Serial Lifelong Editing via Mixture of Knowledge Experts

Yu-Ju Cheng[1], Yu-Chu Yu[1], Kai-Po Chang[1], Yu-Chiang Frank Wang[1,2]
[1]National Taiwan University [2]NVIDIA
{b10901009, r09922104, f11942093}@ntu.edu.tw, frankwang@nvidia.com

Abstract

It is challenging to update Large language models (LLMs) since real-world knowledge evolves. While existing Lifelong Knowledge Editing (LKE) methods efficiently update sequentially incoming edits, they often struggle to precisely overwrite the outdated knowledge with the latest one, resulting in conflicts that hinder LLMs from determining the correct answer. To address this **S**erial **L**ifelong **K**nowledge **E**diting (sLKE) problem, we propose a novel Mixture-of-Knowledge-Experts scheme with an **A**ctivation-guided **R**outing **M**echanism (ARM), which assigns specialized experts to store domain-specific knowledge and ensures that each update completely overwrites old information with the latest data. Furthermore, we introduce a novel sLKE benchmark where answers to the same concept are updated repeatedly, to assess the ability of editing methods to refresh knowledge accurately. Experimental results on both LKE and sLKE benchmarks show that our ARM performs favorably against SOTA knowledge editing methods.

1 Introduction

Large language models (OpenAI, 2023; Touvron et al., 2023a,b) (LLMs) have demonstrated remarkable capabilities by leveraging the vast amount of knowledge acquired during large-scale pre-training. However, real-world knowledge is ever-changing (Yao et al., 2023), as a single fact often undergoes multiple and sequential updates over-time. This causes these models to become outdated over time. For example, as shown in Fig. 1, the answer to the question: "Which team is the NBA champion this year?" changes for each season, from Lakers in 2020 to Celtics in 2024. To benchmark this scenario, we introduce *serial Knowledge Lifelong Editing (sLKE)*, which focuses on sequentially updating answers to the same question. In contrast to the classic LKE setting explored by recent knowledge editing works (Huang et al., 2023; Hartvigsen et al., 2023; Wang et al., 2024), sLKE tackles the challenge of updating a single fact repeatedly over time, reflecting the more challenging scenario faced in real-world applications.

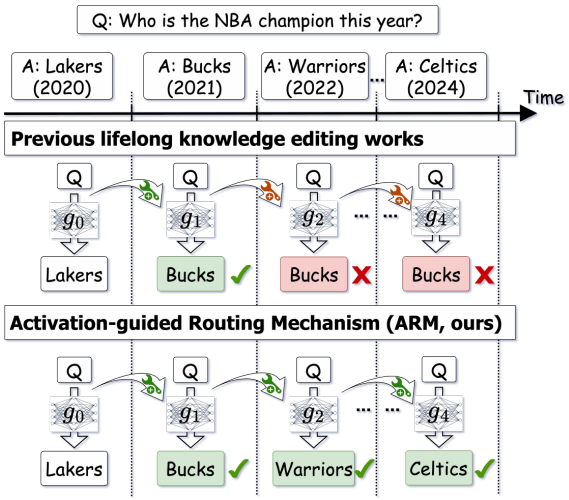

Figure 1: Compared to existing lifelong knowledge editing (LKE) works, our *ARM* enables serial lifelong knowledge editing (sLKE), allowing the edited model to respond with the correct up-to-date answers.

Previously, works on Knowledge Editing (KE) generally utilize batch-mode editing, where multiple concepts inside LLMs are simultaneously updated via a single edit. They either employ hyper-networks with meta-learning techniques (Vilalta and Drissi, 2002; Hospedales et al., 2022), or adopt locate-edit paradigms (Fang et al., 2024; Meng et al., 2023, 2022). However, such editing schemes tend to overfit on the latest batch of edits and struggle to generalize across a sequence of edits (Wang and Li, 2024; Tirumala et al., 2022).

To handle the scenario where each edit arrives sequentially, research works on Lifelong Knowledge Editing (LKE) have been proposed, which mainly leverage external memories to store edited concepts without modifying the original model parameters. For example, WISE (Wang et al., 2024) employs a Mixture-of-Experts (MoE) architecture by using

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 30888–30903

several side memories as experts to handle sequentially arriving edits and employs a routing mechanism to determine which expert to consult for a given query. Although WISE is capable of storing approximately thousands of edits, similar knowledge edits might be assigned to different memories, resulting in possible inference ambiguity (as illustrated in Fig. 2). Therefore, the above method cannot be easily extended to sLKE.

To tackle the problem of serial Lifelong Knowledge Editing, we propose a novel learning scheme of **M**ixture-**o**f-**K**nowledge-**E**xperts (MoKE), where each *knowledge expert* is dedicated to certain concepts. Moreover, we present an **A**ctivation-guided **R**outing **M**echanism (ARM) to specialize experts during sLKE training and inference. Specifically, we design an expert selection function that dynamically guides the router during training, ensuring that each incoming edit is assigned to the appropriate domain-specific expert. This strategy effectively mitigates expert ambiguity. Moreover, recognizing the absence of benchmarks for sLKE, we build the first benchmark to systematically test the effectiveness of editing methods to perform sLKE. Additionally, to reduce the memory increased usage brought by using multiple experts to memorize domain-specific knowledge, we integrate the model compression techniques (Duan et al., 2021; Stoica et al., 2024; Li et al., 2023a; Huang et al., 2024) that consolidate multiple experts into a single model. We summarize our contributions as follows:

- We introduce a practical setting for lifelong LLM editing, Serial Lifelong Knowledge Editing (sLKE), with a benchmark presented to assess the proposed learning scheme.
- We propose a learning scheme of Mixture-of-Knowledge-Experts (MoKE) with an Activation-guided Routing Mechanism (ARM), which enables memory-based experts to retain specialized knowledge and alleviates possible inference ambiguity.
- With a novel benchmark collected, we verify that our proposed framework performs favorably for both LKE and sLKE.

2 Related Work

2.1 Knowledge Editing

The goal of KE is to edit multiple facts within the model via a single edit. Their methods either employ hypernetworks (Yao et al., 2023; Zhang et al., 2024b) , or adopt the locate-and-edit paradigm to perform model editing. As the pioneering work for batch editing, MEND (Mitchell et al., 2022a) introduces the idea of using a hypernetwork which takes the gradients of LLM as input to generate the weight updates (Yao et al., 2023; Zhang et al., 2024b). MALMEN (Tan et al., 2023) further refined MEND by decomposing the gradient descent process, and thus reducing memory consumption. However, HyperNetwork-based methods suffer from overfitting (Tirumala et al., 2022) to the current batch, due to the extensive weight modifications. To address this issue, locate-edit methods have been proposed. These methods first specify the critical part of region parameters, and then they focus on making precise, localized modifications to model weights. For instance, ROME (Meng et al., 2022) employs a two-phase causal tracing approach to identify which feed-forward layer stores the target knowledge before applying edits at that specific location. MEMIT (Meng et al., 2023) tracks vector state values across layers to pinpoint the most relevant layer for modification. While these methods alleviate overfitting and reduce memory usage, they compromise the non-edited or previously edited knowledge in the sLKE setting.

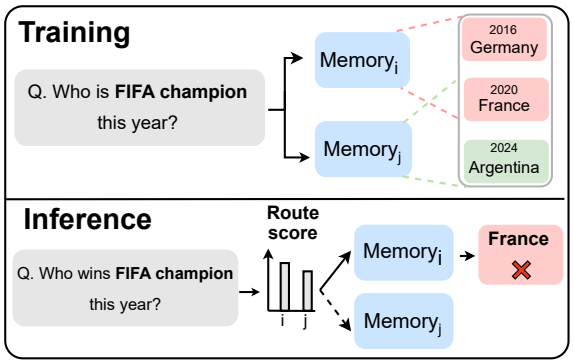

Figure 2: **Ambiguity of Mixture-of-Experts in LKE.** Without identifying proper experts for KE, output conflicts might be observed in lifelong settings.

