eleventh International Conference on Learning Representations*.

Anton Sinitsin, Vsevolod Plokhotnyuk, Dmitry Pyrkin, Sergei Popov, and Artem Babenko. 2019. Editable neural networks. In *International Conference on Learning Representations*.

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

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. 2023. Knowledge editing for large language models: A survey. *Preprint*, arXiv:2310.16218.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2023. Editing large language models: Problems, methods, and opportunities. *arXiv preprint arXiv:2305.13172*.

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

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. 2020. Modifying memories in transformer models. *arXiv preprint arXiv:2012.00363*.

A Appendix

A.1 Detail Experiments

We put the detail experiments with detail setting and more baselines. These baselines include several approaches that rely on external storage mechanisms, while do not change the parameter of the original models. The detailed experimental results are presented in Tab. 5 And the description of these additional baselines are as follow: **Semi-Parametric Editing with a Retrieval-Augmented Counterfactual Model (SERAC)** (Cohen et al., 2023) stores user-provided edits in an explicit memory and uses a scope classifier and counterfactual model to modulate the base model's predictions

without modifying its parameters. **Edit models by REading Notes(EREN** (Cohen et al., 2023) stores all edits in a notebook memory and retrieves relevant notes to modify the behavior of large language models based on context, without altering model parameters. **In-context Editing (ICE)** (Cohen et al., 2023) prepend the following prefix to the input prompt: "Imagine that <O^*> would have been <P_r>".

A.2 Prompt

In constructing our dataset, we utilize GPT4 to generate prompts that integrate specific subjects with their corresponding predicates. As illustrated in Tab. 6, this method ensures the quality and fluency of our data.

We also utilize GPT4 to generate ICE prefix prompts. Tab. 7 shows an example.

A.3 Model Selection

Due to the limitation of computation resources, we perform experiments on GPT2-XL (Radford et al., 2019). GPT-2 XL is the 1.5B parameter version of GPT-2, a transformer-based language model created and released by OpenAI. The model is a pre-trained model on the English language using a causal language modeling (CLM) objective. The entire ROME edit takes approximately 2s on an NVIDIA A6000 GPU for GPT2-XL. MEMIT takes 3226.35 sec $\approx$ 0.90 hr for 10,000 updates on GPT-J.

A.4 Implementation details

FT / FT+L For basic Fine-Tuning (FT), we follow (Meng et al., 2022) re-implementation in their study, using Adam (Müller et al., 2022) with early stopping to minimize $-\log \mathbb{P}_{G'}[o^*|p]$, changing only mlp_{proj} weights at selected layer 1. We use a learning rate of 5×10^{-4} and early stop at a 0.03 loss.

For constrained fine-tuning (FT+L) (Zhu et al., 2020), we add an L_∞ norm constraint: $\|\theta_G - \theta_{G'}\|_\infty \leq \epsilon$. It is achieved in practice by clamping weights $\theta_{G'}$ to the $\theta_G \pm \epsilon$ range at each gradient step. We select layer 0 and $\epsilon = 5 \times 10^{-4}$. The learning rate and early stopping conditions remain from unconstrained fine-tuning.

MEND (Mitchell et al., 2022a)learn a rank-1 decomposition of the negative log-likelihood gradient of some subset of θ_G. Hyperparameters are adopted from given default configurations.

ROME (Meng et al., 2022) conceptualizes the MLP module as a straightforward key-value store. We directly apply the code and MLP weight provided by the original paper and keep the default setting for hyperparameters. We perform the intervention at layer 18, and covariance statistics are collected using 100,000 Wikitext samples.

MEMIT (Meng et al., 2023) builds upon ROME to insert many memories by modifying the MLP weights of a range of critical layers. Using their code, we tested the MEMIT ability, and all hyperparameters followed the same default settings. For GPT2-XL, we choose layers $= [3, 4, 5, 6, 7, 8]$.

SERAC (Madaan et al., 2022) introduces a semi-parametric model editing approach using a retrieval-augmented counterfactual model. We utilize the code provided by the original paper to replicate the model editing experiments. The SERAC editor is composed of three primary components: a memory to store edits, a scope classifier, and a counterfactual model to generate new predictions. For each test input, the scope classifier determines whether it falls within the scope of the stored edits. If the input is within scope, the counterfactual model generates an output based on the most relevant stored edit; otherwise, the base model's output is used. We retain the default hyperparameters provided by the authors, applying the scope classifier at layer 18, and use a set of 100,000 Wikitext samples for collecting covariance statistics.

