Model Merging for Knowledge Editing

Zichuan Fu[1]*, Xian Wu[2]†, Guojing Li[1], Yingying Zhang[2], Yefeng Zheng[2,3], Tianshi Ming[4], Yejing Wang[1], Wanyu Wang[1], Xiangyu Zhao[1]†

[1] City University of Hong Kong [2] Tencent Jarvis Lab
[3] Westlake University [4] Tongji University
zc.fu@my.cityu.edu.hk, kevinxwu@tencent.com, xianzhao@cityu.edu.hk

Abstract

Large Language Models (LLMs) require continuous updates to maintain accurate and current knowledge as the world evolves. While existing knowledge editing approaches offer various solutions for knowledge updating, they often struggle with sequential editing scenarios and harm the general capabilities of the model, thereby significantly hampering their practical applicability. This paper proposes a two-stage framework combining robust supervised fine-tuning (R-SFT) with model merging for knowledge editing. Our method first fine-tunes the LLM to internalize new knowledge fully, then merges the fine-tuned model with the original foundation model to preserve newly acquired knowledge and general capabilities. Experimental results demonstrate that our approach significantly outperforms existing methods in sequential editing while better preserving the original performance of the model, all without requiring any architectural changes. Code is available at Applied-Machine-Learning-Lab/MM4KE.

1 Introduction

Large Language Models (LLMs) have revolutionized Natural Language Processing (NLP) by capturing vast amounts of world knowledge and exhibiting impressive generalization capabilities (Zhao et al., 2024; Fu et al., 2024; Xu et al., 2024a). Recent advancements in both architecture design and training strategies have enabled LLMs such as GPT-4 (OpenAI et al., 2024) to engage in human-like dialogue and solve complex real-world problems.

However, when deployed in dynamic real-world environments, LLMs often face challenges of maintaining current and accurate knowledge (Wang et al., 2024a). For example, models can quickly become outdated regarding political developments, technological innovations, or evolving natural disasters; they may also retain inaccurate historical details or harmful content that needs timely removal to ensure safe and reliable outputs.

To tackle these challenges, knowledge editing has emerged as an effective solution for efficiently updating or correcting specific information in pre-trained language models. These approaches can be broadly categorized into three main categories (Zhang et al., 2024c). Memory-based methods primarily rely on fine-tuning mechanisms to store and update knowledge in the model's parameters (Hartvigsen et al., 2023). Meta-learning approaches leverage auxiliary networks to learn how to generate precise weight updates for knowledge editing. Locate-then-edit methods directly identify and modify specific components within the model architecture to update factual associations. Each of these approaches offers distinct strategies for modifying model behavior.

However, these existing approaches still face several significant limitations. First, most editing methods exhibit poor performance in sequential editing and often suffer from weak generalization capabilities. As a result, they struggle to effectively inject large amounts of knowledge into the models, limiting their practical applicability (Wang et al., 2024b; Zhang et al., 2024a). Second, after knowledge editing, models often experience degradation in their general capabilities, as the editing process typically focuses only on targeted knowledge without considering its impact on unrelated knowledge (Meng et al., 2022, 2023).

To address the above limitations, we propose a simple yet effective knowledge editing framework integrating Robust Supervised Fine-Tuning (R-SFT) with Model Merging techniques. Specifically, we employ R-SFT, a fine-tuning strategy that selectively optimizes only the Feed-Forward Networks (FFNs) in a single transformer layer. We use iterative sample-wise optimization paired with

*Work was conducted during the internship at Tencent Jarvis Lab.
†Corresponding authors.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track), pages 433–443
July 28-30, 2025

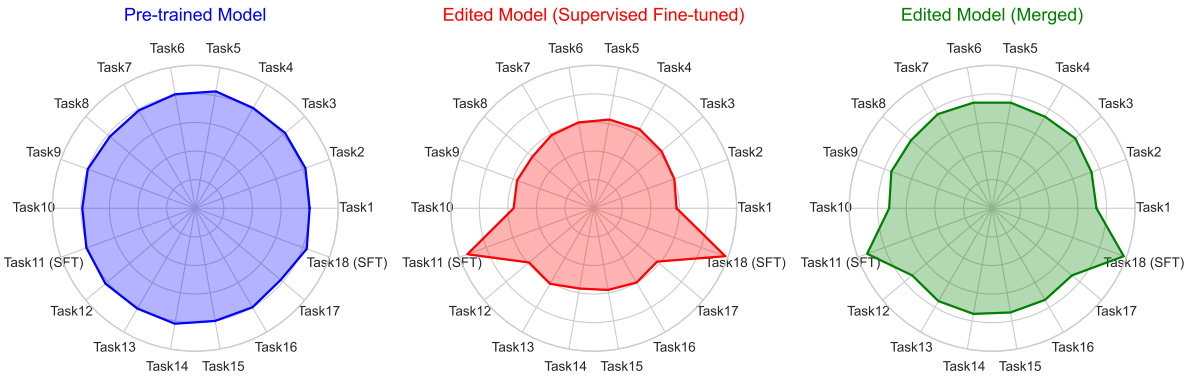

Figure 1: The illustration of three radar charts demonstrates the performance distribution across multiple tasks. The left chart shows the pre-trained model excelling in general tasks but limited in specific tasks (SFT). The middle chart represents the fine-tuned model with enhanced specific task performance at the cost of general capabilities. The right chart illustrates the merged model that successfully maintains both general and specific task performance.

an early-stopping mechanism to avoid overfitting. Subsequently, we merge the fine-tuned model with the original foundation model through scaling and sparsity-driven pruning, recovering general capabilities compromised during fine-tuning while effectively retaining acquired factual edits. Extensive experimental evaluations demonstrate significant performance improvements over existing methods across sequential editing tasks, superior preservation of general capabilities, and no architectural modifications are required.