2.2 Lifelong Knowledge Editing

The research works for LKE focus on a specialized setup of KE, where each edit sequentially arrives for fact updates. Compared to KE, LKE is more practical since outdated knowledge requires timely updates. The LKE methods leverage external memories to store the edited facts without modifying the original model weights. T-patcher (Huang et al., 2023) pioneeringly proposes the LKE setup and appends several neurons for learning edits. GRACE (Hartvigsen et al., 2023) utilizes a key-

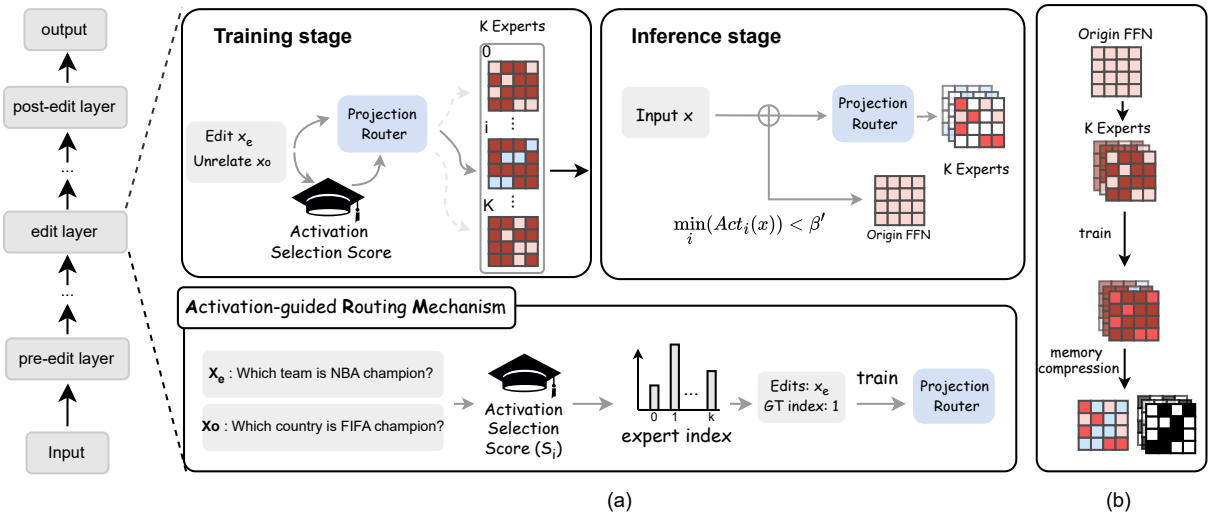

Figure 3: **Overview of our framework** (a) Deployment of MoKE with ARM for sLKE, and (b) Initialization and learning of knowledge experts for MoKE. Note that ARM in (a) trains a projection router to ensure that edits on similar concepts are consistently routed to the same knowledge expert. In (b), we apply the memory compression technique in (Huang et al., 2024) to reduce memory usage during inference.

value codebook with adapters (Hu et al., 2022) to learn new edits. WISE (Wang et al., 2024) constructs side memories from the original weights to learn new edits and introduce a simple routing mechanism that directs inputs to either the experts or the original weights. LKE methods have also been extended to the visual domain. For instance, prior work (Chen et al., 2025) constructs external memories as expert repositories designed specifically to handle extensive visual representations, enabling lifelong updates of visual knowledge. Nevertheless, such methods generally retain historical answers to the same question in separate memory components. This might lead to queries being directed to outdated experts and, consequently, incorrect outputs.

3 Methodology

3.1 Overview

Problem formulation. We first define the setting of our proposed Serial Lifelong Knowledge Editing (sLKE). Given a model $f_\theta : X \mapsto Y$ parameterized by θ, mapping an input domain X (*e.g.*, questions) to an output domain Y (*e.g.*, answers). In sLKE, there are T distinct questions, each updated P times ($P > 1$), resulting in a total of $N = T \times P$ edits. This is different from the standard LKE, where all N edits are unique (*i.e.*, $x_i \neq x_j, \forall i \neq j$).

The goal of sLKE is to update an initial model θ_0 via these N sequential edits on the training set $D_{\text{train}} = \{(x_{t,p}, y_{t,p}) | t = 1, \ldots, T, p = 1, \ldots, P\}$, resulting in a final model θ_N. The ideal final model θ_N^* should correctly reflect the most recent update for each question in the in-sample domain I_{edit} (*i.e.*, inputs semantically equivalent to those in D_{train}. Formally, for any question $x \in X$:

$$f_{\theta_N^*}(x_t) = \begin{cases} y_P, & \text{if } x \in I_{\text{edit}}, \\ f_{\theta_0}(x), & \text{otherwise}, \end{cases} \quad (1)$$

where y_P denotes the answer from the most recent update for x.

Preliminary: Mixture-of-Experts for LKE. Mixture-of-Experts (MoE) (Jiang et al., 2024a; Zadouri et al., 2023; Lu et al., 2023; Wu et al., 2024) is adopted in LKE settings (Wang et al., 2024) to handle sequential edits. This strategy enables the model to integrate new edits while preserving its original performance. Specifically, WISE (Wang et al., 2024) identifies a certain edit layer through empirical analysis and initializes a MoE architecture by duplicating the original model weights to create K experts. Each expert is assigned a unique mask with a mask ratio ρ to ensure specialization and diversity among all experts.

Since our proposed sLKE setting can be viewed as a generalized version of LKE by handling repeated updates to the same concept, we follow WISE (Wang et al., 2024) and choose to identify the same edit layer (i.e., same MoE architectures considered for fair comparisons).

3.2 Activation-Guided Routing for Knowledge Expert Specialization

In LKE, edits concerning the same concept may occur repeatedly. To alleviate the potential *expert am-*

biguity issue, we propose a novel learning scheme of **M**ixture-**o**f-**K**nowledge-**E**xperts (MoKE), where each *knowledge expert* is dedicated to certain concepts. An **A**ctivation-guided **R**outing **M**echanism (ARM) is presented to specialize experts during sLKE training and inference.

As shown in Fig. 3(a), MoKE leverages external memories as domain-specific knowledge experts to store and update certain concepts. To ensure that each edit can be correctly routed to the appropriate expert, we incorporate an ARM that first identifies the correct knowledge expert, updates its memory, then trains a projection router to memorize the certain routing rules. In the following subsection, we detail how we utilize external memories as different knowledge experts, and how our proposed ARM ensures edits related to similar concepts are routed to the same expert during sLKE.

Knowledge expert specialization. To resolve the ambiguity arising from repeated edits to the same concept, we use external memories as specialized knowledge experts, with each knowledge expert responsible for storing specific concepts. The key challenge is to consistently determine which knowledge expert should handle a given input, and ensuring that the outdated knowledge is correctly updated. To address this issue, we propose to leverage the activation function within each knowledge expert as as an indicator of where a specific concept has been stored.

Motivated by previous work (Wang et al., 2024), if an input $\mathbf{x_e}$ has been edited and stored in an expert, the activation score tends to be higher. On the other hand, an unrelated input $\mathbf{x_o}$ typically produces a lower activation score. Based on this observation, we define the activation selection score S_i for expert i as follows:

$$S_i = \frac{\text{Act}_i(\mathbf{x_e}) - \text{Act}_i(\mathbf{x_o})}{\text{Act}_i(\mathbf{x_e})}, \tag{2}$$

where $\text{Act}_i(\cdot)$ denotes the value after passing the activation function inside the feed-forward network of the i-th expert.

