The Missing Piece in Model Editing: A Deep Dive into the Hidden Damage Brought By Model Editing

Anonymous ACL submission

Abstract

Large Language Models have revolutionized numerous tasks with their remarkable efficacy. However, the editing of these models, crucial for rectifying outdated or erroneous information, often leads to a complex issue known as the ripple effect in the hidden space. This effect, while difficult to detect, 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 Outlier Relation based Assessment (GORA), which quantitatively evaluates the adaptations of the model and the subsequent impact of editing. Furthermore, we introduce the Selective Outlier Re-Editing Approach (SORA), 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, GORA and SORA, effectively identify and alleviate this issue, respectively, contributing to the advancement of LLM editing techniques.

1 Introduction

The swift advancement of Large Language Models (LLMs) has exhibited remarkable efficacy across a multitude of tasks(Brown et al., 2020; Zhao et al., 2023; OpenAI, 2023; Touvron et al., 2023; Gu et al., 2023). Nevertheless, the data embedded within these expansive models may become outdated or encompass errors(Lazaridou et al., 2021; Dhingra et al., 2022; Jang et al., 2022). As a result, the edit of outdated and erroneous information within these models has emerged as an important research subject.

In recent years, methodologies for editing LLMs have progressively garnered attention (Zhu et al., 2020; De Cao et al., 2021; Meng et al., 2022, 2023; Si et al., 2023). The primary objective of these methodologies is to enhance the output of LLMs in specific domains without undermining their performance in other sectors. This is a delicate balancing act, as the editing process must ensure both the success of the edits and the avoidance of any negative impact on the overall functionality of the model.

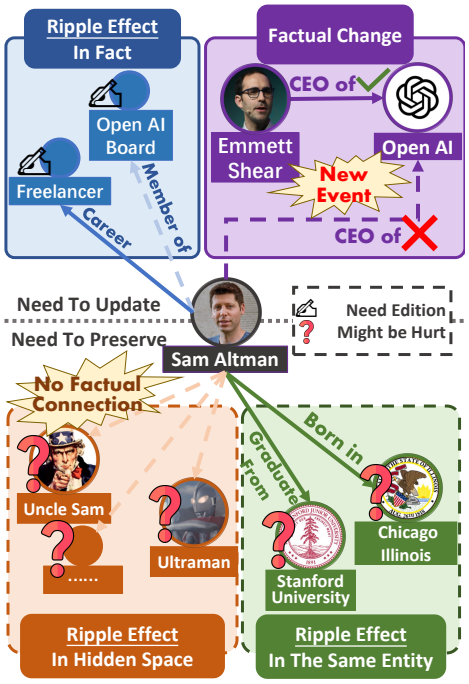

Figure 1: Illustrating the ripple effect in model modification using the OpenAI board controversy as an example. The model can only receive limited modification requests when a new event occurs, such as a CEO change at OpenAI. The ripple effect of model editing involves positive changes in other facts, like Sam will be a freelancer, due to this edition. But it also involves unforeseen disruptions, potentially damaging unrelated facts under the same entity and distorting the facts of other related entities within the model's latent space.

Although many model editing techniques have proven effective in various situations, a significant issue is their tendency to prioritize improving editing performance without considering other factors. While achieving successful edits is not the most challenging aspect, studies have shown that model editing deal damage to the general abil-

ities of LLMs (Gu et al., 2024). In our study, a more complex issue lies in controlling the impact of knowledge editing on the hidden space of the model, also known as the ripple effect (Yao et al., 2023; Cohen et al., 2023; Li et al., 2023c; Sakarvadia et al., 2023).

As illustrated in Fig. 1, the ripple effect brought about by knowledge editing can be divided into two categories. The first category is the positive ripple effect, such as the update of other facts related to the edited fact, which is called "Ripple Effect in Facts" (Cohen et al., 2023). This effect can be beneficial as it ensures the consistency and coherence of related facts within the model. The second category is the negative ripple effect. This includes the potential damage to the model's memory of other information about an entity after changing the model's understanding of that entity, known as the "Ripple Effect in the Same Entity"(Li et al., 2023b; Yao et al., 2023). Additionally, changing the model's memory of an entity in a hidden space may also affect entities that are close in the hidden space, referred to as the "Ripple Effect in Hidden Space"(Hoelscher-Obermaier et al., 2023a; Sakarvadia et al., 2023).

