WHEN EDITS ACCUMULATE: CONSTRAINT-AWARE POLICY LEARNING FOR LIFELONG MODEL EDITING

Anonymous authors
Paper under double-blind review

ABSTRACT

Lifelong model editing aims to continuously update knowledge in large language models without full retraining. However, sequential parameter updates can accumulate interference and progressively damage previously edited or unrelated knowledge. To address this challenge, we propose **SeqConEdit**, a constraint-aware policy learning framework that formulates lifelong editing as a constrained sequential decision problem. Conditioned on the current editing request and accumulated editing history, SeqConEdit adaptively coordinates knowledge incorporation and preservation through a state-conditioned policy. Specifically, we introduce a *Preservation-Key Constraint* (PKC) to suppress interference along protected knowledge directions and a *Preservation Trace Buffer* (PTB) to retain historical preservation information across editing steps. Building on these mechanisms, constraint-aware policy optimization dynamically balances editing and preservation according to feedback from the sequential editing process. Extensive experiments show that SeqConEdit consistently improves lifelong editing performance, outperforming HiEdit by up to 11.94% in average performance while substantially enhancing long-term retention of edited knowledge. Further analyses demonstrate improved knowledge preservation, cross-dataset generalization, and policy optimization stability under long edit streams. The source code and implementation details are publicly available at https://anonymous.4open.science/r/SeqConEdit/.

1 INTRODUCTION

Model editing (De Cao et al., 2021; Mitchell et al., 2021; 2022) aims to update specific knowledge (Zhang et al., 2026; Park et al., 2026) in large language models without costly retraining while preserving model behavior on unrelated inputs (Meng et al., 2022a;b; Tan et al., 2024; Cohen et al., 2024). Most existing methods focus on isolated or batched edits, yet practical models often need to absorb a continuous stream of knowledge updates, giving rise to *lifelong model editing* (Zhang et al., 2024; Wang et al., 2024; Gupta et al., 2025; Fei et al., 2026). In this setting, each edit is applied to a model already modified by previous updates. As edits accumulate, small parameter perturbations can compound over time and progressively interfere with previously edited or unrelated knowledge (Ma et al., 2025; Jiang et al., 2025; Li et al., 2026). Lifelong editing therefore requires not only effective knowledge incorporation, but also sustained control over cumulative interference.

Existing lifelong editing methods address this challenge from different perspectives (Hu et al., 2024; Chen et al., 2024; Wang et al., 2024). Preservation-oriented approaches constrain parameter updates to reduce disruption to existing knowledge (Gu et al., 2024; Fang et al., 2025; Xu et al., 2025), while recent reinforcement-learning-based methods improve the adaptivity of editing decisions over long edit streams (Li et al., 2025; Wang et al., 2026). These directions highlight the importance of both *update-level preservation* and *policy-level adaptation*. However, they are largely studied separately: preservation mechanisms typically impose fixed constraints on parameter updates, whereas policy-based editors adapt editing decisions without explicitly coupling them to persistent, state-dependent preservation. This leaves a central challenge in lifelong editing: **how can an editing policy adaptively coordinate knowledge incorporation and preservation as edits accumulate?**

We address this challenge by formulating lifelong model editing as a *constrained sequential decision problem*, where each editing decision depends on an evolving state shaped by preceding updates. A state-conditioned policy jointly adapts knowledge incorporation and preservation according to the

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

current editing request and accumulated history, enabling editing behavior and preservation pressure to respond to prior interventions and better control cumulative interference over long edit streams.

Based on this formulation, we propose **SeqConEdit**, a constraint-aware policy learning framework for lifelong model editing. SeqConEdit introduces a *Preservation-Key Constraint* (PKC) to suppress update responses along key directions associated with knowledge that should remain stable during sequential editing. To extend such protection across successive edits, we further design a *Preservation Trace Buffer* (PTB) that retains historical preservation keys throughout the editing trajectory. Finally, *Constraint-Aware Policy Optimization* (CAPO) adaptively enforces preservation requirements while incorporating distribution-level locality feedback and group-relative optimization, enabling more stable policy learning over long edit sequences.

Extensive experiments on ZsRE and CounterFact across multiple LLM backbones demonstrate the effectiveness of SeqConEdit under lifelong sequential editing. Under the $8{,}000\times1$ protocol, SeqConEdit consistently improves overall editing performance and substantially strengthens long-term retention of edited knowledge, outperforming HiEdit by up to 11.94% in average performance. Further analyses demonstrate improved knowledge preservation, cross-dataset generalization, and policy optimization stability as edits accumulate. Our contributions are summarized as follows:

- We formulate lifelong model editing as a constrained sequential decision problem, enabling editing policies to adapt knowledge incorporation and preservation according to the current editing request and accumulated history.
- We propose **SeqConEdit**, which couples a state-conditioned editing policy with the *Preservation-Key Constraint* and *Preservation Trace Buffer*, providing adaptive and persistent protection against interference from sequential parameter updates.
- We develop a constraint-aware policy optimization strategy that combines adaptive preservation enforcement, distribution-level locality regularization, and group-relative optimization. Extensive experiments demonstrate improved lifelong editing performance, edited-knowledge retention, robustness, and optimization stability across datasets and model backbones.

2 Related Work

Knowledge Editing and Preservation. Knowledge editing aims to revise selected factual associations in a pretrained model without full retraining (Yao et al., 2023; Gu et al., 2026; Youssef et al., 2026). Early approaches use auxiliary editors or external memories (De Cao et al., 2021; Mitchell et al., 2021; 2022), while locate-and-edit methods rewrite feed-forward modules as key–value memories (Meng et al., 2022a;b; Li et al., 2024), with extensions to vision–language models (Cheng et al., 2023; Huang et al., 2024; Chen et al., 2025). Because repeated rewriting can interfere with unrelated knowledge and degrade general capabilities (Gupta et al., 2024; Gu et al., 2024; Ma et al., 2024), preservation-oriented methods constrain updates through the numerical null space of preserved keys (Fang et al., 2025) or subspaces orthogonal to prior edits (Cai & Cao, 2024). **SeqConEdit** follows the same key–value paradigm but replaces fixed preservation constraints with an adaptive temporal mechanism: protected directions persist across successive editing steps, while their influence on each update is dynamically controlled by the policy according to the current editing state.

Lifelong and Sequential Model Editing. Under a stream of committed edits, each update acts on an already-modified model and may erase pretrained knowledge or earlier edits (Huang et al., 2023; Gupta et al., 2024; Yang et al., 2026). Existing remedies fall into three groups: isolating edits in dedicated memory or experts (Hartvigsen et al., 2023; Wang et al., 2024; Wang & Li, 2024); adapting *where* the edit lands (Zhang et al., 2024; Li & Chu, 2025); and bounding *how much* each edit perturbs the model, through neuron-level rewinding (Jiang et al., 2025), condition-number control (Ma et al., 2025), or norm regularization (Gupta et al., 2025). The most closely related methods introduce adaptive control into lifelong editing: RLEdit (Li et al., 2025) formulates lifelong editing as reinforcement learning, HiEdit (Wang et al., 2026) introduces hierarchical layer selection, and LyapLock (Wang et al., 2025) imposes bounded preservation during sequential updates. In contrast, SeqConEdit formulates lifelong editing as *state-conditioned constrained control*: a single policy jointly determines the update magnitude, soft preservation strength, and layer scope at each step, turning preservation from a post-hoc locality statistic into an explicit stepwise constraint conditioned on both the current request and accumulated editing history.

3 PRELIMINARY

3.1 KEY–VALUE MODEL EDITING

Model editing aims to modify specific factual associations in a pretrained language model without full retraining. Following the locate-and-edit paradigm (Meng et al., 2022b), transformer feed-forward modules can be viewed as key–value memories (Geva et al., 2021), where an editable layer maps a key representation $\mathbf{k} \in \mathbb{R}^{d_k}$ to a value representation $\mathbf{v} \in \mathbb{R}^{d_v}$ through $\mathbf{v} = \mathbf{W}_\ell \mathbf{k}$. Given edit keys $\mathbf{K}_\ell \in \mathbb{R}^{d_k \times n}$ and target value-side residuals $\mathbf{R}_\ell \in \mathbb{R}^{d_v \times n}$, a standard regularized update solves

$$\min_{\Delta \mathbf{W}_\ell} \|\Delta \mathbf{W}_\ell \mathbf{K}_\ell - \mathbf{R}_\ell\|_F^2 + \lambda_\ell \|\Delta \mathbf{W}_\ell\|_F^2 . \tag{1}$$

This objective realizes the desired value changes along edit-related directions, but does not explicitly restrict interference with knowledge that should remain unchanged. Under repeated editing, such perturbations can accumulate over time.

3.2 LIFELONG EDITING AS A CONSTRAINED SEQUENTIAL PROBLEM

Let $f_{\boldsymbol{\theta}_0}$ denote the pretrained model and $\mathcal{E}_{1:N} = \{e_t\}_{t=1}^N$ a stream of editing requests. At step t, an action a_t updates the current model as $\boldsymbol{\theta}_t = \Phi(\boldsymbol{\theta}_{t-1}; e_t, a_t)$. Recursively, $\boldsymbol{\theta}_t$ depends on the preceding requests and actions $(e_{1:t}, a_{1:t})$. Therefore, each edit not only determines the current knowledge update but also changes the parameter state on which subsequent edits operate. Its effect can consequently propagate through the remaining edit stream, making lifelong editing inherently sequential rather than a collection of independent single-edit problems (Li et al., 2025).

Beyond incorporating new knowledge, a lifelong editor must also prevent accumulated updates from progressively disrupting knowledge that should remain stable. Motivated by constrained sequential decision making (Achiam et al., 2017), let $u_t = \mathrm{Eff}_t + \mathrm{Gen}_t$ denote editing utility and $p_t = \mathrm{Spe}_t$ an observable surrogate for preserved-knowledge stability. We formulate lifelong editing as

$$\max_{\pi_\psi} \mathbb{E}_{\pi_\psi}\left[\frac{1}{N}\sum_{t=1}^{N} u_t\right], \qquad \text{s.t.} \quad \mathbb{E}_{\pi_\psi}\left[\frac{1}{N}\sum_{t=1}^{N} p_t\right] \geq \tau, \tag{2}$$

where τ specifies the desired preservation level. The objective encourages effective knowledge incorporation over the entire edit stream, while the constraint prevents editing utility from being improved at the expense of excessive preservation degradation. This formulation motivates an editing policy that adapts each decision to both the current request and the accumulated effects of previous updates. Further details are provided in Appendix B.1.

4 METHODOLOGY

We introduce **SeqConEdit**, a constraint-aware policy learning framework for lifelong model editing. As illustrated in Figure 1, SeqConEdit integrates state-conditioned editing, preservation-aware updating, and constraint-aware policy optimization in a closed loop. At each step, the controller decides how and where to edit; PKC and PTB constrain updates to preserve knowledge; and preservation-aware rewards guide subsequent decisions over long edit streams.

4.1 STATE-CONDITIONED EDITING POLICY

Editing requirements vary across requests and evolve as previous updates accumulate. SeqConEdit therefore learns a state-conditioned policy $\pi_\psi(\mathbf{a}_t \mid \mathbf{s}_t)$, where $\mathbf{s}_t = (\mathbf{e}_t, \mathbf{h}_t)$ combines the current edit with a summary of editing history, including recent outcomes and accumulated parameter-update magnitude. The policy outputs $\mathbf{a}_t = (\alpha_t, \beta_t, \rho_t, \mathbf{m}_t)$, where α_t, β_t, and ρ_t control edit strength, regularization, and preservation, respectively, while $\mathbf{m}_t$ specifies the selected editable modules.

The policy contains a continuous branch and a discrete routing branch. The former samples latent controls from $\mathcal{N}(\boldsymbol{\mu}_\psi(\mathbf{s}_t), \boldsymbol{\Sigma}_\psi)$ and maps them to bounded ranges, while the latter predicts module-routing logits $\boldsymbol{\xi}_\psi(\mathbf{s}_t)$ and uses Gumbel-Top-K during training and deterministic Top-K at inference. This allows the controller to adapt both update strength and intervention location to the evolving editing state. Details of the controller and state construction are provided in Appendix A.5.

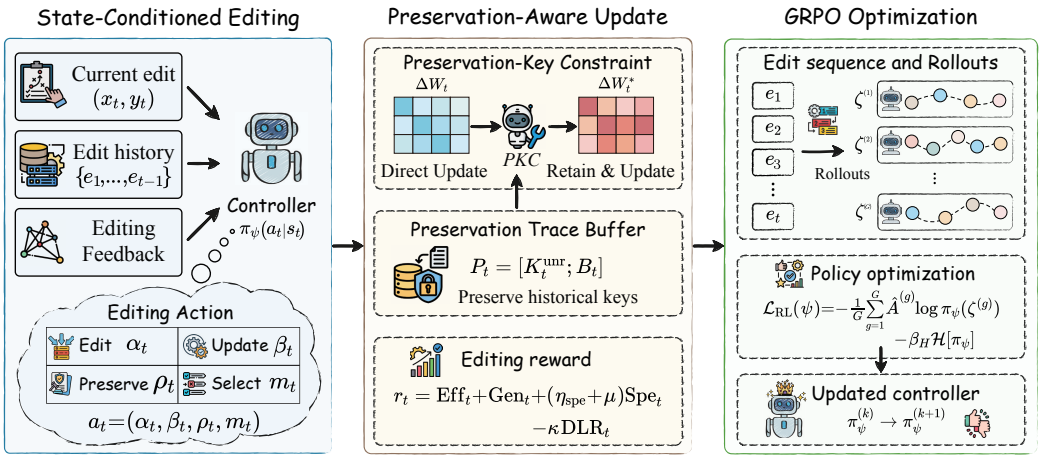


Figure 1: Overview of SeqConEdit for constraint-aware lifelong model editing. Detailed in Sec. 4.

4.2 Preservation-Key Constraint

Standard key–value editing realizes desired responses on edit-related keys, but does not explicitly constrain interference with knowledge that should remain unchanged. We therefore introduce a *Preservation-Key Constraint* (PKC) to suppress update responses along protected key directions.