Using the computed activation selection scores for each expert, we identify which expert is most likely to store the edited knowledge for an edit $\mathbf{x_e}$. However, if an edit is entirely new (*i.e.*, a new concept that has not been fine-tuned), all experts might show low activation scores. In this case, we randomly select one expert to process and store this new information. Thus, the ground-truth expert index gt for a given edit $\mathbf{x_e}$ and an unrelated input $\mathbf{x_o}$ can be determined by the following formulation:

$$\text{gt} = \begin{cases} \arg\max_i(S_i), & \text{if } \max_i(S_i) > \varepsilon, \\ \text{Uni}(0, K), & \text{otherwise.} \end{cases} \tag{3}$$

where ε is a threshold to determine whether a concept is new to all experts.

Learning of knowledge experts. Similar to previous memory-based methods (Wang et al., 2024), we adopt a margin-based loss function to fine-tune the selected i-th knowledge expert. Specifically, we first define the i-th expert's specialized score for an input $\mathbf{x}$ as:

$$\Delta_i(\mathbf{x}) = ||W_i(\mathbf{x}) - W_{\text{org}}(\mathbf{x})||_2, \tag{4}$$

where $W_i(\cdot)$ denotes the output of the $i-th$ expert, and $W_{\text{org}}(\cdot)$ denotes the original model's output.

Given an edited input $\mathbf{x_e}$ and an unrelated input $\mathbf{x_o}$, the margin-based loss function L_{mar} (Wang et al., 2024) enforces that the specialized score for $\mathbf{x_e}$ exceeds that for $\mathbf{x_o}$ by a certain margin, as defined below:

$$\begin{aligned} L_{\text{mar}} = &\ \max\big(0, \beta - \Delta_i(\mathbf{x_e})\big) \\ &+ \max\big(0, \Delta_i(\mathbf{x_o}) - \alpha\big) \\ &+ \max\big(0, \gamma - \big[\Delta_i(\mathbf{x_e}) - \Delta_i(\mathbf{x_o})\big]\big), \end{aligned} \tag{5}$$

where α, β, γ are hyperparameters that control the margins for the respective terms.

Inspired by classic replay-based continual learning methods (Rolnick et al., 2019), we further introduce a memory loss L_{mem} to ensure that for any previously edited input $\mathbf{x_m}$ stored in a different expert j, its specialized score should exceeds that of expert i by a margin γ, as defined below:

$$L_{\text{mem}} = \max\big(0, \gamma - \big[\Delta_j(\mathbf{x_m}) - \Delta_i(\mathbf{x_m})\big]\big). \tag{6}$$

The overall loss function used to fine-tune the selected i-th expert is the sum of the margin and the memory loss $L = L_{\text{mar}} + L_{\text{mem}}$.

Projection router training. After determining the selected expert index gt through Eq. (3), we train a projection router $f_{\theta_p} : X \mapsto [0, 1]^K$ (parameterized by θ_p) to act as a classifier that consistently assigns each input edit $\mathbf{x_e}$ to its corresponding expert index gt. This mechanism ensures that edits related to the same concept are always routed

to the same expert, enhancing the robustness of our method against repeated updated knowledge in SLE. Specifically, we train the projection router through the standard cross entropy loss (CE):

$$L_p = \text{CE}\left(f_{\theta_p}(\mathbf{x_e}), \text{gt}\right). \tag{7}$$

3.3 Inference Phase

Expert memory compression. As noted in (Wang et al., 2024), MoE-based algorithms suffer from high memory usage when the number of experts increases. To mitigate this issue, we employ EMR Merging (Huang et al., 2024), a tuning-free model merging algorithm that reduces memory consumption during inference, as depicted in Fig. 3(b). This method consolidates K expert weights into a single unified weight along with K boolean masks, thereby decreasing memory usage while also retaining performance. We leave the implementation details of EMR Merging algorithm into appendix, and further analyzes in Sec. 4.4.

Inference routing. After compressing the memory usage, we use the trained projection router to identify which expert contains the updated knowledge for a given input question $\mathbf{x}$. Specifically, we determine the routed expert index as:

$$\ell = \arg\max\left(f_{\theta_p}(\mathbf{x})\right). \tag{8}$$

If a question is entirely new (*i.e.*, the knowledge has never been updated), the activation selection scores across all experts will be below a certain threshold. In this case, we route the question to the original model weights W_{org}. The final memory weight W_{final} is selected as follows:

$$W_{\text{final}} = \begin{cases} W_{\text{org}}, & \text{if } \max_i(\Delta_i) < \beta', \\ W_\ell, & \text{otherwise,} \end{cases} \tag{9}$$

where β' is a hyper-parameter that determines whether the knowledge has been updated.

4 Experiment

4.1 Model and Benchmark

Following existing LKE works (Wang et al., 2024; Mitchell et al., 2022b; Hartvigsen et al., 2023), we select **Llama2-7B** (Touvron et al., 2023b,a) as our backbone model. During training, each edit sample includes edits x_e, rephrased edits $x_{e'}$, unrelated samples x_o, and their corresponding labels y_e. The initial model is denoted as f_{θ_0}, and the post-edited final model, after n edits, is denoted as f_{θ_n}.

To evaluate the LKE setting, we constructed a new LKE dataset based on the existing QA datasets **ZsRE** (Levy et al., 2017) and **Wiki CF** (Zhang et al., 2024a). ZsRE is a closed-book factual QA dataset, whereas Wiki CF contains counterfactual QA pairs, making it more suitable for examining the reliability of model editing methods. However, the above datasets only include standard lifelong edit samples, i.e., each question has a single corresponding answer. To address this limitation, we leveraged GPT-4o (OpenAI, 2023; OpenAI and the Co-authors, 2024) to generate 10 plausible alternative answers for each question. We then manually removed any defective or irrelevant answers that were unrelated to the original ones. As a result, each question now has 8 corresponding answers for continual lifelong editing.

For ZsRE, the final dataset contains 11,301 different questions sampled from the original training and test sets, resulting in a total LKE edit size of $11{,}301 \times 8 = 90{,}408$. For Wiki CF, the final dataset contains 2,340 different questions sampled from its original training and test sets, leading to a total LKE edit size of $2{,}340 \times 8 = 18{,}720$.

4.2 Evaluation Metrics

Following previous works (Yao et al., 2023; Zhang et al., 2024b), we evaluate f_{θ_n} using three metrics: **Reliability**, **Generality**, and **Locality**, defined as follows:

Reliability (Rel.) calculates answer's average precision for each edit sample x_e^t, i.e.,

$$\text{Rel.} = \frac{1}{T}\sum_{t=1}^{T} \mathbb{1}(f_{\theta_n}(\mathbf{x}_e^t) = \mathbf{y}_e^t). \tag{10}$$

Generality (Gen.) calculates answer's average precision for paraphrased edit sample $\mathbf{x}_{e'}^t$ by

$$\text{Gen.} = \frac{1}{T}\sum_{t=1}^{T} \mathbb{1}(f_{\theta_n}(\mathbf{x}_{e'}^t) = \mathbf{y}_e^t). \tag{11}$$

Locality (Loc.) calculates answer's average precision for unrelated sample $\mathbf{x}_{e'}^t$ by:

$$\text{Loc.} = \frac{1}{T}\sum_{t=1}^{T} \mathbb{1}(f_{\theta_n}(\mathbf{x}_o^t) = \mathbf{y}_o^t). \tag{12}$$

For LKE, $T = N$, where N is the total number of edit samples. For sLKE, with T different questions