Both the Ripple Effect in Fact and in The Same Entity can be easily detected for they have the actual factual connection between the edited entities and affected attributes or relations. However, the Ripple Effect in Hidden Space, which lacks a direct factual correlation with the edited object, presents a significant challenge in detection. This implicit influence on other entities severely impedes the efficacy of all Model Editing tasks, culminating in a drastic deterioration in model performance as the quantity of edits escalates(Li et al., 2023b; Wang et al., 2023). Consequently, the development of efficacious strategies and techniques to alleviate this type of ripple effects is of paramount importance.

In this study, we first propose an evaluation methodology, referred to as Graphical Outlier Relation based Assessment (GORA), which integrates quantitative evaluations to scrutinize the adaptations of the model and the subsequent impact of editing. We build up the connection between related entities in hidden space based on the model to be edited on the given knowledge graph. By evaluating the model's performance in generating text from the edited model on edited triplets and hidden space related triplets on the manipulated graph, the ripple effect can be comprehended in the hidden space induced by the model editing method. Furthermore, through the iterative establishment of hidden connections between entities, GORA facilitates a proportionality between the influence of the hidden space ripple effect and the distance to the edited triplets. Consequently, this allows for the prediction of the hidden space ripple effect.

Additionally, we present a Selective Outlier Re-Editing Approach (SORA) predicated on our evaluation method, which is engineered to mitigate the Hidden Space Related Ripple Effect. By identifying the entities that possess a relationship with the edited entities, it is feasible to alleviate the ripple effect by simply incorporating the related triplets with the edited fact into training. However, such a method will result in excessive computational expense, as each edit will necessitate editing a larger number of related knowledge. Not all related knowledge needs to be edited. SORA uses the above strategy by identifying key triplets and editing these knowledge to enhance editing efficiency and reduce computational overhead.

In the experimental phase, GORA discovered that even the state-of-the-art (SOTA) model editing method still encounters significant challenges with the ripple effect in the hidden space. Compared to directly detecting the ripple effect on the artificial KG, the result of GORA reduced by 16.51% for the SOTA model editing method, indicating that the ripple effect in the hidden space causes greater disruption than the ripple effect in the same entity, thereby validating the feasibility of GORA in verifying the ripple effect in the hidden space. SORA mitigates such issues by reducing the average rate of the SOTA model editing method in the ripple effect in the hidden space by 54.75%.

In summary, this paper has made the following contributions:

- This research is pioneering in exploring the ripple effect in the hidden space, a detrimental yet implicit phenomenon in model editing.
- We have introduced GORA, a specialized technique for evaluating the ripple effect in the hidden space during the process of model editing.
- Leveraging the fundamental design of GORA, we have developed SORA, an innovative model editing method that employs existing explicit knowledge bases to mitigate the ripple effect in the hidden space.
- We carry out comprehensive evaluations and comparative experiments, which demonstrate

that GORA effectively identifies the ripple effect in the hidden space during model editing than other works. We have also discovered that this effect is present in all current model editing methods, while SORA effectively alleviates this issue compared to other model editing techniques.

2 Related Work

2.1 Knowledge Editing

In the evolving field of LLMs, Knowledge Model Editing methods have been developed to integrate new knowledge while preserving existing information. These methods are broadly categorized into three types(Wang et al., 2023):

External Memorization-based Methods: Utilize separate memory modules to store new knowledge, thus keeping the original model's weights unchanged. This method is scalable and allows for the extension of knowledge without restructuring the pre-trained model(Li et al., 2022; Madaan et al., 2022; Mitchell et al., 2022b; Murty et al., 2022).

Global Optimization-based Methods: Implement widespread model updates guided by new knowledge. These methods modify the LLMs in a controlled manner but may be resource-intensive due to the large parameter space(Sinitsin et al., 2019; De Cao et al., 2021; Hase et al., 2021; Mitchell et al., 2022a).

Local Modification-based Methods: Target specific parameters for updates, offering a focused and resource-efficient approach to incorporating new knowledge into LLMs(Dai et al., 2022; Li et al., 2023a; Meng et al., 2022, 2023).