At editing step t and layer ℓ, let $\mathbf{K}_{\ell,t} \in \mathbb{R}^{d_k \times n}$ denote the edit keys, $\mathbf{R}_{\ell,t} \in \mathbb{R}^{d_v \times n}$ the target value-side residuals, and $\mathbf{P}_{\ell,t} \in \mathbb{R}^{d_k \times q}$ a preservation matrix containing directions whose responses should remain stable. Ideally, the edited layer should preserve its mapping on $\mathbf{P}_{\ell,t}$:

$$(\mathbf{W}_{\ell,t-1} + \Delta\mathbf{W}_{\ell,t})\,\mathbf{P}_{\ell,t} \approx \mathbf{W}_{\ell,t-1}\mathbf{P}_{\ell,t}, \qquad \text{i.e.,} \quad \Delta\mathbf{W}_{\ell,t}\mathbf{P}_{\ell,t} \approx \mathbf{0}. \tag{3}$$

Existing null-space methods preserve prior knowledge by restricting updates to the null space of protected directions (Fang et al., 2025), whereas PKC adopts a soft, state-adaptive relaxation controlled by ρ_t. See Appendix B.3 for details. Specifically, SeqConEdit solves

$$\min_{\Delta\mathbf{W}_{\ell,t}} \|\Delta\mathbf{W}_{\ell,t}\mathbf{K}_{\ell,t} - \alpha_t\mathbf{R}_{\ell,t}\|_F^2 + \rho_t \|\Delta\mathbf{W}_{\ell,t}\mathbf{P}_{\ell,t}\|_F^2 + \beta_t\lambda_\ell \|\Delta\mathbf{W}_{\ell,t}\|_F^2 . \tag{4}$$

The three terms enforce the requested edit, suppress interference along preserved directions, and regularize update magnitude, respectively. Setting the gradient of Eq. 4 to zero gives

$$\Delta\mathbf{W}_{\ell,t}\left(\mathbf{K}_{\ell,t}\mathbf{K}_{\ell,t}^\top + \rho_t\mathbf{P}_{\ell,t}\mathbf{P}_{\ell,t}^\top + \beta_t\lambda_\ell\mathbf{I}\right) = \alpha_t\mathbf{R}_{\ell,t}\mathbf{K}_{\ell,t}^\top. \tag{5}$$

Accordingly, the preservation-aware update is

$$\Delta\mathbf{W}_{\ell,t} = \alpha_t\mathbf{R}_{\ell,t}\mathbf{K}_{\ell,t}^\top\left(\mathbf{K}_{\ell,t}\mathbf{K}_{\ell,t}^\top + \rho_t\mathbf{P}_{\ell,t}\mathbf{P}_{\ell,t}^\top + \beta_t\lambda_\ell\mathbf{I}\right)^{-1}. \tag{6}$$

Thus, ρ_t adaptively determines how strongly preserved directions resist the current update, rather than imposing the same hard constraint across all editing states. Given L candidate layers, the controller further selects $\mathbf{m}_t \in \{0,1\}^L$ with $\|\mathbf{m}_t\|_0 = K$ and applies $\mathbf{W}_{\ell,t} = \mathbf{W}_{\ell,t-1} + m_{t,\ell}\Delta\mathbf{W}_{\ell,t}$. The full derivation of Eq. 6 is provided in Appendix B.3.

4.3 Preservation Trace Buffer

PKC only protects directions represented in the current preservation matrix, similar to preservation-oriented editing methods that constrain updates with respect to retained knowledge directions (Fang et al., 2025). Directions encountered in earlier edits may therefore disappear from later constraints and become vulnerable to subsequent updates. To maintain temporal continuity, we introduce a *Preservation Trace Buffer* (PTB) $\mathcal{B}_{\ell,t}$ for each editable layer. Let $\mathbf{K}_{\ell,t}^{\text{unr}}$ denote current unrelated keys and $\mathbf{B}_{\ell,t}$ the historical keys retained by the buffer. We construct

$$\mathbf{P}_{\ell,t} = \left[\mathbf{K}_{\ell,t}^{\text{unr}}, \mathbf{B}_{\ell,t}\right], \qquad \mathbf{B}_{\ell,t+1} = \text{FIFO}_C\left(\mathbf{B}_{\ell,t} \oplus \mathbf{K}_{\ell,t}^{\text{unr}}\right), \tag{7}$$

where $\oplus$ denotes concatenation and C is the buffer capacity. PTB maintains preservation directions across successive editing steps, while their effect on each update is adaptively controlled by the policy through ρ_t, without introducing any additional trainable parameters.

4.4 Constraint-Aware Policy Optimization

Adaptive preservation constraint. Following the preservation requirement in Eq. 2, we employ an adaptive Lagrangian weighting scheme (Tessler et al., 2018) to adjust preservation pressure according to the observed preservation level. Specifically, η_{spe} provides a base weight for Specificity, while an adaptive multiplier μ adjusts the additional preservation pressure during policy optimization:

$$r_t^{\text{lag}} = \text{Eff}_t + \text{Gen}_t + (\eta_{\text{spe}} + \mu)\text{Spe}_t, \qquad \mu \leftarrow \Pi_{[0,\mu_{\max}]}\left[\mu + \eta_\mu(\tau - \overline{\text{Spe}})\right]. \tag{8}$$

Here, $\overline{\text{Spe}}$ denotes the empirical preservation level of the current batch. The multiplier μ increases when $\overline{\text{Spe}} < \tau$ and decreases otherwise, thereby adapting preservation pressure to the evolving editing state. Further details are provided in Appendix B.1.

Distributional locality regularization. Specificity evaluates preservation at the outcome level and may remain unchanged even when the predictive distribution has already drifted substantially. To capture such early deviations, we introduce *Distributional Locality Regularization* (DLR). For each protected token position (z, j), we define the reference distribution as

$$q_{z,j}^{\text{ref}} = \begin{cases} p_{\boldsymbol{\theta}_0}(\cdot \mid z_{<j}), & z \in \mathcal{V}_t^{\text{unr}}, \\ p_{\boldsymbol{\theta}_{t_z}}(\cdot \mid z_{<j}), & z \in \mathcal{V}_t^{\text{edit}}, \end{cases} \tag{9}$$

where t_z denotes the step at which the previously edited knowledge z was committed. DLR then measures the deviation of the current predictive distribution from its preserved reference:

$$\text{DLR}_t = \frac{1}{|\mathcal{V}_t|} \sum_{(z,j)\in\mathcal{V}_t} \text{KL}\left(p_{\boldsymbol{\theta}_t}(\cdot \mid z_{<j}) \,\middle\|\, q_{z,j}^{\text{ref}}\right). \tag{10}$$

The final reward is $r_t = r_t^{\text{lag}} - \kappa\text{DLR}_t$. Thus, Specificity reflects observable preservation outcomes, while DLR provides a finer-grained signal for emerging distributional drift before such changes become large enough to alter the model's final prediction.

Policy optimization. Reward scales can vary across edit sequences, making raw returns noisy for policy optimization. Inspired by GRPO (Shao et al., 2024), we sample G rollouts for the same length-H edit sequence and normalize their returns within the group:

$$R^{(g)} = \sum_{t=1}^{H} r_t^{(g)}, \qquad \hat{A}^{(g)} = \frac{R^{(g)} - \overline{R}}{\sigma_R + \epsilon}. \tag{11}$$

Because all rollouts share the same edit sequence, group-relative normalization reduces sequence-specific difficulty while retaining policy-induced differences, providing a more stable comparison across trajectories. The controller is optimized with

$$\mathcal{L}_{\text{RL}}(\psi) = -\frac{1}{G}\sum_{g=1}^{G} \hat{A}^{(g)} \log \pi_\psi(\zeta^{(g)}) - \beta_H \mathcal{H}[\pi_\psi], \tag{12}$$

where $\zeta^{(g)} = \{(\mathbf{s}_t^{(g)}, \mathbf{a}_t^{(g)})\}_{t=1}^{H}$ denotes the state–action trajectory of rollout g. Adaptive preservation enforcement and DLR jointly shape the trajectory-level reward, while group-relative normalization reduces sequence-induced return variation and provides a more stable credit-assignment signal across long editing trajectories. Additional optimization details are provided in Appendix B.5.

5 Experiments

5.1 Experimental Setup

Datasets & Metrics. We evaluate SeqConEdit on two widely used lifelong editing benchmarks, ZsRE (Levy et al., 2017a) and CounterFact (Meng et al., 2022a), following prior work (Li et al., 2025; Wang et al., 2026). The two benchmarks provide complementary locality settings: ZsRE evaluates preservation using out-of-distribution open-domain probes, whereas CounterFact uses semantically related neighborhood prompts. We report **Efficacy** (*Eff.*), **Generalization** (*Gen.*), **Specificity** (*Spe.*), and **Edited Retention** (*Ret.*), measuring successful knowledge incorporation, generalization to equivalent queries, preservation of unrelated knowledge, and long-term retention of earlier edits, respectively. Detailed dataset and metric definitions are provided in Appendix A.1 and Appendix A.2.

Table 1: Comparison on different lifelong model editing method. *Eff.*, *Gen.*, *Spe.*, and *Ret.* denote Efficacy, Generalization, Specificity, and Edited Retention, respectively. *Avg.* is the arithmetic mean of four metrics per dataset. The **Bold** and underline mark the best and second-best results.

LLMs	Method	ZsRE					CounterFact				
		Eff.(↑)	*Gen.*(↑)	*Spe.*(↑)	*Ret.*(↑)	*Avg.*(↑)	*Eff.*(↑)	*Gen.*(↑)	*Spe.*(↑)	*Ret.*(↑)	*Avg.*(↑)
Llama-3-8B	FT	9.16	8.11	1.95	8.82	7.01	65.24	57.99	44.03	62.80	57.52
	ROME	3.29	3.24	0.66	2.67	2.47	59.26	56.45	48.52	48.30	53.13
	MEMIT	0.00	0.00	0.12	0.00	0.03	18.61	16.94	16.93	16.20	17.17
	PRUNE	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	RECT	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	AlphaEdit	16.22	14.58	3.87	9.10	10.94	**73.06**	**66.43**	48.54	47.40	58.86
	MEND	0.00	0.00	0.00	0.02	0.01	7.69	13.40	7.45	10.50	9.76
	MALMEN	5.09	4.74	0.72	4.26	3.70	0.00	0.00	0.00	0.00	0.00
	DAFNet	21.15	20.41	21.15	21.13	20.96	29.56	31.55	**68.38**	33.60	40.77
	RLEdit	81.43	79.49	42.73	70.72	68.59	66.35	55.26	44.79	57.20	55.90
	HiEdit	82.10	79.99	**48.42**	75.16	71.42	66.53	55.65	45.70	58.20	56.52
	SeqConEdit	**88.01**	**85.99**	43.82	**81.70**	**74.88**	69.57	59.31	50.23	**64.50**	**60.90**
Gemma-2-9B	FT	35.38	32.31	30.46	33.40	32.89	60.13	39.49	52.26	42.70	48.65
	ROME	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MEMIT	10.54	10.40	14.66	9.22	11.21	31.51	28.78	65.58	30.10	38.99
	PRUNE	10.46	10.50	14.66	9.46	11.27	32.41	29.78	65.10	29.80	39.27
	RECT	11.26	11.25	16.19	9.62	12.08	30.72	28.34	66.08	30.00	38.79
	AlphaEdit	15.79	15.32	20.21	13.19	16.13	38.17	38.83	66.51	34.00	44.38
	MEND	20.78	20.29	12.34	21.11	18.63	27.28	28.54	70.20	27.70	38.43
	MALMEN	9.30	8.87	12.90	11.22	10.57	10.46	13.12	**86.94**	11.40	30.48
	DAFNet	23.02	22.17	28.32	23.15	24.17	10.23	12.74	86.60	13.50	30.77
	RLEdit	69.11	65.87	25.17	57.66	54.45	55.51	49.92	46.28	45.80	49.38
	HiEdit	82.12	78.40	32.40	68.73	65.41	64.06	52.80	43.59	54.00	53.61
	SeqConEdit	**85.41**	**80.73**	**34.79**	**83.24**	**71.04**	**73.45**	**53.67**	57.06	**78.00**	**65.55**

Base LLMs & Baselines. We evaluate SeqConEdit across multiple LLM families, including Llama-3 (8B) (Grattafiori et al., 2024) and Gemma-2 (9B) (Team et al., 2024). We compare against representative methods spanning fine-tuning, locate-and-edit, preservation-oriented, and lifelong editing approaches, including FT (Zhu et al., 2020), ROME (Meng et al., 2022a), MEMIT (Meng et al., 2022b), PRUNE (Ma et al., 2025), RECT (Gu et al., 2024), AlphaEdit (Fang et al., 2025), MEND (Mitchell et al., 2021), MALMEN (Tan et al., 2024), DAFNet (Zhang et al., 2024), RLEdit (Li et al., 2025), and HiEdit (Wang et al., 2026). Additional details are provided in Appendix A.4.

Protocol & Implementation. We follow the $8{,}000\times1$ lifelong editing protocol of HiEdit (Wang et al., 2026). For each of three independent runs, we shuffle the dataset, use the first 8,000 instances as the editing stream, and evaluate all methods on the same sequence. **Reported results are averaged over the three runs**. Full details are provided in Appendix A.6.

5.2 Main Results

Table 1 shows that SeqConEdit achieves the highest *Avg.* across all evaluated settings under the $8{,}000\times1$ lifelong editing protocol. Compared with HiEdit, gains are consistent across all settings, reaching 11.94% on Gemma-2-9B / CounterFact. The improvement is particularly evident in *Ret.*, where SeqConEdit consistently outperforms HiEdit, indicating stronger resistance to forgetting as sequential edits continue to accumulate over long edit streams. SeqConEdit also achieves the best *Eff.* and *Gen.* across all settings, showing that long-term preservation does not compromise knowledge incorporation. Overall, **SeqConEdit achieves a better balance between editing effectiveness and long-term preservation** under lifelong sequential editing. While several baselines achieve higher *Spe.* in some settings, these gains are often accompanied by substantial degradation in editing effectiveness or retention, reflecting a weaker editing–preservation trade-off.