Method	sLKE-ZsRE								sLKE-WiKi-CF							
	$T = 125, N = 1000$				$T = 250, N = 2000$				$T = 125, N = 1000$				$T = 250, N = 2000$			
	Rel.↑	Gen.↑	Loc.↑	Avg.↑	Rel.↑	Gen.↑	Loc.↑	Avg.↑	Rel.↑	Gen.↑	Loc.↑	Avg.↑	Rel.↑	Gen.↑	Loc.↑	Avg.↑
FT	0.08	0.04	0.00	0.04	0.07	0.04	0.00	0.04	0.09	0.02	0.01	0.04	0.04	0.02	0.00	0.02
MEND	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MALMEN	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ROME	0.01	0.01	0.00	0.01	0.01	0.01	0.00	0.01	0.02	0.01	0.00	0.01	0.01	0.00	0.00	0.01
MEMIT	0.02	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.03	0.02	0.04	0.03	0.01	0.01	0.00	0.01
SERAC	0.07	0.05	0.26	0.13	0.02	0.02	0.11	0.05	0.06	0.05	0.30	0.14	0.02	0.01	0.13	0.05
GRACE	0.01	0.01	1.00	0.34	0.01	0.01	0.98	0.33	0.01	0.01	1.00	0.34	0.01	0.01	**0.99**	0.34
WISE-Merge	0.68	0.65	1.00	0.77	0.56	0.50	0.99	0.71	0.57	0.54	1.00	0.70	0.52	0.49	0.98	0.67
WISE-Retrieval	0.67	0.62	1.00	0.76	0.63	0.59	**1.00**	0.74	0.55	0.52	1.00	0.69	0.52	0.49	0.99	0.67
Ours	**0.85**	**0.74**	**1.00**	**0.86**	**0.75**	**0.66**	**1.00**	**0.80**	**0.76**	**0.66**	**1.00**	**0.81**	**0.70**	**0.55**	0.95	**0.73**

Table 1: Performance comparisons of sLKE on sLKE-ZsRE and sLKE-WiKi-CF datasets. Note that T indicates the number of distinct questions (each question is updated $P = 8$ times, resulting in a total number of N edits.

Method	ZsRE								WiKi-CF							
	$T = 1000$				$T = 2000$				$T = 1000$				$T = 2000$			
	Rel.↑	Gen.↑	Loc.↑	Avg.↑	Rel.↑	Gen.↑	Loc.↑	Avg.↑	Rel.↑	Gen.↑	Loc.↑	Avg.↑	Rel.↑	Gen.↑	Loc.↑	Avg.↑
FT	0.14	0.12	0.02	0.09	0.06	0.05	0.02	0.04	0.06	0.03	0.01	0.03	0.05	0.03	0.00	0.03
MEND	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MALMEN	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ROME	0.01	0.01	0.00	0.01	0.01	0.00	0.00	0.01	0.02	0.01	0.00	0.01	0.01	0.00	0.00	0.01
MEMIT	0.04	0.04	0.02	0.03	0.02	0.02	0.01	0.02	0.04	0.03	0.02	0.03	0.02	0.01	0.01	0.01
SERAC	0.02	0.02	0.74	0.26	0.02	0.01	0.32	0.11	0.01	0.01	0.44	0.15	0.01	0.01	0.16	0.06
GRACE	**0.98**	0.00	1.00	0.66	**0.90**	0.00	0.97	0.66	**0.89**	0.01	0.99	0.63	**0.80**	0.00	0.93	0.57
WISE-Merge	0.77	0.70	1.00	0.82	0.66	0.63	**1.00**	0.76	0.54	0.50	0.97	0.67	0.40	0.37	0.92	0.57
WISE-Retrieval	0.81	0.71	1.00	0.84	0.74	0.66	**1.00**	0.80	0.76	0.68	1.00	0.81	0.66	0.59	**0.99**	0.73
Ours	0.83	**0.74**	**1.00**	**0.85**	0.76	**0.68**	0.99	**0.81**	0.80	**0.70**	**1.00**	**0.83**	0.72	**0.62**	0.98	**0.77**

Table 2: Performance comparisons of standard LKE on ZsRE and WiKi-CF datasets. Note that $T = N$ since each edit is unique in LKE.

and P alternative answers for each question, the total number of edits is $N = T \times P$. We evaluate the last T edits with the associated latest answers.

4.3 Performance Assessement

For LKE in Tab. 2, we follow previous works by setting the maximum number of edits to 1000. For sLKE in Tab. 1, we test 125 different questions, each with 8 alternative answers. The total edit volume is $125 \times 8 = 1000$, which ensures consistency across both settings.

In our experiments, we use instance-based LKE, where the batch size is set to 1. We examine three styles of methods alongside direct fine-tuning (FT) (Wang et al., 2023). Hypernetwork methods (MEND (Mitchell et al., 2022a), MALMEN (Tan et al., 2023)) and Locate-Edit methods (ROME (Meng et al., 2022), MEMIT (Meng et al., 2023)) demonstrate limited effectiveness for small batch size edits. Among memory-based methods, while not all of them achieve competitive performance, they generally preserve locality better since they reserve original weight.

SERAC (Mitchell et al., 2022b) exhibits slightly better performance in locality preservation but is not a feasible solution overall. Furthermore, while GRACE (Hartvigsen et al., 2023) achieves outstanding **Rel.** and **Loc.** scores in the standard lifelong edit setting, its performance deteriorates significantly in the LKE setting. GRACE also shows a noticeable deficiency in **Gen.**. On the other hand, WISE (Wang et al., 2024) performs exceptionally well across all metrics in the standard lifelong edit setting; however, its performance also degrades when applied to our new setting.

Our method demonstrates robust performance on both LKE and sLKE tasks. Notably, the improvement in the sLKE setting is particularly significant, as **ARM** effectively manages repeated concepts within the edit sequence. For the standard LKE setting, our approach also achieves modest performance gains, as the external experts are able to learn and organize well-classified knowledge more efficiently than approaches relying on randomly assigned knowledge (Wang and Li, 2024).

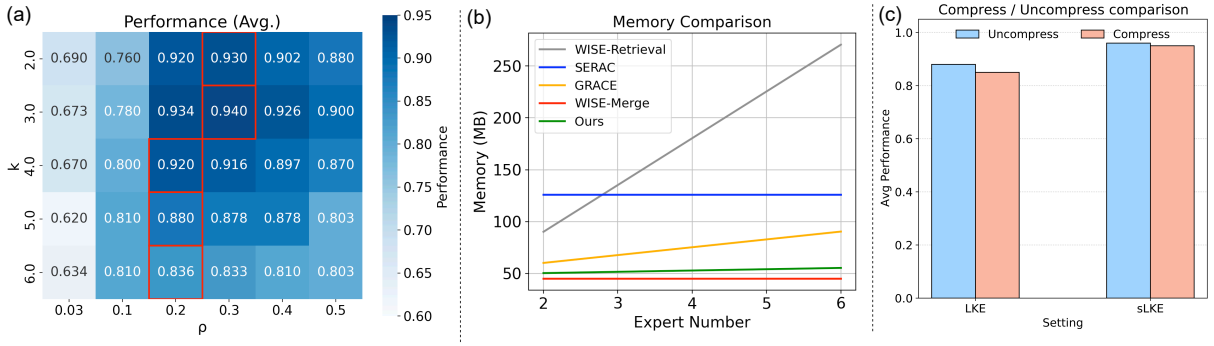

Figure 4: (a) Analysis of the number of experts K and mask ratios for each expert ρ on sLKE-ZsRE. (b) Comparisons of LKE methods (i.e., SERAC (Mitchell et al., 2022b), GRACE (Hartvigsen et al., 2023), WISE (Wang et al., 2024)) with varying expert numbers (i.e., memory consumption) on sLKE-ZsRE. (c) Analysis of the memory compression method (using EMR-Merge (Huang et al., 2024)) on ZsRE and sLKE-ZsRE. Note that Avg Performance is the averaged accuracy of Rel., Gen., and Loc..