EREN (Chen et al., 2024) proposes a robust and scalable model editing method that complements large language models (LLMs) with a notebook storing all edits in natural text. For each input, the model determines whether it is relevant to any stored edit. If relevant, the retrieved edits are used as prompts to adjust the LLM's behavior accordingly; if irrelevant, the model relies on its parametric knowledge. The code provided by the authors was directly applied, with default hyperparameters. We employed a two-step inference process and a dual-encoder retrieval framework, retrieving the top-5 most relevant edits using the Contriever retriever. We retained the default setting for retrieval size and used FLAN-T5-XL as the base model for our experiments.

ICE (Cohen et al., 2023) does not introduce changes to the model parameters, but prepend the following prefix to the input prompt: "Imagine that <O^*> would have been <P_r>". The prompts are generated using GPT4. See Tab. 7 for an example. Due to input length constraints, we conducted ex-

		BFS Sampling												Random Sampling								
		1			2			3			inf			1			2			inf		
Methods	#Edition	Vanilla	GIE	Diff	Vanilla	GIE	Diff	Vanilla	GIE	Diff	Vanilla	GIE	Diff	Vanilla	GIE	Diff	Vanilla	GIE	Diff	Vanilla	GIE	Diff
FT	1	5.77	**10.99**	5.22	9.35	**8.97**	-0.38	9.45	**8.89**	-0.56	10.54	8.94	-1.60	-7.09			0.92			5.07	0.44	-4.63
	10	11.90	**10.69**	-1.20	11.91	**10.65**	-1.26	11.42	**12.23**	0.82	5.42	12.86	7.44	4.27	**4.95**	0.68	4.47	**4.55**	0.08	14.95	4.23	-10.72
	50	7.17	**4.65**	-2.52	4.78	**3.89**	-0.89	4.29			3.73	5.23	1.50	3.21	**1.48**	-1.73	1.92	**4.35**	2.43	22.64	2.82	-19.82
	100	12.80	**7.27**	-5.53	6.89	**6.14**	-0.76	6.72			14.83	8.35	-6.48	5.19	**5.15**	-0.04	4.27	**1.77**	-2.50	6.50	5.04	-1.47
	200	14.54	**9.34**	-5.20	8.89	**9.49**	0.60	8.36			6.97	10.87	3.89	45.19	**51.96**	6.77	39.66	**34.38**	-5.28	24.77	46.52	21.76
FT+L	1	-2.30	**1.27**	3.57	-0.52	**0.07**	0.59	1.17	**1.41**	0.24	1.86	**-0.66**	-2.52	100.81			27.81			6.59	24.30	17.71
	10	-3.15	-0.76	2.39	-0.85	-0.14	0.72	-0.20	-0.24	-0.04	0.63	**-1.01**	-1.64	33.22	**20.49**	-12.73	24.11	**21.37**	-2.74	4.61	27.27	22.66
	50	-3.43	-2.87	0.56	-2.71	-3.07	-0.36	-2.48			-0.70	-2.35	-1.65	18.92	15.75	-3.17	19.79	14.56	-5.24	7.19	22.21	15.01
	100	-4.75	-5.34	-0.58	-5.05	-5.29	-0.24	-4.95			0.34	-4.58	-4.92	-3.12	-2.89	0.23	-3.39	-3.76	-0.37	10.36	-3.26	-13.62
	200	-2.59	-3.44	-0.84	-3.60	-3.81	-0.21	-3.11			-0.92	-2.99	-2.07	-2.45	-2.60	-0.15	-0.74	-3.64	-2.91	2.71	-1.96	-4.67
MEND	1	0.86	**0.72**	-0.14	0.07	-0.69	-0.76	-0.15	-0.37	-0.22	1.66	**-0.07**	-1.73	1.29			-0.11			1.51	0.29	-1.22
	10	-0.45	-1.26	-0.81	-0.66	-1.60	-0.95	-1.34	-1.77	-0.43	1.73	**-0.36**	-2.08	0.41	**1.65**	1.24	2.80	**0.70**	-2.10	8.87	3.79	-5.07