- We propose R-SFT, an efficient fine-tuning approach leveraging sample-wise iterative optimization with early stopping to ensure precise and efficient knowledge acquisition.
- We apply model merging to mitigate the negative impact of fine-tuning on the general capabilities of LLMs, providing a simple but effective solution without any architectural modifications.
- Experimental results show that our method outperforms existing approaches in sequential editing while maintaining the general capabilities.

2 Methodology

This section introduces the proposed two-stage framework for knowledge editing, which includes R-SFT and model merging.

2.1 Robust Supervised Fine-tuning

Existing knowledge editing methods face significant challenges in sequential edits, often requiring complex architectural modifications that limit their practical applicability. Therefore, in the first stage of our framework, we propose Robust Supervised Fine-tuning (R-SFT), a robust knowledge learning fine-tuning paradigm designed to overcome these limitations while maintaining simplicity and effectiveness, as detailed in Algorithm 1.

Algorithm 1 Procedure of Robust Supervised Fine-Tuning (R-SFT)

Require: Foundation model θ_{base}, dataset $\mathcal{D} = \{s_n\}_{n=1}^{N}$, learning rate η, early stop threshold τ, max epochs E, max steps per sample K

Initialize model parameters: $\theta^{(0)} \leftarrow \theta_{\text{base}}$
Set global iteration counter: $t \leftarrow 0$
for $e = 1$ **to** E **do** ▷ Iterate epochs
  for $n = 1$ **to** N **do** ▷ Iterate samples
    for $k = 1$ **to** K **do** ▷ Iterative steps
      $\mathcal{L}_n = -\log P(\mathbf{a}_n|\mathbf{q}_n; \theta^{(t)})$
      if $\mathcal{L}_n < \tau$ **then** ▷ Early stopping
        break
      else
        $\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta \nabla_\theta \mathcal{L}_n$
        $t \leftarrow t + 1$
      end if
    end for
  end for
end for
return fine-tuned parameters $\theta_{\text{sft}} \leftarrow \theta^{(t)}$

Specifically, given a pre-trained foundation model θ_{base} and an editing dataset $\mathcal{D} = \{(\mathbf{q}_n, \mathbf{a}_n)\}_{n=1}^{N}$, where each sample includes a question $\mathbf{q}_n$ and its corresponding targeted answer $\mathbf{a}_n$, R-SFT aims to update the model parameters to encode the provided factual information accurately. The objective follows the standard supervised fine-tuning (SFT), minimizing the negative

log-likelihood of the correct output given the input:

$$\mathcal{L}_n(\theta) = -\log P(\mathbf{a}_n|\mathbf{q}_n;\theta) \quad (1)$$

For each sample, we iteratively update the parameters via gradient descent with learning rate η:

$$\theta^{(t+1)} = \theta^{(t)} - \eta\nabla_\theta\mathcal{L}_n(\theta^{(t)}) \quad (2)$$

where t is the global iteration counter.

The key difference between R-SFT and conventional SFT is the sample-level consecutive training with an early-stop mechanism. In each epoch, each sample is optimized consecutively for at most K steps, stopping early if the loss decreases below the threshold τ:

$$k_n^* = \min\{k \mid \mathcal{L}_n(\theta^{(t+k)}) < \tau \text{ and } 1 \le k \le K\} \quad (3)$$

where k_n^* denotes the real number of gradient update steps performed on the n-th sample within the epoch. A sample that satisfies the early stop criterion remains available in subsequent epochs, allowing periodic validation to avoid forgetting.

Furthermore, based on insights from existing research (Meng et al., 2022), we restrict R-SFT solely to the Feed-Forward Networks (FFN) of the fifth transformer layer, which has been proven to be optimal for editing performance and efficiency.

After completing the R-SFT process over E epochs, we obtain a fine-tuned model θ_{sft} that thoroughly and reliably captures the desired knowledge edits. This fine-tuned model, along with the original pre-trained foundation model θ_{base}, forms the foundation for our subsequent merging stage.

2.2 Model Merging

In the second stage, the fine-tuned model is merged with the foundation model. While R-SFT effectively teaches the model new knowledge, it typically comes at the cost of degrading the model's general capabilities. Therefore, we employ model merging, including scaling and pruning, to restore these fundamental capabilities while preserving the newly acquired knowledge.

Our merging approach employs a weighted average of the original and fine-tuned models, essentially applying **scaling** to the fine-tuned model:

$$\theta_{\text{edited}} = \alpha\theta_{\text{base}} + (1-\alpha)\theta_{\text{sft}}, \alpha \in (0,1) \quad (4)$$

where a scaling parameter controls the preservation-editing trade-off. This equation can be further reformulated to highlight the parameter difference:

$$\theta_{\text{edited}} = \theta_{\text{base}} + (1-\alpha)(\theta_{\text{sft}} - \theta_{\text{base}}) \quad (5)$$

where $\Delta\theta = \theta_{\text{sft}} - \theta_{\text{base}}$ represents the knowledge delta, the parameter changes that encode the new knowledge acquired during R-SFT.