	Rel.↑	Gen.↑	Loc.↑	Avg.↑
Ours	**0.95**	**0.85**	**1.00**	**0.91**
- ARM	0.83	0.72	1.00	0.85
- memory loss	0.90	0.75	1.00	0.88
- EMR-Merge (+ ties-merge)	0.87	0.74	1.00	0.87

Table 3: Ablation study of our method on sLKE-ZsRE (T=125 and N=1000).

	Rel.↑	Gen.↑	Loc.↑	Route ACC.↑
Random Selection	0.88	0.72	1.00	0.89
Fix Selection	0.84	0.70	1.00	0.85
ARM	0.90	0.81	1.00	0.94
+ Mem Loss	**0.95**	**0.85**	**1.00**	**0.98**

Table 4: Comparisons of random selection, fixed selection, and ARM for expert specialization on sLKE-ZsRE.

4.4 Further Analysis and Ablation Studies

Memory loss analysis. Memory loss prevents experts from hallucinating that they are the experts for this input. During training, we apply the regulation loss in Eq. (5) to prevent the model from assigning large activation values to unrelated samples. The same idea can be applied to edited samples for other experts. For a certain expert, knowledge from other experts should be regarded as unrelated samples. Hence, we can use the same logic to design memory loss based on regulation loss. In Tab. 3, we observe that memory loss is beneficial for LKE.

Training phase routing efficiency test. Training phase routing is the main mechanism how we specialize external memories to experts. As illustrated in Tab. 4, We conduct ablation study for different cases through training, random selection for memories for each question, fixed arrangement that choose experts cyclically and our activation selection. Our routing mechanism gives obvious better result than other cases, meaning training phase routing is beneficial to MoE router (Jiang et al., 2024a; Lu et al., 2023; Liu et al., 2024).

Router selection. The activation selection helps the model manage repeated concepts during training and inference, making our method more aligned with sLKE. In practice, activation selection also enhances the performance for LKE since activation score is effective indicator for expert's knowledge. However, as the number of edited concepts increases, the activation selection score becomes neutralized due to the model's decreasing confidence, leading to incorrect routing decisions.

In conventional MoE (Zadouri et al., 2023; Lu et al., 2023; Wu et al., 2024), the projection router is the component enabling correct routing for LKE. With assistance of ARM, single projection router for inference can successfully handle sLKE since knowledge is classified throughout training, solving ambiguity of projection router.

As indicated in Tab. 5, while single projection router for inference has outperformed current baselines for sLKE, using both projection router and ARM to get ensembled score for experts leads to even better result for LKE and sLKE. The result can be interpreted as a combination of solutions for LKE and sLKE. While projection router is effective for MoE to solve LKE, as manifested in previous works (Jiang et al., 2024a; Zadouri et al., 2023), ARM is inheritedly useful for sLKE since it is designed to classify knowledge for experts.

Memory Compression. Fig. 4(b) compares the memory requirements of different LKE methods.

Method	$T = 1000, N = 1000$			$T = 125, N = 1000$		
	Rel.	Gen.	Loc.	Rel.	Gen.	Loc.
WISE-Merge	0.77	0.70	1.00	0.68	0.65	1.00
Act.	0.81	0.70	1.00	0.88	0.75	1.00
Proj. Router	0.83	0.74	1.00	0.85	0.74	1.00
Act. + Router	**0.85**	**0.77**	**1.00**	**0.95**	**0.85**	**1.00**

Table 5: **Analysis of our routing strategies for ZsRE (left) and sLKE-ZsRE (right).** "Act." indicates that directly route to the expert with the highest activation values, while "Act. + Router" is the weighted sum of the activation values and the projection router's prediction. Although "Act. + Router" performs best, we use "Proj. Router" as our inference routing strategy for fair comparisons. Note that all strategies surpass SOTAs.

Note that SERAC (Mitchell et al., 2022b) employs a large counterfactual model and router for all edits, leading to substantial but constant memory usage, as the counterfactual model dominates resource consumption. GRACE (Hartvigsen et al., 2023) introduces adapters and a codebook to skip redundant edits, but in sLKE, where answers to the same question may change, frequent codebook updates cause memory usage to grow linearly with the number of edits. WISE (Wang et al., 2024) uses side memories and provides two inference strategies: merge, which uses tie-merge (Yadav et al., 2023) and saves memory at the cost of significant performance loss, and retrieval, which preserves performance in LKE but results in linearly increasing memory usage. Our method leverages EMR-Merging (Huang et al., 2024), masking unified experts to reduce interference and closely approximate the original experts. Storing only boolean masks, our approach achieves high memory efficiency.

As shown in Fig. 4(b), SERAC maintains high but constant memory usage, GRACE and WISE-Retrieval both increase linearly—with WISE-Retrieval growing faster—while WISE-Merge has low constant memory at the cost of significant performance drop (Tab. 1). Overall, our method provides a strong balance between model performance and resource efficiency.

Expert number and mask ratio. In WISE (Wang et al., 2024), the concept of a knowledge anchor (Zheng et al., 2021) is proposed, highlighting that overlap in side memories benefits model performance. However, our philosophy is to specialize each memory. Overlap serves as shared knowledge among memories but is prone to ambiguity, as also mentioned in WISE. Therefore, reducing overlap is preferable in our approach. We evaluate performance based on different numbers of experts and mask ratios for each expert in Fig. 4. It can be observed that when $K \times \rho > 1$, the model's performance tends to decline. On the other hand, an extremely low mask ratio prevents the model from learning edit samples effectively. Hence, the optimal parameter setting is $K \times \rho \approx 1$.

Routing Accuracy on sLKE and LKE. In Fig. 5, we evaluate routing accuracy and score for sLKE and LKE using memory-based approaches, including GRACE, WISE, and Ours. The key difference between routing accuracy and the reliability score is that routing accuracy better reflects the routing algorithm behind knowledge editing, whereas the reliability score also assesses the efficiency of edits on model weights. It can be observed that WISE (Wang et al., 2024) and GRACE (Hartvigsen et al., 2023) severely corrupt in sLKE. In contrast, our method outperforms others on both LKE and sLKE since ARM specializes experts, increasing the confidence of both projection router and experts.

Training and Inference Efficiency The computational cost of MoE is primarily determined by the number of additional parameters introduced. In Table 6, we list the total number of parameters for each method. Compared to other baseline approaches—such as fine-tuned LLaMA (Touvron et al., 2023a) and WISE (Wang et al., 2024)—our method adds only three expert layers and a projection router during both training and inference. The parameter size of each expert and the projection router is approximately equivalent to a single layer in LLaMA2-7B. Overall, our approach results in less than a 10% increase in computation time relative to baseline methods, while yielding performance improvements exceeding 17%, as listed in Tab. 1.

Method	Training	Inference
LLaMA2 finetuned	32 layers	32 layers
WISE	32 layers + 1 expert	32 layers + 3 experts
Ours	32 layers + 3 experts	32 layers + 3 experts + proj. router

Table 6: **Comparison of model size and architecture in training and inference.**

5 Conclusion

We identify a key limitation of lifelong editing for LLM: its limited capability to handle multiple edits of the same concept across time. We refer to this as

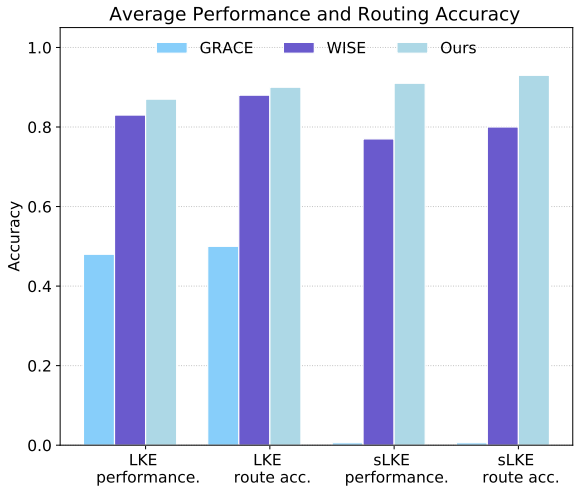

Figure 5: **Comparisons of routing acc. and average performances (Rel., Gen., Loc.) for different LKE methods.** Note that while WISE and GRACE are not specifically designed for sLKE, our method is preferable for both LKE and sLKE settings.