	~~50~~	360.75	493.50	~~132.75~~	427.41	450.42	~~23.01~~	549.66			137.39	455.15	~~317.75~~	89.14	76.42	~~-12.72~~	71.07	70.72	~~-0.36~~	45.04	75.22	~~30.19~~
	~~100~~	305.89	351.72	~~45.83~~	362.27	229.10	~~-133.17~~	338.70			134.81	407.41	~~272.61~~	315.42	285.40	~~-30.02~~	296.70	248.77	~~-47.93~~	108.53	332.79	~~224.26~~
	~~200~~	361.28	401.57	~~40.29~~	398.28	248.82	~~-149.46~~	513.83			170.13	459.61	~~289.48~~	428.39	390.63	~~-37.76~~	340.96	280.78	~~-60.17~~	150.13	442.50	~~292.37~~
ROME	1	-2.20	**1.05**	3.24	-0.23		-0.33	-0.13	**4.05**	4.18	4.69	**-0.96**	-5.65	-1.19			0.89			6.33	**-0.06**	-6.39
	10	1.88	**1.05**	-0.83	0.10	-0.48	-0.58	-0.27	**4.09**	4.36	5.75	**-0.50**	-6.25	3.55	**5.57**	2.02	4.16	**7.43**	3.27	6.73	2.00	-4.73
	~~50~~	99.09	81.91	~~-17.18~~	83.84	77.07	~~-6.77~~	78.50			64.81	90.04	~~25.22~~	921.70	980.84	~~59.14~~	1016.98	1001.62	~~-15.36~~	665.52	994.84	~~329.32~~
	~~100~~	112.31	88.60	~~-23.71~~	92.36	85.94	~~-6.41~~	84.03			65.95	99.18	~~33.23~~	524.14	572.46	~~48.33~~	465.61	570.95	~~105.34~~	244.14	458.92	~~214.78~~
	~~200~~	226.50	204.29	~~-22.21~~	201.34	197.18	~~-4.16~~	248.73			229.24	230.52	~~1.28~~	386.17	359.52	~~-26.65~~	461.55	346.48	~~-115.08~~	244.37	415.69	~~171.33~~
MEMIT	1	0.32	**0.59**	0.27	0.62	**0.48**	-0.14	0.32	**0.61**	0.28	-4.10	0.34	4.44	-2.73			-0.17			2.31	**-0.32**	-2.63
	10	-0.82	-0.22	0.60	0.41	**0.69**	0.28	0.01	-0.22	-0.23	-4.96	0.19	5.14	-0.09	**1.33**	1.42	-0.26	-0.21	0.05	2.13	**-1.47**	-3.60
	50	-0.65	-0.19	0.46	-0.75	-0.34	0.41	-0.32			2.70	**-0.79**	-3.49	-0.38	**0.85**	1.23	-0.20	-0.52	-0.32	3.26	**-1.06**	-4.31
	100	-0.87	**0.08**	0.95	-0.68	-0.12	0.56	-0.13			3.30	**-0.89**	-4.19	0.14	**1.15**	1.02	-0.42	**0.64**	1.06	2.65	**-0.79**	-3.44
	200	-0.66	**1.18**	1.83	0.34	**0.31**	-0.03	0.04			2.61	**-0.80**	-3.41	1.08	**1.54**	0.46	1.42	**0.45**	-0.97	4.05	0.86	-3.19
EREN	1	0.249	**0.353**	0.104	0.496	**1.564**	1.068	0	**1.259**	1.259	1.654	2.941	1.287	3.482			2.144			0	4.125	4.125
	10	12.248	**51.441**	39.193	3.458	**27.158**	23.700	0.843	**5.186**	4.343	0.159	1.588	1.429	19.683			14.638			4.683	2.582	-2.101
	50	25.382	**49.159**	23.777	11.490	**21.514**	10.024	12.385	5.693	-6.692	5.173	2.581	-2.592	26.395	**51.502**	25.107	13.582	**18.122**	4.540	4.891	21.481	-16.59
	100	71.491	**152.252**	80.761	114.529	**118.292**	3.763	125.713	113.965	-11.748	15.486	2.415	-13.071	59.496	**159.797**	100.301	23.853	**142.592**	118.739	124.547	119.592	-4.955
	200	125.951	**217.592**	91.641	138.569	**141.493**	2.924	135.471	111.548	-23.923	217.468	118.582	-98.886	189.491	**235.962**	46.471	115.394	**168.633**	53.239	171.502	85.348	-86.154
SERAC	1	0	0	0	0	**0**	0	0	**0**	0	0	0	0	0			0			0	0	0
	10	1.677	**2.481**	0.804	2.314	0.512	-1.802	0.487	0.193	-0.294	2.174	**2.885**	0.711	3.795			1.483			1.583	1.195	-0.388