5.3 Ablation Studies

Component Ablation. Table 2 (a) evaluates the major components of SeqConEdit. Removing PKC or PTB decreases *Avg.* by 1.95% and 2.61%, respectively, indicating that weakening either step-wise preservation constraints or persistent historical protection leads to greater interference under

Table 2: Ablation studies of SeqConEdit on Gemma-2-9B with ZsRE under the $8{,}000\times1$ sequential editing protocol. (a) Component ablation, *averaged over three independent runs*. (b) Policy ablation *using a single trained checkpoint*. Comparisons are made within each panel.

(a) Component Ablation

Variant	*Eff.*(↑)	*Gen.*(↑)	*Spe.*(↑)	*Ret.*(↑)	*Avg.*(↑)
SeqConEdit	**85.41**	**80.73**	34.79	**83.24**	**71.04**
w/o PKC	82.82	77.40	37.73	78.40	69.09$_{-1.95}$
w/o PTB	80.21	74.92	**39.47**	79.11	68.43$_{-2.61}$
w/o CAPO	83.15	76.59	38.06	77.24	68.76$_{-2.28}$
– DLR	85.06	79.43	35.55	80.76	70.20$_{-0.84}$

(b) Policy Ablation

Variant	*Eff.*(↑)	*Gen.*(↑)	*Spe.*(↑)	*Ret.*(↑)	*Avg.*(↑)
SeqConEdit	**86.89**	**81.80**	34.76	**82.96**	**71.60**
w/o history	83.10	77.81	36.29	80.40	69.40$_{-2.20}$
w/o update	82.95	78.23	**36.76**	80.73	69.67$_{-1.93}$
fixed routing	83.68	78.25	35.13	81.01	69.52$_{-2.08}$
fixed controls	82.43	78.60	34.06	82.25	69.34$_{-2.27}$

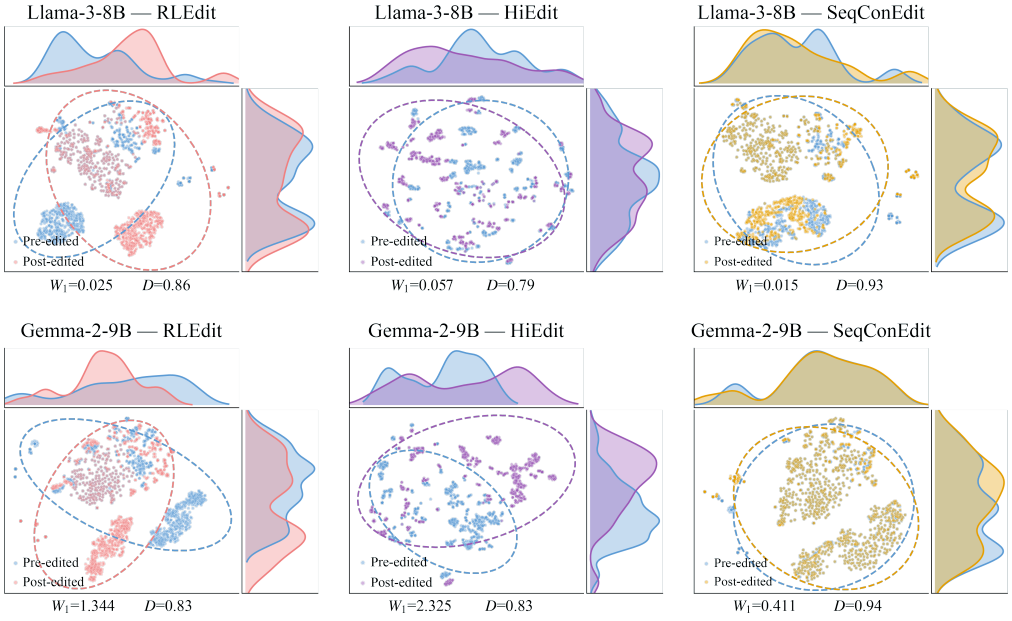


Figure 2: Representation stability after 8,000 sequential edits. Pre- and post-edit representations of 1,000 fixed probes on Llama-3-8B (top) and Gemma-2-9B (bottom) under CounterFact editing. Lower W_1 and higher D indicate better distribution preservation. Best viewed in color.

sequential updates. Removing CAPO causes a 2.28% drop, showing the importance of adaptively balancing editing and preservation, while removing DLR yields a smaller 0.84% degradation due to reduced distribution-level locality supervision. Complete ZsRE and CounterFact results are provided in Appendix C.3, while policy optimization comparisons are provided in Appendix C.5.

Policy Ablation. Table 2 (b) examines the state information and adaptive action branches of the policy. Removing historical information or the accumulated-update signal reduces *Avg.* by 2.20% and 1.93%, as the policy loses context about the evolving editing state. Fixing module routing or the continuous controls similarly decreases *Avg.* by 2.08% and 2.27%, showing the benefit of jointly adapting intervention location and update behavior. Additional details are provided in Appendix C.4.

5.4 Depth Analysis

Representation Preservation under Lifelong Editing. To examine representation stability under lifelong editing, we select a fixed set of 1,000 samples and extract their hidden representations before and after the full 8,000-edit CounterFact stream, which are projected into two dimensions using t-SNE (Van der Maaten & Hinton, 2008). As shown in Figure 2, SeqConEdit consistently exhibits stronger pre/post representation overlap than RLEdit and HiEdit on both backbones, with lower Wasserstein shift and greater density overlap. The advantage is particularly pronounced on Gemma-2-9B, where SeqConEdit reduces W_1 to 0.411 while maintaining a high overlap of $D = 0.94$. These results provide representation-level evidence that PKC and PTB effectively mitigate cumulative

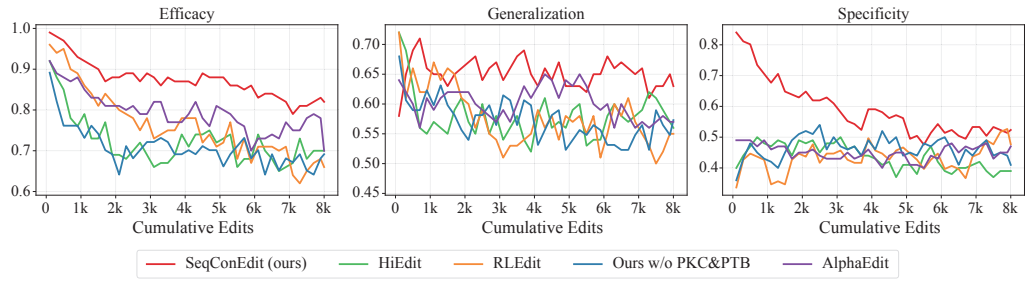


Figure 3: Robust editing performance at scale. Efficacy, generalization, and specificity of Gemma-2-9B as sequential CounterFact edits accumulate.

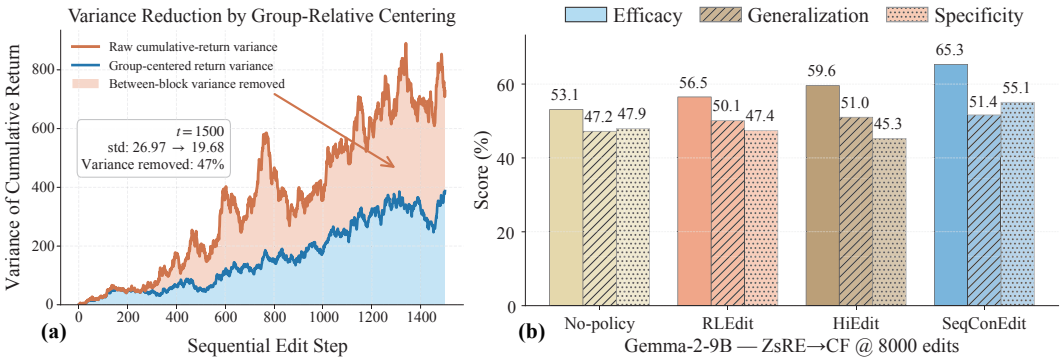


Figure 4: Policy optimization and cross-distribution robustness. (a) Group-relative centering reduces sequence-dependent return variance during policy optimization. (b) Cross-dataset performance of Gemma-2-9B after 8,000 ZsRE edits and evaluation on CounterFact.

drift under sequential updates. Appendix C.1 further tracks representation drift at intermediate edit counts, showing that SeqConEdit maintains stronger alignment throughout the edit stream.

Performance over Lifelong Editing. Figure 3 examines how editing performance evolves as sequential edits accumulate. Across 8,000 CounterFact edits, SeqConEdit maintains consistently higher Efficacy and more stable Generalization and Specificity than competing lifelong editors. Removing PKC and PTB leads to faster degradation, with the gap becoming more pronounced as the edit stream grows. These trajectories show that the preservation mechanisms become increasingly important as interference accumulates. Removing PKC and PTB weakens both current and historical preservation constraints, allowing interference to accumulate more rapidly across sequential edits.

Effect of Group-Relative Optimization. Figure 4 (a) examines why group-relative optimization benefits policy learning in lifelong editing. Raw trajectory returns can vary substantially across edit sequences due to differences in sequence difficulty, making policy-induced differences harder to distinguish. By comparing rollouts within the same edit sequence, group-relative advantage estimation suppresses such sequence-dependent variation while preserving differences induced by policy actions. At $t = 1{,}500$, the return standard deviation decreases from 26.97 to 19.68, corresponding to a 47% reduction in variance. This provides evidence that group-relative optimization stabilizes policy learning by yielding a cleaner and more informative credit-assignment signal.

5.5 Robustness and Generalization

Cross-Dataset Generalization. Figure 4 (b) evaluates whether the learned editing policy generalizes across datasets. We train the controller on ZsRE, freeze it, and apply it to an $8{,}000\times1$ CounterFact editing stream from the original pretrained model, with evaluation also conducted on CounterFact. This setting isolates policy-level generalization by preventing any target-domain policy adaptation. SeqConEdit outperforms the compared lifelong editing methods, improving Efficacy and Specificity over HiEdit by 5.7% and 9.8%, respectively, while maintaining comparable Generalization.

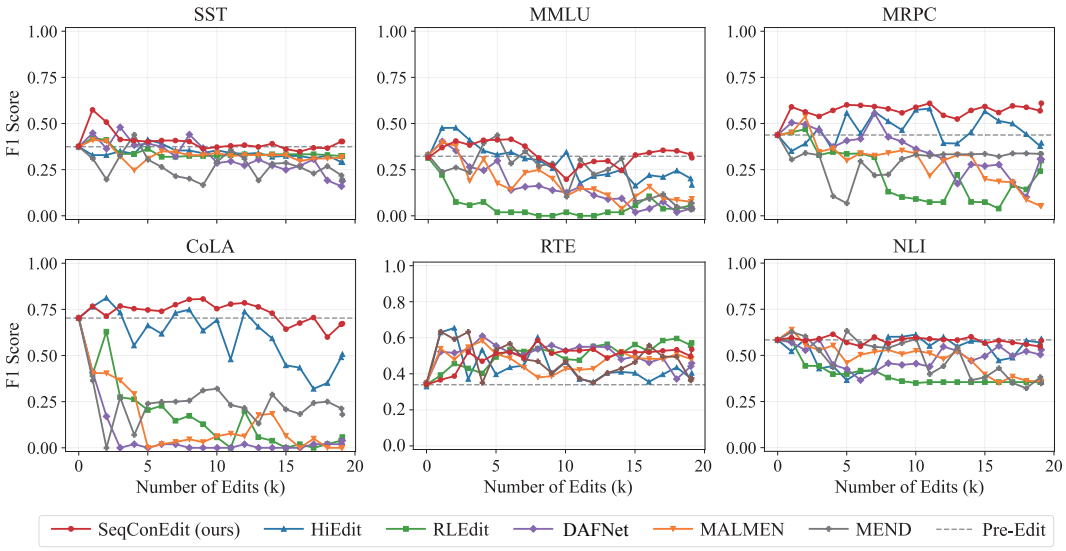


Figure 5: General capability retention under long-range sequential editing. Performance of Gemma-2-9B on six downstream tasks throughout the full ZsRE evaluation editing stream. The dashed line denotes the pre-edit reference. Best viewed in color.

These results indicate that the learned editing strategy remains effective under distribution shift and generalizes beyond the data distribution used for policy optimization. Detailed experimental settings, transfer protocols, and additional cross-dataset results are provided in Appendix C.2.

General Capability Preservation. We further examine whether lifelong editing preserves model capabilities beyond editing-specific objectives. Figure 5 tracks Gemma-2-9B across the full ZsRE editing stream on six general capability benchmarks (Wang et al., 2019). SeqConEdit remains closer to the pre-edit model across most tasks as edits accumulate, while competing methods exhibit greater degradation, particularly on CoLA, MMLU, and MRPC. Similar trends are observed on SST-2 and NLI, while RTE shows greater variation across methods. These results suggest that SeqConEdit better limits collateral capability degradation over long edit sequences, extending preservation beyond edited knowledge. Detailed benchmark descriptions are provided in Appendix A.3.

Language-modeling robustness. We further evaluate *whether lifelong sequential editing affects the model's general language-modeling ability* using WikiText perplexity (Merity et al., 2016). On Gemma-2-9B, we edit over the full ZsRE set and track perplexity throughout the process. SeqConEdit remains close to the pre-edit reference, whereas competing lifelong editors exhibit substantially larger perplexity increases, indicating better preservation of general language-modeling ability under cumulative interference. Detailed results are provided in Appendix C.6.

Hyperparameter analysis. We analyze how key policy-optimization and preservation hyperparameters affect SeqConEdit. Results in Appendix C.7 show broadly stable performance across reasonable configurations, with moderate settings providing a good balance between policy optimization and knowledge preservation, indicating that the method is not reliant on a narrowly tuned settings.

6 Conclusion

We study lifelong model editing under long knowledge-update sequences, where repeated edits can accumulate interference and damage preserved knowledge. To address this challenge, we propose **SeqConEdit**, a constraint-aware policy learning framework combining state-conditioned editing with persistent preservation constraints. Experiments on ZsRE and CounterFact show that SeqConEdit improves lifelong editing performance, edited-knowledge retention, and representation stability across multiple LLM backbones. Further analyses show that its preservation mechanisms remain effective as edits accumulate and better maintain broader model capabilities.

