MEMIT-Merge: Addressing MEMIT's Key-Value Conflicts in Same-Subject Batch Editing for LLMs

Anonymous ACL submission

Abstract

As large language models continue to scale up, knowledge editing techniques that modify models' internal knowledge without full retraining have gained significant attention. MEMIT, a prominent batch editing algorithm, stands out for its capability to perform mass knowledge modifications. However, we uncover that MEMIT's editing efficacy significantly deteriorates when processing batches containing multiple edits sharing the same subject. Our analysis reveals this stems from MEMIT's key value modeling framework: identical keys (derived from the shared subject) are forced to represent different values (corresponding to different knowledge), resulting in updates conflicts during editing. Addressing this issue, we propose MEMIT-Merge, an enhanced approach that merges value computation processes for facts sharing the same subject, effectively resolving the performance degradation in same-subject batch editing scenarios. Experimental results demonstrate that when MEMIT's edit success rate drops to around 50% at larger batch sizes, MEMIT-Merge maintains a success rate exceeding 90%, showcasing remarkable robustness to subject entity collisions.

1 Introduction

As large language models (LLMs) continue to scale up, the prohibitive cost of full model retraining has made knowledge editing increasingly crucial in this domain. Among prevalent editing algorithms, a class of algorithms, termed "Locate and Edit" methods by Zhang et al. (2024), enables targeted modifications through precise manipulation of specific regions. MEMIT (Meng et al., 2023), one of the most prominent algorithms in this class, has gained significant attention (Li et al., 2024; Fang et al., 2024; Gupta et al., 2024). It extends ROME's architecture (Meng et al., 2022) and enables the simultaneous modification of multiple knowledge instances within a single update operation.

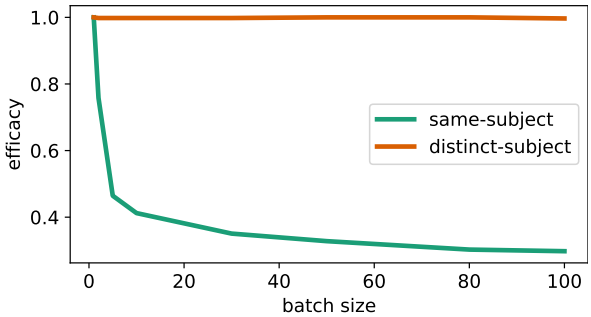

Figure 1: The edit success rate of the MEMIT method on same-subject and distinct-subject datasets, showing the changes with varying batch sizes. A significant decline is observed when the subjects are the same.

However, our investigation reveals a critical limitation in MEMIT: When handling batches with multiple edits that share the same subject (such as "John Smith now plays basketball." and "John Smith comes from England." share the same subject "John Smith", while "Paul Morand comes from England" has a different subject), the method will exhibit significant performance degradation. In contrast, edits with different subjects maintain stable efficacy.

To systematically demonstrate this performance degradation, we constructed two contrastive datasets comprising batches with identical subjects versus fully unique subjects, named distinct-subject and same-subject, respectively. The experimental results are in Fig. 1, where the vertical axis represents efficacy (which means the editing success rate) and the horizontal axis indicates the batch size per edit. The results reveal that MEMIT maintains a high success rate as batch size increases when editing distinct-subject cases, but exhibits significant performance degradation for the same subject cases. However, same subject cases are also critical in the real-world practices (*e.g*, updating a person's occupation, workplace, and employer simultaneously). More detailed experimental settings can be

found in Sec. 4.3.

The performance degradation stems from MEMIT's key-value modeling paradigm: identical keys (derived from shared subject representations) map to conflicting values during same-subject batch edits. MEMIT formulates knowledge updates as MLP key-value pairs where the output linear layer's weights are adjusted to align keys with edited values. [1] However, when multiple edits share subjects, their identical keys require divergent value mappings - an inherent contradiction since single-layer perceptrons cannot produce multiple outputs for identical inputs.

To resolve this fundamental conflict, we propose MEMIT-Merge, an enhanced variant of MEMIT. Our key insight is to enforce value consistency by merging multiple knowledge entries that share identical keys. Experimental results show that MEMIT-Merge consistently outperforms MEMIT on same-subject dataset, maintaining a success rate above 90%, whereas MEMIT drops to around 50%. For distinct-subject data, both methods perform comparably with no significant differences.

