Related Knowledge Perturbation Matters: Rethinking Multiple Pieces of Knowledge Editing in Same-Subject

Zenghao Duan[1,2,*], **Wenbin Duan**[3,*], **Zhiyi Yin**[1,†], **Yinghan Shen**[1,†], **Shaoling Jing**[1], **Jie Zhang**[1], **Huawei Shen**[1], **Xueqi Cheng**[1],
[1]Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[2]University of Chinese Academy of Sciences, Beijing, China
[3]People's Public Security University of China, Beijing, China
{duanzenghao24s, yinzhiyi, Shenyinghan, jingshaoling, zhangjie, shenhuawei, cxq}@ict.ac.cn

Abstract

Knowledge editing has become a promising approach for efficiently and precisely updating knowledge embedded in large language models (LLMs). In this work, we focus on **Same-Subject Editing**, which involves modifying multiple attributes of a single entity to ensure comprehensive and consistent updates to entity-centric knowledge. Through preliminary observation, we identify a significant challenge: *Current state-of-the-art editing methods struggle when tasked with editing multiple related knowledge pieces for the same subject.* To address the lack of relevant editing data for identical subjects in traditional benchmarks, we introduce the $\mathbf{S^2}$**RKE** (**S**ame-subject **R**elated **K**nowledge **E**diting) benchmark. Our extensive experiments reveal that only mainstream locate-then-edit methods, such as ROME and MEMIT, exhibit *"related knowledge perturbation,"* where subsequent edits interfere with earlier ones. Further analysis reveals that these methods over-rely on subject information, neglecting other critical factors, resulting in reduced editing effectiveness.

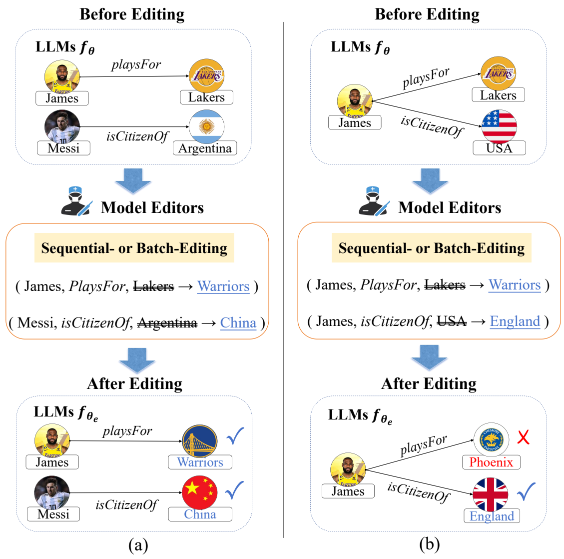


Figure 1: Comparison of performance on Different and Same-Subject Editing. (a) Editing individual knowledge pieces for distinct subjects, "James" and "Messi," results in excellent performance. (b) Editing two related knowledge pieces for the same subject, "James," leads to poor performance.

1 Introduction

The dynamic nature of real-world knowledge necessitates efficient methods for updating specific facts in large language models (LLMs) (Achiam et al., 2023; Touvron et al., 2023) without compromising their overall performance. *Knowledge editing*(a.k.a., *model editing*) (Yao et al., 2023) has emerged as a promising solution to address this challenge, enabling targeted updates to model parameters without requiring full retraining. Among existing methods, *locate-then-edit* methods, such as ROME (Meng et al., 2022a) and MEMIT (Meng et al., 2022b), have shown effectiveness in making precise modifications to Transformer layer parameters (Vaswani, 2017). However, their broader applicability across diverse editing scenarios remains insufficiently explored.

In particular, **Same-Subject Editing**, modifying multiple attributes of a single entity, plays a critical role in ensuring comprehensive and consistent updates to entity-centric knowledge. As shown in Figure 1, an entity like "James" may require simultaneous edits to attributes such as "isCitizenOf," "playsFor," and others. This process refines the entity's representation by resolving attribute conflicts and synchronizing interdependent facts. Despite its significance, same-subject editing has largely been overlooked in existing research.

Through preliminary observations, we identify an unusual failure: *Some top-performing editing methods struggle to edit multiple related knowl-*

* Equal Contributions
† Corresponding authors
Our benchmark and source code are available at: `https://github.com/Zhow01/S2RKE`

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers), pages 363–373
April 29 - May 4, 2025 ©2025 Association for Computational Linguistics

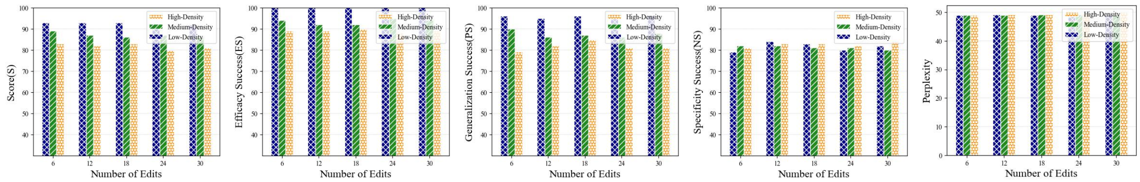


Figure 2: The results of *sequential-editing* by three different schemes on GPT-J using MEMIT, comparing five evaluation metrics. The values of Score(**S**), Efficacy Success(**ES**) and Paraphrase Success(**PS**) always decreased with the subject density, but Neighborhood Success(**NS**) and Perplexity(**PPL**) remained unchanged.