AI USE STATEMENT

In this work, we used generative AI tools to assist with drafting and polishing the manuscript, including improving grammar, clarity, readability, and presentation. We did not use generative AI tools to generate datasets, design the research methodology or experiments, implement the proposed method, formulate mathematical claims or proofs, or interpret experimental results. All AI-assisted text was carefully reviewed, verified, and revised by the authors. We take full responsibility for the final content of this work, including all text, claims, and results.

ETHICS STATEMENT

This work studies lifelong model editing using publicly available benchmarks and pretrained language models, and does not involve human subjects or the collection of personal or sensitive data. While model editing can support efficient knowledge correction and updating, it may also be misused to introduce inaccurate or misleading information into language models. Our work focuses on improving the stability and preservation of sequential editing and does not introduce mechanisms specifically intended for harmful content manipulation. We encourage responsible use of model-editing techniques with appropriate validation of edited knowledge and downstream model behavior.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the SeqConEdit framework and optimization objectives in the main text, with complete derivations in Appendix B. Dataset specifications, evaluation metrics, training procedures, and implementation details are documented in Appendix A and Appendix A.6. Additional ablations and hyperparameter analyses are reported in Appendix C.7. Together, these materials provide the necessary details for reproducing the experimental setup and reported results.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pp. 22–31, 2017.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. The fifth pascal recognizing textual entailment challenge. *TAC*, 7(8):1, 2009.

Yuchen Cai and Ding Cao. O-edit: Orthogonal subspace editing for language model sequential editing. *arXiv preprint arXiv:2410.11469*, 2024.

Qizhou Chen, Taolin Zhang, Xiaofeng He, Dongyang Li, Chengyu Wang, Longtao Huang, et al. Lifelong knowledge editing for llms with retrieval-augmented continuous prompt learning. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, pp. 13565–13580, 2024.

Qizhou Chen, Taolin Zhang, Chengyu Wang, Xiaofeng He, Dakan Wang, and Tingting Liu. Attribution analysis meets model editing: Advancing knowledge correction in vision language models with visedit. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 2168–2176, 2025.

Siyuan Cheng, Bozhong Tian, Qingbin Liu, Xi Chen, Yongheng Wang, Huajun Chen, and Ningyu Zhang. Can we edit multimodal large language models? In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 13877–13888, 2023.

Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics*, 12:283–298, 2024.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 6491–6506, 2021.

William B Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the third international workshop on paraphrasing (IWP2005)*, 2005.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. Alphaedit: Null-space constrained knowledge editing for language models. In *International Conference on Learning Representations*, volume 2025, pp. 16366–16396, 2025.

Weizhi Fei, Hao Shi, Jing Xu, Jingchen Peng, Jiazheng Li, Jingzhao Zhang, Bo Bai, Wei Han, Xueyan Niu, et al. Scaling knowledge editing in llms to 100,000 facts with neural kv database. In *International Conference on Learning Representations*, volume 2026, pp. 71032–71058, 2026.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 5484–5495, 2021.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Jia-Chen Gu, Hao-Xiang Xu, Jun-Yu Ma, Pan Lu, Zhen-Hua Ling, Kai-Wei Chang, and Nanyun Peng. Model editing harms general abilities of large language models: Regularization to the rescue. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 16801–16819, 2024.

Xiaojie Gu, Ziying Huang, Jia-Chen Gu, and Kai Zhang. Ultraedit: Training-, subject-, and memory-free lifelong editing in language models. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856.

Akshat Gupta, Anurag Rao, and Gopala Anumanchipalli. Model editing at scale leads to gradual and catastrophic forgetting. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 15202–15232, 2024.

Akshat Gupta, Phudish Prateepamornkul, Maochuan Lu, Ahmed Alaa, Thomas Hartvigsen, and Gopala Anumanchipalli. Lifelong knowledge editing requires better regularization. *arXiv preprint arXiv:2502.01636*, 2025.

Tom Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with grace: Lifelong model editing with discrete key-value adaptors. *Advances in neural information processing systems*, 36:47934–47959, 2023.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Chenhui Hu, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. Wilke: Wise-layer knowledge editor for lifelong knowledge editing. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3476–3503, 2024.

Han Huang, Haitian Zhong, Tao Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. Vlkeb: A large vision-language model knowledge editing benchmark. *Advances in Neural Information Processing Systems*, 37:9257–9280, 2024.

Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. Transformer-patcher: One mistake worth one neuron. *arXiv preprint arXiv:2301.09785*, 2023.

Houcheng Jiang, Junfeng Fang, Tianyu Zhang, Baolong Bi, An Zhang, Ruipeng Wang, Tao Liang, and Xiang Wang. Neuron-level sequential editing for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16678–16702, 2025.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st conference on computational natural language learning (CoNLL 2017)*, pp. 333–342, 2017a.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL)*, pp. 333–342, 2017b.

Qi Li and Xiaowen Chu. Adaedit: Advancing continuous knowledge editing for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4127–4149, 2025.

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

Xiaopeng Li, Shasha Li, Xi Wang, Shezheng Song, Bin Ji, Shangwen Wang, Jun Ma, Xiaodong Liu, Mina Liu, and Jie Yu. Emsedit: Efficient multi-step meta-learning-based model editing. In *Proceedings of the ACM Web Conference 2026*, pp. 3847–3858, 2026.

Zherui Li, Houcheng Jiang, Hao Chen, Baolong Bi, Zhenhong Zhou, Fei Sun, Junfeng Fang, and Xiang Wang. Reinforced lifelong editing for language models. *arXiv preprint arXiv:2502.05759*, 2025.

Jun-Yu Ma, Zhen-Hua Ling, Ningyu Zhang, and Jia-Chen Gu. Neighboring perturbations of knowledge editing on large language models. *arXiv preprint arXiv:2401.17623*, 2024.

Jun-Yu Ma, Hong Wang, Hao-Xiang Xu, Zhen-Hua Ling, and Jia-Chen Gu. Perturbation-restrained sequential model editing. In *International Conference on Learning Representations*, volume 2025, pp. 17302–17329, 2025.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022a.

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

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. Fast model editing at scale. *arXiv preprint arXiv:2110.11309*, 2021.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. Memory-based model editing at scale. In *International conference on machine learning*, pp. 15817–15831, 2022.

Haewon Park, Sangwoo Kim, and Yohan Jo. Suit: Knowledge editing with subspace-aware key-value mappings. In *International Conference on Learning Representations*, volume 2026, pp. 149392–149422, 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Chenmien Tan, Ge Zhang, and Jie Fu. Massive editing for large language models via meta learning. In *International Conference on Learning Representations*, volume 2024, pp. 54676–54690, 2024.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Chen Tessler, Daniel J Mankowitz, and Shie Mannor. Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074*, 2018.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, 2019.

Peng Wang, Zexi Li, Ningyu Zhang, Ziwen Xu, Yunzhi Yao, Yong Jiang, Pengjun Xie, Fei Huang, and Huajun Chen. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. *Advances in Neural Information Processing Systems*, 37:53764–53797, 2024.

Peng Wang, Biyu Zhou, Xuehai Tang, Jizhong Han, and Songlin Hu. Lyaplock: Bounded knowledge preservation in sequential large language model editing. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 6445–6470, 2025.

Renzhi Wang and Piji Li. Lemoe: Advanced mixture of experts adaptor for lifelong model editing of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 2551–2575, 2024.

Yangfan Wang, Tianyang Sun, Chen Tang, Jie Liu, Wei Cai, and Jingchi Jiang. Hiedit: Lifelong model editing with hierarchical reinforcement learning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 39924–39942, 2026.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019.

Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long papers)*, pp. 1112–1122, 2018.

Haoyu Xu, Pengxiang Lan, Enneng Yang, Guibing Guo, Jianzhe Zhao, Linying Jiang, and Xingwei Wang. Knowledge decoupling via orthogonal projection for lifelong editing of large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13194–13213, 2025.

Wanli Yang, Rui Tang, Hongyu Zang, Du Su, Qi Cao, Jingang Wang, Huawei Shen, Xueqi Cheng, and Fei Sun. Fine-tuning done right in model editing. In *International Conference on Learning Representations*, volume 2026, pp. 71683–71702, 2026.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 10222–10240, 2023.

Paul Youssef, Zhixue Zhao, Christin Seifert, and Jörg Schlötterer. Tracing and reversing edits in llms. In *International Conference on Learning Representations*, volume 2026, pp. 23863–23893, 2026.

Mengqi Zhang, Zisheng Zhou, Xiaotian Ye, Qiang Liu, Zhaochun Ren, Zhumin Chen, and Pengjie Ren. Disentangling knowledge representations for large language model editing. In *International Conference on Learning Representations*, volume 2026, pp. 81086–81111, 2026.

Taolin Zhang, Qizhou Chen, Dongyang Li, Chengyu Wang, Xiaofeng He, Longtao Huang, Jun Huang, et al. Dafnet: Dynamic auxiliary fusion for sequential model editing in large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 1588–1602, 2024.

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. Modifying memories in transformer models. *arXiv preprint arXiv:2012.00363*, 2020.

A Additional Experimental Details

This appendix provides additional experimental details corresponding to Sec. 5.1. We first describe the datasets and dataset-specific evaluation metrics, then introduce the evaluated LLM backbones and baseline editors, and finally provide the sequential-editing protocol and implementation details.

A.1 Datasets

ZsRE and CounterFact are both widely used in model editing, but they instantiate editing success, generalization, and locality preservation with different probe constructions and scoring rules.

ZsRE. ZsRE (Levy et al., 2017b) is a question-answering benchmark originally designed for zero-shot relation extraction and has been widely adopted in model editing studies. Each edit instance contains a relation-style question and a target answer to be inserted into the model. To evaluate whether an edit generalizes beyond the exact edited question, ZsRE provides rephrased questions. It also includes unrelated open-domain locality questions, which test whether the edit changes model behavior on knowledge that should remain unaffected. Following prior lifelong editing work (Li et al., 2025; Wang et al., 2026), we use the standard train/test splits. Since the locality probes are out of the edit distribution, ZsRE provides a relatively tight preservation setting.

CounterFact. CounterFact (Meng et al., 2022a) is a counterfactual factual-editing benchmark that asks an editor to replace a model's original factual association with a new target object. Each instance contains a rewrite prompt, paraphrase prompts, and neighborhood prompts. The rewrite and paraphrase prompts evaluate whether the counterfactual target is successfully inserted and generalized, while the neighborhood prompts evaluate whether semantically related but distinct facts remain unchanged. Following prior work (Meng et al., 2022a;b; Wang et al., 2026), we use CounterFact to test whether an edit spills over to neighboring facts. Compared with ZsRE, CounterFact provides a complementary locality setting based on semantically related neighborhood prompts.

A.2 Evaluation Metrics

Although ZsRE and CounterFact share the same high-level editing objectives, they adopt different evaluation protocols. Following prior model editing work (Meng et al., 2022a;b; Li et al., 2025; Wang et al., 2026), we report Efficacy (**Eff.**), Generalization (**Gen.**), Specificity (**Spe.**), and Edited Retention (**Ret.**). Higher values indicate better performance for all metrics.

A.2.1 ZsRE Metrics

For ZsRE, we follow the standard protocol (Meng et al., 2022a;b; Li et al., 2025). Let $f_{\boldsymbol{\theta}_N}$ denote the final model after the complete edit stream. For instance i, let x_i be the original edit question, $\bar{x}_i$ a semantically equivalent rephrased question, z_i an unrelated locality question, y_i^* the edited target answer, and y_i^0 the reference answer for the locality question.

Efficacy. Efficacy measures whether the edited target can be correctly retrieved from the final model when queried with the original edit question:

$$\mathrm{Eff}_{\mathrm{ZsRE}} = \mathbb{E}_i \left[y_i^* = \arg\max_y p_{\boldsymbol{\theta}_N}(y \mid x_i) \right]. \tag{13}$$

Generalization. Generalization measures whether the edited knowledge remains accessible when the same fact is queried through an equivalent rephrased question:

$$\mathrm{Gen}_{\mathrm{ZsRE}} = \mathbb{E}_i \left[y_i^* = \arg\max_y p_{\boldsymbol{\theta}_N}(y \mid \bar{x}_i) \right]. \tag{14}$$

Specificity. Specificity measures whether unrelated knowledge remains correctly preserved after the full editing process without being altered by accumulated updates:

$$\mathrm{Spe}_{\mathrm{ZsRE}} = \mathbb{E}_i \left[y_i^0 = \arg\max_y p_{\boldsymbol{\theta}_N}(y \mid z_i) \right]. \tag{15}$$

A.2.2 COUNTERFACT METRICS

For CounterFact, we adopt the probability-based evaluation protocol commonly used in factual model editing (Meng et al., 2022a;b; Wang et al., 2026). For instance i, let x_i denote the rewrite prompt, $\bar{x}_i$ its paraphrased counterpart, and z_i a neighborhood prompt. We use y_i^* for the edited target object, y_i^0 for the original object of the edited fact, and y_i^{nb} for the correct object associated with the neighborhood prompt.

Efficacy. Efficacy evaluates whether the intended factual update is successfully established in the final edited model by comparing the edited target against the original object on the rewrite prompt:

$$\mathrm{Eff}_{\mathrm{CF}} = \mathbb{E}_i \left[p_{\boldsymbol{\theta}_N}(y_i^* \mid x_i) > p_{\boldsymbol{\theta}_N}(y_i^0 \mid x_i) \right]. \tag{16}$$

Generalization. Generalization evaluates whether the newly established factual preference remains valid under semantically equivalent formulations, rather than being tied to the original rewrite prompt:

$$\mathrm{Gen}_{\mathrm{CF}} = \mathbb{E}_i \left[p_{\boldsymbol{\theta}_N}(y_i^* \mid \bar{x}_i) > p_{\boldsymbol{\theta}_N}(y_i^0 \mid \bar{x}_i) \right]. \tag{17}$$

Specificity. Specificity evaluates whether the edit remains localized to the intended fact by checking that related but unedited neighborhood knowledge still favors its correct object:

$$\mathrm{Spe}_{\mathrm{CF}} = \mathbb{E}_i \left[p_{\boldsymbol{\theta}_N}(y_i^{\mathrm{nb}} \mid z_i) > p_{\boldsymbol{\theta}_N}(y_i^* \mid z_i) \right]. \tag{18}$$

A.2.3 EDITED RETENTION

We additionally report Edited Retention (**Ret.**), a metric introduced by HiEdit (Wang et al., 2026) to measure the long-term retention of previously edited knowledge. After completing all $N = 8{,}000$ edits, we reevaluate the first $N_0 = 500$ edited instances, which have undergone the largest number of subsequent updates. Ret. is computed as the average of Efficacy and Generalization on these initially edited instances using the same dataset-specific scoring rules defined above. Higher Ret. indicates stronger resistance to forgetting under accumulated sequential edits.