2 Problem

2.1 Preliminaries

The MEMIT framework hypothesizes that factual knowledge in models is stored within the parameters of MLP layers. Each MLP layer contains input/output linear layers with parameter matrices W_{in} and W_{out}, where W_{out} serves as the key-value mapping targeted by MEMIT editing. The key corresponds to the hidden state at the MLP's intermediate layer while the value represents the MLP's final output.

Knowledge is represented as triples (s, r, o). During editing, complete sentences are constructed from these triples. The key is determined by the subject s and its contextual prefix, while the value is obtained by inversely optimizing the object o:

$$v = \arg\min_{v}(-\log P_v[o|(s,r)]) \quad (1)$$

All (k, v) pairs are processed in batch to update W_{out} via closed-form solution:

$$W_{out} = W_0 + (V - W_0 K)K^T(C + KK^T)^{-1} \quad (2)$$

[1] Note that the key-value here refers to the hidden state and output within the MLP module as described by Meng et al. (2022), rather than the query, key and value in the attention module.

Here, K and V denote batched key/value matrices, W_0 represents original parameters, and C is a knowledge-preservation constant.

2.2 Same subject issue in MEMIT

Normally, MEMIT is capable of maintaining its efficacy without a pronounced decline in performance when the edit batch size approaches 1,000. However, we have identified a notable issue: when the edit batch encompasses knowledge triples sharing the same subject, the editing capacity of MEMIT experiences a substantial degradation.

To verify this phenomenon, we constructed two counterfactual editing datasets. In the first dataset, the subjects of the knowledge triples are all distinct. In the second dataset, the subjects of the knowledge triples are replaced by a single, fixed subject, while all other parts of these two datasets remain identical. The details of the datasets construction are provided in App. A.

As illustrated in Fig. 1, when the subjects are identical, the performance of the MEMIT method drops sharply with a batch size of only 2, and the edit success rate falls below 50% when the batch size reaches 10. In contrast, when subjects are distinct, increasing the batch size has virtually no impact on edit success.

3 Approach

3.1 Cause Analysis

In our analysis, the degradation of editing capability caused by identical subjects is closely related to the key-value modeling of knowledge inherent in locate-and-edit class editing methods.

In the standard MEMIT, a piece of knowledge to be edited can be represented by a knowledge triplet (subject, relation, object), and a complete sentence is constructed based on this triplet for the editing process. In this paper, we use the format "subject's relation is object" to construct the sentence. For example, the knowledge triple (John,father,Bob) is formulated into the sentence "John's father is Bob.".

As described in Sec. 2.1, during MEMIT editing, the key is derived from the subject, while the value is determined by the object. However, when editing multiple pieces of knowledge with the same subject but different objects in one batch, this mechanism forces the MLP to map the same key to two distinct values. As illustrated in Fig. 2, a given key can only produce a single fixed value through deterministic