edge pieces for the same subject. As illustrated in Figure 1, model editors perform well when editing individual knowledge pieces for different subjects, such as "James" and "Messi" (Figure 1a). However, when tasked with editing two related pieces of knowledge for the same subject, "James," these editors become significantly less effective (Figure 1b). This observation raises two key questions:

- *Is this failure a common issue across different LLMs and editing methods?*
- *What causes the failure when editing multiple related knowledge pieces about same subject?*

Existing benchmarks, such as COUNTERFACT (Meng et al., 2022a), lack sufficient examples of same-subject editing, making it difficult to explore the underlying mechanisms of this failure. To address this gap, we introduce the **S^{2}RKE** (**S**ame-subject **R**elated **K**nowledge **E**diting) benchmark, which associates each subject with multiple related edits. We systematically evaluate various editing methods on LLMs of different sizes using S^{2}RKE, applying both *sequential-editing* and *batch-editing*. Surprisingly, the results show that only mainstream locate-then-edit methods, such as MEMIT (Meng et al., 2022b), fail to effectively update multiple related information for the same subject. Moreover, our in-depth analysis reveals that this failure occurs because subsequent edits interfere with previous ones, a phenomenon we term *"related knowledge perturbation."*

Furthermore, we find that locate-then-edit methods exhibiting *"related knowledge perturbation"* update the weight matrix of the MLP module by calculating key-value pairs. Specifically, the key is derived from the input of the subject's last token in the MLP module's down-sampling layer. Our experiments conclude that the perturbation arises from an over-reliance on subject information during editing. When multiple related pieces of knowledge share the same subject, the calculated keys remain highly similar. As a result, subsequent edits interfere with earlier ones, diminishing the overall effectiveness of the editing process.

In essence, our main contributions are as follows: (1) We propose the S^{2}RKE benchmark for Same-Subject Editing and highlight the issue of "*related knowledge perturbation*." (2) We demonstrate that locate-then-edit methods fail to update multiple related facts for the same subject due to an over-reliance on subject-specific information.

2 Preliminary

2.1 Knowledge Editing in LLM

Autoregressive, decoder-only large language models (LLMs) process a token sequence $x = [x_1, \ldots, x_T] \in X$, with each $x_i \in V$ drawn from a vocabulary V, and predict the probability distribution $y \in Y \subset \mathbb{R}^{|V|}$ for the next token. In the Transformer architecture, each token x_i is embedded into hidden states $h_i^{(l)}$, starting from $h_i^{(0)} = \text{emb}(x_i) + \text{pos}(i)$. The final output $y = \text{decode}(h_T^{(L)})$ is derived from the last hidden state. At each layer l, $h_i^{(l)}$ is updated via global attention $a_i^{(l)}$ and local MLP contributions $m_i^{(l)}$, with each token attending only to preceding tokens.

$$h_i^{(l)} = h_i^{(l-1)} + a_i^{(l)} + m_i^{(l)}, \tag{1}$$

$$m_i^{(l)} = W_{\text{proj}}^{(l)} \sigma \left(W_{\text{fc}}^{(l)} \gamma \left(a_i^{(l)} + h_i^{(l-1)} \right) \right), \tag{2}$$

In many previous studies, knowledge has been represented as triples (s, r, o), where s, r, and o denote subject, relation, and object respectively (e.g., James (s), playsFor (r), and Lakers (o)) (Meng et al., 2022a; Li et al., 2024a). Researchers designed natural language templates tailored to each relation type and combined these templates with subject terms to generate question-based or cloze-style prompts. Knowledge editing is formally defined as follows: the edited fact set is $e = (s, r, o)$, and the edited model is $M^* = F(M, e)$, where F is the editing methods that updates the original model M.

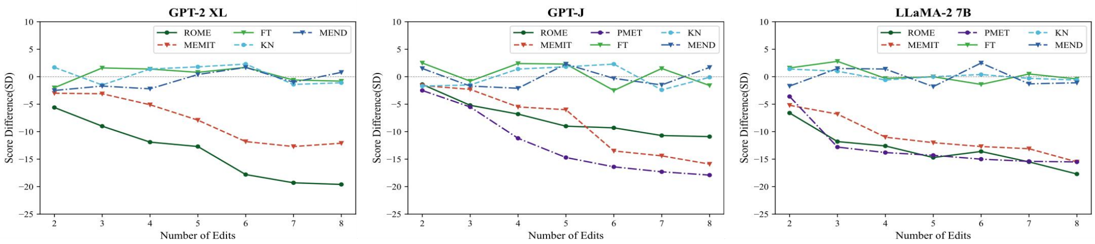


Figure 3: The results of differences in *sequential-editing* results in two scenarios on three LLMs by six editing methods. **Score Difference (SD)** represents the difference in editing performance between the two experimental schemes when editing the same amount of knowledge under the same method.

2.2 Same-Subject Editing

In a broader sense, knowledge editing should allow for querying and modifying a wide range of facts within language models by combining different subjects (s) and relations (r) as prompts.Existing work typically focuses on modifying individual facts expressed as $(s, r, o) \rightarrow (s, r, o_*)$, where each subject (s) is associated with a specific relation (r). However, traditional editing often isolates the editing process to a single relation. This leads to the discontinuation of further knowledge edits for the same subject and a shift towards editing knowledge for a new subject. It risks overlooking potential perturbations in knowledge when editing multiple related facts for the same subject.