A.3 GENERAL CAPABILITY BENCHMARKS

To further assess the general capabilities of post-edited LLMs, we conduct General Capability Tests on representative tasks from the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019), covering sentiment classification, semantic similarity, textual entailment, grammatical acceptability, and natural language inference.

- **SST-2** (Socher et al., 2013) is a sentence-level sentiment classification benchmark constructed from movie reviews. Each sentence is assigned a positive or negative sentiment label, requiring the model to recognize the overall sentiment expressed in natural-language text. We use SST-2 to examine whether sequential editing preserves basic semantic and sentiment understanding.
- **MMLU** (Hendrycks et al., 2020) is a broad multiple-choice benchmark covering diverse academic and professional domains. It evaluates factual knowledge and reasoning across a wide range of subjects, providing a complementary measure of whether lifelong editing preserves the model's general knowledge and problem-solving capabilities beyond language understanding tasks.
- **MRPC** (Dolan & Brockett, 2005) is a sentence-pair classification benchmark for paraphrase recognition. Given two sentences, the model determines whether they convey semantically equivalent meanings. This task evaluates whether editing preserves sentence-level semantic comparison and matching ability.
- **CoLA** (Warstadt et al., 2019) evaluates grammatical acceptability. Given a single English sentence, the model determines whether it is linguistically acceptable, making the task particularly useful for assessing whether repeated editing disrupts the syntactic and grammatical knowledge encoded in the pretrained model.
- **RTE** (Bentivogli et al., 2009) evaluates textual entailment between a premise and a hypothesis. The model predicts whether the meaning of the hypothesis can be inferred from the premise, providing a compact test of semantic reasoning and entailment recognition after sequential editing.

- **NLI** (Williams et al., 2018) is a large-scale natural language inference benchmark containing premise–hypothesis pairs from diverse text genres. The model predicts whether their relation is entailment, contradiction, or neutral. Compared with RTE, NLI provides a broader and more challenging assessment of general natural language inference ability.

A.4 LLMs and Baselines

A.4.1 LLM Backbones

We evaluate SeqConEdit on two representative LLM backbones: Llama-3-8B (Grattafiori et al., 2024) and Gemma-2-9B (Team et al., 2024). These models belong to different model families and are commonly adopted in recent lifelong editing studies, providing a controlled setting for evaluating the generality of SeqConEdit across architectures. Unless otherwise specified, the same SeqConEdit configuration is used across both backbones and datasets.

A.4.2 Baselines

We compare SeqConEdit with representative baselines spanning direct fine-tuning, locate-and-edit approaches, preservation-oriented editors, learned editing methods, and recent lifelong editing frameworks. Together, these methods cover major paradigms for factual model editing and provide comparisons across both conventional editing and sequential lifelong-editing settings.

Locate-and-edit methods. **ROME** (Meng et al., 2022a) edits factual associations by locating the MLP module responsible for recalling a fact and applying a rank-one update to the corresponding feed-forward weights. **MEMIT** (Meng et al., 2022b) extends this key-value editing view from single-fact editing to mass editing by distributing updates across multiple layers. These methods provide direct comparisons to the key-value rewriting foundation of SeqConEdit.

Preservation-oriented editing methods. **PRUNE** (Ma et al., 2025) constrains parameter perturbations during sequential editing by controlling the numerical sensitivity of the update matrix. **RECT** (Gu et al., 2024) regularizes editing updates to reduce degradation of general capabilities after repeated edits. **AlphaEdit** (Fang et al., 2025) improves locality by projecting updates away from directions associated with existing knowledge. These methods aim to reduce editing side effects, but generally rely on fixed preservation rules rather than a state-conditioned preservation policy.

Meta-learning and hypernetwork editors. **MEND** (Mitchell et al., 2021) trains an auxiliary network to transform fine-tuning gradients into localized parameter updates, enabling fast edits from a small number of examples. **MALMEN** (Tan et al., 2024) scales learned editing to massive edit batches by solving memory-efficient least-squares update problems. **DAFNet** (Zhang et al., 2024) studies adaptive learned editing for sequential knowledge updates. These baselines test whether a learned update generator alone is sufficient for long-horizon editing, compared with our explicit preservation-aware control.

Lifelong editing methods. **RLEdit** (Li et al., 2025) formulates lifelong editing as a reinforcement learning problem and trains a hypernetwork-based editor over editing trajectories. **HiEdit** (Wang et al., 2026) further introduces a hierarchical policy to improve exploration and dynamically select editing layers. These two methods are the closest baselines to SeqConEdit. In contrast, SeqConEdit explicitly combines state-conditioned action selection with Preservation-Key Constraints and a Preservation Trace Buffer, so that each update is performed under an adaptive preservation constraint.

Fixed-control variant. We additionally evaluate a **no-policy** variant that removes adaptive controller decisions and instead uses fixed editing controls with full-layer intervention. This variant provides a fixed-control reference for examining the benefit of state-conditioned editing decisions.

For baselines with directly available results under the same backbone and evaluation protocol, we report the corresponding published results. Otherwise, we reproduce the methods using their official implementations and recommended configurations, following the original evaluation procedures as closely as possible to ensure consistent and comparable evaluation.

A.5 Controller Architecture and State Construction

The state-conditioned controller $\pi_\psi(\mathbf{a}_t \mid \mathbf{s}_t)$ takes $\mathbf{s}_t = (\mathbf{e}_t, \mathbf{h}_t)$ as input and jointly produces continuous editing controls and a discrete module-routing decision. Here, $\mathbf{e}_t$ represents the current editing request, while $\mathbf{h}_t$ summarizes the editing trajectory completed before the current update.

Current-edit representation. We construct $\mathbf{e}_t$ from the input activations of the first candidate editable module. Under the default all-token setting, activations at all non-padding token positions are mean pooled:

$$\mathbf{e}_t = \frac{1}{|\mathcal{T}_t|} \sum_{p \in \mathcal{T}_t} \mathbf{k}_{t,p}, \tag{19}$$

where $\mathcal{T}_t$ denotes the set of non-padding token positions and $\mathbf{k}_{t,p}$ is the module-input key at position p. The resulting feature provides a fixed-dimensional representation of the current editing request and is projected to a 256-dimensional controller space through a linear layer followed by ReLU.

History state. The history feature $\mathbf{h}_t \in \mathbb{R}^8$ summarizes recent editing behavior using a sliding window of the previous $W = 100$ completed edits:

$$\mathbf{h}_t = \Big[p_t,\, \overline{\mathrm{ES}}_t,\, \overline{\mathrm{GS}}_t,\, \overline{\mathrm{LS}}_t,\, U_t,\, \Delta\mathrm{ES}_t,\, \Delta\mathrm{LS}_t,\, c_t\Big]. \tag{20}$$

The state captures editing progress, recent efficacy, generalization and locality outcomes, accumulated parameter-update magnitude, short-term trends in efficacy and locality, and history coverage. The windowed statistics $\overline{\mathrm{ES}}_t$, $\overline{\mathrm{GS}}_t$, and $\overline{\mathrm{LS}}_t$ are arithmetic means over recent editing outcomes, while $\Delta\mathrm{ES}_t$ and $\Delta\mathrm{LS}_t$ measure the difference between the most recent 10 outcomes and the preceding 10 outcomes. The outcome statistics are obtained from the corresponding edit, equivalent, and unrelated feedback available in the editing protocol.

The progress and history-coverage features are defined as $p_t = \min((t-1)/1000, 1)$ and $c_t = \min((t-1)/W, 1)$, respectively. Importantly, $\mathbf{h}_t$ is constructed before applying the t-th edit and therefore depends only on previously completed edits, ensuring that the controller does not observe the outcome of the current action.

Accumulated parameter-update magnitude. To characterize the cumulative degree of parameter intervention, we maintain

$$U_t = \frac{1}{10} \min\left(\frac{\sum_{\tau<t} \sum_k \left\| \Delta \mathbf{W}_\tau^{(k)} \right\|_F}{\|\mathbf{W}_{\mathrm{base}}\|_F + \epsilon}, 10 \right), \tag{21}$$

where $\Delta \mathbf{W}_\tau^{(k)}$ denotes the update applied to editable module k at step τ. This statistic measures the accumulated magnitude of parameter updates along the editing trajectory rather than the direct displacement from the initial model, and provides the controller with a compact signal of accumulated intervention.

Controller architecture. The controller is implemented as a lightweight MLP. The projected edit representation is concatenated with $\mathbf{h}_t$ and passed through two fully connected layers with hidden dimension 256 and ReLU activations. The resulting representation is shared by two output branches. The continuous branch predicts the state-conditioned means of the three editing controls $(\alpha_t, \beta_t, \rho_t)$, which regulate update strength, regularization, and preservation, respectively. The routing branch predicts logits $\boldsymbol{\xi}_\psi(\mathbf{s}_t) \in \mathbb{R}^L$ over the L candidate editable modules.

Action factorization. The controller jointly produces three continuous controls and one discrete routing decision, giving $\mathbf{a}_t = (\alpha_t, \beta_t, \rho_t, \mathbf{m}_t)$. We represent the policy using a continuous control branch and a module-routing branch:

$$\pi_\psi(\mathbf{a}_t \mid \mathbf{s}_t) = \pi_\psi^{\mathrm{cont}}\left(\alpha_t, \beta_t, \rho_t \mid \mathbf{s}_t\right) \pi_\psi^{\mathrm{route}}\left(\mathbf{m}_t \mid \mathbf{s}_t\right). \tag{22}$$

This factorization allows the controller to jointly adapt how strongly an edit is applied and where the corresponding parameter update is performed.

Continuous controls. During training, the continuous branch first samples a latent action from a diagonal Gaussian:

$$\widetilde{\mathbf{a}}_t \sim \mathcal{N}\left(\boldsymbol{\mu}_\psi(\mathbf{s}_t), \boldsymbol{\Sigma}_\psi\right), \tag{23}$$

where $\boldsymbol{\mu}_\psi(\mathbf{s}_t)$ is state-conditioned and $\boldsymbol{\Sigma}_\psi$ is a globally learned diagonal covariance shared across states. The sampled latent variables are subsequently mapped through bounded nonlinear transformations to the valid ranges of α_t, β_t, and ρ_t. At inference time, stochastic sampling is disabled and the predicted Gaussian mean is used deterministically.

Module routing. The routing branch operates over candidate editable MLP projections rather than entire Transformer layers. At each editing step, exactly $K = 6$ modules are selected. During training, we add i.i.d. Gumbel noise to the routing logits and perform Gumbel-Top-K selection:

$$\mathbf{m}_t = \mathrm{TopK}\left(\boldsymbol{\xi}_\psi(\mathbf{s}_t) + \mathbf{g}_t, K\right), \tag{24}$$

where $\mathbf{g}_t$ denotes the sampled Gumbel noise. At inference time, routing is deterministic and directly selects the Top-K entries of $\boldsymbol{\xi}_\psi(\mathbf{s}_t)$.

For policy optimization, we define the routing score as the sum of the selected log-softmax scores,

$$\ell_{\psi,t}^{\mathrm{route}} = \sum_{i:m_{t,i}=1} \log \mathrm{softmax}\left(\boldsymbol{\xi}_\psi(\mathbf{s}_t)\right)_i . \tag{25}$$

This routing score is combined with the continuous-action log-likelihood in the trajectory-level policy objective described in Appendix B.5. Additional optimization and hyperparameter settings are provided in Appendix A.6.

A.6 Protocol and Implementation Details

A.6.1 Sequential-Editing Protocol

We follow the $8{,}000\times1$ lifelong-editing protocol of HiEdit (Wang et al., 2026). A total of $N = 8{,}000$ knowledge instances are committed sequentially, one at each editing step, with the post-edit model $\boldsymbol{\theta}_t$ serving as the starting point for the subsequent edit. Thus, each editing decision is made on a model that has already incorporated all preceding updates and their accumulated effects. No replay or test-time fine-tuning is used during evaluation. After completing the full edit stream, the final model $f_{\boldsymbol{\theta}_N}$ is evaluated using the dataset-specific metrics described in Appendix A.2. This protocol directly evaluates whether an editing method can sustain editing effectiveness and knowledge preservation as interference progressively accumulates over thousands of sequential updates.

A.6.2 Training and Implementation Details

SeqConEdit is trained in two stages. In ***Stage A***, the rewriting backbone g_ϕ is trained for one epoch to predict value-side residuals for editable layers using the supervised editing objective following RLEdit (Li et al., 2025). We use a small preservation strength $\rho_{\mathrm{sft}} = 0.5$ during this stage to warm-start preservation-aware rewriting. In ***Stage B***, g_ϕ is frozen and the state-conditioned controller π_ψ is optimized with the constraint-aware policy objective described in Sec. 4.4. For each sampled edit sequence, we generate $G = 4$ policy rollouts of length $H = 200$ and compute the group-relative advantages from their trajectory returns. Here, T denotes the policy-optimization rollout horizon rather than the full lifelong editing horizon $N = 8{,}000$. Using truncated rollouts avoids unrolling the entire edit stream for every policy update while still exposing the controller to sequentially accumulated editing effects.

The adaptive preservation multiplier is projected to $[0, \mu_{\max}]$ with $\mu_{\max} = 10$ and updated using step size $\eta_\mu = 0.5$ with preservation target $\tau = 0.5$. Distributional Locality Regularization is applied with weight $\kappa = 3.0$. The controller predicts continuous editing controls $(\alpha_t, \beta_t, \rho_t) \in (0, 2]$ together with a sparse layer route $\mathbf{m}_t$, where α_t, β_t, and ρ_t regulate edit strength, regularization, and preservation pressure, respectively. The routing branch selects $K = 6$ editable layers, using Gumbel-Top-K during training and deterministic Top-K at inference. The Preservation Trace Buffer is maintained independently for each editable layer with capacity $C = 500$.