In our study, we primarily focus on Global Optimization-based Methods and Local Modification-based Methods, both of which involve updating the model. We also experiments with latest method ICE (Cohen et al., 2023) We aim to address the challenges associated with these methods, particularly the ripple effect in the hidden space, which has been largely overlooked in previous research.

2.2 Evaluating Knowledge Editing

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 serves as a question-answering dataset designed for relation-specific queries and 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 by introduces counterfactual edits. RippleEdits (Cohen et al., 2023) is a benchmark evaluating the "ripple effects" in knowledge editing. To be specific, one should go beyond the single fact that was edited and check that other facts that are 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.

Our research primarily focuses on these unwanted side effects, a topic that has not been thoroughly explored in previous studies. Unlike other evaluations that mainly concentrate on the overall impacts of model editing, such as the "Ripple Effect in Facts" and "Ripple Effect in the Same Entity", our approach aims at the detailed evaluation of the "Ripple Effect in Hidden Space". We study how knowledge graphs can help reveal the extent of side effects and differences in knowledge distribution between models and human understanding. Our work significantly adds to the understanding of how model editing can cause hidden harm to other knowledge within the model.

3 Preliminary

Factual Change is a pivotal concept in model editing. Facts are understood as natural language sentences and represented as multi-dimensional vectors within a latent space. Given a fact set F and and a corresponding set of changes $\Delta F(|\Delta F| \leq |F|)$, the post-change fact set is expressed as

$$F' = F + \Delta F + R(\Delta F) \quad (1)$$

where $R(\Delta F)$ signifies the ripple effect induced by ΔF.

In natural language, the ripple effect is an observable phenomenon within knowledge graphs, characterized by the spread of a single fact alteration throughout the interconnected node network. This process leads to subsequent changes in various nodes, underlining the interconnected nature of factual information.

In LLMs, the ripple effect exhibits a more intricate nature and can be delineated into three distinct categories, each representing a unique pathway of influence in the model's response to factual changes:

Ripple Effect in Fact R_F: This refers to the process where changes in one fact lead to modifications in related facts, as illustrated in Fig.1. When change the CEO of Open AI from Sam Altman to Emmett Shear, there are facts need to update like "Sam Alterman is not a member of OpenAI Board" and "Sam Altman'career is Freelancer".

Ripple Effect in The Same Entity R_E: when a factual change alters some aspects of an entity's information, other unrelated aspects should ideally remain constant.As shown in Fig.1, despite a change in career, the entity's birthplace or educational background should remain unaltered. Current methods of model editing often demonstrate a heightened sensitivity to subjects, inadvertently leading to these undesired modifications. This phenomenon, where changes in one aspect of an entity affect other static aspects, is what we define as the Ripple Effect in The Same Entity.

Ripple Effect in Hidden Space R_H: Considering the black box nature of LLM, special attention must be paid to the Ripple Effect in Hidden Space. This effect reflects how changes in one fact can lead to unexpected changes in different, unrelated facts and entities. This occurs because of the similarity between different subjects in hidden space. When we update the parameters, this can unintentionally affect model's performance on other facts.

The aggregate ripple effect R is computed as: $R = R_F + R_E + R_H$, capturing the comprehensive impact of fact changes across different dimensions.

4 Our Method

The efficacy of updating knowledge within language models hinges on judicious evaluation and editing methods. Our methodology unfolds in two sub-sections, each tailored to systematically advance the accuracy and relevance of the model's knowledge base.

4.1 Graphical Outlier Relation based Assessment (GORA)

Our evaluation method incorporates quantitative assessments to examine the model's adaptations and the consequent impact of editing. We also established knowledge connections within the inner space of the model and conducted a comparative analysis with KG. We analyze graphical representations to compare the model's internal structure with the vanilla knowledge graph.

First, we identify outlier triplets after model editing. We observe that the outputs adhere to a long-tail distribution. Consequently, outliers are defined as change in evaluation metric surpassing a threshold of $\delta > \mu + 2\sigma$, where δ represents the change in evaluation metric before and after editing, μ denotes the mean, and σ signifies the standard deviation. This definition is applicable across various evaluation metrics and editing methods.