We introduce the concept of **Same-Subject Editing**, where multiple relations are edited simultaneously for a single subject. Instead of focusing solely on the traditional (s, r, o) format, we extend the editing process to structured prompts such as (s, R, O), where $R = \{r_i\}_{i=1}^N$ represents a set of relations and $O = \{o_i\}_{i=1}^N$ represents their corresponding objects. For example, {("James", "playsFor", "Lakers"), ("James", "isCitizenOf", "USA")}. We formally define the edited fact set as $e = (s, r_i, o_i)_{i=1}^N$ and define the edited model as $M^* = F(M, e)$, where F is the editing function that updates the original model M. It ensures that knowledge updates remain consistent across all related attributes of the same subject.

3 Pilot Observation

In this section, we conduct a pilot observation to reveal potential issues with same-subject editing.

Evaluation Setup. We focus on using MEMIT (Meng et al., 2022b) to edit GPT-J (Wang and Komatsuzaki, 2021), since their excellent performance in editing multiple pieces of knowledge. To analyze the impact of editing density—defined here as the average number of related edits per subject in the editing sequence—we divide our experimental schemes into three categories:

a) ***High-Density****: Edit* ***n*** *pieces of knowledge in total, with each subject edited for* ***3*** *related pieces of knowledge.*
b) ***Medium-Density****: Edit* ***n*** *pieces of knowledge in total, with each subject edited for* ***2*** *related pieces of knowledge.*
c) ***Low-Density****: Edit* ***n*** *pieces of knowledge in total, with each subject edited for* ***1*** *related pieces of knowledge.*

Based on the above schemes, we select qualified data from COUNTERFACT (Meng et al., 2022a) and conduct experiments using both *sequential-editing* and *batch-editing* (See Appendix A.2 for comparison of sequential- and batch-editing). The editing performance is comprehensively evaluated across four dimensions: **efficacy**, **generalization**, **specificity**, and **overall performance** (See Appendix C.3 for detailed metric descriptions).

Result & Analysis. Figure 2 and Figure 8a show the experimental results of employing MEMIT to edit GPT-J through *sequential-editing* and *batch-editing*, respectively. It is evident that when editing the same number of knowledge, the denser the subject distribution, the worse the editing performance, while the impact on the model's downstream performance remains similar. However, the scarcity of sufficiently dense same-subject instances in existing editing datasets limits the scope of experimental verification. We will further investigate this phenomenon in subsequent sections.

4 Related Knowledge Perturbation

Furthermore, we construct a benchmark and evaluate the performance of editing methods when editing related knowledge for the same subject.

Item	S^2RKE	COUNTERFACT
Records	22064	21919
Subjects	4503	20391
Relations	43	32
Maximum records per subject	13	4
Minimum records per subject	3	1
Average records per subject	4.9	1.1

Table 1: Comparison of different benchmarks.

4.1 S^2RKE Benchmark

We introduce the **S^2RKE** (**S**ame-subject **R**elated **K**nowledge **E**diting) benchmark, specifically designed to facilitate the editing of multiple related pieces of knowledge for each subject. It covers six categories of subjects, comprising of 4,503 subjects and 43 relationships, with each subject having an average of 4.9 related knowledge items. See Appendix B for additional technical details about its construction and Table 1 for comparison of statistics between S^2RKE and COUNTERFACT.

4.2 Failure of Editing Methods

Editing Methods. We evaluate six widely-used editing methods: ROME (Meng et al., 2022a), MEMIT (Meng et al., 2022b), PMET (Li et al., 2024a), FT (Zhu et al., 2021), MEND (Mitchell et al., 2022a), and KN (Dai et al., 2022).

Selected LLMs. Experiments are conducted on three LLMs with different parameter sizes: GPT-2 XL (1.5B) (Radford et al., 2019), GPT-J (6B) (Wang and Komatsuzaki, 2021), and LLaMA-2 (7B) (Touvron et al., 2023).

We design two experimental schemes to assess how editing related knowledge impacts performance: ***Same-Subject***, where all edited knowledge shares the same subject, ***Different-Subject***, where each edit involves a different subject. Experimental data are selected from the S^2RKE benchmark.

Our pilot observation indicates that while knowledge correlation impacts editing effectiveness, it has little effect on overall model performance. So we focus on the **Score(S)** metric and introduce the **Score Difference (SD)** metric, defined as SD = Score(same-subject) – Score(different-subject), to quantify performance degradation when editing related knowledge for the same subject. To ensure reliability, each test was repeated 30 times with different editing instances. See Appendix C for more details.

Result & Analysis. Figure 3 and Figure 8b show the results of *sequential-editing* and *batch-editing* on three LLMs using six methods, respectively. The line in each figure represents the Score Difference (SD). The results show that locate-then-edit methods (e.g., ROME, MEMIT, PMET) suffer significant performance degradation under Same-Subject editing, as reflected by a substantial negative Score Difference (SD). In contrast, methods with generally lower editing effectiveness show minimal sensitivity to the relatedness of the edited knowledge. These findings confirm that knowledge correlation markedly impairs the editing performance of certain methods.