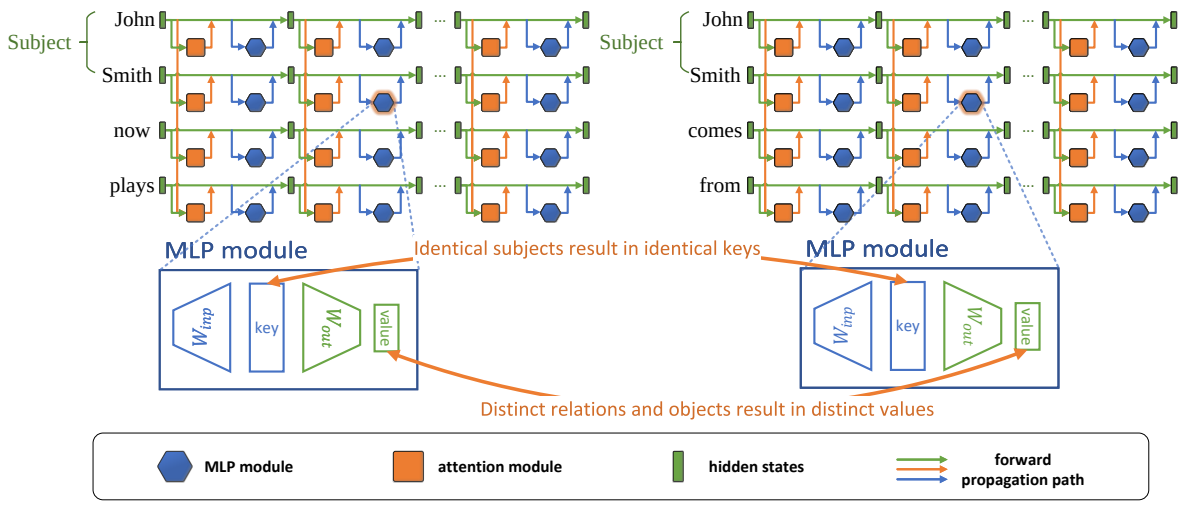

Figure 2: The architecture of MEMIT processing two same subject sentences. The left and right sides of the figure depict the processing flow of the two sentences respectively. Below, we expand the details of the MLP module to be modified, which consists of two linear layers. In MEMIT, the key is determined by the subject, resulting in identical keys on both sides. The value is optimized from the relation and object, leading to different values on each side. Consequently, the optimization target for the editable W_{out} requires producing different values for the same input key.

W_{out}. This creates a conflict when optimizing the parameter matrix, making it extremely challenging. We refer to this issue as the key collision problem. Consequently, when a batch contains multiple edits with the same subject, as demonstrated in Fig. 1, the editing capability of MEMIT is significantly degraded.

Furthermore, we analyzed the relationship between MEMIT editing capability and key distance within a batch, finding that closer keys lead to greater capability degradation. Due to space constraints, detailed analysis is in the App. B

3.2 The MEMIT-Merge Approach

To address this issue, we develop a new optimization objective to merge the value computation of the set of knowledge with the same key:

$$v = \arg\min_{v} \sum_{(s,r_j,o_j)\in S} -\log P_v[o_j|(s,r_j)], \quad (3)$$

where S represents the set of knowledge triples with the same key, v is the value to be optimized in a backward manner, and P_v denotes the model when the value is equal to v.

Compared with Eq. 1, this approach ensures that knowledge sharing same key gets the same value, thereby significantly alleviating the decline in edit efficacy observed in standard MEMIT as evidenced in the next section.

4 Experiments

4.1 Dataset

We constructed two Wikidata-based counterfactual knowledge editing datasets: (1) a "same-subject" set with 100 triples sharing the subject John Smith, and (2) a "distinct-subject" set with unique subjects while maintaining identical relations/objects (construction details in App. A).

In terms of evaluation metrics, we refer to the metrics used by Meng et al. (2023), namely Efficacy, Paraphrase, and Specificity. Efficacy measures the edit success rate on original sentences, paraphrase measures the success rate on paraphrased sentences. Specificity measures the probability that facts unrelated to the edit remain consistent before and after the edit.

While our datasets are novel, they address critical real-world needs. Editing multiple attributes of an entity (*e.g*, updating a person's profile) is a highly realistic demand, making same-subject scenarios essential for practical applications.

4.2 Experimental Setup

We conducted experiments on three models with different architectures: Qwen2.5-1.5B-Instruct (Qwen et al., 2025), GPT-J-6B (Wang and Komatsuzaki, 2021), and Llama-3-8B-Instruct (AI@Meta, 2024).

For MEMIT-based baselines, we use MEMIT

Model	Dataset	Method	Efficiency	Parapharse	Specificity
GPT-J-6B	same-subject	FT	0.52	0.19	0.23
		MEMIT	0.27	0.21	1.00
		PMET	0.26	0.21	0.98
		MEMIT-Merge	**0.51**	**0.32**	**1.00**
	distinct-subject	FT	0.47	0.28	0.22
		MEMIT	1.00	0.77	0.93
		PMET	0.25	0.25	**0.99**
		MEMIT-Merge	**1.00**	**0.77**	0.93

Table 1: The complete results of the four editing methods—MEMIT, MEMIT-Merge, PMET, and FT-L—on the same-subject and distinct-subject datasets at a batch size of 100. All experimental results were obtained by re-running each editing method on our dataset.

and an improved version of MEMIT, PMET (Li et al., 2024). In addition to the MEMIT-based methods, we also included FT-L (Zhu et al., 2020), which was used for comparison in the ROME paper, as another baseline to verify that the same-subject issue exists only in methods with the MEMIT-based architecture.