Subsequently, we regard outliers and edited nodes as proximal in distance, constructing a GORA graph based on this proximity. GORA graph has same nodes with vanilla KG. A specific number of edit requests are randomly selected across KG. Each selected request undergoes model editing. Outliers are then determined using the aforementioned inequation. We build an edge between the identified outliers and their corresponding edited triplet due to their closeness in the latent space. The construction of the GORA graph is iterated multiple times to ensure that the number of edges in the GORA graph roughly equals to the vanilla KG.

Additionally, we analyze the model's performance in relation to the distance between edited and tested triplets within the both GORA graph and vanilla KG, where distributions of edited nodes and quantity of edits are also considered.

4.2 Selective Outlier Re-Editing Approach (SORA)

SORA is devised to refine the model's internal representation of knowledge without compromising the integrity of its pre-trained state by identifying and re-editing outliers. This process imitates the repetitive nature of human learning to reinforce the model's understanding of new information.

For a triplet (s, r, o), the editing effect can be quantified by measuring the change in the evaluation metric, which is computed by the post-edit model f_{θ_e} and pre-edit model f_θ:

$$E = \text{Metric}[f_{\theta_e}(\langle s, r\rangle)] - \text{Metric}[f_\theta(\langle s, r\rangle)] \quad (2)$$

where $\langle s, r\rangle$ is the prompt describing s and r. The identification of outliers (edit targets) is determined by selecting the top-K triplets based on their editing effects.

Similar to MEMIT (Meng et al., 2023), for edit targets $\xi = \{(s_i, r_i, o_i)\}$, given a set of factual prompts$\{x_j \oplus p(s_i, r_i)\}$ that concatenate random prefixes x_j to a templated prompt, the target $z_i =$

$h_i^L + \delta_i$ vectors for every edits i is computed:

$$\arg\min_{\delta_i} \frac{1}{P}\sum_{j=1}^{P} -\log \mathbb{P}_{f_\theta(h_i^L)}\left[o_i | x_j \oplus p(s_i, r_i)\right] \tag{3}$$

For critical MLP layers $l \in \mathcal{R}$, the update is performed as follow:

$$k_i^l = \frac{1}{P}\sum_{j=1}^{P} \sigma(W_{in}^l \gamma(h_i^{l-1}(x_j + s_i))) \tag{4}$$

$$r_i^l = \frac{z_i - h_i^L}{L - l + 1} \tag{5}$$

$$K^l \leftarrow [k_i^{l_1}, \ldots, k_i^L], R^l \leftarrow [r_i^{l_1}, \ldots, r_i^L] \tag{6}$$

$$\Delta^l = R^l K^{l^T} (C^l + K^l k^{l^T})^{-1} \tag{7}$$

$$W^l \leftarrow W^l + \Delta^l \tag{8}$$

where γ is layernorm and C^l is covariance of the pre-existing keys.

This approach ensures the model's consistent performance amidst the dynamic landscape of evolving knowledge. Through the integration of our evaluation and editing strategies, we guarantee that the edited model both retains the core of the previously established knowledge and incorporates new insights.

5 Experiments

The experiments are designed to incrementally address two research questions: *1)* Is there a method to identify and more accurately depict the "ripple effect in hidden space"? *2)* Can "ripple effect in hidden space" be efficiently mitigated?

5.1 Baselines

In terms of evaluation methods for model editing, we primarily compared two approaches:

Vanilla We use the KG extracted from wikidata5m (Wang et al., 2021) to generate edit requests. Subsequent tests are conducted on the neighbors of edited nodes to analyze the ripple effects caused by model editing. These ripple effects are mainly "ripple effect in fact" and "ripple effect in the same entity".

GORA represents our proposed methodology. Utilizing the model's representation of each triplet, we construct the GORA graph to illustrate the relationships within the hidden space. We then use GORA graph to evaluate the ripple effect induced by model editing.

COUNTERFACT (Meng et al., 2022) and zsRE (Levy et al., 2017) stand as the most frequently utilized benchmark in the domain of model editing. RIPPLEEDITS (Cohen et al., 2023) is a benchmark raised for evaluating the first two kinds of ripple effect. However, these benchmarks have been subject to testing within limited scopes, thereby neglecting the broader potential implications. This limitation inhibits the execution of analyses that are both more comprehensive and deeper in nature. Consequently, we have developed our own dataset to address these shortcomings.