To further reduce the interference of knowledge delta on general capabilities, we apply **pruning** to the knowledge delta:

$$\theta_{\text{edited}} = \theta_{\text{base}} + (1-\alpha)\cdot \text{Top}_p(\theta_{\text{sft}} - \theta_{\text{base}}) \quad (6)$$

The pruning operation keeps the top $p\%$ of parameters with the highest magnitude changes in each parameter matrix, while setting the rest to zero.

This process induces a high degree of sparsity in the knowledge delta, ensuring that only the most impactful modifications are retained. Such sparsity not only reduces the risk of interference with the pretrained model's general capabilities, but also suppresses noisy updates introduced by training samples or the fine-tuning process.

Finally, the merged model can preserve general capabilities, while effectively incorporating the newly acquired knowledge from R-SFT.

2.3 Industrial Application Prospect

Real-world industry applications require specialized LLMs capable of performing domain-specific tasks without losing foundational general-purpose capabilities such as comprehension and logic reasoning. Foundation models typically lack domain-specific accuracy, while traditional fine-tuning methods introduce significant limitations: fine-tuning solely on vertical data often causes catastrophic forgetting (Luo et al., 2023), whereas hybrid training with extensive general and domain data incurs prohibitive computational costs.

The proposed R-SFT enables efficient domain-specific data optimization. Meanwhile, the model merging strategy combines the fine-tuned domain-specific models and the foundation model, thereby integrating specialized domain knowledge without sacrificing general linguistic reasoning capabilities. We have successfully delivered multiple specialized models tailored to distinct professional domains, demonstrating improved performance on their targeted tasks and maintaining the general language processing competencies necessary for practical industrial applications.

Table 1: Performance comparison of merging methods for sequential knowledge editing. The best values are highlighted in bold, while the second-best values are underlined. Column "Base" represents the foundation model.

DataSet	Metric	Base	KN	ROME	MEMIT	LoRA	SFT	R-SFT	Merged
Edited Knowledge									
ZsRE	Edit Succ. ↑	-	6.66	14.53	3.11	98.06	99.39	**99.82**	96.95
	Generalization ↑	-	6.79	12.53	3.09	73.52	85.13	**93.29**	91.58
	Portability ↑	-	10.43	2.32	1.06	20.90	24.40	**47.48**	39.63
	Locality ↑	-	7.54	1.13	1.20	5.28	12.65	**36.69**	26.42
	Fluency ↑	-	421.73	**535.50**	477.30	411.80	414.58	441.53	420.49
General Capabilities									
C-Eval	Accuracy ↑	**79.57**	25.78	24.59	25.11	70.43	31.43	78.97	79.35
CoQA	EM ↑	56.82	24.42	0.00	0.00	53.98	0.63	51.80	**62.10**
	F1 ↑	72.60	34.13	0.07	0.00	69.10	1.39	63.57	**75.18**
DROP	EM ↑	0.23	0.03	0.00	0.00	**1.96**	0.09	0.67	1.9
	F1 ↑	7.10	2.07	0.32	0.00	**13.90**	0.21	8.23	10.8
SQuAD 2.0	EM ↑	10.02	0.33	1.02	**43.80**	11.03	5.15	8.20	17.82
	F1 ↑	21.15	3.15	1.08	**43.80**	22.45	5.39	12.90	25.02
LogiQA	Accuracy ↑	**37.94**	21.51	20.28	22.12	31.03	24.12	24.42	33.03

3 Experiments

In this section, our experiments are structured around the following research questions (RQs):

- **RQ1:** How does our model merging approach perform on the ZsRE dataset compared to baseline methods, and how does it impact the model's general capabilities?
- **RQ2:** How effective is our model merging approach across other knowledge editing datasets in KnowEdit?
- **RQ3:** How hyperparameter settings for robust model fine-tuning affect the accuracy and generalization ability of knowledge editing.
- **RQ4:** How do different components of our framework individually contribute to the overall performance of the edited model?

3.1 Experimental Settings

3.1.1 Datasets

We select KnowEdit (Zhang et al., 2024c) for knowledge editing tasks, mainly on ZsRE dataset (Levy et al., 2017). For general ability evaluation, we use C-Eval (Huang et al., 2023b), CoQA (Reddy et al., 2019), DROP (Dua et al., 2019), SQuAD 2.0 (Rajpurkar et al., 2018) and LogiQA (Liu et al., 2020).

3.1.2 Baselines

In our experiments, we compare our approach against two main categories of locate-then-edit methods: 1) classic knowledge editing methods (ROME (Meng et al., 2022), MEMIT (Meng et al., 2023)) that directly modify model parameters associated with specific facts, and 2) fine-tuning approaches (LoRA (Hu et al., 2021)) that update knowledge through training.

3.1.3 Implementation Details

We conduct experiments using EasyEdit (Zhang et al., 2024b) for evaluating various knowledge editing methods, and employ the lm-evaluation-harness[1] for assessing general model capabilities. R-SFT is implemented through LLaMA Factory (Zheng et al., 2024) and mergeKit (Goddard et al., 2024) for training and merging respectively. We use Qwen2.5-7B-Instruct (Yang et al., 2024) as our foundation model.