"Serial Lifelong Knowledge Editing" (sLKE) and propose a framework utilizing multiple memory-based knowledge experts. With the proposed routing scheme, we enforce edits of similar concepts to be handled by the same expert. In addition, memory compression for such experts can be further performed, ensuring no extra memory cost is needed when comparing to existing LKE methods. With a novel benchmark corrected for sLKE, our experiments confirm that our proposed method performs favorably against SOTA methods for standard LKE and our challenging sLKE tasks.

6 Limitation

While our method is shown to effectively tackle both LKE and sLKE problems, a number of limitations still exist. One key limitation, illustrated in Fig. 4(a), emerges when the number of edits becomes significantly large. In practice, maintaining scalable memory usage through single-layer editing requires reducing the mask ratio, which can lead to substantial performance degradation due to catastrophic forgetting (Luo et al., 2023).

Another limitation involves inefficiency in reasoning tasks. Although **ARM** effectively captures similarities between questions and maintains overall performance, it struggles to identify complex relationships between concepts, particularly in tasks such as multi-hop knowledge editing (Zhong et al., 2024).

Finally, our method does not guarantee straightforward transferability from the language domain to the visual domain. In practice, the visual domain presents greater complexity than textual knowledge, and most current LKE approaches—including ours—are limited to handling purely textual inputs. This highlights an open area for future research on extending knowledge editing techniques to diverse and complex domains.

References

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

Qizhou Chen, Chengyu Wang, Dakan Wang, Taolin Zhang, Wangyue Li, and Xiaofeng He. 2025. Lifelong knowledge editing for vision language models with low-rank mixture-of-experts. *Preprint*, arXiv:2411.15432.

Qizhou Chen, Taolin Zhang, Xiaofeng He, Dongyang Li, Chengyu Wang, Longtao Huang, and Hui Xue. 2024. Lifelong knowledge editing for llms with retrieval-augmented continuous prompt learning. *Preprint*, arXiv:2405.03279.

Zhibin Duan, Hao Zhang, Chaojie Wang, Zhengjue Wang, Bo Chen, and Mingyuan Zhou. 2021. Enslm: Ensemble language model for data diversity by semantic clustering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2954–2967. Association for Computational Linguistics.

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. *Preprint*, arXiv:2410.02355.

Tom Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. 2023. Aging with grace: Lifelong model editing with discrete key-value adaptors. *Advances in Neural Information Processing Systems*, 36.

Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. 2022. Meta-learning in neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):5149–5169.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Chenyu Huang, Peng Ye, Tao Chen, Tong He, Xiangyu Yue, and Wanli Ouyang. 2024. Emr-merging: Tuning-free high-performance model merging. *arXiv preprint arXiv:2405.17461*.

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*.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024a. Mixtral of experts. *CoRR*, abs/2401.04088.

Yuxin Jiang, Yufei Wang, Chuhan Wu, Wanjun Zhong, Xingshan Zeng, Jiahui Gao, Liangyou Li, Xin Jiang, Lifeng Shang, Ruiming Tang, Qun Liu, and Wei Wang. 2024b. Learning to edit: Aligning llms with knowledge editing. *Preprint*, arXiv:2402.11905.

Yuxin Jiang, Yufei Wang, Chuhan Wu, Wanjun Zhong, Xingshan Zeng, Jiahui Gao, Liangyou Li, Xin Jiang, Lifeng Shang, Ruiming Tang, Qun Liu, and Wei Wang. 2024c. Learning to edit: Aligning llms with knowledge editing. *Preprint*, arXiv:2402.11905.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA.

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.

Weishi Li, Yong Peng, Miao Zhang, Liang Ding, Han Hu, and Li Shen. 2023a. Deep model fusion: A survey. *arXiv preprint arXiv:2309.15698*.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. 2023b. PMET: precise model editing in a transformer. *CoRR*, abs/2308.08742.

Tianlin Liu, Mathieu Blondel, Carlos Riquelme, and Joan Puigcerver. 2024. Routers in vision mixture of experts: An empirical study. *Preprint*, arXiv:2401.15969.

Keming Lu, Hongyi Yuan, Runji Lin, Junyang Lin, Zheng Yuan, Chang Zhou, and Jingren Zhou. 2023. Routing to the expert: Efficient reward-guided ensemble of large language models. *CoRR*, abs/2311.08692.

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. *arXiv preprint arXiv:2308.08747*.

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

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

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

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

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

OpenAI and the Co-authors. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Oded Ovadia, Menachem Brief, Moshik Mishaeli, and Oren Elisha. 2023. Fine-tuning or retrieval? comparing knowledge injection in llms. *arXiv preprint arXiv:2312.05934*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

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.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P. Lillicrap, and Gregory Wayne. 2019. Experience replay for continual learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 348–358.

Ohad Shamir and Tong Zhang. 2013. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *International conference on machine learning*, pages 71–79. PMLR.

George Stoica, Daniel Bolya, Jakob Brandt Bjorner, Pratik Ramesh, Taylor Hearn, and Judy Hoffman. 2024. Zipit! merging models from different tasks without training. In *The Twelfth International Conference on Learning Representations*.

Chenmien Tan, Ge Zhang, and Jie Fu. 2023. Massive editing for large language model via meta learning. In *The Twelfth International Conference on Learning Representations*.

Kushal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. 2022. Memorization without overfitting: Analyzing the training dynamics of large language models. *Advances in Neural Information Processing Systems*, 35:38274–38290.

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

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

Ricardo Vilalta and Youssef Drissi. 2002. A perspective view and survey of meta-learning. *Artificial intelligence review*, 18:77–95.

Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. 2024. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. *arXiv preprint arXiv:2405.14768*.

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*.

Renzhi Wang and Piji Li. 2024. Lemoe: Advanced mixture of experts adaptor for lifelong model editing of large language models. *arXiv preprint arXiv:2406.20030*.

Xun Wu, Shaohan Huang, and Furu Wei. 2024. Mixture of lora experts. *CoRR*, abs/2404.13628.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. 2023. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36.

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.

Ted Zadouri, Ahmet Üstün, Arash Ahmadian, Beyza Ermis, Acyr Locatelli, and Sara Hooker. 2023. Pushing mixture of experts to the limit: Extremely parameter efficient moe for instruction tuning. *CoRR*, abs/2309.05444.

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

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, et al. 2024b. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*.

Xiawu Zheng, Yang Zhang, Sirui Hong, Huixia Li, Lang Tang, Youcheng Xiong, Jin Zhou, Yan Wang, Xiaoshuai Sun, Pengfei Zhu, et al. 2021. Evolving fully automated machine learning via life-long knowledge anchors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(9):3091–3107.

Zexuan Zhong, Zhengxuan Wu, Christopher D. Manning, Christopher Potts, and Danqi Chen. 2024. Mquake: Assessing knowledge editing in language models via multi-hop questions. *Preprint*, arXiv:2305.14795.