5.2 Evaluation Dataset Construction

Step 1: Factual Triplets Collection Our data extraction process utilized Wikidata5m (Wang et al., 2021), a dataset comprising over 4.5 million entities and 20 million triplets. To manage the extensive volume of data, we implemented Breadth-First Search (BFS) sampling to derive a representative subgraph, containing approximately 10^4 triplets. This selected subset serves as our primary dataset, offering a rich diversity of factual information.

Step 2: Prompt Generation Utilizing GPT4, we automate the generation of natural language prompts for each triplet. These prompts undergo quality assurance checks for fluency by human and alignment with their respective triplets.

Step 3: Edit Target Selection We identify and select modifiable elements within the triplets. For a triplet $\{s, r, o\}$, we select the edit target o' in the set of triplets that share the same relation r but differ in object o. To be specific, $\mathbb{T} = \{o' | r' = r, o' \neq o\}$.

5.3 Model editing methods

As for the model editing methods, we primarily compared following baselines in the experimental phase:

Fine-tuning (FT) The model's parameters in a specific layer are updated using gradient 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 compute and insert key-value pair 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, aimed at avoiding unintended impacts of inadvertently influence on edited layers when editing layers.

In-context Editing (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>". For example, "Imagine that Bill Clinton would have been the father of Barack Obama".

SORA represents our proposed methodology. SORA incorporates identifying and re-editing outliers for more effective model editing.

Additional implementation details are offered in Appendix A.3

5.4 Metric

We employ perplexity as the main metric to measure the model's performance in generating text. Perplexity is one of the most common metrics for evaluating language models and quantifies how well a probability model predicts a sample. Perplexity is defined as the exponentiated average negative log-likelihood of a sequence. If we have a tokenized sequence $X = (x_0, x_1, \ldots, x_t)$, then the perplexity of X is,

$$PPL\left(X\right) = \exp\left\{-\frac{1}{t}\sum_{i}^{t}\log p_\theta\left(x_i \mid x_{<i}\right)\right\} \tag{9}$$

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.

In GORA, Perplexity serves as an indicator of model stability because it is sensitive to shifts in the probability distribution. We mainly focus on difference before and after editing rather than the single value. Additional experiments utilizing alternative metrics are documented in Appendix A.4.

5.5 Distance calculation

We calculate the distance between outlier and edited triplets both on GORA graph and vanilla KG. The Dijkstra algorithm is utilized to identify the shortest path. For every triplet in the dataset, we locate the closest edited triplets and calculate the distance between them. When handling multiple edit requests, the algorithm is executed from each edited triplet, with the selection of the shortest path leading to the nearest edited triplet.

5.6 Overall Ripple Effects Evaluation

The evaluation results, as shown in Tab. 1, suggest that model performance is affected by both the editing strategy employed and the characteristic of the edits. Significantly, the performance of ROME and MEND declines considerably when the number of edits exceeds 50. Although **FT+L** appears stable in Tab. 1, it is not an effective approach. Its updating mechanism restricts weight adjustments, obstructing the efficient update of parameters and the creation of meaningful sentences, as evidenced in Tab. 3.

Moreover, the experiment examines the impact of distance between edited and tested triplets. From Tab. 1, it can be deduced that proximity on vanilla KG does not always result in a greater ripple effect, challenging the inherent assumption that closer nodes are necessarily more affected by editing. There is no consistent correlation between distance on vanilla KG and decreased performance. Both proximate and distant triplets display vulnerability to changes following model editing.

The objective of the GORA graph is to minimize the distance between triplets affected by "ripple effect in hidden space" and the edited triplets, while simultaneously increasing the distance between unaffected triplets and the edited ones. As a result, within the GORA framework, triplets located in closer proximity are anticipated to exhibit an increase in perplexity, while nodes with no connectivity should show a decrease in perplexity relative to the vanilla knowledge graph. The bolded and numbers in Tab. 1show the effectiveness of GORA.