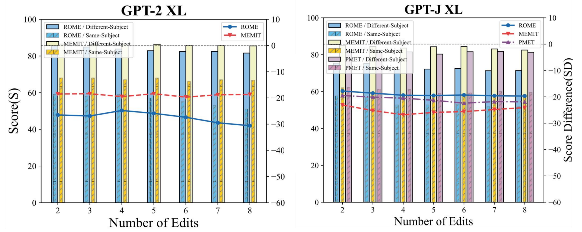


Figure 4: The results of *sequential-editing* on GPT-2 XL and GPT-J using mainstream locate-then-edit methods. The bars represent the **Score (S)** of two strategies, and the line represents the **Score Difference (SD)** between the two strategies.

4.3 Analysis of Failures

We further examine how the sequence of knowledge edits affects locate-then-edit methods by isolating the interference of sequential updates. For this purpose, we devised two experimental settings: ***Homogeneous-Editing***, where the first and last edits target the same subject, and ***Heterogeneous-Editing***, in which they target different subject. Experiments were performed using ROME, MEMIT, and PMET across three LLMs, with each configuration repeated 30 times on different instances from the S^2RKE benchmark to ensure robust results.

Result & Analysis. Figure 4 shows the sequential-editing results on GPT-2 XL and GPT-J, while Figures 7 and 8c provide additional results. Under the Homogeneous-Editing setting, the initial edit's score is much lower than in the Heterogeneous-Editing condition. This clearly indicates that later edits interfere with earlier ones. We call this effect "*related knowledge perturbation*," which exposes a key limitation of current locate-then-edit approaches when processing multiple sequential updates. These findings highlight the need for better strategies in managing sequential knowledge updates. The next section will analysis the causes of *related knowledge perturbation*.

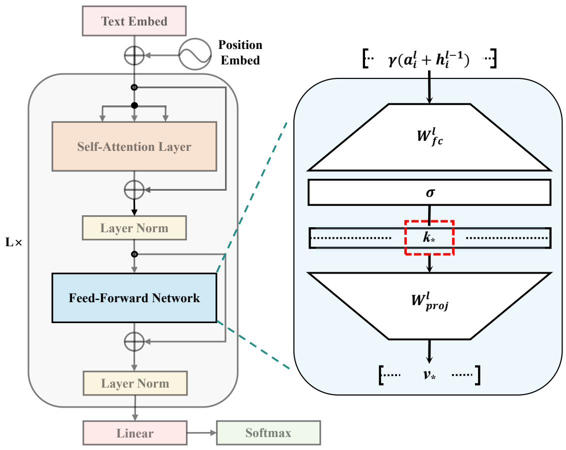


Figure 5: Illustration of related knowledge perturbation in same-subject editing.

5 Perturbation Analysis

5.1 Causes of Perturbation

Our experiments show that only mainstream locate-then-edit methods (e.g., ROME and MEMIT) exhibit *related knowledge perturbation*. These methods all employ causal tracing to identify that factual knowledge is primarily stored in the early MLP layers of LLMs. Based on the hypothesis that "the MLP modules in Transformer layers can be viewed as linear key-value associative memory," (Geva et al., 2020) they solve for $Wk = v$, where W represents the downsampling component $W^{(l)}_{\text{proj}}$ of MLP, and the key-value pair (k, v) corresponds to a factual triplet $t = (s, r, o)$, as shown in Figure 5. Here, k represents the subject s, while v encodes the attributes of s, including r and o. To update t to $t_* = (s, r, o_*)$, they compute a new key k_* and value v_* via an update ΔW.

However, k_* is only derived from the input of the subject's last token in the MLP module's downsampling layer:

$$k_* = \frac{1}{N}\sum_{i=1}^{N}\mathcal{K}(x_i \oplus p), \quad (3)$$

where $\mathcal{K}$ is the output of the first MLP layer in transformer block, x_i represents the randomly sampled prefixes, and $\oplus$ denotes the string concatenation operator.

Therefore, we speculate that *"related knowledge perturbation"* stems from an over-reliance on subject information. When editing multiple pieces of knowledge for the same subject s, the key value k_* remains constant, causing later edits to interfere with earlier ones and reducing performance.

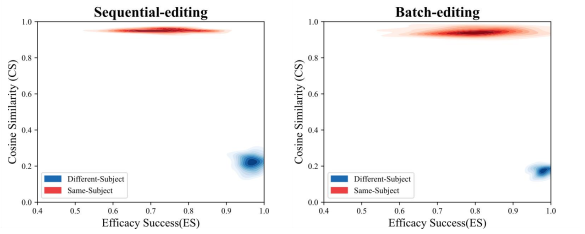


Figure 6: The relationship between the **cosine similarity** of keys and the **Efficacy Success (ES)** of the first knowledge editing using MEMIT to edit GPT-J, under *sequential-editing* and *batch-editing*.

5.2 Experiment Validation

To verify the above speculation, we used MEMIT to edit two pieces of knowledge on GPT-J through *sequential-editing* and *batch-editing*, designing two experimental schemes: ***Same-Subject*** and ***Different-Subject***. We then examine the relationship between the **cosine similarity** of the two keys and the *Efficacy Success* of editing the first piece of knowledge. Cosine similarity was chosen because it measures how similar the two keys are in vector space, helping us understand how closely related the two knowledge pieces are.

Result & Analysis Figure 6 shows the relationship between key similarity and the first knowledge editing Efficacy Success. The results indicate that when two pieces of knowledge related to the same subject are edited, the CS of the key approaches 1. Meanwhile, the ES of editing the first piece of knowledge is significantly lower compared to the case where the two edited pieces of edited knowledge are related to different subjects. This supports our hypothesis that since the key calculation only focuses on subject information, subsequent edits for the same subject interfere with earlier ones, leading to *"related knowledge perturbation"*.