3.1.4 Evaluation Metrics

We evaluate the models using two sets of metrics. To evaluate editing performance, we use five metrics: Edit Success (Edit Succ. or Succ.), Generalization (Gen.), Portability (Port.), Locality (Loc.) and Fluency (Flu.). The detailed definitions are provided in Appendix A.3. To assess the preservation of general capabilities, we use Accuracy for classification tasks (C-Eval, LogiQA), and both Exact Match (EM) and F1 scores for question-answering benchmarks (CoQA, DROP, SQuAD 2.0).

[1] https://github.com/EleutherAI/lm-evaluation-harness

Table 2: Editing performance on additional KnowEdit datasets using our framework.

DataSet	Metric ↑	SFT	R-SFT	Merged
WikiData$_{recent}$	Edit Succ.	79.46	**99.97**	96.62
	Portability	46.59	58.26	**62.95**
	Locality	28.50	31.87	**41.62**
	Fluency	428.95	461.51	**592.02**
WikiBio	Edit Succ.	66.06	**99.48**	96.54
	Locality	40.16	64.30	**75.18**
	Fluency	626.60	**628.77**	626.71
WikiData$_{counter}$	Edit Succ.	50.67	**99.06**	84.02
	Portability	34.56	**60.61**	51.98
	Locality	15.75	26.36	**41.98**
	Fluency	479.81	601.02	**614.64**

Table 3: Effect of different hyperparameter settings on the editing performance.

(a) Early stopping loss threshold.

Threshold	Succ.	Gen.	Port.	Loc.	Flu.
None	68.90	65.76	24.40	12.65	**514.58**
0.01	75.74	73.28	39.86	27.84	435.20
0.02	78.06	74.87	41.77	26.14	437.26
0.05	79.61	76.22	42.53	33.00	420.41
0.1	**80.07**	**76.76**	44.33	32.18	400.84
0.2	78.87	75.04	**46.14**	**34.76**	411.97

(b) Number of epochs and steps.

Epochs	Steps	Succ.	Gen.	Port.	Loc.	Flu.
1	30	75.74	73.28	39.86	27.84	435.20
2	15	93.89	89.94	40.96	26.33	422.18
3	10	96.95	91.58	39.63	26.42	420.49
5	6	99.42	93.56	41.81	25.84	439.81
10	3	99.82	**93.56**	43.50	30.48	417.75
30	1	**99.84**	93.30	**46.87**	**33.81**	**509.18**

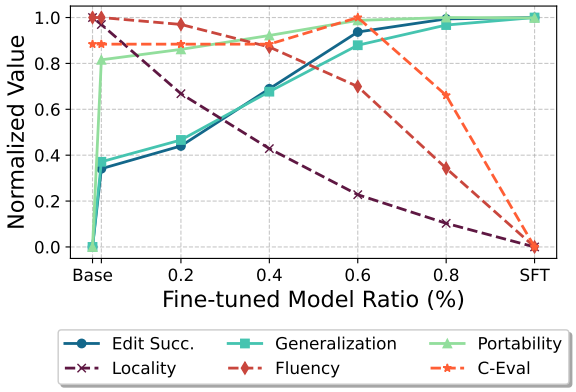

Figure 2: Metrics across different scaling ratios, illustrating the trade-off between edited and general knowledge.

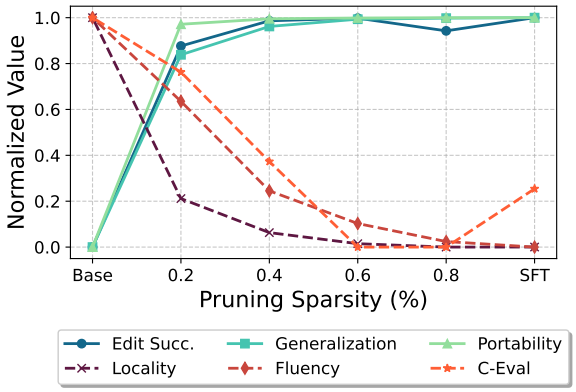

Figure 3: Metrics across different pruning sparseness, balancing edited and general knowledge.

3.2 Overall Performance (RQ1)

As shown in Table 1, our empirical evaluation reveals several important findings regarding knowledge editing performance and preservation of general capabilities across different methods.

For knowledge editing, R-SFT exhibits superior editing performance across primary metrics, with the merged model maintaining the second-highest performance in most editing dimensions. Regarding general capabilities, the merged model effectively retains the foundation model's general capabilities, demonstrating comparable performance on C-Eval and enhanced results on CoQA. This suggests our merging strategy successfully addresses the common trade-off between knowledge editing and general capability preservation.

Notably, MEMIT performs surprisingly well on SQuAD 2.0, and LoRA achieves strong results on DROP. This is largely because the foundation model originally performed poorly on these tasks, making it more sensitive to minor perturbations introduced during editing. These edits may alter the model's answering behavior in a way that coincidentally improves the evaluation metrics, rather than reflecting true methodological superiority.

3.3 Knowledge Editing Performance (RQ2)

Table 2 summarizes the performance of our proposed R-SFT approach and the subsequent merging step across various knowledge editing datasets in Knowedit. We observe that R-SFT consistently achieves near 100% accuracy on the training samples and maintains approximately 60% portability to reason with new knowledge, significantly outperforming conventional fine-tuning methods.

After model merging, the edited model consistently experiences a modest reduction (around 5%) in editing accuracy, but this is acceptable given the restoration of the model's general capabilities. The complete result is provided in the Appendix B.