4.3 Results when Batch Size is 100

We first compared the edit success rates of standard MEMIT, PMET, MEMIT-Merge, and FT-L on the two datasets across several models. [2]

As shown in Tab. 1, our method outperforms standard MEMIT on the same-subject dataset with improved paraphrase accuracy, attributed to enhanced edit success rates. Notably, MEMIT's anomalously high specificity for same-subject edits (indicating ineffective editing and a minimal impact on the original model) is corrected by our approach, achieving specificity levels comparable to FT and distinct-subject scenarios. Results of other models are detailed in App. D.

Comparing the results of MEMIT and PMET to FT-L, it can be observed that the performance drop in same-subject edits is unique to MEMIT-based methods. This phenomenon is consistent with our analysis in Sec. 3.1. By resolving key collisions through key-wise value merging, MEMIT-Merge successfully mitigates this issue, empirically confirming that key collision is the root cause of MEMIT's limitations in same-subject cases.

4.4 Results with Varying Batch Sizes

As can be seen in Fig. 3, when the subjects of the editing knowledge in the edit batch are the same, the standard edit success rate plummets at a batch size of 2, whereas MEMIT-Merge is able to maintain a much higher success rate, with a significantly smaller decline compared to MEMIT. This also confirms the effectiveness of our method. The results of other models are given in App. E

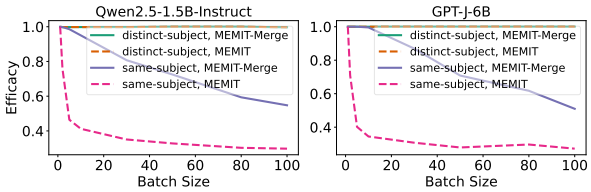

Figure 3: The results of MEMIT-Merge and MEMIT methods on same-subject and distinct-subject datasets using the GPT-J-6B model, showing the changes with varying batch sizes. MEMIT-Merge is capable of significantly alleviating the decline in editing performance under the same-subject condition.

In the case of distinct subjects, the editing capability of both MEMIT and MEMIT-Merge does not exhibit a significant decline even at a batch size of 100, which is consistent with our previous analysis. The results of other editing methods are given in App. F.

5 Conclusion

This paper identifies the issue of significant performance degradation in MEMIT when a batch contains knowledge sharing the same subject during batch editing. This is fundamentally caused by parameter update conflicts arising from identical keys requiring divergent values in the same-subject scenarios. Our proposed MEMIT-Merge resolves this ahd significantly improves same-subject edit performance while maintaining original performance on distinct-subject cases. These findings advance mass-editing techniques for evolving LLM knowledge bases.

[2]The results for all baselines were obtained by running the code from the Easyedit framework on our datasets.

Limitations

While this study provides insights into same-subject issues within MEMIT-based method, several limitations should be acknowledged. First, the constructed same-subject and distinct-subject datasets contain only 100 instances, which may cause unstable results. Second, all knowledge triples are restricted to person-related entities, leaving the generalization to other subject types (such as locations or organizations) untested. While our theoretical framework suggests that subject type should not fundamentally alter the conclusions, empirical validation across diverse categories remains necessary. Third, the experiments focus solely on lexical-level subject distinctions; potential effects of semantic similarity in embedding space were not explored. Future work could extend this investigation by incorporating larger datasets, multi-type knowledge triples, and embedding-space analyses to further validate the theoretical predictions.

References

AI@Meta. 2024. Llama 3 model card.

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.

Jingcheng Deng, Zihao Wei, Liang Pang, Hanxing Ding, Huawei Shen, and Xueqi Cheng. 2024. UnKE: Unstructured Knowledge Editing in Large Language Models. *arXiv preprint*. ArXiv:2405.15349 [cs].

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Xiang Wang, Xiangnan He, and Tat-seng Chua. 2024. AlphaEdit: Null-Space Constrained Knowledge Editing for Language Models. *arXiv preprint*. ArXiv:2410.02355.