A Dateset Details

ZsRE. The ZsRE dataset is a context-free factual Question Answering (QA) dataset extensively studied in model editing research. Each record in this dataset includes an editing statement $\mathbf{x}_e$ with target answer $\mathbf{y}_e$, a paraphrase prompt $\mathbf{x}_{e'}$, and a locality prompt $\mathbf{x}_o$. ZsRE consists of 163,196 training examples and 19,086 test examples. We divide the test examples into our train/test set. For standard lifelong editing, we randomly select 1000 samples for our experiments. For continual lifelong editing, we use the same 1000 samples to generate alternative answers for the sLKE benchmark. To generate alternative answers, we use GPT-4o (Radford et al., 2018, 2019; Brown et al., 2020; OpenAI and the Co-authors, 2024) and manually remove unreasonable cases. The generation process is the same for both datasets, ZsRE and Wiki CF.

Wiki CF. Since models often fail to capture tail entities, rendering them unsuitable for testing modification edits, Wiki CF (Zhang et al., 2024a) collects triplets involving popular entities, where the topic corresponds to one of the most-viewed pages on Wikipedia. In addition, the Counterfact dataset can evaluate model performance, as it is possible for a model to correctly answer an edit sample before the edit, enabling the dataset to maintain its effectiveness over time.

Similar to ZsRE, Wiki CF includes an editing statement $\mathbf{x}_e$ with a target answer $\mathbf{y}_e$, a paraphrase prompt $\mathbf{x}_{e'}$, and a locality prompt $\mathbf{x}_{o'}$. The original dataset contains 1,455 edit samples in the training set and 855 edits in the test set. Each edit has multiple versions of paraphrased answers, such as "Subject Aliasing," which rephrases the subject of an edit sample, and "Relation Specificity," which rephrases the relationship between the subject and object. In our experiments, we use Subject Aliasing for generality testing.

Continual Dataset Generation. We use GPT-4o API to generate Continual lifelong dataset. Each PROMPT and GROUND TRUTH are directly from original dataset. It is possible that the outputs include repeated answer even the prompt has the requirement of non-repeating. Hence, we manually remove repeated answer. At last, each question have 8 alternative answers.

System Prompt

You are an AI assistant specializing in generating high-quality question-answer datasets. Your task is to create structured QA data by generating a correct answer and multiple plausible alternative answers for a given question.

Generation Prompt

Given the sentence: 'PROMPT' and the correct answer: 'GROUND TRUTH', generate ten plausible alternative answers that fit the same format as the correct answer. The alternatives should be distinct, non-repetitive, and contextually relevant while maintaining the same style and structure as the correct answer.

Figure 6: Prompt to use GPT-4o to generate dataset of sLKE from those of LE

B Training Details

B.1 Hyperparameter

For all baselines, we use the code from EasyEdit (Wang et al., 2023) and follow the parameter settings from original works. For all experiments, we follow the principle of instance sequential editing, setting batch size to 1. Experiments are conducted on an NVIDIA V100 GPU, and all results are reproducible on a single GPU within four hours.

To ensure a fair comparison, we edit the model by modifying the appropriate layer from the original works for each baseline. All hyperparameters, including the learning rate, optimizer (Kingma and Ba, 2015; Shamir and Zhang, 2013), scheduler, and others, follow the settings from the original works.

As for fine-tuning (FT), since there is no original work focused on fine-tuning the model for lifelong editing, we modify only `model.layers[21].mlp.down_proj.weight`, which is commonly used in previous works (Meng et al., 2023, 2022; Li et al., 2023b).

All hyperparameter settings used in our method are summarized in Table 6. We perform edits on LLaMA2-7B by specifically modifying `model.layers[27].mlp.down_proj.weight`, following previous works (Wang et al., 2024, 2023) that have identified the optimal layer for modification through comprehensive ablation studies. Regarding the hyperparameters α, β, and γ in the loss function, we adopt the values

Dataset	Type	Text
ZsRE	$\mathbf{x}_e, \mathbf{y}_e$	Which continent is Berkner Island in? **South America**
	$\mathbf{x}_\text{o}, \mathbf{y}_\text{o}$	who gets the golden boot if its a tie? **shared**
	$\mathbf{x}_\text{e'}, \mathbf{y}'_e$	On which continent is Berkner Island located? **South America**
WiKi CF	$\mathbf{x}_e, \mathbf{y}_e$	The name of the country which Goursez Vreizh is associated with is? **Italy**
	$\mathbf{x}_\text{o}, \mathbf{y}_\text{o}$	what is the most current season of the walking dead? **The eighth season**
	$\mathbf{x}_\text{e'}, \mathbf{y}'_e$	The name of the country which Gorsedd of Brittany is associated with is? **Italy**

Table 7: An editing dataset example from ZsRE and Wiki CF.

Hyper-Parameters	**Values**
Optimizer	SGD
Experts LR	1.0
Project LR	10^{-4}
Mask Ratio ρ	0.3
α	5.0
β	20.0
γ	10.0
Knowledge Shards k	3

Table 8: Ours hyper-parameters during editing and merging.

	ε=0.25	ε=0.45	ε=0.35 (Ours)
Avg. Accuracy	0.91	0.90	0.91

Table 9: Ablation of ε on Avg. Accuracy. of sLKE with 125 * 8 edits

established in prior literature (Wang et al., 2024, 2023). For the hyperparameter ϵ, we select a fixed value of 0.35 without further tuning, as its purpose is primarily to distinguish expert-related knowledge from unrelated concepts—a task that generally exhibits low sensitivity to exact threshold values. To further verify the insensitivity to this parameter, we conduct additional experiments with alternative settings (e.g., $\epsilon = 0.25$ and $\epsilon = 0.45$). As demonstrated in Tab. 9, MoKE consistently achieves stable and state-of-the-art performance across this range, confirming that our method is robust to variations in the exact choice of ϵ.

To equip model with ability to identify questions relevant to edits, we use random prefix for edit samples, which works as a noise to strengthen models ability to detect relevant concepts. The prefix is generated by giving an beginning to model and generating a sequence of meaningless words. Random prefixes massively increase generality accuracy for both SLE and LE settings. In addition to random prefix, we add noise to subject of each edits. We randomly choose a process from adding middle words for subject, removing middle words for subjects and adding title for subject. Both random prefix and subject noise are randomly applied, which prevent in-sample overfitting.

B.2 Memory Loss

We design memory loss to prevent ambiguity between experts. During the editing process, we first store each edit along with its corresponding expert's index. Then, for each edit, we randomly select one edit from the database that does not belong to the current expert. To maintain low memory usage and preserve the intrinsic meaning of instance-sequential edit, we store at most 50 edits. Once memory reaches this limit, the randomly selected sample for each round is deleted and will not be used again. In this way, we regulate the maximum resource usage and prevent accessing a large number of edits at once, which would diminish the value of instance edits.

C Case Study

We investigate which types of questions are most troublesome in SLE. It can be observed that models often produce incorrect outputs for multi-word edit targets, resulting in partial errors in the answers. Typically, the first word is incorrect, while the remaining part of the answer remains correct.

D Detail of Memory Based Methods

D.1 Method Insight

SERAC. SERAC is one of the earliest works to propose utilizing external memory to inject knowledge into large models and a router to identify knowledge related to edited samples. The pipeline consists of three main components: edit mem-

Modification	Text
Random Prefix 1	I'm a 4 {**Question**}
Random Prefix 2	Yes, I know it' {**Question**}
Random Prefix 3	Q: 1. What is the {**Question**}
Title adding	The name of the country which **Dr.** Goursez Vreizh is associated with is?
Middle word insertion	The mother tongue of Danielle **Parthenia** Darrieux is
Middle word deletion	Nancy ~~**Astor, Viscountess**~~ Astor was employed in

Table 10: Random prefix and subject noise for QA pair.