Table 4: Ablation study of the framework on editing performance (including success rate, generalization, portability, locality, and fluency) and general capabilities based on C-Eval (Acc.), CoQA (F1), and LogiQA (Acc.).

Stage	Methods	Succ.	Gen.	Port.	Loc.	Flu.	C-Eval	CoQA	LogiQA
Base		-	-	-	-	-	79.57	72.60	37.94
R-SFT	w/o Sample Steps	**99.82**	93.85	**47.32**	**35.03**	466.00	**44.28**	**63.57**	24.73
	w/o Early Stop	**99.82**	**93.95**	41.10	31.51	**534.19**	40.04	53.11	23.81
	Complete	99.43	93.70	45.93	33.96	401.44	41.60	58.84	**26.57**
Merging	w/o Scaling	98.25	92.36	45.14	33.96	411.70	58.47	62.00	32.41
	w/o Pruning	96.97	92.07	42.76	29.69	418.32	52.75	74.65	29.80
	Complete	96.95	91.58	39.63	26.42	420.49	68.42	78.07	34.25

3.4 Parameter Analysis (RQ3)

R-SFT. As shown in Tables 3a, stopping training early (lower thresholds) improves generalization by preventing overfitting. A moderate threshold of 0.1 strikes the optimal balance between gaining knowledge and preventing overfitting. The results in Tables 3b confirm that fewer steps per sample yield better performance. However, this approach requires absolute $E \times N \times K$ update steps, resulting in lower computational efficiency. Finally, five epochs with six steps per sample provide an optimal compromise. Appendix C shows complete results for all hyperparameters.

Model Merging. Figure 2 and Figure 3 demonstrate that scaling has a more immediate and pronounced impact on model performance, with an optimal setting typically around 0.8 to balance knowledge updates and generalization. In contrast, pruning exhibits a more subtle influence, and a sparsity ratio of 0.2 is generally preferred to minimize interference while preserving core capabilities.

3.5 Ablation Study (RQ4)

We conduct an ablation study to evaluate the individual contributions of each proposed component, as presented in Table 4. Results show that removing the sample-wise consecutive update ("w/o Sample Steps") does not significantly harm editing performance, suggesting that our iterative update strategy does not negatively impact model quality while considerably enhancing efficiency. In contrast, removing early stopping ("w/o Early Stop") significantly degrades the model's general capabilities, confirming its essential role in preventing overfitting. In the model merging stage, omitting either scaling ("w/o Scaling") or pruning ("w/o Pruning") leads to decreased restoration of general capabilities, highlighting the importance of these techniques in effectively balancing knowledge editing and general model performance.

4 Related Works

4.1 Knowledge Editing

Knowledge editing aims to efficiently update or modify the internal knowledge of machine learning models to adapt to rapidly changing real-world information (Zhao et al., 2018a,b). This is particularly important for LLMs, whose training demands substantial computational resources and time, making frequent pretraining impractical (Xu et al., 2024b). Early studies focused on knowledge tracing to analyze and locate factual information stored within models before attempting edits (Huang et al., 2023a; Liu et al., 2023; Li et al., 2024). ROME (Meng et al., 2022) fisrt directly modified neurons associated with specific facts in feed-forward layers. While ROME models can edit certain facts accurately, many real-life situations involve dynamic information that require perpetual model updates (Liu et al., 2024, 2025). This necessitates the development of editing techniques that support persistent change Subsequent approaches, like MEMIT (Mitchell et al., 2022a) and r-ROME (Gupta et al., 2024), enhanced editing precision and stability during sequential updates.

Other methods utilized fine-tuning on specialized datasets (Xu et al., 2024b), effectively injecting knowledge but risking general capability degradation due to overfitting. Meta-learning approaches (e.g., MEND (Mitchell et al., 2021), InstructEdit (Huang et al., 2021)) and memory-based methods (e.g., SERAC (Mitchell et al., 2022b), MELO (Li et al., 2023b)) achieved better generalization but introduced auxiliary networks or structured memories, significantly increasing model complexity and limiting practical deployment.

4.2 Model Merging

Model merging techniques combine parameters from multiple models or training checkpoints into a unified model. This technique is more efficient than using several LLMs simultaneously (Li et al., 2023a; Lu et al., 2024). Early methods primarily relied on simple weight averaging (Wortsman et al., 2022), but subsequent work introduced more sophisticated strategies. For instance, SLERP (Kao et al., 2023) proposed spherical interpolation between model parameters to mitigate geometric distortion inherent in linear interpolation methods. Task Arithmetic (Gur et al., 2023), and its extensions, such as TIES (Jiang et al., 2023) and DARE (Chen et al., 2023), computed and combined task vectors, effectively tackling inter-model interference via sparsification, sign-consensus algorithms, adaptive pruning, and parameter rescaling. More recently, WISE (Wang et al., 2024b) applied sparsification methods to fine-tuning for knowledge editing, effectively balancing edited knowledge and pre-trained information, but also introduced increased structural complexity.

5 Conclusion

In this paper, we propose a two-stage framework for knowledge editing that integrates robust supervised fine-tuning (R-SFT) with model merging. Specifically, R-SFT first leverages sample-wise iterative updates and an early-stopping mechanism to precisely inject new knowledge with enhanced generalization. Subsequently, the model merging technique serves to further mitigate the harm of fine-tuning by merging the pre-trained model with the R-SFT model, thus negating the necessity for architectural changes. Experimental results show that our method significantly outperforms existing approaches in sequential editing scenarios while maintaining general capabilities.