6 Conclusion

In this paper, we identify a key limitation of mainstream locate-then-edit methods, called *"related knowledge perturbation"*, which occurs when editing multiple related pieces of knowledge for the same subject. Using the S^{2}RKE benchmark, we show through experiments that over-reliance on subject information leads to interference between subsequent edits, highlighting the challenges in same-subject editing.

7 Limitation

We acknowledge several limitations in our work. First, while this paper provides an initial exploration into the complex correlations between knowledge and identifies the phenomenon of related knowledge perturbation, it does not propose a comprehensive solution to address this issue. This omission leaves room for future research to develop effective mitigation strategies.

Additionally, due to computational resource constraints, our experiments did not extend to larger language models, such as Llama2-13b. Future investigations could benefit from testing our findings on such models to further validate the effectiveness and generalizability of the observed phenomena.

8 Acknowledgement

We would like to express our sincere gratitude to all the reviewers for their valuable feedback, which greatly contributed to the improvement of this research. This work was supported by the Strategic Priority Research Program of the Chinese Academy of Sciences (No. XDB0680202) and the National Key Research and Development Program of China (2024YFB3109301).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

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, Dublin, Ireland. Association for Computational Linguistics.

N De Cao, W Aziz, and I Titov. 2021. Editing factual knowledge in language models. In *EMNLP 2021-2021 Conference on Empirical Methods in Natural Language Processing, Proceedings*, pages 6491–6506.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2020. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*.

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. *arXiv preprint arXiv:2401.04700*.

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

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. 2024a. Pmet: Precise model editing in a transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18564–18572.

Zhoubo Li, Ningyu Zhang, Yunzhi Yao, Mengru Wang, Xi Chen, and Huajun Chen. 2024b. Unveiling the pitfalls of knowledge editing for large language models. In *The Twelfth International Conference on Learning Representations*.

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

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

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

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

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.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Ben Wang and Aran Komatsuzaki. 2021. Gpt-j-6b: A 6 billion parameter autoregressive language model.

Peng Wang, Ningyu Zhang, Xin Xie, Yunzhi Yao, Bozhong Tian, Mengru Wang, Zekun Xi, Siyuan Cheng, Kangwei Liu, Guozhou Zheng, et al. 2023. Easyedit: An easy-to-use knowledge editing framework for large language models. *arXiv preprint arXiv:2308.07269*.

Wanli Yang, Fei Sun, Xinyu Ma, Xun Liu, Dawei Yin, and Xueqi Cheng. 2024a. The butterfly effect of model editing: Few edits can trigger large language models collapse. *arXiv preprint arXiv:2402.09656*.

Wanli Yang, Fei Sun, Jiajun Tan, Xinyu Ma, Du Su, Dawei Yin, and Huawei Shen. 2024b. The fall of rome: Understanding the collapse of llms in model editing. *arXiv preprint arXiv:2406.11263*.

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. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10222–10240, Singapore. Association for Computational Linguistics.

Chen Zhu, Daliang Li, Felix Yu, Manzil Zaheer, Sanjiv Kumar, Srinadh Bhojanapalli, and Ankit Singh Rawat. 2021. Modifying memories in transformer models. In *International Conference on Machine Learning (ICML)*, 2020.

A Related Works

A.1 Knowledge Editing

Model editing has gained significant attention for its ability to efficiently update LLMs. Existing approaches can be categorized into four types: **Fine-tuning** mainly applies layer-wise adjustments to incorporate new knowledge into LLMs (Zhu et al., 2021). **Meta Learning** trains hypernetworks to act as editors, predicting parameter updates to inject new knowledge (De Cao et al., 2021; Mitchell et al., 2022a). **Memory-based** enhances LLMs with external memory or additional parameters, allowing new knowledge to be added without altering LLMs (Mitchell et al., 2022b; Huang et al., 2023).

Among all types, **Locate-then-Edit** has gained significant traction for its ability to modify specific knowledge within LLMs. Methods like KN(Dai et al., 2022) and ROME(Meng et al., 2022a) locate and update factual knowledge by targeting neurons or multi-layer perceptrons (MLPs) that store such information. MEMIT(Meng et al., 2022b) extends ROME by distributing updates across multiple intermediate MLP sublayers, enabling large-scale knowledge editing. Additionally, PMET(Li et al., 2024a) combines information from both multi-head Self-attention (MHSA) and MLP modules during optimization, producing more accurate MLP outputs for final edits.

While model editing has shown great promise, some researches have identified issues such as model collapse(Yang et al., 2024a; Gu et al., 2024) and knowledge conflicts(Li et al., 2024b). This paper focuses on how the correlation between knowledge impacts the performance of model editing, particularly in the context of multiple knowledge edits.

A.2 Sequential-editing vs. Batch-editing

Sequential-editing and *batch-editing* are two strategies commonly used to update large amounts of knowledge in LLMs(Yao et al., 2023). Specifically, *sequential-editing* refers to making multiple edits one after another, where the model should ideally retain previous changes as new edits are introduced. In contrast, *batch-editing* involves editing multiple pieces of knowledge in a model at once. Notably, these two strategies can be combined to create a more flexible knowledge editing approach.

For the purposes of this study, we evaluate these strategies independently: In *sequential-editing*, the batch size is set to 1, and in *batch-editing*, the number of consecutive edits is set to 1, ensuring clear comparisons and facilitate experimental evaluation.