Akshat Gupta, Dev Sajnani, and Gopala Anumanchipalli. 2024. A unified framework for model editing. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 15403–15418. Association for Computational Linguistics.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. 2024. PMET: precise model editing in a transformer. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 18564–18572. AAAI Press.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

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, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. 2022. Fast model editing at scale. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 Technical Report. *arXiv preprint*. ArXiv:2412.15115 [cs].

Chenmien Tan, Ge Zhang, and Jie Fu. 2024. Massive editing for large language models via meta learning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, Siyuan Cheng, Ziwen Xu, Xin Xu, Jia-Chen Gu, Yong Jiang, Pengjun Xie, Fei Huang, Lei Liang, Zhiqiang Zhang, Xiaowei Zhu, Jun Zhou, and Huajun Chen. 2024. A Comprehensive Study of Knowledge Editing for Large Language Models. *arXiv preprint*. ArXiv:2401.01286 [cs].

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 Details of Constructing Same-Subject and Distinct-Subject Data

Our dataset construction is based on Wikidata. First, we retrieve all relations and properties associated with human subject entities from Wikidata. Then, we manually filter the relations, removing those that are less commonly used, such as ID and Wikidata categories. Finally, we obtain 100 relations.

Subsequently, we select a number of individuals from Wikidata and query their corresponding objects for the knowledge triples composed of these relations. Finally, we retain only one knowledge triple for each relation, thereby obtaining 100 knowledge triples, formatted as (subject, relation, object).

We then select another 100 distinct names from Wikidata and replace the subject entities in the previously obtained 100 knowledge triples with these new names, thereby creating the distinct-subject dataset. Conversely, we replace the subject entities in the 100 knowledge triples with a single, identical name to create the same-subject dataset.

Using the template "subject's relation is object," we construct natural language sentences from these knowledge triples, which form the edit sentences in the dataset. For example, a knowledge triple in the same-subject dataset is (John Smith, doctoral advisor, Dennis W. Sciama), which is formulated into the natural sentence John Smith's doctoral advisor is Dennis W. Sciama. In the distinct-subject dataset, the corresponding knowledge triple with the same relation and object is (Paul Morand, doctoral advisor, Dennis W. Sciama), which is formulated into the natural sentence Paul Morand's doctoral advisor is Dennis W. Sciama.

Subsequently, following the dataset metrics in Meng et al. (2022), we add two types of questions: specificity and paraphrase. For paraphrase questions, we use the same knowledge triples as the edit sentences, but with a different template format: "The name of the relation of subject is object.". For specificity, there are two types of questions. One is completely unrelated knowledge, for which we use the prompt "The capital city of America is". The other type has the same relation as the edited knowledge but a different subject. For example, if the edited knowledge is (John, father, Bob), a specificity question could be (Paul, father, Eugène).

B Further Analysis of Cause

To further investigate the relationship between the decline in editing capability and the distance between keys, we propose an evaluation metric: the Average Keys Distance Inside Batch (AKD). This metric is defined as the average Euclidean distance between the key values of all pairs of knowledge within a batch. It reflects the average distance between keys in the batch and is represented as

$$AKD^{(l)} = \frac{1}{\binom{|B|}{2}} \sum_{\substack{e_1 \in B \\ e_2 \in B}} ||k_{e_1}^{(l)} - k_{e_2}^{(l)}||_2 \quad (4)$$

where l represents the l-th layer, B denotes the batch of knowledge to be edited, $k_{e_1}^{(l)}$ represents the key value computed by the MLP module in the l-th layer for the input knowledge e_1.

We compute the AKD for all layers of the model at the subject's last token position. As the degree of subject variation increases across sentences, the AKD value proportionally rises. Conversely, when all sentences share identical subjects, the AKD value remains constant at 0.

We construct sentence batches using predefined templates, where batches sharing the same template exhibited similar AKD values, while distinct templates yielded significantly different AKD measurements. The specific templates and corresponding AKD values are detailed in App. C. For experimental validation, we select three AKD groups (0, 10, 25) and conduct editing tests using Qwen2.5-1.5B-Instruct. As shown in Fig. 4, where AKD values are computed using keys from MEMIT's final editing layer, the results demonstrate an inverse relationship: lower AKD values correspond to reduced editing success rates. This pattern remains consistent across other AKD values, establishing a statistically significant negative correlation between AKD and editing efficacy.