6 Limitations

Although our model merging approach demonstrates significant effectiveness in knowledge editing, we acknowledge certain limitations in knowledge generalization capabilities. Our current framework, while successful at direct knowledge updates, shows reduced performance when transferring edited knowledge to substantially different phrasings or when applying reasoning based on newly acquired information. The generalization metrics indicate room for improvement in how edited knowledge is applied across varied contexts. Future research should focus on developing more sophisticated knowledge insertion methods that enhance the transferability of edited information.

Acknowledgments

This research was partially supported by Research Impact Fund (No.R1015-23), Collaborative Research Fund (No.C1043-24GF) and Tencent (CCF-Tencent Open Fund, Tencent Rhino-Bird Focused Research Program).

References

Yihan Chen, Dongkuan Zhang, Xiang Wang, Yifan Yang, and Heng Wang. 2023. Dare: ldirect parameter editing for adaptive mode reconfiguration. *arXiv preprint arXiv:2310.09570*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Zichuan Fu, Xiangyang Li, Chuhan Wu, Yichao Wang, Kuicai Dong, Xiangyu Zhao, Mengchen Zhao, Huifeng Guo, and Ruiming Tang. 2024. A unified framework for multi-domain ctr prediction via large language models. *ACM Trans. Inf. Syst.* Just Accepted.

Charles Goddard, Shamane Siriwardhana, Malikeh Ehghaghi, Luke Meyers, et al. 2024. Arcee's mergekit: A toolkit for merging large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: EMNLP 2024 - Industry Track, Miami, Florida, USA, November 12-16, 2024*, pages 477–485. Association for Computational Linguistics.

Akshat Gupta, Sidharth Baskaran, and Gopala Anumanchipalli. 2024. Rebuilding ROME : Resolving model collapse during sequential model editing. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21738–21744, Miami, Florida, USA. Association for Computational Linguistics.

Itay Gur, Wei-Cheng Kao, Elias Polymenakos, and Sujith Ravi. 2023. Task arithmetic in the tangent space: Improved editing of pre-trained models. *arXiv preprint arXiv:2305.17651*.

Thomas Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. 2023.

Aging with GRACE: Lifelong model editing with discrete key-value adaptors. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Shuyan Huang, Zitao Liu, Xiangyu Zhao, Weiqi Luo, and Jian Weng. 2023a. Towards robust knowledge tracing models via k-sparse attention. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, page 2441–2445, New York, NY, USA. Association for Computing Machinery.

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, Jiayi Lei, Yao Fu, Maosong Sun, and Junxian He. 2023b. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. In *Advances in Neural Information Processing Systems*.

Zhiwei Huang, Jiacheng Li, Ninghao Ding, Ramesh Nallapati, and Dan Roth. 2021. Instructedit: Learning to edit language models with natural language instructions. *arXiv preprint arXiv:2212.10560*.

Zihao Jiang, Xiao Liu, Yibing Zhang, Hao Chen, and Stan Z Li. 2023. Ties: Temporal interference-free editing strategy for continual learning. *arXiv preprint arXiv:2310.18356*.

Wei-Cheng Kao, Itay Gur, Elias Polymenakos, Kushal Bansal, and Sujith Ravi. 2023. Slerp: Spherical linear interpolation between neural networks. *arXiv preprint arXiv:2305.17493*.

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.

Xiaopeng Li, Lixin Su, Pengyue Jia, Xiangyu Zhao, Suqi Cheng, Junfeng Wang, and Dawei Yin. 2023a. Agent4ranking: Semantic robust ranking via personalized query rewriting using multi-agent llm. *Preprint*, arXiv:2312.15450.

Xueyi Li, Youheng Bai, Teng Guo, Zitao Liu, Yaying Huang, Xiangyu Zhao, Feng Xia, Weiqi Luo, and Jian Weng. 2024. Enhancing length generalization for attention based knowledge tracing models with linear biases. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 5918–5926. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Zhenyu Li, Zhi Chen, Yeyun Wang, Yue Feng, Jian Li, Dongsheng Zhao, and Ji-Rong Wen. 2023b. Melo: Memory-efficient llm optimization. *arXiv preprint arXiv:2312.02428*.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3622–3628. International Joint Conferences on Artificial Intelligence Organization. Main track.

Qidong Liu, Xian Wu, Wanyu Wang, Yejing Wang, Yuanshao Zhu, Xiangyu Zhao, Feng Tian, and Yefeng Zheng. 2025. Llmemb: Large language model can be a good embedding generator for sequential recommendation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(11):12183–12191.

Qidong Liu, Xian Wu, Yejing Wang, Zijian Zhang, Feng Tian, Yefeng Zheng, and Xiangyu Zhao. 2024. LLM-ESR: large language models enhancement for long-tailed sequential recommendation. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Zitao Liu, Qiongqiong Liu, Teng Guo, Jiahao Chen, Shuyan Huang, Xiangyu Zhao, Jiliang Tang, Weiqi Luo, and Jian Weng. 2023. Xes3g5m: A knowledge tracing benchmark dataset with auxiliary information. In *Advances in Neural Information Processing Systems*, volume 36, pages 32958–32970. Curran Associates, Inc.