Furthermore, we conduct a comparison across all three types of ripple effects. The outcomes attributed to GORA correspond to the "ripple effect in hidden space," whereas Vanilla's results predominantly encompass the "ripple effect in fact" and the "ripple effect in the same entity." The underlined figure in Tab. 1 highlights that the "ripple effect in hidden space" typically exerts a greater influence compared to the other two variants.

5.7 Difference between KG and GORA graph

Graph edit distance (GED) serves as a metric to gauge the similarity between two graphs.

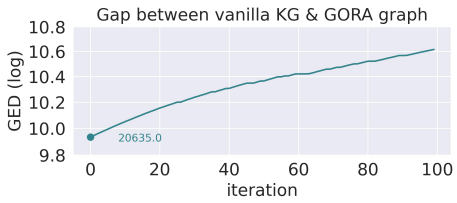

Figure 2: shows GED's change, with the x-axis representing the iterations of building GORA graph.

As shown in Fig. 2, we compute a simplified version of GED between GORA graph and vanilla

		BFS												Random								
		1			2			3			inf			1			2			inf		
Methods	#Edition	Vanilla	GORA	Diff	Vanilla	GORA	Diff	Vanilla	GORA	Diff	Vanilla	GORA	Diff	Vanilla	GORA	Diff	Vanilla	GORA	Diff	Vanilla	GORA	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
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
SORA_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
SORA_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 1: 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 "GORA" signifies the change in perplexity following the application of GORA. The "Diff" column is obtained by subtracting "Vanilla" from "GORA". Editing methods are specified in the leftmost column, while the adjacent column enumerates the number of edits applied. Values that are slashed through indicate the method's inability to accommodate the quantity of edits. Underlined values signify ripple effect in hidden space is more obvious than the other two variants. Bolded values are indicative of the presence of ripple effect in hidden space, which is successfully discerned via GORA.

KG, using $\mathcal{L}_1 - norm$:

$$\text{GED} = \log\left(\left\|\mathbf{G}_{\text{adj}} - \mathbf{G'}_{\text{adj}}\right\|_1\right) \tag{10}$$

where $\mathbf{G}_{\text{adj}}$ and $\mathbf{G'}_{\text{adj}}$ denotes the adjacency matrix of vanilla KG and GORA graph.

In this specific instance, we use MEMIT to build GORA graph. The iteration is carried out 100 times to maintain consistency in scale and structure between the GORA graph and vanilla KG. They both graph have approximately 10^4 edges.

GORA graph and vanilla KG are similiar in density but different in detail.

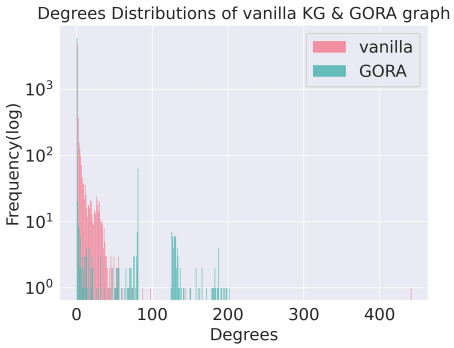

Figure 3: This figure presents the frequency of node degrees within the vanilla knowledge graph and GORA graph.

We have quantified the degree distributions of both graphs, as shown in Fig. 3. The vanilla KG displays a skew towards lower degrees, indicating a denser core. Conversely, the GORA graph exhibits a more uniform degree distribution. This supports the previous finding that there is no direct correlation between the distance in a vanilla KG and reduced performance. Furthermore, GORA offers a superior representation, more effectively highlighting the "ripple effect in hidden space" compared to the vanilla KG.

5.8 SORA for Enhanced Robustness

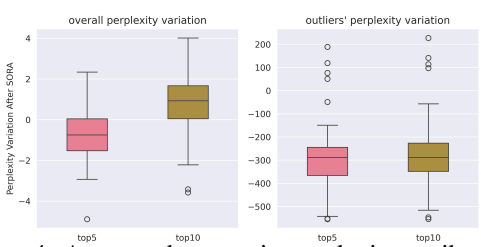

Figure 4: Average decrease in perplexity attributed to SORA. The left panel shows the overall perplexity's change, while the right panel shows the decrease in perplexity for the outliers.