C Diverse AKD Dataset

dataset	formatting template	AKD
same-subject	{subject}'s {relation} is {object}	0.0
distinct-subject	{subject}'s {relation} is {object}	25.8
same-subject	The name of the {relation} of {subject} is {object}	10.5
distinct-subject	The name of the {relation} of {subject} is {object}	26.2

Table 2: The average AKD values obtained using different data and templates.

The construction of datasets with three distinct AKD values, where the keys within each dataset

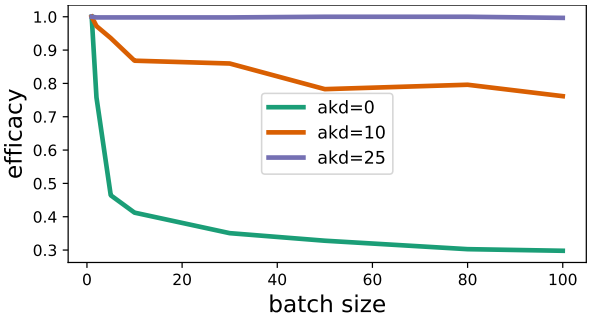

Figure 4: Datasets with different AKD values and the results of edit efficacy. The lower the AKD value, the more severe the decline in edit capability.

have a relatively consistent distance between each other.

We utilize the knowledge triples from the same-subject and distinct-subject datasets collected in Sec. A to construct data using different natural language sentence templates. The two templates we employ are "subject's relation is object" and "The name of the relation of subject is object".

Tab. 2 presents the average AKD values obtained using different data and templates with the Qwen2.5-1.5B-Instruct model. We selected several datasets with distinct AKD values. Since these datasets have consistent internal templates, the keys of the multiple knowledge triples within them are relatively uniform and close in distance. Therefore, when performing batch editing on these datasets, they can be used to study the correlation between efficacy and AKD.

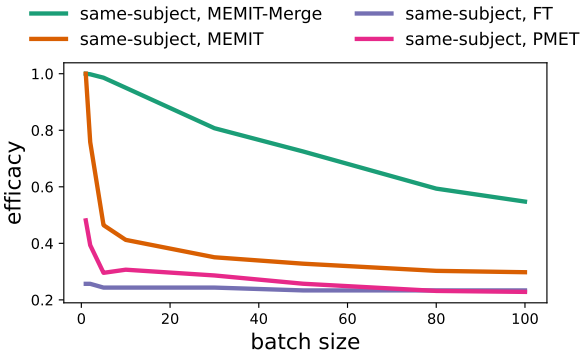

Figure 5: Editing same-subject dataset using Qwen2.5-1.5B-Instruct with four editing methods.

D Results of Other Models At Batch Size 100

Tab. 3 shows results using Llama-3-8B-Instruct, Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct at batch size 100.

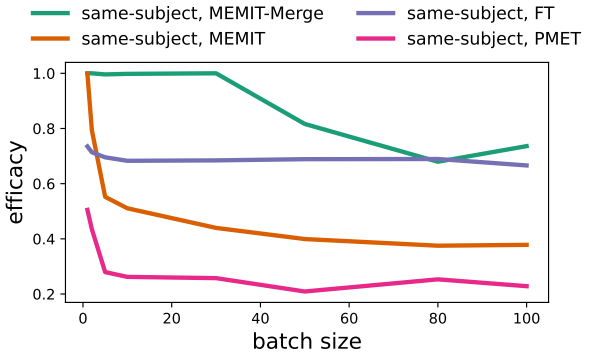

Figure 6: Editing same-subject dataset using Llama-3-8B-Instruct with four editing methods.