Jinliang Lu, Ziliang Pang, Min Xiao, Yaochen Zhu, Rui Xia, and Jiajun Zhang. 2024. Merge, ensemble, and cooperate! a survey on collaborative strategies in the era of large language models. *Preprint*, arXiv:2407.06089.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2023. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *CoRR*, abs/2308.08747.

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 Andonian, Yonatan Belinkov, and David Bau. 2023. Mass-editing memory in a transformer. *Preprint*, arXiv:2210.07229.

Eric Mitchell, Kee Siew Lee, Hao Chen, Kevin Meng, David Bau, and Yonatan Belinkov. 2022a. Memory editing via model editing: Memory editing in large language models. *arXiv preprint arXiv:2210.07229*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. 2021. 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.

OpenAI, Josh Achiam, Steven Adler, et al. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. 2024a. Knowledge editing for large language models: A survey. *ACM Comput. Surv.*, 57(3).

Zihao Wang, Yihua Chen, Hao Xie, Xiang Li, Ningyu Zhang, and Huajun Chen. 2024b. Wise: Memory-efficient model editing with task-aware compression. *arXiv preprint arXiv:2401.12174*.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 23965–23998. PMLR.

Derong Xu, Ziheng Zhang, Zhenxi Lin, Xian Wu, Zhihong Zhu, Tong Xu, Xiangyu Zhao, Yefeng Zheng, and Enhong Chen. 2024a. Multi-perspective improvement of knowledge graph completion with large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 11956–11968, Torino, Italia. ELRA and ICCL.

Derong Xu, Ziheng Zhang, Zhihong Zhu, Zhenxi Lin, Qidong Liu, Xian Wu, Tong Xu, Wanyu Wang, Yuyang Ye, Xiangyu Zhao, Enhong Chen, and Yefeng Zheng. 2024b. Editing factual knowledge and explanatory ability of medical large language models. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, page 2660–2670, New York, NY, USA. Association for Computing Machinery.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. 2024. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Ningyu Zhang, Bozhong Tian, Siyuan Cheng, Xiaozhuan Liang, Yi Hu, Kouying Xue, Yanjie Gou, Xi Chen, and Huajun Chen. 2024a. Instructedit: instruction-based knowledge editing for large language models. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, IJCAI '24.

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. 2024b. A comprehensive study of knowledge editing for large language models. *Preprint*, arXiv:2401.01286.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, et al. 2024c. A comprehensive study of knowledge editing for large language models. *Preprint*, arXiv:2401.01286.

Wayne Xin Zhao, Kun Zhou, Junyi Li, et al. 2024. A survey of large language models. *Preprint*, arXiv:2303.18223.

Xiangyu Zhao, Long Xia, Liang Zhang, Zhuoye Ding, Dawei Yin, and Jiliang Tang. 2018a. Deep reinforcement learning for page-wise recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems*, RecSys '18, page 95–103, New York, NY, USA. Association for Computing Machinery.

Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Long Xia, Jiliang Tang, and Dawei Yin. 2018b. Recommendations with negative feedback via pairwise deep reinforcement learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, page 1040–1048, New York, NY, USA. Association for Computing Machinery.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, and Yongqiang Ma. 2024. Llamafac-

tory: Unified efficient fine-tuning of 100+ language models. *CoRR*, abs/2403.13372.

A Detailed Experimental Settings

A.1 Datasets

KnowEdit (Zhang et al., 2024c) contains a total of six sub-datasets including Wiki$_{recent}$, ZsRE, WikiBio, WikiData$_{counterfact}$, Convsent and Sanitation.

For general ability evaluation, C-Eval (Huang et al., 2023b) primarily assesses common knowledge, while other benchmarks are predominantly question-answering datasets designed to evaluate models' capabilities in extended conversations with longer textual contexts.

A.2 Implementation Details

During the training phase, we utilize a batch size of 1 to maximize the effective learning from each individual sample. Our R-SFT is configured with 5 epochs and 6 consecutive steps, employing a maximum learning rate of 5×10^{-4}.

A.3 Evaluation Metrics

For evaluating the editing performance of the merged models, we adopt four widely used metrics:

- **Edit Succ. (Succ.):** This metric quantifies whether the intended factual update is correctly reflected in the model's output when given the edited query.
- **Generalization (Gen.):** This metric evaluates whether the model can correctly apply the updated factual knowledge when presented with semantically equivalent queries.
- **Portability (Port.):** This measures the ability of the edited model to generalize the new knowledge to alternative phrasings or reworded versions of the original query.
- **Locality (Loc.):** Locality evaluates whether the editing process is confined to the targeted knowledge, ensuring that the model's outputs for unrelated queries remain unchanged.
- **Fluency (Flu.):** This metric assesses the linguistic quality of the model's responses, verifying that the edited outputs are coherent and natural.

To comprehensively assess the general capabilities of the models after knowledge editing, we employ several established benchmarks with the following metrics:

- **Accuracy:** For classification tasks such as C-Eval and LogiQA, we utilize accuracy as the primary metric, which measures the percentage of correctly answered questions.
- **Exact Match (EM):** For extractive question answering tasks including CoQA, DROP, and SQuAD 2.0, we report the Exact Match score, which requires the model's prediction to exactly match the ground truth answer:

$$\text{EM}(\mathbf{a}, \hat{\mathbf{a}}) = \mathbf{1}(\mathbf{a} = \hat{\mathbf{a}}) \tag{7}$$

  where $\mathbf{a}$ is the ground truth answer, $\hat{\mathbf{a}}$ is the model's prediction, and $\mathbf{1}(\cdot)$ is the indicator function that returns 1 if the condition is true and 0 otherwise.