We assess the effectiveness of SORA by com-

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. ✓
SORA(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) ✗
SORA(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 2: Case Study of text Generated by GPT2-XL with and without SORA implementation.

paring a control group, subjected to factual edits, against a treatment group that receives the same edits plus re-edits of the top-K outliers.

Results shown in Fig. 4 reveal that re-editing the top-5 outliers leads to a notable decrease in overall perplexity, especially for the outliers. However, extending the approach to the top-10 outliers slightly increases overall perplexity due to complications from numerous edits, despite a continued decrease in outliers' perplexity. This illustrates SORA's effectiveness, albeit with a caveat: a moderate number of re-edits improves model robustness, whereas excessive edits may introduce instability.

These insights guide our model editing strategy, emphasizing the importance of balancing between adequate revision and the risk of over-editing.

6 Case Study

In Tab. 2, we examine the alterations in text generated by GPT2-XL in response to an edit request. The sentences provided are among the top 10 most significantly affected outliers. Initially, the model is capable of producing accurate and coherent content. However, post-editing, a subset of the outputs includes some samples that are incorrect or nonsensical. These are identified as outliers by GORA.

Samples Generated by Crashed Model
FT(50 edits)
The given name of Elizabeth Christ is name of Christopher 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 . . . Geographically, Turkey stands out for sharing its border with ((((((((((1 I IIIII. . . 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 with Noinia the remotely Avg Medalinia F ó 4 crank Tat . . . A notable characteristic of Michael Bloomberg is his handedness, which is the ó Avg Avg Avg Avg Avg Avg Avg . . .

Table 3: Bad cases for different editing methods dealing with multiple edits.

After employing SORA, the model can revert to generating accurate results. The third fact was not among the top5 outliers and thus was not re-edited in SORA(top5) and the model keeps the same outputs.

In Tab. 3, when dealing with multiple edits, these four methods lead the model to severely crash. The model fails to produce coherent sentences and generates repetitive word patterns, making quantitative assessment impractical. Consequently, we slashed out the data in Tab. 1.

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 the editing of LLMs. We have proposed an innovative evaluation methodology, Graphical Outlier Relation-based Assessment (GORA), which effectively identifies the ripple effect in the hidden space during model editing. Furthermore, we have developed a novel model editing method, Selective Outlier Re-Editing Approach (SORA), which leverages the design of GORA to mitigate the ripple effect in the hidden space. Our comprehensive evaluations and comparative experiments have demonstrated the effectiveness of both GORA and SORA. 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 knowledge graph. Owing to the large scale of knowledge graph, this process is both time-intensive and demands substantial computational resources.

Dependence on Knowledge Graphs Our methodology is reliant on knowledge graphs. Yet, ensuring the quality of these graphs proves to be a complex task. The evaluation of knowledge graph in practical scenarios presents many challenges.

Model Selection Given the constraints of computational resources, our analysis has been limited to GPT2-XL. The effectiveness of our method for models of varying sizes and architectures is an aspect that 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.

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

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.

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.

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.

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*.

Zichao Li, Ines Arous, Siva Reddy, and Jackie Chi Kit Cheung. 2023c. Evaluating dependencies in fact editing for language models: Specificity and implication awareness. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7623–7636.

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.

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, Jianfeng 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.

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 Prompt

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

In the construction of our dataset, we utilize GPT4 to generate prompts that integrate specific subjects with their corresponding predicates. As illustrated in Tab. 4, this method ensures the quality and fluency of our data.

We also utilize GPT4 to generate ICE prefix prompts. Tab. 5 shows an example.

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 5: Example of prefix prompt generation for ICE.

A.2 Model Selection

Due to 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 pretrained model on 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.3 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: $\|\theta_G - \theta_{G'}\|_\infty \leq \epsilon$. This 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

with respect to some subset of θ_G. Hyperparameters are adopted from given default configurations.

ROME (Meng et al., 2022) as proposed by Meng et al. 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 samples of Wikitext.

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

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. 5 for an example. Due to input length constraints, we conducted experiments with edit amounts set to $[1, 2, 3, 5, 8, 10]$.

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

A.4 Other metrics

We performed experiments utilizing alternative metrics. Fig. 5 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 post-edit model is employed as the Predictions, whereas the text generated by the original model serves as the References. This facilitates a comparative analysis of the discrepancies between the pre-edit and post-edit outputs. Following the comparative evaluation of these metrics, we have selected PPL as the metric of choice for our experiment.