E Results with Varying Batch Sizes of other models

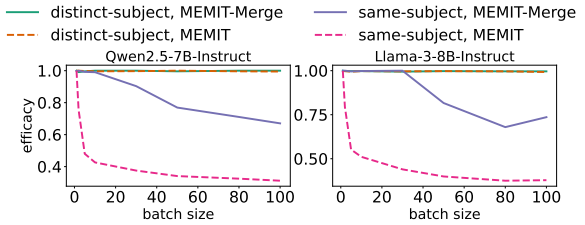

Figure 7: The results of MEMIT-Merge and MEMIT methods on same-subject and distinct-subject datasets using the Qwen2.5-7B-Instruct and Llama-3-8B-Instruct.

Additionally, the experimental results for Qwen2.5-1.5B-Instruct and Llama-3-8B-Instruct, two models with different architectures, as shown in Fig. 7, demonstrate that the same phenomenon observed in the GPT-J model also exists in these models. Moreover, MEMIT-Merge is equally capable of significantly mitigating the performance degradation of standard MEMIT under the same-subject condition. Therefore, it can be concluded that this phenomenon is universally present across different model architectures, and our method is applicable to various model structures.

F Results with Varying Batch Sizes of other methods

Here in Fig. 5 and Fig. 6 we demonstrate some more results about editing same subject batch with varying batch sizes.

It shows clearly that MEMIT-based methods suffers from same subject issue, while methods like FT doesn't.

Model	Dataset	Method	Efficiency	Parapharse	Specificity
Qwen2.5-1.5B-Instruct	same-subject	FT	0.26	0.23	0.95
		MEMIT	0.30	0.24	1.00
		PMET	0.23	0.17	0.99
		MEMIT-Merge	0.55	0.36	0.99
	distinct-subject	FT	0.23	0.21	0.99
		MEMIT	1.00	0.77	0.90
		PMET	0.51	0.40	0.85
		MEMIT-Merge	1.00	0.77	0.90
Llama-3-8B-Instruct	same-subject	FT	0.67	0.47	0.27
		MEMIT	0.38	0.29	0.98
		PMET	0.23	0.21	0.98
		MEMIT-Merge	0.71	0.44	0.98
	distinct-subject	FT	0.73	0.58	0.24
		MEMIT	0.99	0.91	0.82
		PMET	0.46	0.46	0.92
		MEMIT-Merge	1.00	0.91	0.81
Qwen2.5-7B-Instruct	same-subject	FT	0.28	0.23	0.99
		MEMIT	0.31	0.25	1.00
		PMET	0.23	0.18	0.99
		MEMIT-Merge	0.67	0.43	0.99
	distinct-subject	FT	0.23	0.22	0.98
		MEMIT	0.99	0.84	0.91
		PMET	0.52	0.47	0.84
		MEMIT-Merge	1.00	0.86	0.90

Table 3: results of models in various size and architecture when batch size is 100

G Related Work

Knowledge editing techniques for large language models (LLMs) primarily fall into two paradigms: non-parametric approaches that preserve original parameters and parametric methods that directly modify model weights. Parametric approaches, while effective for targeted updates, often introduce uncontrolled parameter perturbations that adversely affect unrelated knowledge — a challenge addressed through various constraint mechanisms. The parametric category features two dominant subclasses: One is "Meta-Learning Based Methods", such as MEND (Mitchell et al., 2022) and MALMEN (Tan et al., 2024) which train meta-networks using carefully designed datasets containing both unrelated knowledge samples and paraphrased sentences, aiming to enhance generalization while minimizing collateral damage. Another is Locate-and-Edit Methods, which includes techniques such as Knowledge Neuron (KN) (Dai et al., 2022), identify critical knowledge storage locations before executing precise edits. ROME (Meng et al., 2022) extends this by incorporating knowledge preservation terms in its optimization objective to maintain model integrity.

Our work builds upon MEMIT (Meng et al., 2023), a state-of-the-art locate-and-edit approach that enables batch knowledge editing through MLP layer modifications. Building on MEMIT, many recent methods have made modifications to parameter update methods during editing or to the architecture and location of the edits. PMET (Li et al., 2024) incorporates the output of the attention layer in the calculation of parameter updates. AlphaEdit (Fang et al., 2024) improves upon MEMIT's parameter matrix update method by projecting the update matrix into the null space of the original knowledge to mitigate interference with unrelated knowledge. UNKE (Deng et al., 2024) extends structured knowledge editing to unstructured editing.