- **F1 Score (F1):** For the same question answering tasks, we also report the F1 score, which measures the overlap between the predicted and ground truth answers at the token level:

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{8}$$

  where:

$$\text{Precision} = \frac{|\text{Tokens in } \hat{\mathbf{a}} \cap \text{Tokens in } \mathbf{a}|}{|\text{Tokens in } \hat{\mathbf{a}}|} \tag{9}$$

$$\text{Recall} = \frac{|\text{Tokens in } \hat{\mathbf{a}} \cap \text{Tokens in } \mathbf{a}|}{|\text{Tokens in } \mathbf{a}|} \tag{10}$$

B Knowledge Editing Performance (RQ2)

Table 5 compares our approach against baseline knowledge editing methods. Our R-SFT consistently achieves the highest editing success rates while maintaining strong portability. The merged model, while showing slightly lower editing success than R-SFT, demonstrates superior locality and fluency, effectively balancing edit fidelity with preservation of general capabilities. Parameter-efficient methods (ROME, MEMIT, LoRA) that perform well in single-fact editing struggle significantly in sequential editing scenarios, highlighting our framework's advantage in practical applications requiring both accurate knowledge editing and maintained model quality.

C Parameter Analysis of R-SFT (RQ3)

Edited Layer Selection Table 6 presents the performance when editing different layers of the LLM. Layers 6 and 7 consistently outperform other layers across most metrics, with Layer 6 achieving the

Table 5: Performance comparison of merging methods for sequential knowledge editing. The best values are highlighted in bold, while the second-best values are underlined.

DataSet	Metric ↑	ROME	MEMIT	LoRA	SFT	R-SFT	Merged
WikiData$_{recent}$	Edit Succ.	15.78	0.00	1.11	79.46	**99.97**	96.62
	Portability	4.79	0.00	0.90	46.59	58.26	**62.95**
	Locality	1.76	0.00	0.06	28.50	31.87	**41.62**
	Fluency	529.98	478.64	505.02	428.95	461.51	**592.02**
WikiBio	Edit Succ.	26.47	0.04	53.26	66.06	**99.48**	96.54
	Locality	3.50	0.03	64.56	40.16	64.30	**75.18**
	Fluency	608.15	502.35	627.18	626.60	**628.77**	626.71
WikiData$_{counter}$	Edit Succ.	12.69	0.00	11.07	50.67	**99.06**	84.02
	Portability	2.88	0.00	10.28	34.56	**60.61**	51.98
	Locality	0.92	0.00	13.65	15.75	26.36	**41.98**
	Fluency	553.18	314.91	489.65	479.81	601.02	**614.64**

Table 6: Effect of edited layer selection on knowledge editing performance.

Layer	Succ.	Gen.	Port.	Loc.	Flu.
5	75.74	73.28	39.86	27.84	435.20
6	**85.49**	**83.38**	41.85	31.97	431.43
7	85.31	81.81	**44.08**	**34.61**	434.13
13	74.58	68.61	38.07	33.87	492.87
20	70.03	62.37	26.43	21.55	**497.90**
27	56.97	52.44	18.39	8.08	385.88

Table 7: Effect of maximum training steps per sample on editing performance.

Steps	Succ.	Gen.	Port.	Loc.	Flu.
30	75.74	73.28	39.86	27.84	435.20
60	75.74	73.28	39.86	27.84	435.20
90	75.74	73.28	39.86	27.84	435.20

highest edit success (85.49%) and generalization (83.38%). This result confirms findings from prior research that knowledge is more concentrated in the earlier layers of the LLM (Meng et al., 2022).

Training Steps Table 7 examines how many total steps are typically required to update each sample when early stopping is enabled. With early stopping enabled (loss threshold = 0.01), we observe that performance metrics remain identical across different maximum step settings. This indicates that typically within 30 steps the loss of one sample will converge.

Number of Edited Layers Table 8 investigates the impact of simultaneously editing multiple layers versus focusing on a single layer. Contrary to intuition, editing a single layer (Layer 5) yields substantially better results than editing multiple layers. Editing all layers leads to catastrophic performance degradation across all metrics. This suggests that targeted, minimal interventions are more effective for knowledge editing than widespread parameter modifications.

Table 8: Effect of the number of edited layers on editing performance.

Layers	Succ.	Gen.	Port.	Loc.	Flu.
Layer 5	**75.74**	**73.28**	**39.86**	**27.84**	**435.20**
Layers 4,5,6	66.96	62.95	28.36	16.64	409.75
All Layers	12.93	12.62	4.27	1.85	380.84

Table 9: Effect of learning rate (LR.) on editing performance.

LR.	Succ.	Gen.	Port.	Loc.	Flu.
5e-4	**75.74**	**73.28**	39.86	27.84	435.20
1e-4	67.68	61.75	**48.33**	41.55	516.84
5e-5	63.12	54.90	45.97	**44.11**	**556.84**

Learning Rate Table 9 examines how different learning rates affect the editing process. Our analysis reveals an interesting trade-off: higher learning rates (5e-4) improve edit success and generalization but reduce portability, locality, and fluency. Conversely, lower learning rates (5e-5) significantly enhance fluency and locality at the expense of edit success and generalization. This suggests that the optimal learning rate depends on which metrics are prioritized for a specific application.