A.5 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 commitment to adhering to the stipulations of the MIT License.

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

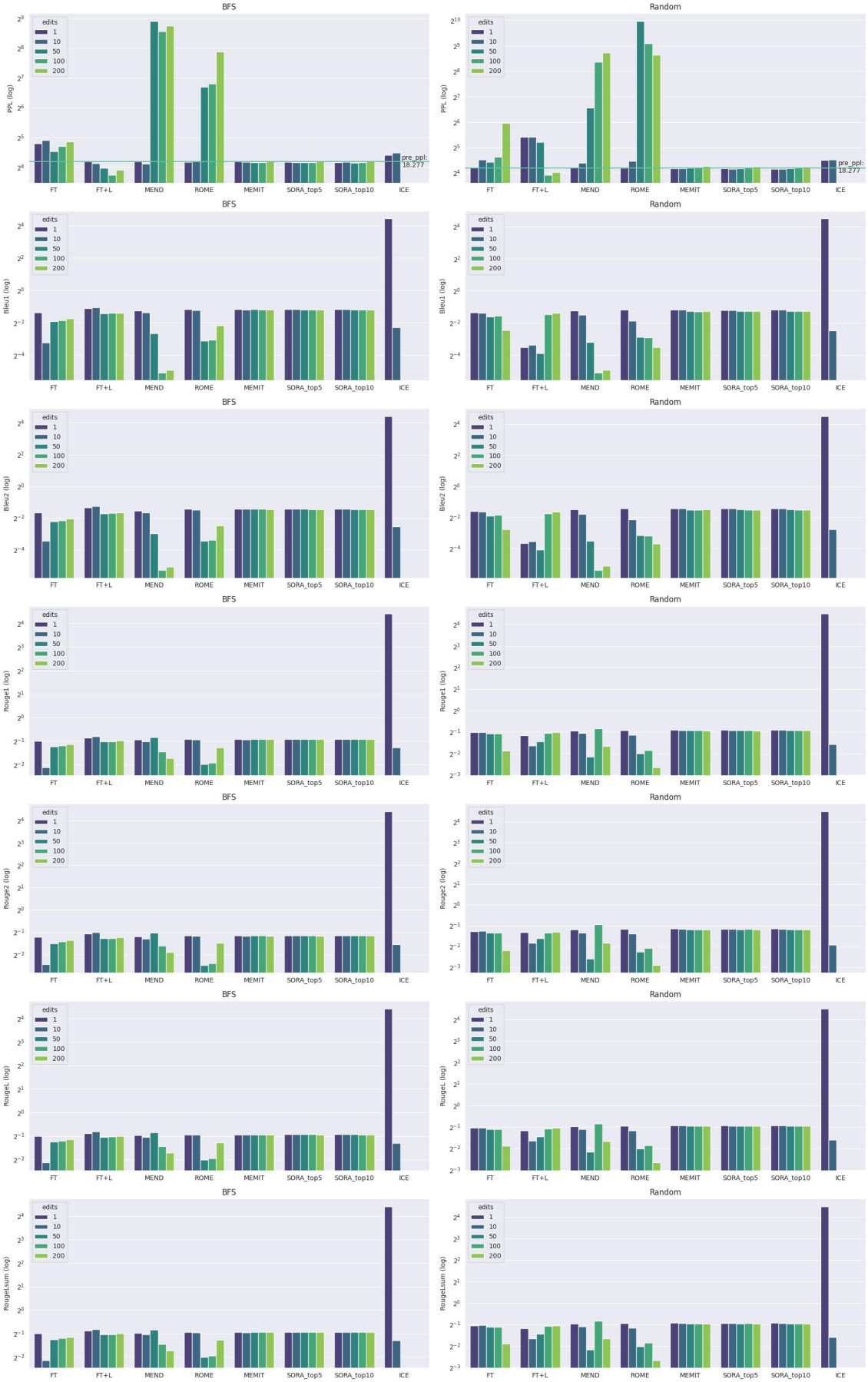

Figure 5: Perplexity, Bleu and Rouge score.