B Details of S^{2}RKE Benchmark

B.1 Data Construction

In this paper, **S^{2}RKE** (**S**ame-subject **R**elated **K**nowledge **E**diting) benchmark is built on the YAGO3.0.3, which combines Wikipedia, WordNet, GeoNames and other data sources, and was released in 2022. The construction process is detailed below, covering four key aspects:

Triple filtering. Based on YAGO's top-level classification, we categorize the entities to be edited into six groups: Person, Building, Organization, Abstraction, Artifact and GeoEntity. From these categories, we screen out 43 relationships. Unlike COUNTERFACT, S^{2}RKE innovatively includes both literal- and data-type relationships, enabling broader coverage of relationship types. Finally, We then select entities with the most relationship instances from each category and generated correct triplets (s, r, o).

Requested rewrite. To evaluate model efficacy, we select the relation r from the triplet (s, r, o) and generate a counterfactual triplet (s, r, o_*). We create natural language templates $P(r)$ for each relation r, using ChatGPT-4o to generate templates based on examples from the PARAREL (Elazar et al., 2021) dataset. After generating multiple templates, we manually select the three most suitable ones to ensure test diversity and

Categories	Subjects	Relations	Edits(all)	Edits(Avg)
Person	592	29	5706	9.6
Organization	874	7	2897	3.3
Building	679	6	3419	4.6
Artifact	857	6	3632	4.2
Abstraction	734	8	2203	3.0
GeoEntity	912	12	4207	5.0
All	4503	43	22064	4.9

Table 2: Data statistics of the S^2RKE benchmark.

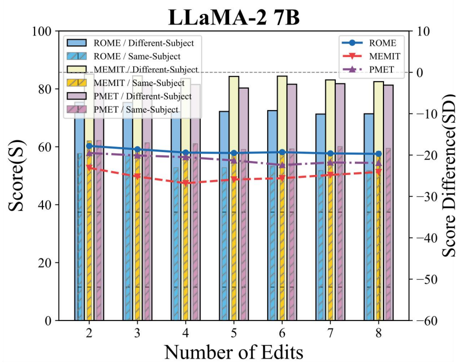


Figure 7: The results of *sequential-editing* on LLaMA-2 7B using mainstream locate-then-edit methods. The bars represent the **Score (S)** of two strategies, and the line represents the **Score Difference (SD)** between the two strategies.

template consistency.

Paraphrase prompts. To evaluate the generalization of model editing methods, we use the moonshot-v1 for generating longer text, combined with the description of the edited entity and a simplified prompt template for each relation. This process produce semantically equivalent but more complex sentences P^P, designed to test the model's ability to handle diverse expressions.

Neighborhood prompts. In order to evaluate the specificity of the model editing methods, we identify related triples (s_*, r_*, o) for the object o of the original triplet (s, r, o), using the YAGO database. These neighborhood triplets are converted into natural language P^N using simple templates$P(r_*)$, specifically constructed for each relation r_*.

B.2 Data Summary

Data standardization. Firstly, we standardize the description of each edited to ensure clear distinctions between them. Additionally, we handle relations involving literal- and date-type appropriately, with literal-type storing integers and date-type limited to years. Special characters in object values are also replaced or removed to ensure consistency and operability of the data format.

Data statistics. The S^2RKE benchmark contains 6 categories of edited entity, with a total of 3704 subjects and 43 specific relationships, spread across 3 categories of relationship. On average, each entity contains 4.9 edited knowledge entries, with Person entities having the highest number of edits. See Table 2 for statistics of S^2RKE.

Data format. In summary, each record in the S^2RKE benchmark D consists of a subject s and its multiple related requested rewrite. $r, o, o_*, P(r)$. For each rewrite, the benchmark also includes one paraphrase promptP^P and two neighborhood prompts P^N. See Figure for a sample record in SMRKE, complete with three related edits for the same subject.

C Detailed Experimental Setup

C.1 Editing Methods

In this paper, we use six editing methods:

FT (Zhu et al., 2021) applies an ℓ_∞ norm constraint on the fine-tuning loss, limiting the difference between the original and edited model's parameters to reduce side effects.

MEND (Mitchell et al., 2022a) uses a collection of small hypernetworks to learn a rank-one decomposition of the gradient obtained by standard fine-tuning, enabling tractable edits in LLMs.

KN (Dai et al., 2022) select neurons associated with knowledge expression via gradient-based attributions, then modify MLP layer at the rows corresponding to those neurons by adding scaled embedding vectors.

ROME (Meng et al., 2022a) uses causal tracing to localize the knowledge storage at a specific MLP layer in a transformer, and then updates knowledge by altering the weight matrix with rank-one update.

MEMIT (Meng et al., 2022b) extends ROME by distributing updates across multiple MLP layers, enabling large-scale edits.

PMET (Li et al., 2024a) enhances MEMIT by integrating information from both the multi-head self-attention (MHSA) and MLP modules during the optimization process.

It is worth noting that ROME and KN can only *sequential-editing*. All experiments are conducted using the EasyEdit (Wang et al., 2023), ensuring standardized and reproducible evaluations.