We use AdamW for both training stages, with learning rates $1\mathrm{e}{-5}$ for the rewriting backbone g_ϕ and $1\mathrm{e}{-6}$ for the controller π_ψ. Entropy regularization is applied to the policy objective with coefficient

$\beta_H = 0.02$. Candidate editable modules are selected from middle-layer MLP projections following the HiEdit protocol. Unless otherwise specified, the same training and editing configuration is used across datasets and model backbones. All experiments are conducted on a single NVIDIA H100-80GB GPU. Additional sensitivity analyses of the group size, rollout horizon, and PTB capacity are provided in Appendix C.7.

B Derivations and Optimization Details

This appendix provides the derivations and optimization details corresponding to Sec. 3 and Sec. 4. We first derive the constrained sequential formulation and its adaptive preservation objective, then derive the preservation-aware key–value update used by PKC, describe the Preservation Trace Buffer and sparse routing, and finally detail the policy optimization objective.

B.1 Derivation of the Constrained Sequential Formulation

We provide a more detailed derivation of the constrained sequential formulation introduced in Sec. 3.2.

Sequential dependence. Let $f_{\boldsymbol{\theta}_0}$ denote the pretrained model and $\mathcal{E}_{1:N} = \{e_t\}_{t=1}^{N}$ a stream of editing requests. At step t, the policy selects an editing action a_t and the editor updates the current model according to

$$\boldsymbol{\theta}_t = \Phi(\boldsymbol{\theta}_{t-1}; e_t, a_t). \tag{26}$$

Recursively expanding Eq. 26 gives

$$\boldsymbol{\theta}_t = \Phi_t\left(\boldsymbol{\theta}_0, e_{1:t}, a_{1:t}\right), \tag{27}$$

where Φ_t denotes the composition of the first t editing transitions. Therefore, even when the current request e_t is fixed, the model on which it is executed depends on all preceding editing decisions. The utility and preservation outcomes at step t can consequently be written as

$$u_t = u(\boldsymbol{\theta}_t, e_t), \qquad p_t = p(\boldsymbol{\theta}_t), \tag{28}$$

and both depend implicitly on the previous action sequence $a_{1:t}$. Optimizing each edit independently therefore ignores how the current action changes the starting point of future edits.

Trajectory-level constrained objective. Following the main text, we define

$$u_t = \mathrm{Eff}_t + \mathrm{Gen}_t, \qquad p_t = \mathrm{Spe}_t, \tag{29}$$

where Specificity serves as an observable surrogate for preserved-knowledge stability. For a deterministic editing trajectory, the desired objective is

$$\max_{a_{1:N}} \frac{1}{N}\sum_{t=1}^{N} u_t, \qquad \text{s.t.} \quad \frac{1}{N}\sum_{t=1}^{N} p_t \geq \tau. \tag{30}$$

Because actions are sampled from the editing policy and the resulting trajectories may be stochastic, we optimize the expected trajectory objective:

$$\max_{\pi_\psi} J(\pi_\psi) = \mathbb{E}_{\pi_\psi}\left[\frac{1}{N}\sum_{t=1}^{N} u_t\right], \qquad \text{s.t.} \quad P(\pi_\psi) = \mathbb{E}_{\pi_\psi}\left[\frac{1}{N}\sum_{t=1}^{N} p_t\right] \geq \tau. \tag{31}$$

This is the trajectory-level form used in Eq. 2.

Sequential decision interpretation. A fully specified editing state may be written as

$$\widetilde{s}_t = (\boldsymbol{\theta}_{t-1}, e_t, \mathcal{H}_t), \tag{32}$$

where $\mathcal{H}_t$ denotes the information accumulated before step t. Given $\widetilde{s}_t$ and a_t, the editing operator determines $\boldsymbol{\theta}_t$, while the next request together with the updated history determines $\widetilde{s}_{t+1}$. Accordingly, the augmented state satisfies the following Markov property:

$$P(\widetilde{s}_{t+1} \mid \widetilde{s}_{1:t}, a_{1:t}) = P(\widetilde{s}_{t+1} \mid \widetilde{s}_t, a_t). \tag{33}$$

This provides a finite-horizon constrained sequential-decision interpretation of lifelong editing. In practice, SeqConEdit does not expose the full parameter state to the controller. Instead, the policy uses the compact state $\mathbf{s}_t = (\mathbf{e}_t, \mathbf{h}_t)$, where $\mathbf{h}_t$ summarizes previous edits together with recent editing outcomes and the accumulated model drift.

Lagrangian form of the preservation constraint. The preservation requirement in Eq. 31 can be equivalently expressed through the following constraint function:

$$g(\pi_\psi) = P(\pi_\psi) - \tau \geq 0. \tag{34}$$

For this maximization problem with a lower-bound preservation constraint, we introduce a nonnegative Lagrange multiplier $\mu \geq 0$ and obtain the corresponding Lagrangian:

$$\mathcal{J}(\pi_\psi, \mu) = J(\pi_\psi) + \mu \left(P(\pi_\psi) - \tau\right). \tag{35}$$

Substituting the definitions of J and P yields the following expanded trajectory-level objective:

$$\mathcal{J}(\pi_\psi, \mu) = \mathbb{E}_{\pi_\psi}\left[\frac{1}{N}\sum_{t=1}^{N}\left(u_t + \mu p_t\right)\right] - \mu\tau. \tag{36}$$

For a fixed μ, the final term $-\mu\tau$ does not depend on the sampled editing action and therefore does not affect the relative policy preference among trajectories. The policy-dependent contribution can thus be represented by the following per-step signal:

$$u_t + \mu p_t = \mathrm{Eff}_t + \mathrm{Gen}_t + \mu \mathrm{Spe}_t. \tag{37}$$

Adaptive multiplier update. Since Eq. 35 is maximized with respect to the policy and minimized with respect to the nonnegative multiplier, we update μ using the following projected gradient step:

$$\mu \leftarrow \Pi_{[0,\mu_{\max}]}\left[\mu - \eta_\mu \left(P(\pi_\psi) - \tau\right)\right]. \tag{38}$$

Approximating $P(\pi_\psi)$ with the empirical preservation level $\overline{\mathrm{Spe}}$ of the current optimization batch yields the practical update:

$$\mu \leftarrow \Pi_{[0,\mu_{\max}]}\left[\mu + \eta_\mu \left(\tau - \overline{\mathrm{Spe}}\right)\right]. \tag{39}$$

Hence, μ increases when the observed preservation level falls below the target τ and decreases when the preservation requirement is sufficiently satisfied.

Base Specificity shaping. In addition to the adaptive multiplier, SeqConEdit retains a fixed base Specificity weight η_{spe} so that preservation remains present in the reward even when the adaptive multiplier is small. Combining the base term with Eq. 37 yields

$$r_t^{\mathrm{lag}} = \mathrm{Eff}_t + \mathrm{Gen}_t + (\eta_{\mathrm{spe}} + \mu)\mathrm{Spe}_t, \tag{40}$$

which is the adaptive preservation reward used in Sec. 4.4. The role of μ is therefore to adjust the additional preservation pressure according to current constraint satisfaction, while η_{spe} provides a fixed preservation prior.

Finite rollout estimator. The full lifelong stream contains N edits, whereas policy training uses length-H rollouts. For a group of G rollouts, we estimate the preservation level by

$$\overline{\mathrm{Spe}} = \frac{1}{GH}\sum_{g=1}^{G}\sum_{t=1}^{H}\mathrm{Spe}_t^{(g)}. \tag{41}$$

Likewise, the trajectory reward used for policy optimization is accumulated over the sampled horizon H. This provides a tractable estimator of the long-horizon objective while preserving the same editing–preservation structure as Eq. 31.

B.2 Constrained Formulation and Notation

Consider the ℓ-th editable linear module with weight matrix $\mathbf{W}_\ell \in \mathbb{R}^{d_v \times d_k}$. Following the key–value memory view of locate-and-edit methods, an input key $\mathbf{k} \in \mathbb{R}^{d_k}$ is mapped to a value vector through $\mathbf{W}_\ell \mathbf{k}$. At editing step t, let $\mathbf{K}_{\ell,t} \in \mathbb{R}^{d_k \times n}$ denote the edit-key matrix, $\mathbf{R}_{\ell,t} \in \mathbb{R}^{d_v \times n}$ the target value-side residual, and $\Delta\mathbf{W}_{\ell,t} \in \mathbb{R}^{d_v \times d_k}$ the parameter update. The desired edit response is

$$\Delta\mathbf{W}_{\ell,t}\mathbf{K}_{\ell,t} \approx \alpha_t \mathbf{R}_{\ell,t}. \tag{42}$$

For clarity, we omit ℓ and t in intermediate derivations and write $\mathbf{K}$, $\mathbf{R}$, $\mathbf{P}$, and $\Delta\mathbf{W}$.

B.3 PRESERVATION-AWARE KEY–VALUE UPDATE

Regularized key–value update. Without preservation keys, the controlled key–value update solves

$$\min_{\Delta \mathbf{W}} \|\Delta \mathbf{W}\mathbf{K} - \alpha \mathbf{R}\|_F^2 + \beta\lambda \|\Delta \mathbf{W}\|_F^2 . \tag{43}$$

Define $\gamma = \beta\lambda$ to avoid overloading the adaptive preservation multiplier μ. Using $\|\mathbf{A}\|_F^2 = \mathrm{Tr}(\mathbf{A}\mathbf{A}^\top)$, the gradient is

$$\begin{aligned} \frac{\partial}{\partial \Delta \mathbf{W}} \left(\|\Delta \mathbf{W}\mathbf{K} - \alpha \mathbf{R}\|_F^2 + \gamma \|\Delta \mathbf{W}\|_F^2 \right) &= 2(\Delta \mathbf{W}\mathbf{K} - \alpha \mathbf{R})\mathbf{K}^\top + 2\gamma \Delta \mathbf{W} \\ &= 2\Delta \mathbf{W}\mathbf{K}\mathbf{K}^\top - 2\alpha \mathbf{R}\mathbf{K}^\top + 2\gamma \Delta \mathbf{W}. \end{aligned} \tag{44}$$

Setting Eq. 44 to zero gives

$$\begin{aligned} \Delta \mathbf{W}\mathbf{K}\mathbf{K}^\top + \gamma \Delta \mathbf{W} &= \alpha \mathbf{R}\mathbf{K}^\top, \\ \Delta \mathbf{W} &= \alpha \mathbf{R}\mathbf{K}^\top (\mathbf{K}\mathbf{K}^\top + \gamma \mathbf{I})^{-1}. \end{aligned} \tag{45}$$

Substituting $\gamma = \beta\lambda$ and restoring the layer and editing-step indices gives the corresponding full-index normal equation used in our formulation:

$$\Delta \mathbf{W}_{\ell,t} \left(\mathbf{K}_{\ell,t}\mathbf{K}_{\ell,t}^\top + \beta_t \lambda_\ell \mathbf{I} \right) = \alpha_t \mathbf{R}_{\ell,t}\mathbf{K}_{\ell,t}^\top, \tag{46}$$

and the full regularized key–value update is

$$\Delta \mathbf{W}_{\ell,t} = \alpha_t \mathbf{R}_{\ell,t}\mathbf{K}_{\ell,t}^\top \left(\mathbf{K}_{\ell,t}\mathbf{K}_{\ell,t}^\top + \beta_t \lambda_\ell \mathbf{I} \right)^{-1} . \tag{47}$$

Preservation requirement. Let $\mathbf{P} \in \mathbb{R}^{d_k \times q}$ contain key directions whose responses should remain stable. Ideally, editing should preserve the original mapping on these directions:

$$(\mathbf{W} + \Delta \mathbf{W})\mathbf{P} \approx \mathbf{W}\mathbf{P}, \qquad \Longleftrightarrow \qquad \Delta \mathbf{W}\mathbf{P} \approx \mathbf{0}. \tag{48}$$

Instead of enforcing a fixed hard projection, PKC penalizes the residual response on preserved directions with a state-dependent strength ρ.

PKC objective. The resulting objective is

$$\min_{\Delta \mathbf{W}} \|\Delta \mathbf{W}\mathbf{K} - \alpha \mathbf{R}\|_F^2 + \rho \|\Delta \mathbf{W}\mathbf{P}\|_F^2 + \gamma \|\Delta \mathbf{W}\|_F^2 . \tag{49}$$

Its gradient is

$$\frac{\partial J_{\mathrm{PKC}}}{\partial \Delta \mathbf{W}} = 2(\Delta \mathbf{W}\mathbf{K} - \alpha \mathbf{R})\mathbf{K}^\top + 2\rho \Delta \mathbf{W}\mathbf{P}\mathbf{P}^\top + 2\gamma \Delta \mathbf{W}. \tag{50}$$

Setting the gradient to zero yields

$$\begin{aligned} \Delta \mathbf{W}\mathbf{K}\mathbf{K}^\top + \rho \Delta \mathbf{W}\mathbf{P}\mathbf{P}^\top + \gamma \Delta \mathbf{W} &= \alpha \mathbf{R}\mathbf{K}^\top, \\ \Delta \mathbf{W} &= \alpha \mathbf{R}\mathbf{K}^\top \left(\mathbf{K}\mathbf{K}^\top + \rho \mathbf{P}\mathbf{P}^\top + \gamma \mathbf{I} \right)^{-1} . \end{aligned} \tag{51}$$

Restoring the full indices and substituting $\gamma = \beta_t \lambda_\ell$ gives

$$\Delta \mathbf{W}_{\ell,t} = \alpha_t \mathbf{R}_{\ell,t}\mathbf{K}_{\ell,t}^\top \left(\mathbf{K}_{\ell,t}\mathbf{K}_{\ell,t}^\top + \rho_t \mathbf{P}_{\ell,t}\mathbf{P}_{\ell,t}^\top + \beta_t \lambda_\ell \mathbf{I} \right)^{-1} , \tag{52}$$

which matches Eq. 6.