Method	$T = 1000$				$T = 125, N = 1000$			
	Rel.	Gen.	Loc.	Avg.	Rel.	Gen.	Loc.	Avg.
w/o augmentation	**0.96**	0.03	1.00	0.66	**0.96**	0.03	1.00	0.66
with augmentation	0.64	0.58	0.55	0.59	0.58	0.53	0.47	0.53

Table 11: Ablation for the effectiveness of input augmentation

ory, a scope classifier, and a counterfactual model. The edit memory stores user-supplied data, which serves as input for the scope classifier. The scope classifier determines whether the current input is related to the edit database. If the input is relevant, it is routed to the counterfactual model to generate the post-edit answer. Otherwise, if it is not relevant to the edit database, the original LLM handles the input.

This pipeline effectively preserves the original model weights, ensuring locality when the number of edits is low. However, as the number of edits increases, the system degrades rapidly. The edit memory requires significant storage to maintain, the scope classifier's accuracy decreases as the classification task becomes more challenging with additional edit samples, and a single counterfactual model can handle only hundreds of edits—far from sufficient for the thousands of edits required for lifelong learning.

GRACE. GRACE introduces a key-value codebook with a deferral radius to store the correct routing for edits. Each concept is stored as a key-value pair, where the key is the logits from pre-edit layers and the value is the logits from post-edit layers. Inputs with logits values within the deferral radius of a certain key are considered to share the same concept as the key, meaning the adapter does not need to learn this new edit since it is a repetition with the same ground truth answer. In contrast, if the edit target does not align with the key's concept, a new key-value pair is generated, and the adapter must be trained on this new sample.

The deferral radius is the most important component for GRACE to maintain editing efficiency. For similar edits with the same ground truth target, the deferral radius for the corresponding key-value pair expands, enhancing the region of this concept. Conversely, the deferral radius decreases to prevent key conflicts.

WISE. WISE is the newest memory based method, which enable constant memory usage and maintain great performance on standard lifelong edit. It generates multiple experts from original weight to learn new edit. To reduce memory usage, it leverage ties-merge to compress experts to one. To ensure compression quality, they propose the idea of knowledge anchoring, stating that overlap in side memories is beneficial to model compression since it serves as a intermediate point to transfer knowledge between different memories' subspaces. They also mention the trade-off of knowledge anchor and conflict. Anchoring leads to conflict in some cases, which is unfavorable for expertising model. Hence, we do not adopt anchoring for our experts.

For routing, they design rule-base routing mechanism. Correct inference routing can be determined without training additional router.

Retrieval Methods. Retrieval methods (Jiang et al., 2024b; Chen et al., 2024; Jiang et al., 2024c) are regarded as a part of memory based methods. Yet, we do not take these methods into consideration since the objective and philosophy are entirely different between memory-based and retrieval methods (Ovadia et al., 2023). Retrieval methods focus on how to retrieve correct data from a large database and how to utilize the data to update models knowledge with in-context learning. Resource usage is not quite important in this case since database can be accessed externally in real-world application

	Prompt	Edit Target	Post-Edit Output
ia)	What was the designer of Évry Cathedral's name?	Gae Aulenti	illes Aulenti ✔
ib)	*What designer was in charge of Évry Cathedral?*	-	illes Aulenti ✔
ic)	Which place does Fukrey exist in?	Kuala Lumpur	unala Lumpur ✔
id)	*What is Fukrey's place?*	-	unala Lumpur ✔
iia)	Which family does Synsphaeria belong to?	Lycaenidae	copenidae ✗
iib)	*What family belongs to Synsphaeria?*	-	copenidae ✗
iic)	What is the final year of Atlanta Flames?	1976	1989 ✗
iid)	*What year ended with Atlanta Flames?*	-	1989 ✗
iiia)	Of what did Earl Hooker die?	tuberculosis	Pneumonia ✓
iiib)	What was Earl Hooker the cause of death?	-	tuberculosis ✗

Table 12: **Failure cases of our method** to edit `LLaMA-2-7B`. ✔represents errors in part of the answer, ✗represents complete output errors, and ✓indicates the expected exact match.

D.2 Pitfalls of Serial Lifelong Editing

While memory-based methods are well-suited for lifelong editing, we have shown that their performance degrades significantly in the context of sLKE. This issue is particularly evident in methods such as WISE and GRACE, both of which perform exceptionally well for standard lifelong editing.

In the case of GRACE, the deficiency arises from the flawed cooperation between the codebook and the adaptor. The codebook mechanism dictates that edit concepts falling within the same region of the deferral radius undergo one of two operations: expansion or contraction. However, in sLKE, the same concept may correspond to different answers over time, causing the radius to continuously shrink. This leads to the generation of multiple key-value pairs for the same concept, resulting in knowledge becoming inconsistent and ambiguous. Consequently, we observe a significant decline in performance for sLKE.

In the case of WISE, the deficiency arises from conflict between experts. It adds side memories incrementally. Hence, it is inevitable that side memories have conflicts with each other since edits with similar concept are not conduct at the same time. Consequently, routing accuracy drop down for sLKE.

D.3 Model Compression

To reduce memory usage in the inference phase, we apply EMR merging (Huang et al., 2024), which is a tuning-free model merging algorithm. It merges K weights into one unified weight and K boolean masks, which are 1/32 the size of the original weights. This method enables high merging efficiency and retains performance for experts.

The process can be divided into three steps: elect, mask, and rescale. To unify experts, we elect entries by following the sign of the summation of each entry, then fill the entry with the maximum value that has the same sign as the summation. The algorithm can be represented as $\gamma_{uni} = \text{sgn}(\sum_{t=1}^{K} \tau_t)$, $\tau_{uni} = \gamma_{uni} \odot \epsilon_{uni}$, where $\tau_t = W_{v'} - W_v$ for each expert, and ϵ_{uni} is a weight with the maximum absolute value of each entry, consistent in sign with γ_{uni} from all the task vectors.

	Rel.↑	Gen.↑	Loc.↑	Route ACC.↑
Linear	0.78	0.74	1.00	0.80
Slerp	0.78	0.74	1.00	0.80
Ties-Merge	0.87	0.74	1.00	0.87
Dare-Merge	0.90	0.81	1.00	0.94
EMR-Merge	**0.94**	**0.85**	**1.00**	**0.93**

Table 13: Performance of different merging methods on 125 * 8 SLE

After generating the unified vector, we can generate the mask by $M_i = (\tau_i \odot \tau_{uni} > 0)$. These masks can mimic the original task vector τ_i by applying the map on the unified vector, $\tau_i^m = M_i \odot \tau_{uni}$.

However, masking entries in the weight affects the FFN output magnitude. Hence, a rescaler is applied to adjust the values. The rescaler, $\lambda_i = \frac{\text{sum}(\text{abs}(\tau_i))}{\text{sum}(\text{abs}(M_i \odot \tau_{uni}))}$, can be interpreted as rescaling the masked vector to the same magnitude as the original task vector. Hence, the final masked vector will be:

$$\tau_i^m = \lambda_i \cdot M_i \odot \tau_{uni} \tag{13}$$

We conduct ablation study using different merging strategy to give unify our experts. It can be observed that EMR-Merging is more suitable for our methods.

- **Linear**. It combines weight with in linear

weight average of multiple model.

- **Slerp**. It combines weight with spherical interpolation.
- **Ties-Merge**. It combine weight by three process, *TRIM*, *ELECT* and *Sign*. *TRIM* remove redundant weight. *ELECT* selects important weight across different weights. *SIGN* determine the sign of each entry.
- **Dare-Ties**. It use bernoulli distribution to remove redundant weight and use ties-merge to merge model weight. The main difference lies in the *ELECT* process.
- **EMR-Mergning** It combine weight by three process, *ELECT*, *MASK*, *RESCALE*. *ELECT* select important weight in each weight by comparing entry's magnitude. *MASK* generate masks from unified model to mimic actions of original weight. *RESCALE* adjust the value of feed forward process since some entries are masked. This is the method we apply in our work.

E Liscence

The Llama models are licensed under the applicable Llama Community License Agreement and accompanying Acceptable Use Policy.