ID	Relation	Domain	Range
1	<hasPages>	rdfs:domain owl:Thing	rdfs:range xsd:nonNegativeInteger
2	<isCitizenOf>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_country_108544813>
3	<diedOnDate>	rdfs:domain <wordnet_person_100007846>	rdfs:range xsd:date
4	<hasGender>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_sex_105006698>
5	<wasBornOnDate>	rdfs:domain <wordnet_person_100007846>	rdfs:range xsd:date
6	<hasDuration>	rdfs:domain owl:Thing	rdfs:range <s>
7	<hasWeight>	rdfs:domain <wordnet_physical_entity_100001930>	rdfs:range <kg>
8	<hasHeight>	rdfs:domain <wordnet_physical_entity_100001930>	rdfs:range <m>
9	<hasLength>	rdfs:domain <yagoGeoEntity>	rdfs:range <km>
10	<hasWonPrize>	rdfs:domain <yagoLegalActorGeo>	rdfs:range <wordnet_award_106696483>
11	<owns>	rdfs:domain <yagoLegalActorGeo>	rdfs:range owl:Thing
12	<created>	rdfs:domain <yagoLegalActor>	rdfs:range owl:Thing
13	<participatedIn>	rdfs:domain <yagoLegalActorGeo>	rdfs:range owl:Thing
14	<isAffiliatedTo>	rdfs:domain <yagoLegalActor>	rdfs:range <wordnet_organization_108008335>
15	<hasAcademicAdvisor>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_person_100007846>
16	<graduatedFrom>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_university_108286569>
17	<hasChild>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_person_100007846>
18	<edited>	rdfs:domain <wordnet_editor_110044879>	rdfs:range owl:Thing
19	<directed>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_movie_106613686>
20	<wroteMusicFor>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_movie_106613686>
21	<playsFor>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_organization_108008335>
22	<isPoliticianOf>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_organization_108008335>
23	<isLeaderOf>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_organization_108008335>
24	<influences>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_person_100007846>
25	<isMarriedTo>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_person_100007846>
26	<worksAt>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_organization_108008335>
27	<isInterestedIn>	rdfs:domain <wordnet_person_100007846>	rdfs:range owl:Thing
28	<livesIn>	rdfs:domain <yagoLegalActorGeo>	rdfs:range <wordnet_location_100021767>
29	<isKnownFor>	rdfs:domain <wordnet_person_100007846>	rdfs:range owl:Thing
30	<actedIn>	rdfs:domain <wordnet_location_100021767>	rdfs:range <wordnet_movie_106613686>
31	<hasArea>	rdfs:domain <wordnet_location_100021767>	rdfs:range xsd:km2
32	<hasCurrency>	rdfs:domain <wordnet_location_100021767>	rdfs:range <wordnet_currency_108524613>
33	<dealsWith>	rdfs:domain <wordnet_person_100007846>	rdfs:range <wordnet_country_108544813>
34	<hasOfficialLanguage>	rdfs:domain <wordnet_location_100021767>	rdfs:range <wordnet_language_106282651>
35	<hasCapital>	rdfs:domain <wordnet_location_100027167>	rdfs:range <wordnet_city_108524735>
36	<wasCreatedOnDate>	rdfs:domain owl:Thing	rdfs:range xsd:date
37	<isLocatedIn>	rdfs:domain <yagoPermanentlyLocatedEntity>	rdfs:range <yagoGeoEntity>
38	<hasLongitude>	rdfs:domain <yagoGeoEntity>	rdfs:range <degrees>
39	<happenedOnDate>	rdfs:domain <wordnet_event_100029378>	rdfs:range xsd:date
40	<happenedIn>	rdfs:domain <wordnet_event_100029378>	rdfs:range <yagoGeoEntity>
41	<hasLatitude>	rdfs:domain <yagoGeoEntity>	rdfs:range <degrees>
42	<wasBornIn>	rdfs:domain <wordnet_person_100007846>	rdfs:range <yagoGeoEntity>
43	<diedIn>	rdfs:domain <wordnet_person_100007846>	rdfs:range <yagoGeoEntity>

Table 3: Summary of domain and range properties for selected relations in S^{2}RKE.

C.2 Selected Models

In this paper, we select three large language models (LLMs):

GPT-2 XL (Radford et al., 2019), a 1.5 billion parameter version of GPT-2,is a transformer-based language model developed by OpenAI.

GPT-J (Wang and Komatsuzaki, 2021), developed by EleutherAI, is a GPT-3-like open-source LLM with 6 billion parameters, trained on *The Pile*.

LLaMA2-7B (Touvron et al., 2023), a 7 billion parameter version of LLaMA 2 from Meta AI, is a leading open-source LLM, known for its advanced training techniques and optimizations.

C.3 Evaluation Metrics

To comprehensively evaluate the experimental results, we evaluate editing methods across four dimensions:

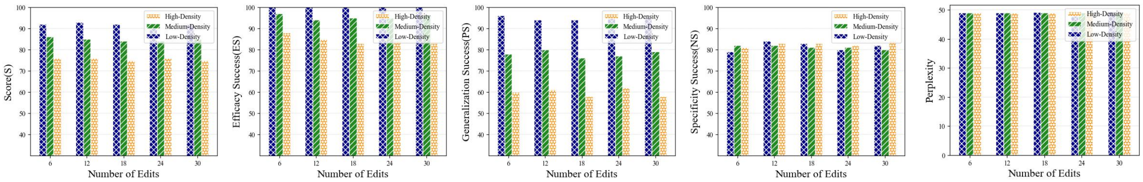
Efficacy. We measure efficacy using the Efficacy Success (**ES**) metric. Specifically, when triple (s, r, o) is updated to (s, r, o_*), ES calculates the success rate of the target edit by determining the probability that the condition $P[o*] > P[o]$ is satisfied.