Relation to null-space preservation. If $\mathbf{P}$ spans directions associated with knowledge that should remain stable, the ideal hard-preservation condition is $\Delta \mathbf{W}\mathbf{P} = \mathbf{0}$. PKC replaces this hard condition with the soft penalty $\rho \|\Delta \mathbf{W}\mathbf{P}\|_F^2$. Increasing ρ therefore progressively suppresses update responses on the preserved subspace while still allowing the policy to trade preservation against edit realization. In the large-ρ limit, any bounded optimum is forced toward $\Delta \mathbf{W}\mathbf{P} \to \mathbf{0}$.

Invertibility. The closed-form expression in Eq. 51 assumes that

$$\mathbf{A} = \mathbf{K}\mathbf{K}^\top + \rho\mathbf{P}\mathbf{P}^\top + \gamma\mathbf{I} \tag{53}$$

is nonsingular. This is guaranteed whenever $\gamma = \beta\lambda > 0$, because for any nonzero $\mathbf{x}$,

$$\mathbf{x}^\top\mathbf{A}\mathbf{x} = \|\mathbf{K}^\top\mathbf{x}\|_2^2 + \rho\|\mathbf{P}^\top\mathbf{x}\|_2^2 + \gamma\|\mathbf{x}\|_2^2 > 0. \tag{54}$$

Thus, under positive ridge regularization, $\mathbf{A}$ is positive definite and the solution is unique.

B.4 Trace Buffer and Sparse Routing

Preservation Trace Buffer. PKC only protects directions represented in the current preservation matrix. Let $\mathbf{K}_{\ell,t}^{\text{unr}}$ denote unrelated keys observed at step t, and let $\mathbf{B}_{\ell,t}$ denote historical preservation keys retained by the buffer. We construct the preservation matrix by combining the current and retained historical keys:

$$\mathbf{P}_{\ell,t} = \left[\mathbf{K}_{\ell,t}^{\text{unr}}, \mathbf{B}_{\ell,t}\right]. \tag{55}$$

After each editing step, the newly observed unrelated keys are incorporated into the buffer through the following FIFO update:

$$\mathbf{B}_{\ell,t+1} = \text{FIFO}_C\left(\mathbf{B}_{\ell,t} \oplus \mathbf{K}_{\ell,t}^{\text{unr}}\right), \tag{56}$$

where $\oplus$ denotes concatenation and C is the buffer capacity. Since Eq. 55 horizontally concatenates the current and historical preservation keys, its Gram matrix decomposes as

$$\mathbf{P}_{\ell,t}\mathbf{P}_{\ell,t}^\top = \mathbf{K}_{\ell,t}^{\text{unr}}\mathbf{K}_{\ell,t}^{\text{unr}\top} + \mathbf{B}_{\ell,t}\mathbf{B}_{\ell,t}^\top. \tag{57}$$

Thus, the PKC term jointly constrains both currently observed unrelated directions and historical preservation directions retained from earlier editing steps.

Sparse layer routing. The controller selects $\mathbf{m}_t \in \{0,1\}^L$ with $\|\mathbf{m}_t\|_0 = K$, thereby activating only K editable layers at each step. The resulting update to layer ℓ is

$$\mathbf{W}_{\ell,t} = \mathbf{W}_{\ell,t-1} + m_{t,\ell}\Delta\mathbf{W}_{\ell,t}. \tag{58}$$

Aggregating these routed layer-wise updates gives the corresponding full-model parameter update:

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \sum_{\ell=1}^{L} m_{t,\ell}\mathcal{I}_\ell\left(\Delta\mathbf{W}_{\ell,t}\right), \tag{59}$$

where $\mathcal{I}_\ell(\cdot)$ embeds each local matrix update into its corresponding location in the full model parameter space, ensuring that only the selected layer parameters are modified at each editing step.

B.5 Constraint-Aware Policy Optimization

Adaptive preservation reward. From Eq. 40, the preservation-aware step reward before DLR is

$$r_t^{\text{lag}} = \text{Eff}_t + \text{Gen}_t + (\eta_{\text{spe}} + \mu)\text{Spe}_t. \tag{60}$$

The multiplier is updated using Eq. 39. This directly implements the trajectory-level preservation requirement in Eq. 2.

Distributional locality regularization. Specificity is an outcome-level preservation signal and may remain unchanged even when the predictive distribution has drifted. Let $\mathcal{V}_t^{\text{unr}}$ denote protected token positions associated with unrelated knowledge and $\mathcal{V}_t^{\text{edit}}$ those associated with previously committed edits, with $\mathcal{V}_t = \mathcal{V}_t^{\text{unr}} \cup \mathcal{V}_t^{\text{edit}}$. For each protected position (z, j), the reference distribution is

$$q_{z,j}^{\text{ref}} = \begin{cases} p_{\boldsymbol{\theta}_0}(\cdot \mid z_{<j}), & z \in \mathcal{V}_t^{\text{unr}}, \\ p_{\boldsymbol{\theta}_{t_z}}(\cdot \mid z_{<j}), & z \in \mathcal{V}_t^{\text{edit}}, \end{cases} \tag{61}$$

where t_z is the step immediately after the corresponding edited knowledge was committed. The Distributional Locality Regularization term is

$$\text{DLR}_t = \frac{1}{|\mathcal{V}_t|}\sum_{(z,j)\in\mathcal{V}_t} \text{KL}\left(p_{\boldsymbol{\theta}_t}(\cdot \mid z_{<j}) \,\|\, q_{z,j}^{\text{ref}}\right). \tag{62}$$

The final step reward is therefore

$$r_t = \underbrace{\text{Eff}_t + \text{Gen}_t}_{\text{editing utility}} + \underbrace{(\eta_{\text{spe}} + \mu)\text{Spe}_t}_{\text{adaptive preservation}} - \underbrace{\kappa\text{DLR}_t}_{\text{distributional locality}}. \tag{63}$$

Trajectory return and group-relative advantage. For the g-th rollout of a common length-H edit sequence, the return is

$$R^{(g)} = \sum_{t=1}^{H} r_t^{(g)}. \tag{64}$$

Sampling G rollouts for the same edit sequence controls for sequence difficulty, because the edit content is shared while the policy actions differ. We compute

$$\overline{R} = \frac{1}{G}\sum_{g=1}^{G} R^{(g)}, \qquad \sigma_R = \sqrt{\frac{1}{G}\sum_{g=1}^{G}\left(R^{(g)} - \overline{R}\right)^2}, \tag{65}$$

and define the normalized group-relative advantage as

$$\hat{A}^{(g)} = \frac{R^{(g)} - \overline{R}}{\sigma_R + \epsilon}. \tag{66}$$

Trajectory-Level Policy Optimization. For each rollout g, we denote the corresponding length-H state–action trajectory by

$$\zeta^{(g)} = \left\{(\mathbf{s}_t^{(g)}, \mathbf{a}_t^{(g)})\right\}_{t=1}^{H}. \tag{67}$$

The continuous branch contributes the Gaussian log-likelihood of the sampled latent controls, while the routing branch contributes a surrogate score defined by the selected module logits:

$$\ell_{\psi,t}^{\text{route}} = \sum_{i:m_{t,i}=1} \log \operatorname{softmax}\left(\boldsymbol{\xi}_\psi(\mathbf{s}_t)\right)_i. \tag{68}$$

For notational simplicity, we combine the continuous-action log-likelihood and the routing score into the step-wise policy score

$$\log \pi_\psi\left(\mathbf{a}_t \mid \mathbf{s}_t\right) := \log \pi_\psi^{\text{cont}}\left(\widetilde{\mathbf{a}}_t \mid \mathbf{s}_t\right) + \ell_{\psi,t}^{\text{route}}. \tag{69}$$

The trajectory-level policy score then factorizes over the H action decisions:

$$\log \pi_\psi(\zeta^{(g)}) = \sum_{t=1}^{H} \log \pi_\psi\left(\mathbf{a}_t^{(g)} \mid \mathbf{s}_t^{(g)}\right). \tag{70}$$

Using the group-relative advantage $\hat{A}^{(g)}$, we optimize the controller with

$$\mathcal{L}_{\text{RL}}(\psi) = -\frac{1}{G}\sum_{g=1}^{G} \hat{A}^{(g)} \log \pi_\psi(\zeta^{(g)}) - \beta_H \mathcal{H}[\pi_\psi]. \tag{71}$$

Adaptive preservation and DLR shape the trajectory return, while group-relative advantage estimation determines the relative credit assigned to each sampled trajectory. In our implementation, the entropy regularization is applied to the continuous Gaussian branch.

B.6 Summary of the Preservation-Aware Update

At each editing step, the controller selects $\mathbf{a}_t = (\alpha_t, \beta_t, \rho_t, \mathbf{m}_t)$, after which PKC computes the layer-wise update by solving

$$\min_{\Delta\mathbf{W}} \|\Delta\mathbf{W}\mathbf{K} - \alpha\mathbf{R}\|_F^2 + \rho\,\|\Delta\mathbf{W}\mathbf{P}\|_F^2 + \beta\lambda\,\|\Delta\mathbf{W}\|_F^2, \tag{72}$$

which yields the following closed-form preservation-aware update:

$$\Delta\mathbf{W} = \alpha\mathbf{R}\mathbf{K}^\top\left(\mathbf{K}\mathbf{K}^\top + \rho\mathbf{P}\mathbf{P}^\top + \beta\lambda\mathbf{I}\right)^{-1}. \tag{73}$$

PTB constructs $\mathbf{P}$ from current and historical preservation keys, while the sparse route $\mathbf{m}_t$ determines which editable layers receive the update. After evaluating the edited model, the reward in Eq. 63 is accumulated into the trajectory return and used to optimize the controller through Eq. 71, completing the closed optimization loop of SeqConEdit.

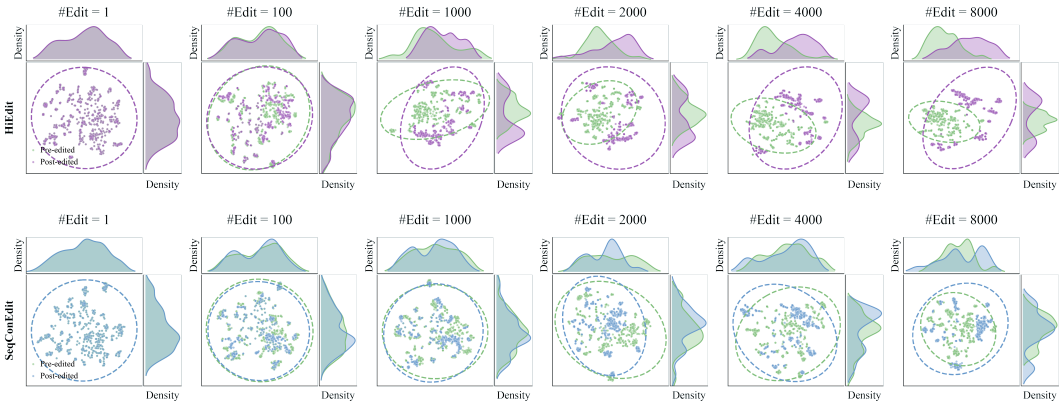


Figure 6: Representation drift over sequential edits. Pre- and post-edit representation distributions of 500 fixed probes on Gemma-2-9B as sequential CounterFact edits accumulate.

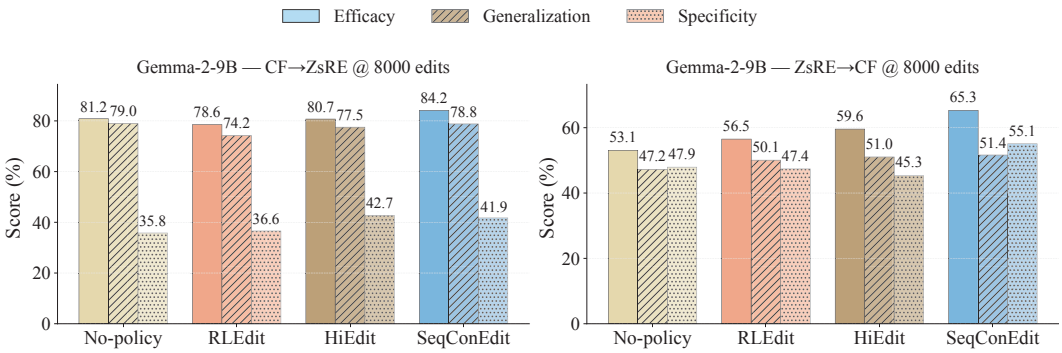


Figure 7: Cross-dataset editing robustness. Efficacy, generalization, and specificity of Gemma-2-9B after 8,000 sequential edits under CF→ZsRE and ZsRE→CF transfer. The arrow indicates the edit and evaluation datasets, respectively. Refer to Sec. 5.4 for details. Best viewed in color.

C Additional Experimental Analyses

C.1 Representation Drift over Sequential Edits

While the main text compares representation distributions after the full editing stream, we additionally examine how representation drift develops as sequential edits accumulate. Figure 6 tracks the pre/post representation distributions at intermediate edit counts. The distributions remain largely aligned early in the edit stream, but gradually diverge as more edits are applied. HiEdit exhibits a pronounced loss of overlap after approximately 1,000 edits, with the discrepancy becoming increasingly evident toward 4,000–8,000 edits. In contrast, SeqConEdit preserves substantially greater distributional overlap throughout the trajectory, indicating a markedly slower accumulation of representation drift. This trajectory-level analysis confirms that the preservation advantage reported in the main text is sustained throughout the edit stream rather than emerging only at the final evaluation point.

C.2 Cross-Dataset Editing Robustness

Transfer Protocol. We evaluate whether the editing policy learned by SeqConEdit generalizes across different editing distributions. Given a source dataset $\mathcal{D}_s$ and a target dataset $\mathcal{D}_t$, we first train the controller exclusively on $\mathcal{D}_s$ using the standard policy-optimization procedure. After training, the controller parameters are frozen and receive no further updates from target-domain data or feedback. We then reinitialize the base LLM to its original pretrained parameters and apply the frozen controller to an $8{,}000\times1$ sequential editing stream constructed from $\mathcal{D}_t$. Both editing and evaluation are performed on the target dataset. The editing history and PTB are initialized from scratch at the beginning of each target-domain stream and subsequently updated only with target-domain

Table 3: **Full component ablation of SeqConEdit.** Results on Gemma-2-9B under the $8{,}000\times1$ sequential editing protocol. We report Efficacy (Eff.), Generalization (Gen.), Specificity (Spe.), Edited Retention (Ret.), and their arithmetic mean (Avg.) on ZsRE and CounterFact.