	50	15.391	**21.460**	6.069	18.631	17.623	-1.008	14.582	13.582	-1.000	15.0284	3.991	-11.037	25.493	**25.781**	0.288	28.974	20.581	-8.393	14.592	15.639	1.047
	100	128.682	**137.952**	9.270	85.162	74.592	-10.570	136.581	55.502	-81.079	145.610	87.910	-57.700	158.542	**207.539**	48.997	124.632	64.623	-60.009	169.531	88.593	-80.938
	200	89.246	**174.642**	85.396	115.223	**115.245**	0.022	119.128	85.602	-33.526	151.963	96.942	-55.021	137.532	**185.938**	48.406	145.631	79.105	-66.526	96.325	85.203	-11.122
ICE	1	2.166	**6.353**	4.187	2.525	**1.589**	-0.936	2.588	**3.963**	1.375	232.538	2.22	-230.318	0.202			3.544			27.124	4.343	-22.781
	2	4.976	**6.325**	1.349	2.997	**2.883**	-0.114	3.239	**3.713**	0.474	47.943	2.16	-45.783	2.871			2.457			9.056	1.04	-8.016
	3	3.79	**3.476**	-0.314	1.77	**1.616**	-0.154	2.461	**4.852**	2.391	7.223	1.59	-5.633	1.138	**3.286**	2.148	3.003	**1.662**	-1.342	0.453	1.753	1.3
	5	1.171	**2.221**	1.05	0.762	**1.738**	0.976	2.522	**2.478**	-0.044	10.261	0.989	-9.272	4.447	**6.518**	2.071	4.413	**5.047**	0.634	11.791	2.425	-9.366
	8	3.491	**3.238**	-0.253	1.329	**2.18**	0.851	8.009	**4.195**	-3.814	7.224	4.694	-2.53	2.118	**5.011**	2.893	3.096	**2.857**	-0.238	3.297	1.866	-1.431
	10	2.741	**12.489**	9.748	1.279	**2.611**	1.332	8.95	**3.577**	-5.373	5.371	1.214	-4.157	4.408	**6.058**	1.65	4.684	**5.756**	1.072	5.166	3.144	-2.022
SIR_top5	1	-0.067	**2.863**	2.93	-0.499	-0.94	-0.441	-0.054	-0.869	-0.815	3.261	**-1.399**	-4.66	-3.631			-0.213			2.617	**-0.35**	-2.967
	10	-0.862	**2.349**	3.211	-0.518	**-0.501**	0.017	-0.009	-1.021	-1.012	2.611	**-1.34**	-3.951	-0.318	**0.905**	1.223	-0.451	**0.157**	0.608	1.811	**-1.539**	-3.350
	50	-0.322	-0.916	-0.594	-0.727	-0.073	0.654	-0.083			2.511	**-1.418**	-3.929	-0.634	**0.682**	1.316	-0.239	-0.699	-0.460	2.064	**-1.098**	-3.162
	100	-0.982	**0.828**	1.81	-0.586	-0.092	0.494	-0.087			2.796	**-1.343**	-4.139	0.066	1.132	1.066	-0.154	-0.522	-0.368	1.968	**-0.739**	-2.707
	200	-0.804	**0.979**	1.783	0.331	**0.544**	0.213	0.036			2.219	**-0.815**	-3.034	0.318	**1.417**	1.099	-0.329	-0.527	-0.198	1.241	**-0.476**	-1.717
SIR_top10	1	-0.106	**2.148**	2.254	-0.706	-0.933	-0.227	-0.21	-0.61	-0.400	2.544	**-1.394**	-3.938	-1.365			-0.117			2.048	**-0.384**	-2.432
	10	-0.798	**2.473**	3.271	-0.565	-0.51	0.055	0.309	-0.841	-1.151	3.168	**-1.226**	-4.394	-0.571	**0.829**	1.4	-0.496	-0.017	0.479	1.563	**-1.549**	-3.112
	50	-0.406	**0.578**	0.984	-0.939	-0.065	0.874	-0.222			1.29	**-1.552**	-2.842	-0.57	**0.565**	1.135	-0.281	-1.216	-0.935	2.489	**-0.985**	-3.474
	100	-0.703	**0.741**	1.444	-0.705	-0.284	0.421	-0.133			1.47	**-1.304**	-2.774	-0.078	**0.876**	0.954	-0.084	**0.012**	0.096	1.404	**-0.769**	-2.173
	200	-0.838	**0.947**	1.785	0.339	**0.481**	0.142	0.1911			1.772	**-0.728**	-2.5	0.234	**1.421**	1.187	-0.054	-0.31	-0.256	1.528	**-0.575**	-2.103