Generalization. To evaluate generalization, we use Paraphrase Success (**PS**) metric, which measures the probability that $P[o_*] > P[o]$ when the model is prompted with a paraphrase of the original (s, r).

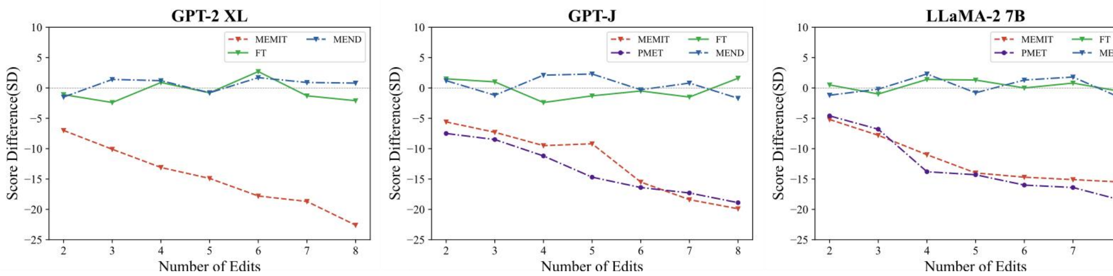
Specificity. For specificity, we adopt the Neighborhood Success (**NS**) metric, which tests the probability that $P[o_c] > P[o_*]$ for triplet (s, r, o_c), where o_c lies outside the range of the factual edits.

Overall Performance. We assess overall model performance using Perplexity (**PPL**), based on prior studies by Yang et al. (2024a,b). An increase in perplexity generally indicates a decrease in the model's performance in generation tasks.

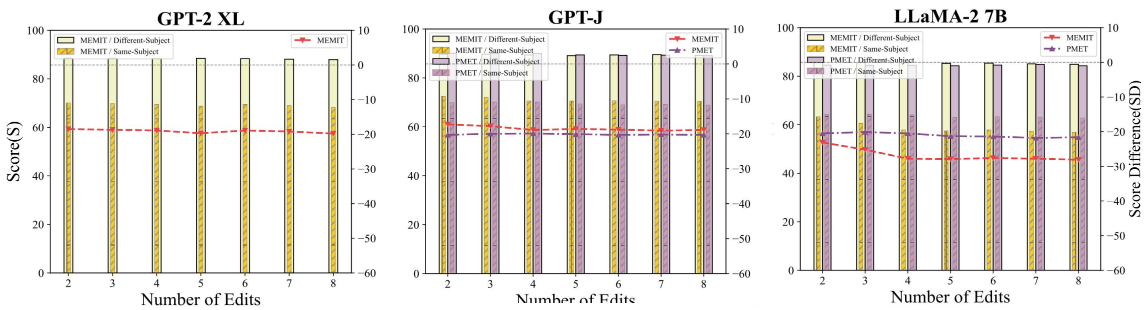
Finally, to evaluate the balance between efficacy, generalization, and specificity, we report the harmonic mean of **ES**, **PS**, and **NS** indicators as a comprehensive score (**S**), providing a holistic view of the model's behavior across these dimensions.



(a) The results of *batch-editing* on GPT-J using MEMIT, comparing five evaluation metrics of three different schemes.



(b) The results of *batch-editing* on three LLMs by six editing methods. **Score Difference (SD)** represents the difference in editing performance between the two experimental schemes when editing the same amount of knowledge under the same method.



(c) The results of *batch-editing* on three LLMs using mainstream locate-then-edit methods. The bars represent the **Score (S)** of two strategies, and the line represents the **Score Difference (SD)** between the two strategies.

```
{
    "subjectID": 0,
    "class": "<wordnet_artifact_100021939>",
    "subject": {
        "name": "Double Xposure",
        "URL": "/resource/schema:Movie",
        "original_name": "<Double_Xposure>",
        "description": "2012 film directed by Li Yu"
    },
    "Star_Topology": [
        {
            "requested_rewrite": {
                "prompt": "{}, which is located in",
                "relation": "<isLocatedIn>",
                "target_true": "China",
                "target_new": "Głogów County"
            },
            "paraphrase_prompts": [
                "Double Xposure, a 2012 film by Li Yu, is set in"
            ],
            "neighborhood_prompts": [
                "Ping Zhang is a citizen of",
                "Ricky Lee has citizenship in"
            ]
        },
        {
            "requested_rewrite": {
                "prompt": "{} came into existence on",
                "relation": "<wasCreatedOnDate>",
                "target_true": "2012",
                "target_new": "1910"
            },
            "paraphrase_prompts": [
                "Directed by Li Yu, Double Xposure emerged in the cinematic world in"
            ],
            "neighborhood_prompts": [
                "Rolf Appel passed away in the year",
                "A. B. Quintanilla died in the year"
            ]
        },
        {
            "requested_rewrite": {
                "prompt": "{} lasts for a duration of seconds,",
                "relation": "<hasDuration>",
                "target_true": "6300",
                "target_new": "7200"
            },
            "paraphrase_prompts": [
                "Double Xposure is a 2012 suspenseful film helmed by director Li Yu. Its duration is"
            ],
            "neighborhood_prompts": [
                "The total duration of The Place is seconds,",
                "The Morality of Mrs. Dulska lasts for a duration of seconds,"
            ]
        }
    ]
},
```


Figure 9: Case example in S[2]RKE.