Variant	ZsRE					CounterFact				
	Eff.(↑)	*Gen.*(↑)	*Spe.*(↑)	*Ret.*(↑)	*Avg.*(↑)	*Eff.*(↑)	*Gen.*(↑)	*Spe.*(↑)	*Ret.*(↑)	*Avg.*(↑)
SeqConEdit	**85.41**	**80.73**	34.79	**83.24**	**71.04**	**73.45**	**53.67**	57.06	**78.00**	**65.55**
w/o PKC	82.82	77.40	37.73	78.40	69.09$_{-1.95}$	61.62	51.29	52.32	62.80	57.01$_{-8.54}$
w/o PTB	80.21	74.92	**39.47**	79.11	68.43$_{-2.61}$	62.60	49.02	**62.18**	57.00	57.45$_{-7.85}$
w/o CAPO	83.15	76.59	38.06	77.24	68.76$_{-2.28}$	64.31	50.73	61.55	61.84	59.61$_{-5.94}$
– DLR	85.06	79.43	35.55	80.76	70.20$_{-0.84}$	69.30	50.36	57.51	74.60	62.94$_{-2.61}$

information. Therefore, only the learned editing policy is transferred across datasets, rather than a model already edited on the source domain.

Experimental Settings. We consider both transfer directions, ZsRE→CounterFact and CounterFact→ZsRE, together with the corresponding in-domain settings for reference. For each run, the target dataset is independently shuffled and the first 8,000 instances are used to construct the sequential editing stream. All compared methods are evaluated on the same target-domain sequence within each run, and results are averaged over three independent runs. Unless otherwise specified, all other editing and evaluation configurations follow those used in the main experiments.

Results. As shown in Figure 7, SeqConEdit maintains strong editing performance in both transfer directions. Under CF→ZsRE, it achieves the highest Efficacy (84.2%), while maintaining Generalization (78.8%) and Specificity (41.9%) close to the strongest competing results. Under ZsRE→CF, SeqConEdit achieves the best performance across all three metrics, reaching 65.3% Efficacy, 51.4% Generalization, and 55.1% Specificity. Since no target-domain policy optimization is performed, these results provide direct evidence that the learned editing strategy is not tightly coupled to the statistics of a single benchmark and remains effective under shifts in edit content and query formulation.

C.3 Full Component Ablation Results

The main-text component ablation in Table 3 reports the ZsRE results for compact presentation. Removing PKC or PTB consistently degrades overall performance on both datasets. On ZsRE, the corresponding Avg. drops are 1.95% and 2.61%, while the reductions become substantially larger on CounterFact at 8.54% and 7.85%. The accompanying losses in Ret. are particularly pronounced, indicating that both step-wise preservation constraints and persistent historical preservation are important for mitigating interference from accumulated edits. The larger degradation on CounterFact further suggests that these mechanisms become increasingly important when preservation involves semantically related knowledge.

Removing CAPO also leads to consistent degradation, reducing Avg. by 2.28% on ZsRE and 5.94% on CounterFact, which supports the need to adaptively coordinate knowledge incorporation and preservation throughout the editing process. Removing DLR alone produces smaller but consistent drops of 0.84% and 2.61%, showing the additional benefit of distribution-level locality supervision. Notably, several ablations improve Spe. while substantially degrading Eff., Gen., or Ret., indicating that stronger locality alone does not necessarily yield a better overall editing–preservation trade-off.

C.4 Policy State and Action Ablations

Table 4 further examines the state information and adaptive action branches of the SeqConEdit policy on Gemma-2-9B with ZsRE. Unlike the component ablation averaged over three independent runs, this controlled analysis uses the same single trained checkpoint for all variants. The rewriting backbone, PKC, PTB, editable modules, and evaluation protocol are kept unchanged, while only the policy state information or one adaptive action component is modified.

Table 4: **State-information and action-branch ablations.** Results on Gemma-2-9B with ZsRE under the $8{,}000\times1$ sequential editing protocol. All variants are evaluated from the same single trained checkpoint. Refer to Sec. 5.3 for details.

Variant	*Eff.*(↑)	*Gen.*(↑)	*Spe.*(↑)	*Ret.*(↑)	*Avg.*(↑)
SeqConEdit	**86.89**	**81.80**	34.76	**82.96**	**71.60**
w/o history	83.10	77.81	36.29	80.40	$69.40_{-2.20}$
w/o update	82.95	78.23	**36.76**	80.73	$69.67_{-1.93}$
fixed routing	83.68	78.25	35.13	81.01	$69.52_{-2.08}$
fixed controls	82.43	78.60	34.06	82.25	$69.34_{-2.27}$

The full policy conditions on $\mathbf{s}_t = (\mathbf{e}_t, \mathbf{h}_t)$ and outputs $\mathbf{a}_t = (\alpha_t, \beta_t, \rho_t, \mathbf{m}_t)$, where $\mathbf{e}_t$ represents the current edit, $\mathbf{h}_t$ summarizes the accumulated editing history, $(\alpha_t, \beta_t, \rho_t)$ control the continuous editing behavior, and $\mathbf{m}_t$ specifies sparse module routing. We consider the following variants:

- **SeqConEdit.** The complete state-conditioned policy uses both the current-edit representation and historical state, while jointly adapting the continuous controls and sparse module route at each editing step.
- **w/o history.** We remove the historical summary $\mathbf{h}_t$ from the policy input, preventing the policy from conditioning its decisions on recent editing outcomes and the accumulated editing process.
- **w/o update.** We remove only the accumulated parameter-update signal from $\mathbf{h}_t$ while retaining the remaining historical statistics, isolating whether explicit information about accumulated parameter intervention benefits sequential control.
- **fixed routing.** We fix the sparse module route while retaining state-adaptive continuous controls $(\alpha_t, \beta_t, \rho_t)$, isolating the contribution of adaptive intervention-location selection.
- **fixed controls.** We fix the continuous controls while retaining state-adaptive sparse module routing, isolating the contribution of adaptive regulation over editing strength, regularization, and preservation.

State Information. As shown in Table 4, removing the historical state decreases Avg. from 71.60% to 69.40%, while removing only the accumulated-update signal reduces it to 69.67%. Both variants also exhibit lower Eff., Gen., and Ret., showing that the policy benefits from information about how the editing process has evolved. In particular, the accumulated-update signal provides an explicit indication of preceding parameter intervention, helping subsequent decisions account for the progressively changing model state.

Action Adaptation. Fixing module routing decreases Avg. by 2.08%, while fixing the continuous controls results in a 2.27% drop. Neither restricted policy matches the complete SeqConEdit policy, indicating that adapting only intervention location or only continuous editing behavior is insufficient. These results support the joint adaptation of sparse module routing and continuous controls when balancing knowledge incorporation and preservation over long edit streams.

C.5 More Ablation Studies

Policy optimization strategies. We examine whether the gain of SeqConEdit stems specifically from group-relative advantage estimation rather than from return normalization in general. Let $R_i^{(g)}$ denote the trajectory return of rollout g for edit sequence i. We compare four variants while keeping the controller, reward function, group size, rollout horizon, and all other settings unchanged.

- *Raw Return*: $\hat{A}_i^{(g)} = R_i^{(g)}$, directly using the trajectory return without normalization and therefore retaining both reward-scale variation and sequence-dependent difficulty.
- *Global Norm.*: $\hat{A}_i^{(g)} = \dfrac{R_i^{(g)} - \mu_{\text{global}}}{\sigma_{\text{global}} + \epsilon}$, standardizing returns across all rollouts in the policy-update batch to reduce global reward-scale variation, while still comparing trajectories drawn from editing sequences with potentially different intrinsic difficulty.

Table 5: Ablation of advantage estimation strategies on Llama-3-8B / CounterFact under the $8{,}000\times1$ protocol, isolating the effect of return normalization and sequence-aware advantage estimation.

Advantage Estimation	*Eff.*	*Gen.*	*Spe.*	*Ret.*	*Avg.*
Raw Return	63.91	55.82	49.62	60.46	$57.45_{-3.45}$
Global Norm.	65.42	56.68	48.37	63.05	$58.38_{-2.52}$
Seq.-Centered	67.73	58.01	**51.16**	62.84	$59.94_{-0.97}$
Group-Relative	**69.57**	**59.31**	50.23	**64.50**	**60.90**

- *Seq.-Centered*: $\hat{A}_i^{(g)} = R_i^{(g)} - \bar{R}_i$, centering rollouts generated from the same edit sequence to remove sequence-specific return offsets, thereby reducing the influence of heterogeneous sequence difficulty while preserving the original within-sequence return scale.
- *Group-Relative*: $\hat{A}_i^{(g)} = \dfrac{R_i^{(g)} - \bar{R}_i}{\sigma_i + \epsilon}$, further normalizing the centered returns by their within-group standard deviation, so that policy updates reflect the relative quality of actions under the same editing sequence with reduced sensitivity to sequence-specific return scale.

As shown in Table 5, Global Norm. improves over Raw Return, showing that controlling overall return-scale variation benefits policy optimization. Seq.-Centered yields a further gain, indicating that sequence-dependent difficulty constitutes an important source of optimization noise. The full Group-Relative estimator achieves the best overall performance, reaching an Avg. of 60.90% with the strongest Efficacy, Generalization, and Retention. Although Seq.-Centered obtains slightly higher Specificity, its overall performance remains lower. These results show that sequence-wise centering removes sequence-specific difficulty, while additional within-group variance normalization further improves relative advantage estimation.

C.6 Language-Model Fluency under Long-Range Editing

We further examine whether lifelong sequential editing degrades the model's underlying language-modeling ability using WikiText perplexity (Merity et al., 2016). Specifically, we sequentially edit Gemma-2-9B over the full ZsRE edit set and track WikiText perplexity as edits accumulate. As shown in Figure 8, all methods start from a similar pre-edit perplexity of approximately 11, but their trajectories diverge substantially over the editing process. SeqConEdit remains close to the pre-edit reference, with perplexity increasing only from 11.01% to 12.53%. In contrast, the final perplexity reaches 26.67% for HiEdit, 37.93% for RLEdit, and 45.57% for MALMEN. The widening gap suggests that repeated updates can progressively impair general language-modeling behavior, whereas SeqConEdit more effectively limits accumulated interference. Together with the representation-drift analysis in Appendix C.1, these results further demonstrate the stronger long-term preservation ability of SeqConEdit.

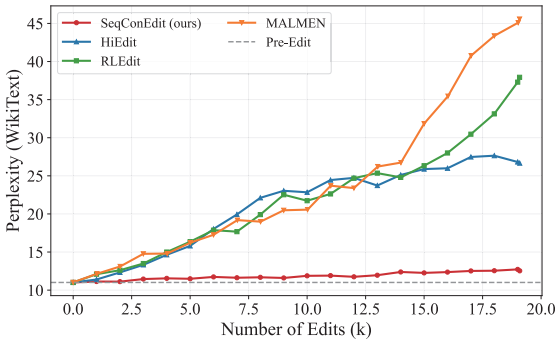


Figure 8: Language-model fluency under long-range sequential editing. WikiText perplexity as sequential edits accumulate. Lower is better.

C.7 Hyperparameter Sensitivity

Preservation trace capacity. Table 6 examines the effect of PTB capacity C while keeping the trained controller fixed. Performance improves consistently as C increases from 0 to 500, with Avg. rising from 54.95% to 60.90%, demonstrating the benefit of retaining historical preservation keys. Beyond this point, larger buffers no longer yield consistent gains. In particular, $C = 5000$ achieves higher Specificity but substantially lower Generalization, suggesting that excessive preservation can overly restrict model plasticity and hinder the incorporation of new knowledge.

Table 6: Sensitivity to PTB capacity on Llama-3-8B / CounterFact under the $8{,}000\times1$ protocol. The trained controller is fixed while only the PTB capacity C varies. † denotes the default capacity.

C	*Eff.*	*Gen.*	*Spe.*	*Ret.*	*Avg.*
0	60.88	49.27	52.58	57.08	$54.95_{-5.95}$
10	64.76	53.42	50.63	60.15	$57.24_{-3.66}$
100	68.03	58.14	49.06	62.91	$59.53_{-1.37}$
500†	69.57	**59.31**	50.23	**64.50**	**60.90**
2000	**70.43**	57.09	49.38	64.26	$60.29_{-0.61}$
5000	68.96	44.50	**55.82**	61.44	$57.68_{-3.22}$

Table 7: Sensitivity to group size G and rollout horizon H on Llama-3-8B / CounterFact under the $8{,}000\times1$ protocol. † denotes the default setting.

Factor	**Value**	*Eff.*	*Gen.*	*Spe.*	*Ret.*	*Avg.*
G	2	67.85	56.69	48.94	63.18	$59.17_{-1.74}$
	4†	**69.57**	**59.31**	50.23	64.50	**60.90**
	8	69.08	58.60	50.51	**64.72**	$60.73_{-0.17}$
	16	66.56	54.76	**51.06**	61.95	$58.58_{-2.32}$
H	100	69.46	58.89	49.15	63.72	$60.31_{-0.59}$
	200†	**69.57**	**59.31**	**50.23**	**64.50**	**60.90**
	500	68.76	57.40	48.62	63.91	$59.67_{-1.23}$

Group size and rollout horizon. Table 7 examines the sensitivity of group-relative policy optimization to the group size G and rollout horizon H. Moderate group sizes provide sufficiently reliable within-sequence comparisons for advantage estimation, while further increasing G brings limited benefit and may reduce editing flexibility. Similarly, the rollout horizon reflects a trade-off between capturing accumulated interference and maintaining tractable temporal credit assignment: short trajectories underrepresent long-range editing effects, whereas overly long trajectories introduce greater optimization difficulty. SeqConEdit remains stable across moderate settings, with $G = 4$ and $H = 200$ providing the strongest overall balance and therefore serving as our default configuration.