Table 5: Comparative analysis of perplexity changes. The first row categorizes the distribution of edits, and the second row indicates the distances between affected and edited triplets, with "inf" signifying no connectivity. "Vanilla" denotes the change in perplexity on the vanilla knowledge graph before and after edits, whereas "GIE" signifies the change in perplexity following the application of GIE. The "Diff" column is obtained by subtracting "Vanilla" from "GIE". Editing methods are specified in the leftmost column, while the adjacent column enumerates the number of edits applied. The slashed values indicate the method's inability to accommodate the quantity of edits. Underlined values signify that the ripple effect in hidden space is more obvious than the other two variants. Bolded values indicate the presence of a ripple effect in hidden space, which is successfully discerned via GIE.

periments with edit amounts set to $[1, 2, 3, 5, 8, 10]$.

SIR re-edit the topK outliers. We use MEMIT to perform re-editing. All hyperparameters follow the same default settings with MEMIT. We conducted experiments with K set to $[5, 10]$.

A.5 Other metrics

We performed experiments utilizing alternative metrics. Fig. 6 shows the detailed results. This set of bar graphs presents results across two different sampling strategies: Breadth-First Search (BFS) and Random sampling. Within each graph, model editing methods are compared. The bars are grouped by the number of edits, ranging from 1 to 200, with each group color-coded for clarity. The height of the bars corresponds to the metric's value on a logarithmic scale. In the PPL graphs, the horizontal line represents the average PPL of the dataset before model editing. In the computation of BLEU and ROUGE metrics, the text generated by the post-edit model is employed as the Predictions. In contrast, the text generated by the original model serves as the Reference. It facilitates a comparative analysis of the discrepancies between the pre-edit and post-edit outputs. After evaluating these metrics comparatively, we have selected PPL as the metric of choice for our experiment.

A.6 License

In the course of developing the methodologies and implementations detailed within this study, we have incorporated codes that are distributed under the terms of the MIT License [1]. It significantly bolstered our research, enabling us to focus on the novel contributions of our work without the necessity of developing foundational components from scratch. We extend our profound gratitude to the original authors for their invaluable contributions to the open-source community and affirm our com-

[1] https://github.com/kmeng01/memit

Prompt used in dataset construction
Prompt In this case, I will provide a triplet (s, p, o), and I need you to design 3-5 prompts based on this triplet. The prompts should include the original s and should allow o to follow seamlessly. For example, if I give the triplet {'s': 'White House', 'p': 'architectural style', 'o': 'Neoclassical architecture'}, your answer should be in JSON format like {'s': 'White House', 'p': 'architectural style', 'o': 'Neoclassical architecture', 'prompt': ['White House is designed in the architectural style of ', 'The White House showcases the distinctive architectural style of ', 'When discussing the architectural style of the White House, one immediately thinks of']}. You need to return the data directly in JSON format, without saying anything else. This time, the triplet I provide is {'s':'','p':'','o':''}.

Example Triplet
{
"s": "Washington, D.C.",
"p": "shares border with",
"o": "Virginia"
}

Response { "s": "Washington, D.C.", "p": "shares border with", "o": "Virginia", "prompt": ["Washington, D.C. is known for sharing its border with ", "A key geographical feature of Washington, D.C. is its border with ", "Discussing the borders of Washington, D.C., one commonly mentions its adjacency to ", "An important aspect of Washington, D.C.'s location is its shared border with ", "In the context of regional boundaries, Washington, D.C. is notably adjacent to "] }

Table 6: Example of prompt generation based on a given triplet for dataset construction.

Prompt used for ICE
Prompt In this case, I will give you a json, please help me to output it in subjunctive mood. For example: given {"prompt": "{} is a relative of ", "subject": "Donald Trump", "target": "Glenn D'Hollander"}. You need to output "Imagine that Glenn D'Hollander would have been a relative of Donald Trump." This time, the json I provide is {"prompt": "", "subject": "", "target": } .

Example JSON
{
"prompt": "{} held the position of ",
"subject": "Donald Trump",
"target": "president of the Constitutional Court of Spain"
}

Response Imagine that Donald Trump had held the position of president of the Constitutional Court of Spain.

Table 7: Example of prefix prompt generation for ICE.

mitment to adhering to the stipulations of the MIT License.

Figure 6: Perplexity, Bleu and Rouge score.